HAL
open science

PUFs: Standardization and Evaluation

Jean-Luc Danger, Sylvain Guilley, Philippe Nguyen, Olivier Rioul

To cite this version:

Jean-Luc Danger, Sylvain Guilley, Philippe Nguyen, Olivier Rioul. PUFs: Standardization and Evaluation. 2nd IEEE Workshop on Mobile System Technologies (MST 2016), Sep 2016, Milano, Italy. 10.1109/MST.2016.11 . hal-02288475

HAL Id: hal-02288475

https://telecom-paris.hal.science/hal-02288475v1

Submitted on 12 Aug 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

PUFs: Standardization and Evaluation

Jean-Luc Danger[1,2], Sylvain Guilley[1,2], Philippe Nguyen[1] and Olivier Rioul[2]

[1] Secure-IC S.A.S., 15 Rue Claude Chappe, Bât. B,
ZAC des Champs Blancs, 35510 Cesson-Sévigné, France.
Email: firstname.lastname@secure-ic.com

[2] LTCI, CNRS, Télécom ParisTech,
Université Paris-Saclay, 75 013 Paris, France.
Email: firstname.lastname@telecom-paristech.fr

***Abstract*—Physically unclonable functions (PUFs) serve as untamperable secrets buried in a device that must meet some properties in order to be securely used in cryptographic protocols. Due to the stealthy nature of PUFs, the verification of these properties is in itself a difficult task.**

In this paper, we survey the current trends regarding those aspects. An international standard working draft is described which handles the accurate and consensual definition of the required properties and also covers the high-level methodology related to the test of PUFs. Some in-depth examples of evaluation procedures are also described, both before the fabrication (illustrated on entropy measurement) and after the fabrication (illustrated on reliability analysis—especially how it withstands aging). These evaluations are carried out on the loop-PUF structure, a delay-PUF for which a simple stochastic model can be derived.

***Index Terms*—Physically Unclonable Functions, Delay-PUF, Loop-PUF, Standardization, Security Requirements, Security Properties, Tests, Evaluation.**

I. Introduction

A physically unclonable function (PUF) is a hardwired secret concealed into some hardware. It is unpredictable by design, and thus shares some commonalities with human biometric traits: for instance, the fingerprints of two real twins are different, despite they share the same genetic material. The PUFs are the equivalent of fingerprints for integrated circuits. The genetic material is the design blueprint (e.g., the GDS2 description of the circuit).

Apart from silicon PUFs, which are integrated into electronic circuits, other kinds of PUFs can be envisioned, such as:

- opportunistic (or intrinsic) PUFs, which exploit the dispersion of properties in a device to extract a unique identifier. Such a PUF has been determined for MEMS [1];
- analogue PUFs, such as coating PUFs, which take advantage of the dispersion of a given quantity like capacitance.

For a more comprehensive overview, we refer the interested reader to the Annex B of [2].

Silicon PUFs are instantiations of the same blueprint through a fabrication process: it is expected that each fabricated PUF is independent from the others. Ideally, each PUF property would be identically distributed among PUFs.

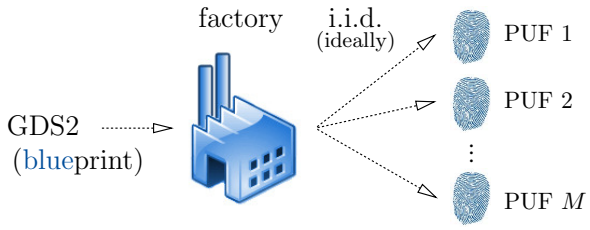

Figure 1. PUFs are instantiations of blueprints by a fab plant

Therefore, it is customary to consider that the fabrication of a batch of $M > 1$ PUFs is equivalent to drawing random variables from a distribution which depends on the blueprint and on the factory. The process is illustrated in Fig. 1. Notice that, ideally, PUFs instances are i.i.d. (independent and identically distributed). However, in practice, this assumption might not hold exactly; still, for the sake for simplicity, we will make this hypothesis in the rest of this paper.

This concept can be used in two different forms:

1) weak PUFs for which only outputs can be measured;
2) strong PUFs which have different outputs depending on chosen inputs.

