HAL
open science

Highly Reliable PUFs for Embedded Systems, Protected Against Tampering

Jean-Luc Danger, Sylvain Guilley, Michael Pehl, Sophiane Senni, Youssef Souissi

To cite this version:

Jean-Luc Danger, Sylvain Guilley, Michael Pehl, Sophiane Senni, Youssef Souissi. Highly Reliable PUFs for Embedded Systems, Protected Against Tampering. INISCOM 2021 : International Conference on Industrial Networks and Intelligent Systems, Apr 2021, Hanoi (Vietnam), Vietnam. pp.167-184, 10.1007/978-3-030-77424-0_14 . hal-03783308

HAL Id: hal-03783308

https://cnrs.hal.science/hal-03783308v1

Submitted on 22 Sep 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.

Highly Reliable PUFs for Embedded Systems, Protected Against Tampering

Jean-Luc Danger[1,2], Sylvain Guilley[2,1][0000−0002−5044−3534], Michael Pehl[3], Sophiane Senni[2], and Youssef Souissi[2]

[1] LTCI, Télécom Paris, Institut Polytechnique de Paris,
Palaiseau, France,
firstname.lastname@telecom-paris.fr
[2] Secure-IC S.A.S.,
Tour Montparnasse, 33 avenue du Maine, 75015, Paris, France,
firstname.lastname@secure-ic.com
[3] Technical University of Munich (TUM),
Department of Electrical and Computer Engineering,
Chair of Security in Information Technology
m.pehl@tum.de

Abstract. Physically Unclonable Functions (PUFs) are well-known to be solutions for silicon-level anti-copy applications. However, as they are sensitive components, they are the obvious target of physical attacks. Thus, they shall be well protected. In this work we discuss the use case of key generation with a Loop PUF. We discuss the Loop PUF's efficiency and efficacy. We analyze it with respect to several known attacks like side-channel and machine learning attacks, and show that in all considered cases it either natively resists or can be protected. We also show that perturbation attempts should be within the scope of likely attacks, hence the PUF shall be protected against tampering attacks as well. Also for this attack scenario we highlight the salient features of the Loop PUF and explain how its mode of operation natively empowers it to resist such attacks.

Keywords: Physically Unclonable Function (PUF), Loop PUF, Internet-of-Things (IoT), Dependability, Anti-Copy, Root-of-Trust, Tamper-Proof.

1 Introduction

Physically Unclonable Functions (PUFs) are hardware structures that allow generating unique per chip values. This property arises from the amplification of tiny manufacturing variations, which become unambiguously quantifiable this way. The unique values derived by a PUF can be seen as "non-stored critical security parameters" and are used in different security applications detailed below.

The PUF's correct operation requires some pre-silicon dimensioning and even some post-silicon configuration (e.g., helper data generation and storage). The former prevents systematic bias and correlations predominantly through careful

place and route, the latter helps to eliminate, e.g., remaining bias [17] and to guarantee high reliability. Although this life cycle management seems complex at a first glance, it fits nicely into that of today's system-on-chip (SoC) deployment model. Therefore, inclusion of PUFs into a SoC is not adding extra steps compared to those already in place in the industry.

PUFs are experiencing many commercial successes as intellectual property (IP) blocks. Large deployment is motivated by the need for defense in depth, but also because the PUFs allows for new use-cases: The key derived from a PUF, e.g., is not *injected*, it is rather *extracted* from silicon, which eases its management.

Owing to this successful dissemination, the standardization of PUFs has been following suit. In particular, at international standards organization (ISO), the project ISO/IEC 20897 is targeting to setting the ground for fair security evaluation of PUFs. Part 1 introduces the security problems and defines the goals. Part 2 tackles test and evaluation methodologies, in particular through quantitative metrics. The achievable security level of a PUF self-evidently depends on the PUF design itself as well as on possibly required processing of the PUF output, the PUF response. But also the use case plays an important role in the security considerations of the PUF based system. In this paper, we focus on the use case of deriving a device unique secret key from a PUF and discuss different aspects of this scenario given a specific PUF primitive, the *Loop PUF*.

Outline. The rest of the work is structured as follows: First, the motivation for using a PUF is provided in Sec. 2 together with an exemplar use case where the PUF is a cornerstone of the security. The model given for this use case simplifies the risk management in that only the PUF security shall be ensured, against all attacks across the board. Sec. 3 introduces the Loop PUF, the PUF used through this work as an architecture which complies to this requirement. The analysis of the reliability and the security of the Loop PUF is carried out afterwards. In Sec. 4 and Sec. 5 the Loop PUF's resistance against side-channel attacks and, respectively, machine learning attacks is discussed. Sec. 6 shows the innate advantage of the Loop PUF even if put at risk by tampering attempts. Eventually, Sec. 7 draws the conclusions of this paper.

2 Use Case

2.1 PUF use cases

Over the last years, many different applications have been suggested for PUFs. The most prominent of them are the use of PUFs in challenge-response protocols [26,33] and as Physically Obfuscated Key (POK) [9,18,13], i.e., to generate and to store keys. Further applications include key-based protocols [1,8] and key-exchange [24]. A generalization of theses and further use cases is provided in [3]. The generalization specifies in particular four PUF applications, which use the PUF

- To generate critical security parameters. This is the scenario, where the PUF is used to store or to derive a secret key.

- As device unfalsified identifier.
- For device authentication through a challenge-response protocol.
- For random source seeding.

2.2 PUF as a Master Key

We focus in the sequel on the first use-case. More precisely, we will investigate an innovative configuration for the case that an Internet of Things (IoT) device and its managing party shall be enabled to share a secret key. The PUF is used in the scenario to store a private key with the goal to reduce the attack surface.

Device Authentication without PUF In order to show the benefit of the PUF, consider the situation **without a PUF** first. This scenario is described in Fig. 1(a): A security owner generates a *secret key*, which is *provisioned* both to the chip and to the chip managing party. Therefore, an untrusted party can compromise the security model. For instance, since the security owner is not authenticated, a corrupted manager can provision several chips with the same key. Therefore, the dishonest manager in this "overbuilding" scenario can build indistinguishable clones of the IoT device when colluding with the foundry.

