2018 21st Euromicro Conference on Digital System Design

An Improved Analysis of Reliability and Entropy for Delay PUFs

Alexander Schaub*, Jean-Luc Danger*§, Sylvain Guilley§*‡ and Olivier Rioul*†

* LTCI, Telecom ParisTech, 75 013 Paris, France
firstname.lastname@telecom-paristech.fr
† CMAP, École Polytechnique, 91 120 Palaiseau, France
olivier.rioul@polytechnique.edu
‡ École Normale Supérieure, 75 005 Paris, France
§ Secure-IC S.A.S. 35 510 Cesson-Sévigné, France

***Abstract*—*Physically unclonable functions* (PUF) have been used in various applications, such as device authentication, secure storage of sensitive data, and anti-counterfeiting. Different applications require various levels of reliability from the PUF. However, as of today, no *predictive* model to characterize the PUF reliability has been developed. This is particularly a problem for PUFs with low error rates, because the lower the error rate, the larger the number of measurements required to obtain a good estimate.**

In this paper, we develop a predictive framework, which enables us to derive a closed-form expression of both *entropy* and *reliability* for several families of delay PUFs: the ring oscillator (RO) PUF, the RO sum PUF as well as the Loop PUF. Improving reliability with bit-filtering, we provide an explicit tradeoff between complexity, reliability and entropy. Error rates as low as 10^{-9} or even lower can be achieved. Our theoretical results are validated by experiments on Loop PUFs implemented in 65 nm CMOS ASIC technology, also used to simulate the behavior of the RO PUF and the RO sum PUF.

I. Introduction

Designing a PUF involves a three-way tradeoff between entropy, reliability and complexity (e.g., circuit size). Firstly, entropy is increased by adding more elements such as RAM cells or oscillators, at the expense of an increased circuit size. Also, reliability is enhanced by error-correcting codes (ECC), but their redundancy generally decreases the entropy. For a given PUF design, it is not obvious how to precisely characterize the tradeoff between these three parameters (entropy, reliability and circuit complexity). For instance, fuzzy extraction [7] using error-correcting codes is implemented in the PUFKY [14] based on the ROPUF [3]. However, for this design, and fuzzy extraction in general, it is very hard to determine the bit error rate (BER) theoretically. Therefore, the actual parameter selection for the fuzzy extractor is not straightforward. Bit-filtering [18] can also improve the reliability but since the number of output bits is reduced as a result of the filtering, this technique also decreases the entropy of the PUF. Thus, this technique, similar to fuzzy extraction, is subject to a tradeoff between reliability and entropy.

The aim of this paper is to build a framework to analyze this tradeoff for delay PUFs. Maes [13] proposed such a framework for the reliability of SRAM PUFs which was *ad hoc* for a given PUF architecture and where the parameters' identification was performed on experimental data. Bhargava et al. [2] also perform filtering to improve the reliability of their PUF design, but provide no theoretical model to predict the reliability that might be obtained. In contrast, we aim at deriving a generic model using elementary assumptions, where the three-way tradeoff is not fully determined by real measurements, but given instead by closed-form expressions involving the signal-to-noise ratio (SNR). In this way, additional estimations of the SNR yield *new predictions* for the tradeoff.

Our framework is applied to three popular delay PUFs: the RO-PUF [19], the RO sum PUF [20] and the Loop PUF [4]. Bit-filtering is the technique chosen here to improve reliability, in a manner similar to the η-out-of-λ scheme of Škoric et al. [18]. Our contributions are as follows:

- a generic tradeoff analysis framework for delay PUFs;
- closed-form expressions for the BER and entropy for these PUFs, with and without bit-filtering;
- an analysis of the RO-PUF, the RO sum PUF and the Loop PUF, using this framework;
- real measurements of the delay PUFs on ASIC confirming our theoretical results.

The remainder of this article is organized as follows. Section II presents a theoretical model for the delay PUFs. Closed-form expressions for reliability and entropy are derived in Section III. This framework is applied to various delay PUFs in Section IV. Section V provides an experimental validation on silicon. Section VII concludes.