We denote the inputs by *challenges* and the outputs by *responses*.

a) Usage of PUFs: Any PUF can serve as a building block for many applications. Weak PUFs can generate cryptographic keys. Strong PUFs can serve as authentication functions (hence the terminology "*challenge*" to designate PUF inputs). Alternatively, PUFs can also be used to seed pseudo-random numbers generators, thereby turning them into true random numbers generators.

b) The Challenge of Evaluating and Testing PUFs: A PUF is intended to provide secret information and its responses should also be unique, as requested by the three use-cases just mentioned (key generation, device authentication and seeding). But how can such a secret information be tested? In the context of PUFs, even the designer (or God!) is not supposed to predict the PUF response(s)! If a mechanism is included to read out PUF responses, then how can we make sure that

this information is not copied and programmed into a PUF simulator which would impersonate a genuine PUF? Besides, other properties do not involve *one single* PUF, but rather *a batch* of PUFs. For this reason, testing is made more complex; usual techniques from pooling can be used, but how can one be sure that a PUF from the test pool has the required properties? In the sequel, we make the distinction between tests *before* fabrication and *after* fabrication.

c) Paper Outline: This paper surveys some recent works and researches made at international level. It is organized as follows. First, we discuss the scope of the standardization at ISO level in Sec. II. Then Sec. III determines which method can be used to assess a given PUF property.

In the remainder of the paper, we discuss how some evaluation method can be used to characterize a given property. First, a stochastic model for the entropy (or randomness) for some PUF families is derived and studied in Sec. IV. We show that despite appearances, more than n bits of entropy can be soaked out of an n-stage PUF where each stage is controlled by a single challenge bit. We then tackle the question of reliability, and show in Sec. V how to predict it. The simulations at transistor-level are validated in real experiments in Sec. VI. Conclusions and perspectives are given in Sec. VII.

II. Standardization

In this section, we will review the state-of-the-art about the standardization process of PUFs. Security requirements are now being discussed at ISO sub-committee ISO/IEC JTC1/SC27 (WG3) in working draft 20897 [2].

A. Scope

We hereby quote the ISO 3rd working draft [2]:

"*The International Standard ISO/IEC 20897 (working draft) specifies the security requirements and the test methods for physically unclonable functions for generating non-stored cryptographic parameters. Cryptographic modules generate the certain class of critical security parameters such as a secret key using a random bit generator within the modules. Such modules may store generated security parameters in embedded non-volatile memory elements. For a higher security, a combination of tamper response and zeroization techniques may be used for protecting stored security parameters from active unauthorized attempts of accessing such parameters. As the reverse-engineering technology advances, however, the risk of theft of such stored security parameters has become higher than ever. The rapidly pervading technology called PUF is promising to mitigate the above-mentioned risks by enabling security parameter management without storing such parameters. PUFs are hardware-based one-way functions providing randomness, repeatability and unpredictability of their outputs and unclonability of the functions themselves, taking advantage of intrinsic subtle variations in the device's physical properties, which are also considered object's fingerprints. A PUF takes a bit sequence as an input or challenge and produces another bit sequence as an output or response. The responses are particular to each of different challenges; the correspondence of the challenge and response sets in a single device is unique to the device. PUFs may be used for key generation or random number generation in cryptographic modules. A PUF's output is expected to be random for different challenges and for inter-modules. Another random number or key to be generated from a PUF cannot be estimated from any other output of a PUF. In order to keep using the same generated key, the same challenge only has to be kept. Even if a challenge is leaked out, the corresponding response cannot be estimated without having the same device activated. For those purposes, however, particular caution must be taken due to the PUF's properties. The raw output of a PUF contains a small amount of errors due to subtle fluctuations in the physical properties exploited. Therefore, typically an error correction scheme is combined with the output of a PUF in order to make the output the same every time the same challenge is given. In order to avoid undesired regeneration of the same key or random number, it is required to maintain the challenge value carefully. The purpose of this International Standard (working draft) is to define the security requirements of batches of PUFs and of single instances of PUF, and to specify how to test those security requirements, for assuring an adequate level of quality of the provided PUFs in cryptographic modules.*"

B. Definitions

For the purposes of the standard (working draft), some terms need to be defined:

1) **Randomness / Unpredictability**: The property that PUFs responses are not predictable (as would be with a serial number, incremented for each device).
2) **Uniqueness**: The property that the probability that a batch of PUFs provides responses with collisions is not higher than pure chance.
3) **Reliability / Stability / Steadiness / Repeatability**: The property that a response for a given input is replied in a deterministic manner. (This property is not only a safety property, because a unreliable PUF can accidentally provide the output of another PUF; alternatively, an erroneous response in the case of a key generation has the effect as a *fault injection* attack).
4) **Diffuseness**: The property that the PUF's response to a challenge is unrelated to the challenge, e.g., it is hard to infer an answer from a set of known challenge response pairs[1].