The regular process, however, consists in injecting a secret key both in the chip and in the manager. After this initial stage, the security owner, i.e., the trusted third party, is no longer needed. Rather, the provisioned chip can directly engage a communication based on a symmetric authentication protocol using the pre-shared key. The symmetric authentication protocol in this scenario is a strict requirement to ensure security both at end-point and at security manager side. One example for a (purely cryptographical) symmetric mutual authentication scheme is that the chip sends a random number[1] to the manager to check whether he is able to encrypt it correctly with the pre-shared secret key (remote authentication in Fig. 1). Vice versa, the manager sends a random number to the chip, to check that the chip encrypts it with the correct secret key (chip/IoT device authentication in Fig. 1).

Beyond the already discussed "overbuilding" risk, the described setup implies that both the end-point and the management company are properly protected against any attacker trying to come into the possession of the shared secret. This means, in particular: Also the keys managing company shall be protected, which happens to be not so obvious as the hack of Gemalto [19] shows, for example.

Device Authentication with PUF: Our Use Case For this reason, there is a trend to transition from symmetric to asymmetric solutions. In the asymmetric case, each party has its own key pair (public and private key). The role of the

[1] The random number is usually termed a "challenge", and the encrypted response using the share secret key is customarily referred to as the "response". Please be aware that those shall not be confused with PUF challenge and responses. Indeed, there is no PUF in Fig. 1(a).

third party is to endorse the chip, by signing its public key, which yields a certificate (in the sense of ITU-T X.509). Similarly, in the situation where the chip is expected to authenticate its manager, the third-party security owner shall also generate a certificate for the manager key pair. Therefore, later on, the third-party is no longer needed, and the chip and the manager can establish a secure channel using the respective key pairs, and can validate the other parties public key thanks to the certificate. For the sake of completeness, let us mention that in asymmetric authentication schemes each party has to prove it knows the private key associated with its public key. Again, and like for the symmetric case, each unilateral authentication is based on a challenge-response protocol [1].

Please note that the asymmetric scenario implies that the chip is able to generate its own key pair. This is not taken for granted. Indeed, in the silicon manufacturing industry, chips are produced to be exact "functional" clones one of each other. Nevertheless, all chips differ from each other in practice due to minuscule manufacturing variations, which are the base to implement a PUF. The role of the PUF in the scenario **with a PUF** is to generate the chip unique private key. As a consequence, the end-point (IoT) device is the sole responsible for managing its root-of-trust key in this asymmetric case and the private key never leaves the device. This is illustrated in Fig. 1(b).

Obviously, in the novel use-case of Fig. 1(b), all the security of the IoT device is concentrated into the PUF itself. I.e., the PUF is the sole IP responsible for its security when in the field. Therefore, the PUF is a potential target of attacks. Notably, the PUF is also prone to physical attacks, since its application in the field allows the device to be tampered with from an attacker's perspective.

2.3 Motivation for this work

The described scenario provides the motivation for this work: First, the PUF in this setting has to provide an unpredictable private key, which remains constant over time. I.e., the PUF shall be unique and reliable and shall not expose any information that helps an attacker to guess the PUF. Second, an attacker should not be able to extract the secret from the device. I.e., the PUF has to resist any form of tampering attempts, in particular it has to resist attacks observing side-channel leakage as well as perturbation attacks.

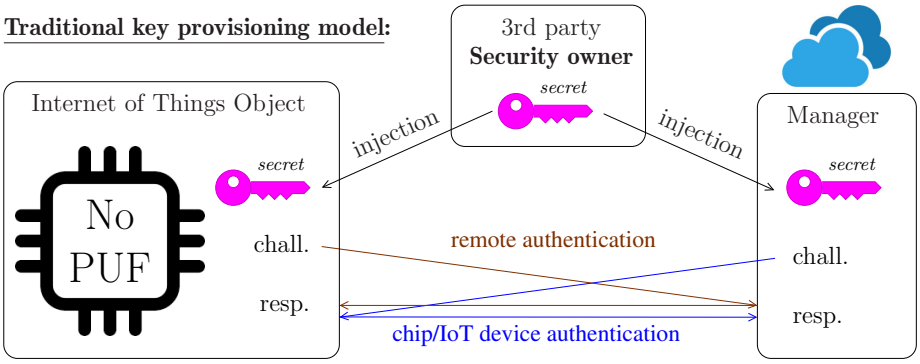
3 One Solution Based on the Loop PUF

The following shows that the requirements provided for the use case in Sec. 2 can be fulfilled with a real PUF. For this purpose, we consider the case of Loop PUF as one PUF primitive.

3.1 The Loop PUF Rationale

The Loop PUF [4] relies on a single loop structure. This loop is composed of n switch elements, which each allow to select a path (direct or crossed). Thus,

(a) Traditional use case (no PUF), with symmetric authentication.



(b) Novel use case implemented by Secure-IC (leveraging a PUF), with asymmetric authentication.

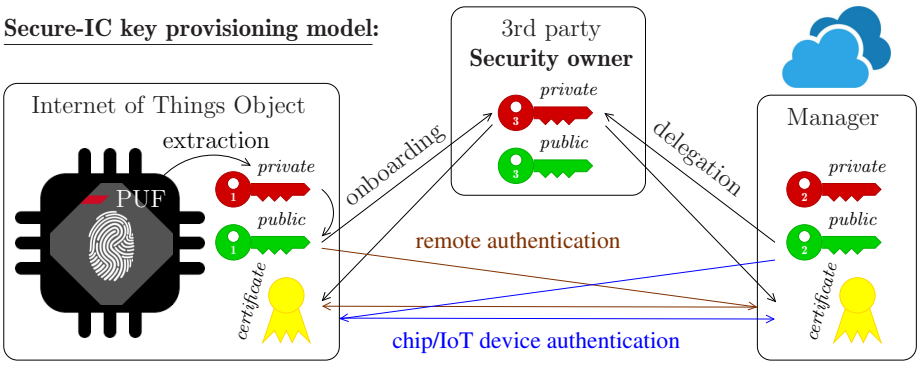


Fig. 1. Provisioning schemes, (a) traditional = symmetrical, (b) promoted by Secure-IC = asymmetrical. The third party is involved only at the key injection/extraction phases; afterwards, the manager is able to uniquely access its IoT object through a challenge/response protocol.

the Loop PUF is strongly related to the Arbiter PUF [23]. However, while the signal traverses the path from the input to the output only once in the Arbiter PUF, it is inverted and feed back for the Loop PUF so that it repeatedly passes through the same structure. This way an oscillator is formed and the delay of a specific configuration is measured multiple times, accumulating process specific variations and decreasing the influence of noise. At the output the number of oscillations in a fixed period of time – the measurement time – is counted, which can be translated into a delay of the structure. The described process allows for the structure to improve the SNR for delay differences by increasing the measurement time.