II. Delay PUF model

In this section, we provide a black-box analysis for a generic delay PUF. Throughout this paper we use the following notations.

n	number of delay elements in the circuit
i	index of a delay element
t	index of a measurement
T	total number of measurements
M	total number of challenges
m	index of a challenge

978-1-5386-7377-5/18/$31.00 ©2018 IEEE
DOI 10.1109/DSD.2018.00096

J	number of circuits
j	index of a circuit
c_i^m	i-th challenge bit
C^m	m-th challenge $C^m = (c_i^m)_i$
$d_{C,t}^j$	total delay for challenge C (at measure t, for circuit j)
$\delta_{C,t}^j$	$\delta_{C,t}^j = d_{C,t}^j - d_{\neg C,t}^j$
δ_C^j	$\delta_C^j = \frac{1}{T}\sum_{t=1}^{T} \delta_{C,t}^j$
Δ_C	random variable modeling δ_C
Z	additive Gaussian measurement noise

For simplification, sub- and superscripts (such as m, t, or j) may be dropped when this does not introduce any confusion.

We model an ideal (noiseless) delay PUF as a deterministic algorithm $\mathcal{P}^I$ that takes a challenge C as input, and outputs a delay difference δ_C:

$$\mathcal{P}^I : C \mapsto \delta_C.$$

This delay is then, in general, discretized in order to extract one (or more) bit(s). Thus, the final output is some function of the measured delay difference. For the sake of simplicity, we consider the $sign$ function as the bit-output of the PUF:

$$b = \text{sign}(\delta_C).$$

The delay difference δ_C for a given challenge stems from a multitude of small delay variations caused by technology dispersion, and is thus seen as a realization of a random variable Δ_C. Similarly to Lim et al. [12], we model this random PUF variable as Gaussian $\Delta_C \sim \mathcal{N}(0, \Sigma^2)$ for some positive deviation $\Sigma > 0$.

Such a delay PUF model is ideal since in practice, measurement noise is always present. Following e.g., [12] we model this noise as additive and independent Gaussian. Our PUF model becomes a *probabilistic* algorithm:

$$\mathcal{P} : C \mapsto \delta_C + Z \qquad b = \text{sign}(\delta_C + Z) \tag{1}$$

where $Z \sim \mathcal{N}(0, \sigma^2)$ for some $\sigma > 0$. Since $\mathcal{P}(C)$ is the sum of a "signal" Δ_C and noise Z, the signal-to-noise ratio (SNR) can be defined

$$\text{SNR} = \frac{\mathbb{E}[\Delta_C^2]}{\mathbb{E}[Z^2]} = \frac{\Sigma^2}{\sigma^2}. \tag{2}$$

and the bit error rate is defined as

$$\text{BER}(\delta_C) = \mathbb{P}(\text{sign}(\delta_C + Z) \neq \text{sign}(\delta_C)). \tag{3}$$

To simplify the reliability analysis, we make the additional assumption that *all PUF responses δ_C are mutually independent*. In general this will only be satisfied approximately. As shown below for each specific PUF, the independence assumption will hold accurately for specific sets of challenges (at the order of n).

In the model proposed by Maes [13], δ_C would correspond to the *process variables* and Z to the *noise variable*. However, the output bits from a delay PUF do not precisely correspond to a measurement of the process variables and further analysis is needed to apply the Maes model to delay PUFs. Furthermore, rather than estimating the BER from experimental data and then find the parameters using a top-down approach, we find it more convenient to derive the BER from measures of simple system parameters such as the SNR, in a bottom-up approach, as described in the next section. We feel that such a determination is better theoretically justified since it requires less *ad hoc* assumptions.

III. Delay PUF Reliability and Entropy

When considering n challenges to generate n response bits, there is a high probability that unreliable response bits are obtained. Katzenbeisser et al. [11] showed that there is 2% to 15% unreliable bits, depending on the environment. Here we consider the proportion of faulty bits, or, equivalently, the average probability that a PUF bit flips, as a metric to characterize the PUF reliability. In contrast to an SRAM PUF, for which only the output bit values are available, delays can be measured in a delay PUF to detect unreliable bits, as we will explain in the next sections.