In addition, let us not forget the **unclonability** (currently addressed marginally in [2]). It is also a property of PUF that satisfies the fact that it must be hard to reproduce the functionality of a PUF.

In the sequel, we will especially focus on two properties of *randomness* and *reliability*. Indeed, unclonability is *design-level* property, which can hardly be apprehended only through attacks. Uniqueness is a consequence of randomness. Diffuseness can be artificially improved at the design-level, e.g., by encrypting or hashing the PUF responses.

[1]Currently, the most powerful such attacks on PUFs rely on machine learning techniques, as for instance illustrated in [3].

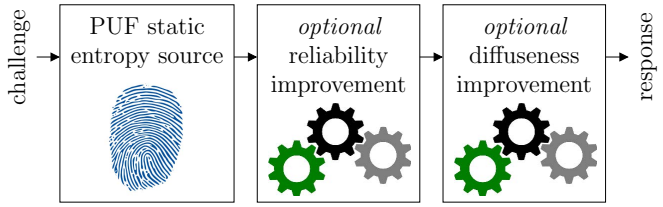

Figure 2. Example of a functional model of a PUF

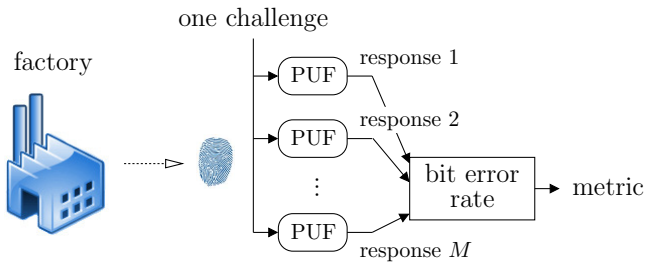

Figure 3. Test of the reliability of one PUF instance

We notice that some notions have been recently analyzed theoretically in [4], where in addition some links are proven. Clearly, this work will help better understand the meaning of the metric and their inter-relationships.

C. Boundaries of the PUF

The PUF itself can be seen as a source of static entropy: all the randomness is supposed to disappear as soon as it is fabricated. The PUF can thus have some defects, it might be interesting to compensate. Therefore, some pre- or post-processing blocks can be added to the PUF. For instance, Fig. 2 illustrates the augmentation of a vanilla PUF with two blocks:

- reliability improvement: this consists in some error correction at the output of the PUF, or in the repetition of some responses to given challenges;
- diffuseness improvement: this block increases the insufficient differences in the responses under different challenges by encrypting or hashing the responses. Other options consists in using the Von Neumann debiasing method.

One example of pre-processing (not illustrated in Fig. 2) would be a selection of suitable challenges, like will be discussed in Sec. IV. Depending on the level of expected confidence in the PUF properties, tests can be carried out only on the static entropy source or on the complete assembly.

D. Test methods

The test method for one PUF or a batch of PUFs is depicted in Fig. 3 regarding the reliability, Fig. 4 regarding the unpredictability, and Fig. 5 regarding the diffuseness.

Example values are:

- **unpredictability**: entropy of 128 bit (or more);
- **reliability**: bit error rate of 10^{-9} (or less);

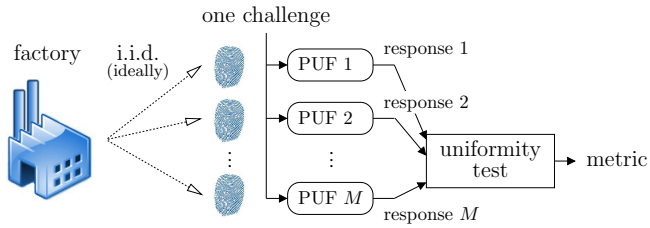

Figure 4. Test of the unpredictability and uniqueness of a batch of M PUFs; the *uniformity test* can also be a measure of *entropy*

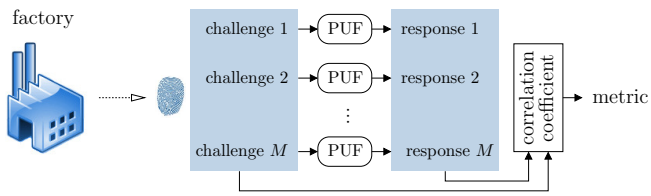

Figure 5. Test of the diffuseness of one PUF instance

- **diffuseness**: Pearson correlation coefficient between challenges and responses of 10^{-9} (or less).