To derive a secret bit from the Loop PUF, it is measured under two different challenges, i.e., two different configurations of the delay path. A bit is derived by comparing the difference of the delays, or equivalently the difference of the counter values, under the two challenges. The suggested method to select challenges for a Loop PUF is to derive the first challenge c_i based on Hadamard codes [27]; The second challenge $\neg c_i$ to compare with is the bitwise inverse of c. This way, as many challenge pairs $(c_i, \neg c_i)$ as stages are derived. Furthermore, the configurations of the Loop PUF between measurements differ as much as possible since all challenges c_i have Hamming distance $\frac{n}{2}$.

The Loop PUF has been well studied and, in particular, comes with a *stochastic model*. This formal model describes its expected properties. The Loop PUF shall be *dependable* [29], in particular:

- Its *entropy* shall be known[4]. An accurate analysis of the entropy of Loop PUFs is provided in [30];
- Its *reliability* shall be known. An accurate analysis of the reliability of Loop PUFs is provided in [28].

3.2 The Loop PUF Life-Cycle

Beyond the reliability achieved for a specific, design-time chosen measurement time, a self-assessment post-silicon test can further reduce the error rate in the Loop PUF's response. This test stage is customarily referred to as the *enrollment* in PUF parlance. This *enrollment* phase is to generate a reference key from the PUF and a "helper data" which is a public word to reconstruct the PUF key which may have faulty bits due to the extraction noise. This enrollment phase happens once just after the fabrication of the device. Fig. 2 illustrates the two phases of enrollment and inference during which the PUF key is reconstructed by means of the helper data.

A simple helper data is to point out the most unreliable PUF bits in order to discard them during the inference phase. As the Loop PUF generates not

[4] Notice that the entropy of the Loop PUF as well as of all strong PUF can be modelled reliably. Indeed, the number of challenge-response pairs is exponential in the number of delay elements. Thus Loop PUFs can be tested in principal using methods such as NIST SP 800-22 if a sufficient amount of carefully selected challenges is used to generate responses.

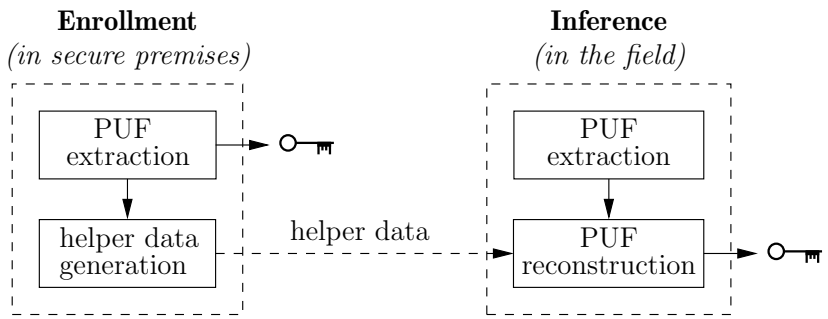


Fig. 2. The two phases of the Loop PUF life cycle to generate a private key.

only bit values but also their associated reliability level, this helper data is easy to obtain. The assessment of reliability by using the helper data to filter out unreliable elements is formalized in [30]. More details regarding the enrollment phases can be found in [25]. An attack on the helper data is possible if the attacker can read and modify it, as explained in [7]. The same paper also shows that the attack is mitigated by using a double metric to extract the PUF response.

An efficient alternative to improve the reliability of the secret derived from the PUF is to generate an helper data by taking advantage of error correcting codes like in [10]. Sec. 4.3 discusses an attack using these helper data.

3.3 Test of Loop PUFs

Obviously, the PUF has become an industrial technology, meaning that it shall be tested. The tests shall consider the following two cases:

1. **Structural integrity test**, based on JTAG, can be implemented on the Loop PUF. This requires a simple mechanism to open the combinational loop. Such test allows to cover:
   - manufacturing issues, and
   - post deployment silicon manipulation detection.
2. **Behavioral test** to check *in situ* for security properties:
   - Health tests, and
   - Entropy test [6], using a partitioning of challenges between *application* and *service* challenges.

Effectively, only with these tests, not only the functional issues, but also security critical deviations in the PUF quality are detected.

3.4 Resistance to Attacks

From the sections above, we conclude, that the Loop PUF as a solution has all the requirements from the functional side and documentation side (for compliance

to standards). It also comes with an accurate calibration procedure and testing capabilities. In the next section we discuss its robustness against attacks. In particular we detail in the next Sec. 4 the resistance to side-channel attack. Since the Loop PUF falls into the category of strong PUFs by nature, we also discuss its robustness to machine learning attacks in Sec. 5. Finally, we provide in Sec. 6 the explanation for the resistance against perturbation attacks.

4 The Loop PUF Security in Front of Side-Channel Attacks

Two scenarios of side-channel attacks are usually envisioned in the PUF context: The prediction of responses

- from unseen challenges based on the previously learned challenge-response pairs (see Sec. 4.1).
- from the power traces of PUF after having learned the relationship between some power traces and some responses (see Sec. 4.2).