E. Other topics of interest regarding the PUFs

The topic of PUF goes far beyond their security. For instance, an interesting problematic regarding their practical deployment is related to their life-cycle. Indeed, before being deployed on the field, a PUF passes through several stages, listed below; These aspects are tackled in the informative annex E of [2]:

1) Design of the rationale;
2) Design of the layout (*including some physical protection, such as active shield [5]*), and analysis of *non-interference* (*see for instance this study [6]*),
3) Tape-out;
4) Physical integrity test, which allows to discard non-functional instances;
5) Characterization,
   - which is required to produce the helper data, and
   - to define the correction capability of the PUF needed to choose the ECC parameters.
6) Save of the characteristics into a tamper-proof area, e.g., protected by HMAC or a digital signature;
7) Deployment.

III. Evaluation Methodology

A. Evaluation techniques

The evaluator has several means to assess the quality of a PUF. Those are represented in Tab. I, with some (non-exhaustive) examples of publications which tackle the question. PUF security metrics can be analyzed at two levels:

- theoretically, by a *stochastic* or *formal* model, and
- empirically, by the estimations on the final product.

Table I
EVALUATION METHODS VS PROPERTIES OF A PUF

Method \ Property	Randomness	Uniqueness	Reliability	Diffuseness	Unclonability
Mathematical method	[7]				
Design analysis	[8]	[8]	[8]		
Simulation			[9]		
Real-world measurements			[9]		
Real-world attacks					[10], [11]

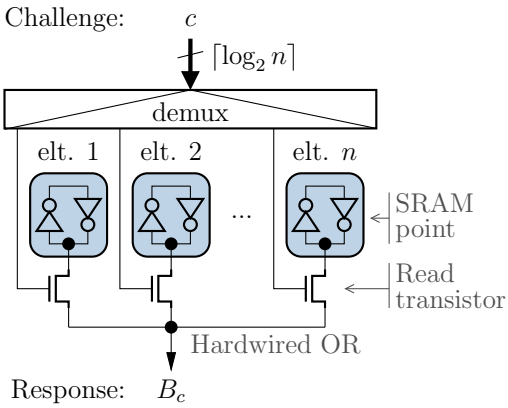

Figure 6. Non-delay PUF: the SRAM-PUF

B. Running example: loop-PUF

In the sequel, we illustrate in particular the evaluation and the test[2] of some PUF security requirements (those represented in gray cell in Tab. I), on the example of a delay-PUF, namely the loop-PUF [12].

The figure 6 depicts an SRAM-PUF, which can be seen as the parallel composition of n single-bit weak PUFs consisting in SRAM memory points.

The loop-PUF is depicted in Fig. 7. Its architecture consists in n single-bit delay elements controlled by one bit of challenge (see Fig. 8), which are chained. Each one-bit element is composed of two identical structures (namely:), obtained by copy-and-paste, and controlled by opposed bits c_i and $\neg c_i$. The reason for this duplication is that the two inputs of a multiplexer are not equivalent in most CMOS design kits (in particular, delay-wise). Therefore, it is important that the delay path travels *successively* through both inputs of the multiplexer (see in Fig. 9 how the total delay is *artificially* balanced).

In both the SRAM-PUF and the loop-PUF structures, the wires represented in bold are buses (vector of bits), whereas thin wires represent bits.

[2]It is customary to use the term *test* for verifications which can be automated by a tool, which contrasts with *evaluation*, for which an expert human being uses his judgment to determine the most likely attack path and the associated verification methodology.

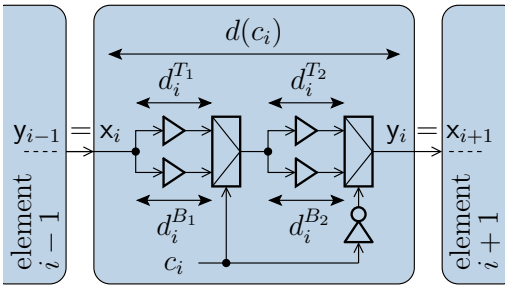

Figure 8. Loop-PUF example: core delay element

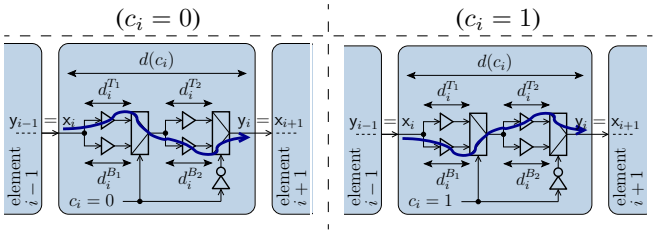

Figure 9. Loop-PUF: core delay element with **balanced paths**

Notwithstanding the sophistication of Fig. 8, the delay element, for a given challenge bit c_i, follows a normal distribution. Notice that the loop-PUF estimates the propagation time of the chain by closing it, to form a loop. This loop is dimensioned in such as way its period $d(c) = \sum_{i=1}^{n} d(c_i)$ is greater than the clock period T_{clk}. The design shown in Fig. 8 features two counters running in parallel:

- one which is *synchronous* with the clock, and
- another one which is *asynchronous* with the clock, but instead incremented each time the *token* in the loop has made two iterations (indeed, the counters trigger on *rising edges* of their sampling signal, hence only on *every other phase*).

Therefore, after N clock cycles,

- the *synchronous* counter simply contains N, whereas
- the *asynchronous* counter contains some integer $D(c)$.

As both counters measure the same time, we have that $D(c)$ is the largest integer such that $2D(c) \times d(c)$ is smaller than $N \times T_{\text{clk}}$. Thus, one can deduce that $D(c)$ is equal to $\left\lfloor N \times \frac{T_{\text{clk}}}{2d(c)} \right\rfloor$. And indeed, one can see in Fig. 8 that this value kept in the *asynchronous* counter is registered as the output after a total of N clock cycles.

Notice that the hardware structure represented in Fig. 7 is not self-contained: it needs to be operated by a protocole, given in Alg. 1. Notice that the result returned at line 5 or Alg. 1 can simplified independently of N and T_{clk} when N grows

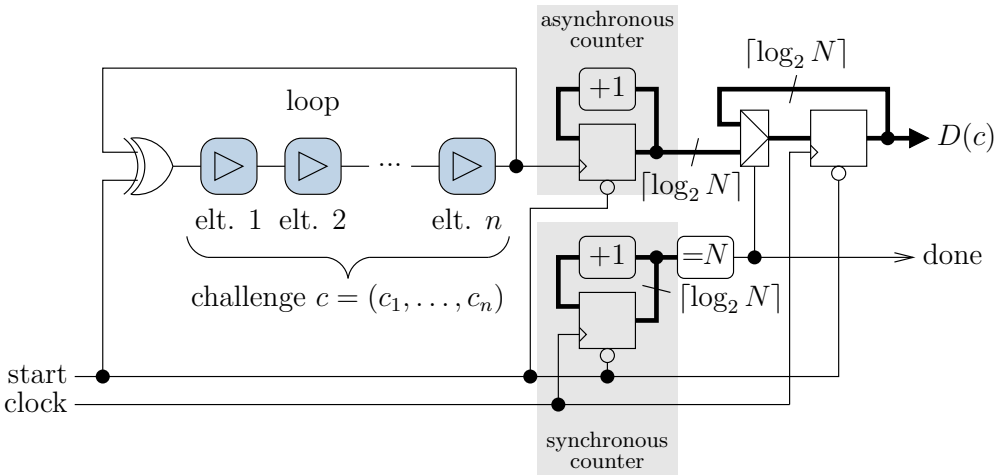

Figure 7. Delay-PUF example: the loop-PUF [12]

larger and larger:

$$\begin{aligned}
\operatorname{sign}(D(\neg c) - D(c)) &= \operatorname{sign}\left(\left\lfloor N\frac{T_{\text{clk}}}{2d(\neg c)}\right\rfloor - \left\lfloor N\frac{T_{\text{clk}}}{2d(c)}\right\rfloor\right) \\
&= \operatorname{sign}\left(\frac{1}{N}\left\lfloor N\frac{T_{\text{clk}}}{2d(\neg c)}\right\rfloor - \frac{1}{N}\left\lfloor N\frac{T_{\text{clk}}}{2d(c)}\right\rfloor\right) \\
&\xrightarrow[N\to+\infty]{} \operatorname{sign}\left(\frac{T_{\text{clk}}}{2d(\neg c)} - \frac{T_{\text{clk}}}{2d(c)}\right) \\
&= \operatorname{sign}\left(\frac{1}{d(\neg c)} - \frac{1}{d(c)}\right) = \operatorname{sign}\left(d(c) - d(\neg c)\right).
\end{aligned}$$