The second scenario can be extended from attacks on the PUF itself to attacks on the processing of the PUF response (see Sec. 4.3)

4.1 Side-Channel Attacks on the Challenge-Response Pairs

This scenario assumes that an attacker can predict the PUF response for new challenge response pairs by observing – in this case via a side-channel – responses of previous challenges. Given these challenge-response pairs, e.g., machine learning can be applied to model the PUF and to eventually predict the responses also for unseen challenges. The attack scenario implies, that an attacker is able to configure the PUF with an unlimited amount of challenges since she is able to first train a model and has challenges left for which the response must be predicted. However, in the use-case of master key generation, this is not the case. In particular, the challenges shall be selected according to the Hadamard code as detailed above. I.e., the key bits are generated from the Loop PUF given a small set of fixed challenges. As a consequence, we do not consider this attack applicable.

4.2 Power Side-Channel Attacks without Knowing Responses

The Loop PUF is designed with a single loop: having two loops would be prone to coupling between them, hence a decrease in reliability (if not a complete undermining of its rationale). However, from the side-channel perspective, measuring fewer oscillators at a time increases the signal-to-noise ratio (SNR) exploitable by an attacker. As a consequence, without surprise, some attacks manage to guess the PUF's frequency [34]. Those attacks on the primitive are getting more and more difficult with age mismatch [16], but remain likely [28]. Though, the direct

readout can be easy countered, by exploiting the PUF's small variations [34] and implementing a temporal masking scheme. Recently demonstrated cross-PUF attacks [15] can be mitigated by this temporal masking scheme, either.

Compared to the Loop PUF, approaches like the parallel PUF [20], the ring oscillators PUF [33, §3], or the TERO PUF [2] have lower SNR for the attacker owing to the parallel processing of such PUFs. Parallelism here means, for example, that for ring oscillator and TERO PUFs, multiple instances are measured in parallel by different counters; Afterwards these counters are compared for bit derivation. Although the SNR for such PUFs is indeed lower, side-channel attacks were also presented for these structures. Effectively, shown attacks identify the counters and attack the PUF primitives by observing its oscillation frequency and oscillation duration [21,35]. While the smaller SNR can make the attack harder in those cases, countermeasures suggested for those PUF types are normally more complex compared to the simple protection mechanism available for the Loop PUF: The suggested method for such PUFs is to interleave counters or to randomly permute the counter usage [21]. Of course, also the simple temporal masking scheme from the Loop PUF can be applied to such PUFs but if an attacker is able to resolve the individual PUF primitives, only the Loop PUF with temporal masking can resist such a side-channel attack.

4.3 Side-Channel Attacks on Error Correction

The key derived from a PUF must be reliable not only under nominal but also under different environmental conditions. Different temperatures, variations of the supply voltages, and aging of the circuit can cause a drift in the PUF response. Thus, although the Loop PUF allows for an enrollment procedure improving its reliability, the secret derived from the PUF might be still subject to post-processing. In particular, error correction is frequently required to derive a sufficiently stable key. This, however, adds another attack vector to the system: Similar to attacks on cryptographical ciphers, where the key-dependent processing is attackable by side-channel attacks, the processing of the secret PUF response can also be attacked. The efficiency of this attack introduced in [22] was demonstrated in [36]. The attack assumes that so called helper data, which is data used to map a random PUF response to the codeword of an error correcting code, are unprotected. I.e., read-out and manipulation of helper data is feasible in this setting. Consequently, from a side-channel perspective the helper data are equivalent to the data input of an algorithm processing a secret. Manipulation of the helper data allows for the derivation of secret dependent hypothesis. As a consequence, correlation power analyses of the error correcting code allows for extracting the secret key.

There are basically two countermeasures against this particular attack: First, write protection of helper data makes this attack impossible, since it hinders an attacker from observing the device under different inputs. Such a write protection might be achieved, e.g., by a locking bit implemented through a fuse blown after the enrollment phase. Second, the codeword masking strategy suggested in [22] hinders the attack. In particular, first order correlation power analysis is

prevented through masking the secret while it is processed by the error correction code.

5 The Loop PUF Security in Front of Machine Learning Attacks

The Loop PUF in this work is used to store a secret key. For this use case, no challenge-response pairs are available to an attacker. Therefore, machine learning attacks were for a long time out of scope in this setting. However, [32] revealed that even in such scenarios machine learning attacks on PUFs with challenge-response behavior are possible. For this purpose, the existence of helper data in the key-storage scenario, which is explained in Section 4.3, is used:

Error correcting codes need redundancy. Consequently, the bits in a codeword have a known, well defined dependency on each other. The redundancy is ensured for PUFs by a helper data defined mapping from PUF response to codeword for which, most frequently, Fuzzy Commitment and Fuzzy Extractor schemes are used. For those schemes, the publicly known helper data reveal if a PUF bit is flipped or not in order to map the PUF response to a codeword. An attacker can now bring this information together: Through her knowledge about the used code, she can combine (XOR) helper data bits in a way so that the result only depends on specific PUF bits. This relation of different PUF bits are labels for a machine learning algorithm, which takes the corresponding publicly known challenges as features. As a consequence, an attacker is able to learn the relation of the bits of a PUF with challenge-response behavior. This way she can ultimately guess the key stored by the PUF or at least reduce the key entropy.

To hinder the attack, [32] suggest, besides others, two methods: (i) A limitation of the amount of challenge-response pairs can hinder the attack. (ii) High rate codes can make the attack more difficult. Countermeasure (i) is of particular interest for the Loop PUF construction. While other PUFs with challenge-response behavior use up to 2^n challenge-response pairs, where n is the number of stages, the Loop PUF uses only a very limited number of challenges. In particular, for the example in Section 6.2, up to $n = 64$ challenges can be used which are selected under consideration of optimal entropy exploitation. This, however, prevents machine learning attack on this PUF type completely. Furthermore, the high reliability achievable by a Loop PUF might allow for preventing the need of a concatenated code, which is capable to correct high errors down to an acceptable key error rate. This contributes to the resistance of the Loop PUF against such machine learning attack even if the number of challenge-response pairs would not be limited. The reason is, that concatenated codes typically comprise a low rate code which frequently contradicts (ii). As a consequence of the discussion we conclude, that the machine learning attack in [32] is not applicable to the Loop PUF construction analyzed in this work.