Hence, approximatively (for large values of N), the setup of Fig. 7 combined with mode of operation of Alg. 1 estimates the outcome of the comparison $d(c) \gtrless d(\neg c)$.

input : Challenge c (a string of n bits)
output: Response B_c

1 Set challenge c
2 Measure $D(c) \leftarrow \lfloor N\frac{T_{\text{clk}}}{2\sum_{i=1}^{n} d(c_i)}\rfloor$ ▷ Using Fig. 7
3 Set challenge $\neg c$
4 Measure $D(\neg c) \leftarrow \lfloor N\frac{T_{\text{clk}}}{2\sum_{i=1}^{n} d(\neg c_i)}\rfloor$ ▷ Using Fig. 7
5 **return** $B_c = \operatorname{sign}(D(\neg c) - D(c))$

Algorithm 1: Protocole to get one bit B_c out the loop-PUF using challenge c

Remark 1. *It can be seen in Fig. 7 that the registered output $D(c)$ is multi-bit. This calls for two comments. First of all, the execution time of Alg. 1, which is about $2N$ (time necessary to get $D(c)$ and $D(\neg c)$), could be reduced to $2N' < 2N$ if the estimation has already converged in such a way the difference between $D(c)$ and $D(\neg c)$ clearly stands out. Second, assuming the execution time for getting $D(c)$ and $D(\neg c)$ remains N, Alg. 1 could benefit from extracting not only one bit as $\operatorname{sign}(D(\neg c) - D(c))$, but more bits of less significant weight.*

We derive some of the loop-PUF metrics, namely its randomness in Sec. IV and its reliability, including its alteration in the presence of aging (both using simulations in Sec. V and on a real ASIC (Application Specific Integrated Circuit) in Sec. VI).

IV. Mathematical model for the entropy

This section emphasizes results obtained and proved in [7] for a n-delay PUF structure and discusses their consequences.

A. Entropy of the SRAM-PUF and the loop-PUF

It is easy to check that the amount of entropy of the SRAM-PUF (Fig. 6) is at most n. In contrast, the amount of entropy of the loop-PUF can be strictly greater than n.

Regarding the loop-PUF, we have mathematically derived its entropy [7] in the case $N \longrightarrow +\infty$ (perfect measurement of $d(c)$). We show that n bits of entropy can be obtained with n challenges if and only if the challenges constitute a Hadamard code. In addition, we demonstrate that adding more challenges results in an entropy strictly greater than n bits. A greedy code construction is provided for this purpose. When n is a power of two, heuristic results indicate the challenge code is constituted of additional non-orthogonal challenges from several other Hadamard matrices.

For example, when $n = 8$ single-bit sources of static entropy are instantiated, the entropy of the SRAM-PUF and the loop-PUF are represented in Fig. 10 and 11 respectively.

B. Discussion

Thus, not only the stochastic model allows to ascertain a given level of entropy (which would be hard to estimate with the test of Fig. 4), but also, it allows to take as much as possible advantage of the available hardware. The stochastic model also allows for a reasoned trade-offs study. In the case of the loop-PUF, the larger n, the more entropy can be recovered per delay element (Fig. 8), but also the slower the measurements ($d(c)$

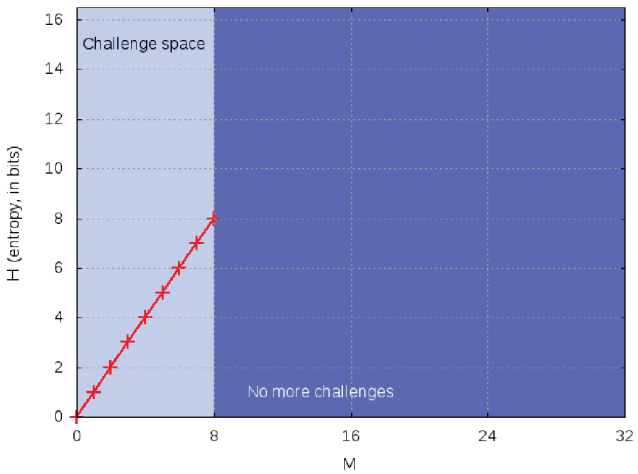

Figure 10. Entropy of the SRAM-PUF, with respect to the number of challenges ($0 \leq M \leq n$)

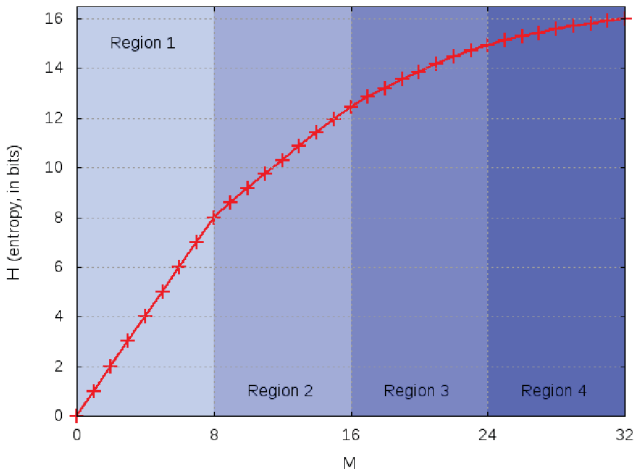

Figure 11. Entropy of the loop-PUF, with respect to the number of challenges ($0 \leq M \leq 2^{n-1}$)

grows linearly with n). Therefore, the parameter n allows to trade speed for entropy, in a non-linear way.

Incidentally, we notice that similar models apply to other PUFs, where the measured quantity depends "*additively*" (as propagation times in a delay-PUF) on the intrinsic properties of the material making up the PUF.

V. Simulation for the Reliability over Aging

We show in [9] using MOSRA (Synopsys "*MOS Reliability Analysis*") to simulate the aging of the loop-PUF that the variance of $d(c_i)$ changes, but not its mean. Hence both the reliability and the entropy of the loop-PUF are steady over time. We also shown that the PUFs using sequential elements (latch or SRAM point) like the arbiter-PUF or the SRAM-PUF are more prone to aging.

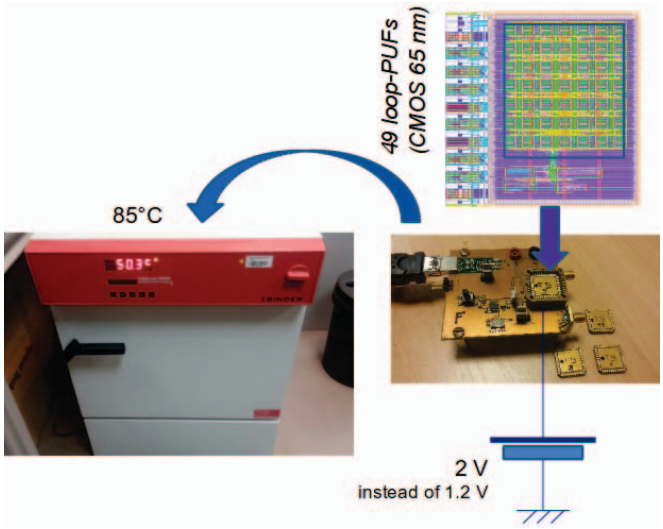

Figure 12. Setup for experimental accelerated aging

VI. Measurement for the Reliability over Aging

The simulation results are reproduced on an ASIC, designed in STMicroelectronics 65 nm process. Measurements presented in [9] (see Fig. 12: the stress is $(2\ \mathrm{V}, 85°\mathrm{C})$ for 23 hours, and then a recovery period at nominal conditions, namely $(1.2\ \mathrm{V}, 20°\mathrm{C})$ for 1 hour) reveal that the aging increases the delays variance (especially in the next weeks which follow fabrication), but that the average delays remains stable. This validates a posteriori the predictive simulations made in Sec. V.

VII. Conclusions and Perspectives

In the way they work, PUFs obviously share some aspects of biometric systems. In this regard, the main problem is the probability of false / true positives / negatives. By mistake, a wrong fingerprint can be considered as correct; therefore, it is important to measure the reliability of PUFs.

PUFs also have some characteristics shared with TRNG (True Random Numbers Generators). Indeed, it is very hard, if not impossible, to decide whether a so-called random sequence of bits has been generated by fresh entropy or by some deterministic process. Regarding TRNG, a solution to strengthen the trust one has in its entropy is the design of a probabilistic model. Provided the real hardware matched the model, and so provided the model is properly parameterized, the properties of the TRNG at stochastic level (or mathematical model) and in real should be alike.