6 The Loop PUF Security in Front of Perturbations Attacks

Until now, the attacker in the previous sections was a passive one. She observed helper data or side-channel leakage in order to learn about the secret key. However, obviously the attacker can also get active. An active attacker can tamper with the device in order to enforce faulty behavior. While other means might be possible, the easiest way to tamper with the device is to intentionally manipulate operational parameters like the ambient temperature or the supply voltage of the device. These attacks we subsume under the term *perturbation attacks* and discuss their impact on the Loop PUF in the following. Please note, that further research is needed to analyze fault injection attacks on the PUF beyond this.

6.1 Criticality of Perturbation Attacks

The criticality of perturbation attacks is best explained with a concrete application scenario in mind. Let the PUF be used as a master key in a device. In this situation, the first operation at boot time is the derivation of this master key. Application code cannot start unless the key has been extracted. But for reliability reasons, most countermeasures are not on by default; They are activated by the application code. Through corrupting the master key, the start of application code is hindered and the countermeasures are not activated at boot. An example for how such a perturbation attack might practically work can be given, e.g., for SRAM PUFs: In this particular case, e.g., irradiating the charge pump can force the SRAM to zero. Clearly, if the SRAM is used as a PUF to store the master key but is forced to zero, the key cannot be derived correctly and the perturbation attack succeeds.

Due to the described process, an attacker disturbing the key derivation process can effectively hinder the activation of countermeasure, thus, enabling further attack. As a consequence perturbation attacks are an enabler for further attack vectors, which allow the attacker for more advanced tampering with the device that would be hindered as soon as countermeasures are active. The resulting criticality of perturbation attacks is qualitatively visualized in Fig. 3. The stages are security sub-system (denoted as "sec. sub-sys.") boot, followed by host system boot. Paradoxically enough, when the host has booted, it has all countermeasures activated (hence is very aware of the threats), and operates under secondary secrets (e.g., obtained thanks to key derivation functions). Therefore, the sweet spot for attacker is when the system starts, because at the same time the master key is handled and no (or few) countermeasures are active yet.

On top of the described scenario, an attacker can also try to gain information and to reduce entropy of the derived key from observing the system with a fault in the PUF response. The example shows, that stressing the PUF in particular at boot is a promising attack strategy for PUFs, which are not carefully protected.

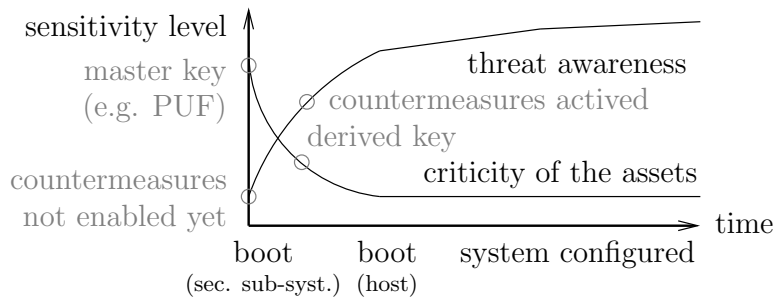


Fig. 3. Qualitative visualization of sensitivity level of keys (decreasing curve) and of the platform sensing capability (increasing curve) as a function of the boot process

6.2 The Loop PUF's Perturbation Resistance

This section shows experimental results to substantiate that the Loop PUF can be implemented so that it is resistant against perturbation attacks. The targeted PUF is an ASIC, implemented in 65 nm CMOS technology from STMicroelectronics [5]. For this purpose, the PUF is placed in a climate chamber (see Fig. 4(a)). The voltage and the temperature within the climate chamber are controlled (see Fig. 4(b)). Secure-IC Analyzr tool [31] is used to setup the experimental conditions and to query the PUF. A temperature range from $-10°$C to $+80°$C is considered. The Loop PUF under consideration is a Loop PUF with $n = 64$ stages, which is at the same time the challenge length. This PUF is configured to oscillate during $2^{18} = 262,144$ clock cycles.

We exemplify the robustness of the Loop PUF against perturbation attacks through manipulation of the ambient temperature for one particular challenge pair $(c, \neg c)$ in accordance to Sec. 3.1. The selected challenge pair in the experiment is $c =$ `0x6996966996696996`, $\neg c =$ `0x9669699669969669`. Please note, that an appropriate challenge selection strategy causes that showing the expected behavior for on particular challenge pair suffices. The enrollment strategy allows than for selecting challenges which behave similarly. The question, which percentage of challenges shows the expected behavior is another research question that is subject to future work.

Fig. 5 shows the result of the experiment. The number of cycles, i.e., the counter values, after applying c and $\neg c$ is shown over the complete temperature range. Obviously the values are drastically affected by the temperature; They significantly decrease with increasing temperature. However, the values for c and $\neg c$ change similarly. In particular, for all temperatures the values for c are significantly larger than that for $\neg c$. As a consequence, the Loop PUF's response, which is defined through this difference, is constant over all temperatures.

To further improve the analysis with respect to noise, another data was collected for $+100°$C, a temperature for which a high noise level can be expected. The results of this experiment are give in Fig. 6. The figure 6 (a) gives actually

(a) PUF evaluation board inside of the climate chamber

(b) Control of the power supply and the climate chamber

Fig. 4. Experimental setup in the PUF characterization laboratory

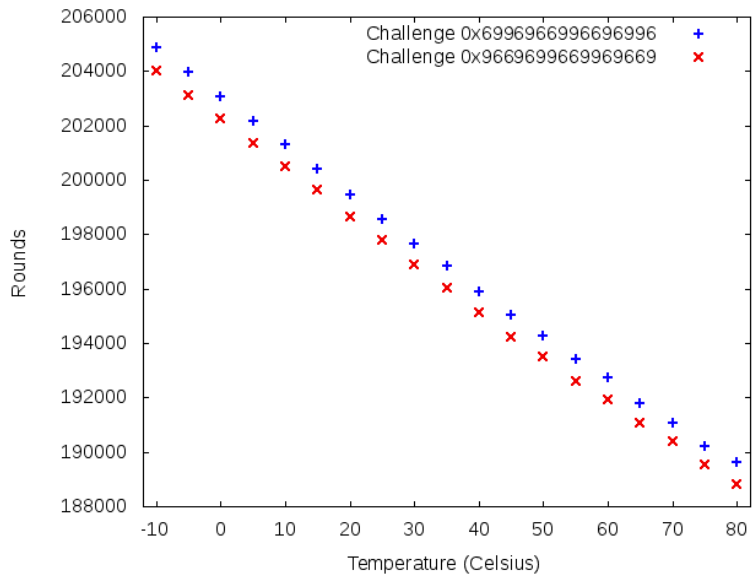


Fig. 5. Number of oscillation cycles (also denoted "rounds") of the PUF for two complementary challenges c and $\neg c$, over temperature