Therefore, taking into account the standardization process and the similarity of evaluation methods with known use-cases, we expect that the trust one has in PUFs will greatly increase in the future, and that as a result its deployment will blossom.

However, we notice that some aspects are rarely discussed in today's open literature, such as the test (before fabrication and when the product is in the field) of PUF functionality and

properties. Ideally, all the stages mentioned in Sec. II-E could be better formalized and evaluated.

Acknowledgments

This work was supported by the KeyHAS project, the R&D program of IITP/MSIP (Study on secure key hiding technology for IoT devices).

References

[1] O. Willers, C. Huth, J. Guajardo, and H. Seidel, "MEMS-based Gyroscopes as Physical Unclonable Functions," *IACR Cryptology ePrint Archive*, vol. 2016, p. 261, 2016. [Online]. Available: http://eprint.iacr.org/2016/261

[2] S. Guilley, S. Hamaguchi, and Y. Kang, "ISO/IEC NP 20897. Information technology – Security techniques – Security requirements, test and evaluation methods for physically unclonable functions for generating nonstored security parameters," http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=69403.

[3] F. Ganji, S. Tajik, F. Fäßler, and J. Seifert, "Strong Machine Learning Attack Against PUFs with No Mathematical Model," in *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, ser. Lecture Notes in Computer Science, B. Gierlichs and A. Y. Poschmann, Eds., vol. 9813. Springer, 2016, pp. 391–411. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-53140-2_19

[4] F. Armknecht, D. Moriyama, A. Sadeghi, and M. Yung, "Towards a Unified Security Model for Physically Unclonable Functions," in *Topics in Cryptology - CT-RSA 2016 - The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29 - March 4, 2016, Proceedings*, ser. Lecture Notes in Computer Science, K. Sako, Ed., vol. 9610. Springer, 2016, pp. 271–287. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-29485-8_16

[5] J. Cioranesco, J. Danger, T. Graba, S. Guilley, Y. Mathieu, D. Naccache, and X. T. Ngo, "Cryptographically Secure Shields," in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2014, Arlington, VA, USA, May 6-7, 2014*. IEEE Computer Society, 2014, pp. 25–31. [Online]. Available: http://dx.doi.org/10.1109/HST.2014.6855563

[6] X. T. Ngo, J. Danger, S. Guilley, T. Graba, Y. Mathieu, Z. Najm, and S. Bhasin, "Cryptographically Secure Shield for Security IPs Protection," *IEEE Trans. Computers*, vol. 66, no. 2, pp. 354–360, 2017. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TC.2016.2584041

[7] O. Rioul, P. Solé, S. Guilley, and J. Danger, "On the entropy of Physically Unclonable Functions," in *IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016*. IEEE, 2016, pp. 2928–2932. [Online]. Available: http://dx.doi.org/10.1109/ISIT.2016.7541835

[8] D. P. Sahoo, P. H. Nguyen, R. S. Chakraborty, and D. Mukhopadhyay, "Architectural Bias: a Novel Statistical Metric to Evaluate Arbiter PUF Variants," Cryptology ePrint Archive, Report 2016/057, 2016, http://eprint.iacr.org/2016/057.

[9] N. Karimi, J.-L. Danger, S. Guilley, and F. Lozac'h, "Predictive Aging of Reliability of two Delay PUFs," in *Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings*, ser. Lecture Notes in Computer Science, vol. 10076. Springer, 2016.

[10] D. Nedospasov, J. Seifert, C. Helfmeier, and C. Boit, "Invasive PUF analysis," in *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography, Los Alamitos, CA, USA, August 20, 2013*, W. Fischer and J.-M. Schmidt, Eds. IEEE, 2013, pp. 30–38, Santa Barbara, CA, USA. [Online]. Available: http://dx.doi.org/10.1109/FDTC.2013.19

[11] C. Helfmeier, C. Boit, D. Nedospasov, and J.-P. Seifert, "Cloning Physically Unclonable Functions," in *Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium on*, 2013, pp. 1–6.

[12] Z. Cherif, J. Danger, S. Guilley, and L. Bossuet, "An easy-to-design PUF based on a single oscillator: The loop PUF," in *15th Euromicro Conference on Digital System Design, DSD 2012, Çeşme, Izmir, Turkey, September 5-8, 2012*. IEEE Computer Society, 2012, pp. 156–162. [Online]. Available: http://dx.doi.org/10.1109/DSD.2012.22