two insights: The two measurement results for c and $\neg c$ follow a global shape. Since c and $\neg c$ are alternately measured, this indicates that the oscillations are affected by the same low frequency noise. The effect is overlaid with higher frequency noise affecting each measurement differently. Nevertheless, while the rounds counter for c is always above 138,000 the corresponding values for $\neg c$ are always below 126,000. This is, the PUF response would be for all measurements the same as under nominal conditions.

For a more quantitative understanding of this finding, a statistical approximation is provided. A histogram of the measurements for c and $\neg c$ is given in Fig. 6 (b). The histograms can be fit by Gaussians. For the sake of simplicity, and since the standard deviations are quite similar, we can approximate both distributions with Gaussian distributions having the same standard deviation σ but different means μ_0 and μ_1. An error occurs, if the difference of the values observed for c and $\neg c$ flips its sign when compared to the nominal reference case, i.e., in our case if the value for c is smaller than the value of $\neg c$.

Lemma 1 ([28, Lemma 1, page 554]). *Under these considerations, the probability for this event, i.e., the error probability, is estimated to be*

$$P_e = \frac{1}{2} - \frac{1}{\sqrt{\pi}} \int_0^{\delta} e^{-t^2} \, dt$$

wherein

$$\delta = \frac{\mu_1 - \mu_2}{\sigma\sqrt{2}}.$$

Notice that P_e is also equal to $\frac{1}{2}(1 - \mathrm{erf}(\delta)) = \frac{1}{2}\mathrm{erfc}(\delta)$, where erf (resp. erfc) is the error function (resp. the complemented error function). For the distributions displayed in Fig. 6, the error probability, estimated with the formula of Lemma 1, is $P_e = 1.37 \times 10^{-264}$.

Please note that the model assumes that the two results of measuring c and $\neg c$ are independent. However, Fig. 6 (a) already revealed that the noise of two subsequent measurements of the Loop PUF under c and $\neg c$ can be correlated (depending on the experimental protocol for the measurements). Therefore, the assumption of independence results likely in a worst case error estimate. A more accurate estimate considering the differences in the standard deviation and the correlation of the noise can be given using [12].

Summarizing, the result shows that the PUF response remains unaltered even if an attacker changes the temperature. The same conclusion can be drawn when the attacker instead manipulated the voltage. The cost for making the Loop PUF so reliable is a relatively long measurement time and possibly a pre-selection of stable challenges. Nevertheless, we can conclude that the Loop PUF indeed can be build so that it is insensitive to perturbation attacks, even if no additional processing step like error correction is used.

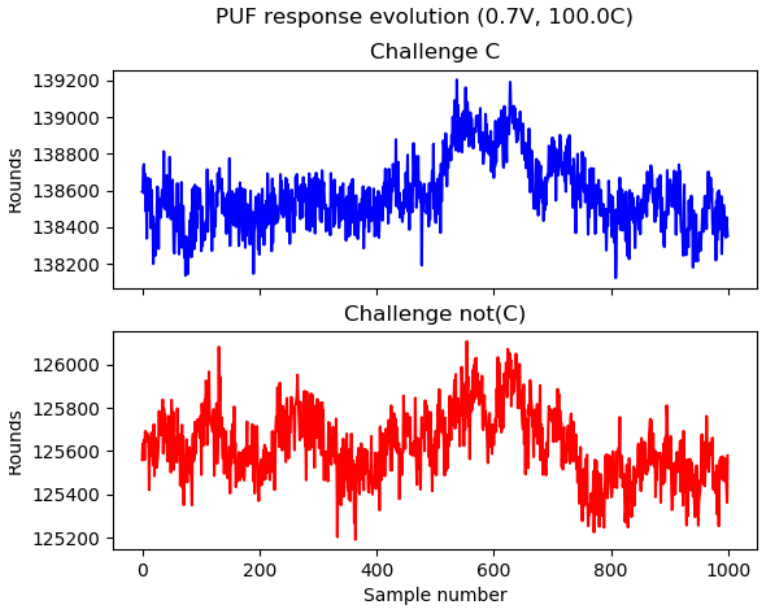
7 Conclusions and Perspectives

This work has discussed the application of PUFs for a specific use case: The Loop PUF is used to generate and to store a device unique key. Besides an explanation of the use case and of the Loop PUF, different attack vectors were discussed. In particular side-channel attacks, machine learning attacks, and perturbation attacks were considered. The results show that for all discussed attacks, the Loop PUF is able to resist or can be protected against. As a consequence, the Loop PUF is suitable for anti-copy application, while being natively anti-tamper at the same time. This is a highly desired feature for master key generation, when no system-level countermeasure is enabled yet.

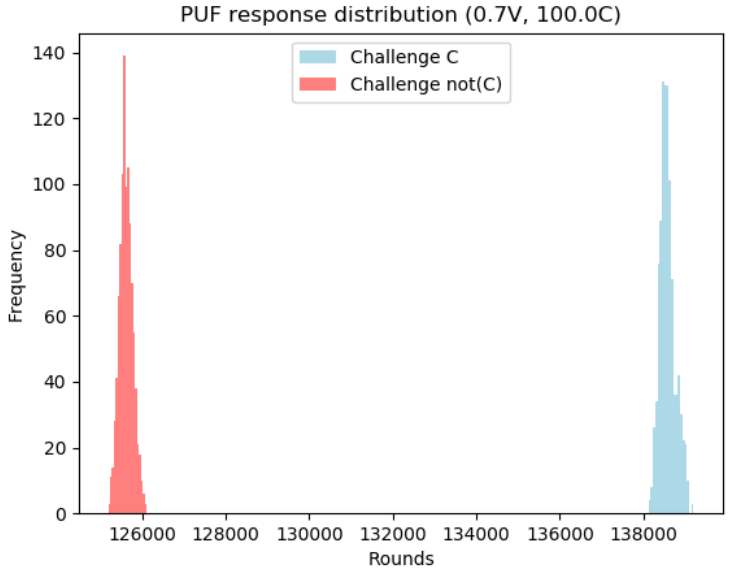
As a perspective, we underline that in the current paper, we assumed only *global* perturbation, which they affect the full chip [11]. However, attacker with advanced or bespoke equipment can try for *local* attacks, which should be subject to future research.

Acknowledgments

This work has benefited from a funding via the bilateral project APRIORI (*Advanced PRivacy of IOT Devices through Robust Hardware Implementations*), from FR-DE cybersecurity 2020 call (MESRI-BMBF), managed by ANR from the French side.



(a) Repeated numbers of measurements (1000 queries)



(b) Histogram of the measurements

Fig. 6. Number of rounds for two complementary 64-bit challenges

References

1. Aysu, A., Gulcan, E., Moriyama, D., Schaumont, P., Yung, M.: End-To-End Design of a PUF-Based Privacy Preserving Authentication Protocol. In: Güneysu, T., Handschuh, H. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2015. pp. 556–576. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
2. Bossuet, L., Ngo, X.T., Cherif, Z., Fischer, V.: A puf based on a transient effect ring oscillator and insensitive to locking phenomenon. Emerging Topics in Computing, IEEE Transactions on **2**(1), 30–36 (March 2014). https://doi.org/10.1109/TETC.2013.2287182
3. Bruneau, N., Danger, J., Facon, A., Guilley, S., Hamaguchi, S., Hori, Y., Kang, Y., Schaub, A.: Development of the Unified Security Requirements of PUFs During the Standardization Process. In: Lanet, J., Toma, C. (eds.) Innovative Security Solutions for Information Technology and Communications - 11th International Conference, SecITC 2018, Bucharest, Romania, November 8-9, 2018, Revised Selected Papers. Lecture Notes in Computer Science, vol. 11359, pp. 314–330. Springer (2018). https://doi.org/10.1007/978-3-030-12942-2_24, https://doi.org/10.1007/978-3-030-12942-2_24
4. Cherif, Z., Danger, J., Guilley, S., Bossuet, L.: 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. pp. 156–162. IEEE Computer Society (2012). https://doi.org/10.1109/DSD.2012.22, http://dx.doi.org/10.1109/DSD.2012.22
5. Cherif, Z., Danger, J.L., Lozac'h, F., Mathieu, Y., Bossuet, L.: Evaluation of Delay PUFs on CMOS 65 nm Technology: ASIC vs FPGA. In: Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy. pp. 4:1–4:8. HASP '13, ACM, New York, NY, USA (2013). https://doi.org/10.1145/2487726.2487730, http://doi.acm.org/10.1145/2487726.2487730
6. Dafali, R., Danger, J.L., Guilley, S., Lozac'h, F.: Embedded test circuit for physically unclonable function (December 1st 2020), USA patent US10855476B2
7. Danger, J., Guilley, S., Schaub, A.: Two-Metric Helper Data for Highly Robust and Secure Delay PUFs. In: IEEE 8th International Workshop on Advances in Sensors and Interfaces, IWASI 2019, Otranto, Italy, June 13-14, 2019. pp. 184–188. IEEE (2019). https://doi.org/10.1109/IWASI.2019.8791249, https://doi.org/10.1109/IWASI.2019.8791249
8. Frisch, C., Tempelmeier, M., Pehl, M.: PAG-IoT: A puf and aead enabled trusted hardware gateway for iot devices. In: 2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). pp. 500–505. IEEE (2020)
9. Gassend, B.: Physical Random Functions (2003), msc thesis, MIT
10. Guajardo, J., Kumar, S.S., Schrijen, G.J., Tuyls, P.: FPGA Intrinsic PUFs and Their Use for IP Protection. In: CHES. pp. 63–80. Lecture Notes in Computer Science, Springer (2007)
11. Guilley, S., Danger, J.L.: Global Faults on Cryptographic Circuits, Chapter 17 of [14]
12. Hiller, M., Sigl, G., Pehl, M.: A new model for estimating bit error probabilities of ring-oscillator PUFs. In: International Workshop on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC). IEEE (2013)
13. Jacob, N., Wittmann, J., Heyszl, J., Hesselbarth, R., Wilde, F., Pehl, M., Sigl, G., Fischer, K.: Securing FPGA SoC configurations independent

of their manufacturers. In: Alioto, M., Li, H.H., Becker, J., Schlichtmann, U., Sridhar, R. (eds.) 30th IEEE International System-on-Chip Conference, SOCC 2017, Munich, Germany, September 5-8, 2017. pp. 114–119. IEEE (2017). https://doi.org/10.1109/SOCC.2017.8226019, https://doi.org/10.1109/SOCC.2017.8226019

14. Joye, M., Tunstall, M. (eds.): Fault Analysis in Cryptography. Information Security and Cryptography, Springer (2012). https://doi.org/10.1007/978-3-642-29656-7, http://dx.doi.org/10.1007/978-3-642-29656-7, ISBN: 978-3-642-29655-0; DOI: 10.1007/978-3-642-29656-7
15. Kroeger, T., Cheng, W., Guilley, S., Danger, J.L., Karimi, N.: Cross-PUF Attacks on Arbiter-PUFs through their Power Side-Channel. In: 51st International Test Conference (ITC) sponsored by IEEE (November 3-5 2020)
16. Kroeger, T., Cheng, W., Guilley, S., Danger, J., Karimi, N.: Effect of aging on PUF modeling attacks based on power side-channel observations. In: 2020 Design, Automation & Test in Europe Conference & Exhibition, DATE 2020, Grenoble, France, March 9-13, 2020. pp. 454–459. IEEE (2020). https://doi.org/10.23919/DATE48585.2020.9116428, https://doi.org/10.23919/DATE48585.2020.9116428
17. Maes, R., van der Leest, V., van der Sluis, E., Willems, F.: Secure Key Generation from Biased PUFs. In: Güneysu, T., Handschuh, H. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2015. pp. 517–534. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
18. Maes, R., Van Herrewege, A., Verbauwhede, I.: PUFKY: A Fully Functional PUF-based Cryptographic Key Generator. In: Proceedings of the 14th International Conference on Cryptographic Hardware and Embedded Systems. pp. 302–319. CHES'12, Springer-Verlag, Berlin, Heidelberg (2012), http://dx.doi.org/10.1007/978-3-642-33027-8_18
19. Magazine, W.: Gemalto Confirms It Was Hacked But Insists the NSA Didn't Get Its Crypto Keys (February 25 2015), https://www.wired.com/2015/02/gemalto-confirms-hacked-insists-nsa-didnt-get-crypto-keys/
20. Majzoobi, M., Koushanfar, F., Potkonjak, M.: Lightweight secure PUFs. In: Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design. pp. 670–673. ICCAD '08, IEEE Press, Piscataway, NJ, USA (2008), http://portal.acm.org/citation.cfm?id=1509456.1509603
21. Merli, D., Heyszl, J., Heinz, B., Schuster, D., Stumpf, F., Sigl, G.: Localized electromagnetic analysis of RO PUFs. In: Proceedings of the IEEE Int. Symposium of Hardware-Oriented Security and Trust. IEEE (Jun 2013)
22. Merli, D., Stumpf, F., Sigl, G.: Protecting PUF error correction by codeword masking. Cryptology ePrint Archive, Report 2013/334 (2013), http://eprint.iacr.org/2013/334
23. Pappu, R., Recht, B., Taylor, J., Gershenfeld, N.: Physical One-Way Functions. Science **297**(5589), 2026–2030 (September 20 2002), DOI: 10.1126/science.1074376
24. Pehl, M., Frisch, C., Feist, P.C., Sigl, G.: KeLiPUF: a key-distribution protocol for lightweight devices using physical unclonable functions. In: 17th ESCAR Europe: embedded security in cars (Konferenzveröffentlichung) (2019). https://doi.org/10.13154/294-6676
25. Pour, A.A., Beroulle, V., Cambou, B., Danger, J.L., Natale, G.D., Hely, D., Guilley, S., Karimi, N.: PUF Enrollment and Life Cycle Management: Solutions and Perspectives for the Test Community. In: 25th IEEE European Test Symposium (May 25-29 2020), Tallinn, Estonia

26. Ranasinghe, D., Engels, D., Cole, P., et al.: Security and privacy: Modest proposals for low-cost RFID systems. In: Auto-ID labs research workshop, Zurich, Switzerland. Citeseer (2004)
27. Rioul, O., Solé, P., Guilley, S., Danger, J.: On the entropy of Physically Unclonable Functions. In: IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016. pp. 2928–2932. IEEE (2016). https://doi.org/10.1109/ISIT.2016.7541835, http://dx.doi.org/10.1109/ISIT.2016.7541835
28. Schaub, A., Danger, J., Guilley, S., Rioul, O.: An Improved Analysis of Reliability and Entropy for Delay PUFs. In: Novotný, M., Konofaos, N., Skavhaug, A. (eds.) 21st Euromicro Conference on Digital System Design, DSD 2018, Prague, Czech Republic, August 29-31, 2018. pp. 553–560. IEEE Computer Society (2018). https://doi.org/10.1109/DSD.2018.00096, http://doi.ieeecomputersociety.org/10.1109/DSD.2018.00096
29. Schaub, A., Danger, J.L., Rioul, O., Guilley, S.: The Big Picture of Delay-PUF Dependability. In: 24th European Conference on Circuit Theory and Design (September 7-10 2020), Sofia, Bulgaria
30. Schaub, A., Rioul, O., Danger, J., Guilley, S., Boutros, J.: Challenge codes for physically unclonable functions with Gaussian delays: A maximum entropy problem. Adv. Math. Commun. **14**(3), 491–505 (2020). https://doi.org/10.3934/amc.2020060, https://doi.org/10.3934/amc.2020060
31. Secure-IC S.A.S.: Analyzr® post-silicon security evaluation platform (2021), https://www.secure-ic.com/solutions/analyzr/
32. Strieder, E., Frisch, C., Pehl, M.: Machine Learning of Physical Unclonable Functions using Helper Data - Revealing a Pitfall in the Fuzzy Commitment Scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems **2021**(2) (2021)
33. Suh, G.E., Devadas, S.: Physical Unclonable Functions for Device Authentication and Secret Key Generation. In: Proceedings of the 44th Design Automation Conference, DAC 2007, San Diego, CA, USA, June 4-8, 2007. pp. 9–14. IEEE (2007). https://doi.org/10.1145/1278480.1278484, http://doi.acm.org/10.1145/1278480.1278484
34. Tebelmann, L., Danger, J.L., Pehl, M.: Self-Secured PUF: Protecting the Loop PUF by Masking. In: Constructive Side-Channel Analysis and Secure Design - 11th International Workshop, COSADE 2020, Lugano, Switzerland, October 5-7, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12244. Springer (October 5-7 2020)
35. Tebelmann, L., Pehl, M., Immler, V.: Side-Channel Analysis of the TERO PUF. In: Polian, I., Stöttinger, M. (eds.) Constructive Side-Channel Analysis and Secure Design. pp. 43–60. Springer International Publishing, Cham (2019)
36. Tebelmann, L., Pehl, M., Sigl, G.: EM Side-Channel Analysis of BCH-based Error Correction for PUF-based Key Generation. In: Proceedings of the 2017 Workshop on Attacks and Solutions in Hardware Security. pp. 43–52. ASHES '17, ACM, New York, NY, USA (2017). https://doi.org/10.1145/3139324.3139328, http://doi.acm.org/10.1145/3139324.3139328