A. Reliability Assessment

The reliability of a delay PUF is directly related to the absolute value $|\delta_C|$ of the delay difference δ_C associated to each challenge C. Indeed, the larger the value, the smaller the probability to have a bit flip of the measured δ_C sign due to measurement error. More formally, if we consider the Gaussian noise $Z \sim \mathcal{N}(0, \sigma^2)$ added to δ_C, the BER is the probability to have a bit flip for challenge C, and is given by the following

Lemma 1. *One has*

$$\text{BER}(\delta_C) = \mathbb{P}\big(\text{sign}(\delta_C + Z) \neq \text{sign}(\delta_C)\big) = Q\Big(\frac{|\delta_C|}{\sigma}\Big), \tag{4}$$

where $Q(x) = \frac{1}{2}\,\text{erfc}(\frac{x}{\sqrt{2}})$.

Proof. Let $Z \sim \mathcal{N}(0, \sigma^2)$ and δ_C be a fixed value. Then

$$\begin{aligned}\text{BER} &= \mathbb{P}[\text{sign}(\delta_C + Z) \neq \text{sign}(\delta_C)] \\ &= \mathbb{P}[(\delta_C + Z > 0, \delta_C < 0)] + \mathbb{P}[(\delta_C + Z < 0, \delta_C > 0)] \\ &= \mathbb{P}[Z > |\delta_C|, \delta_C < 0] + \mathbb{P}[-Z > |\delta_C|, \delta_C > 0] \\ &= \mathbb{P}[Z > |\delta_C|] = Q\Big(\frac{|\delta_C|}{\sigma}\Big)\end{aligned}$$

since Z is symmetrically distributed. □

Figure 1 illustrates the distribution of Δ_C and the noise distribution around the value δ_C associated to the challenge C. In this example, an error occurs when $\delta_C + Z$ is negative.

$\lvert\delta_C\rvert/\sigma$ value	0	1	2	3	4
BER	0.5	$1.6\ 10^{-1}$	$2.3\ 10^{-2}$	$1.3\ 10^{-3}$	$3.2\ 10^{-5}$
$\lvert\delta_C\rvert/\sigma$ value	5	6	7	8	9
BER	$2.9\ 10^{-7}$	$9.9\ 10^{-10}$	$1.3\ 10^{-12}$	$6.2\ 10^{-16}$	$1.1\ 10^{-19}$

Table I
BER FOR ONE BIT ACCORDING TO THE $|\delta_C|/\sigma$ VALUE.

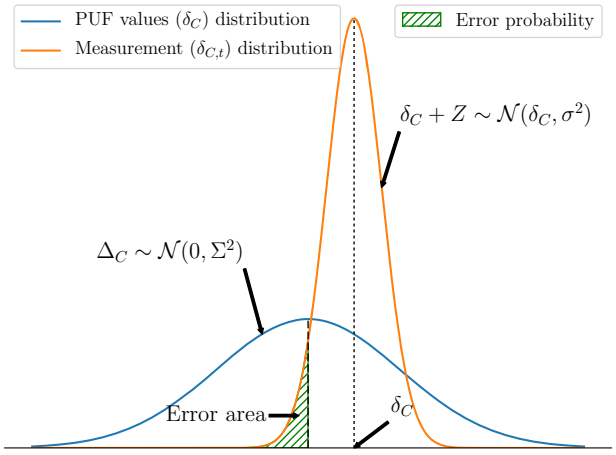

Figure 1. pdf of Δ and noise for a given challenge C.

Table I gives the BER one can expect for a given challenge. For a set of challenges, the BER has to be assessed on all the δ_C values, which are assumed to be independent.

The average proportion of bit flips is the expectation of the BER over Δ_C, and is given by the following

Lemma 2. *One has*

$$\widehat{\mathrm{BER}} = \mathbb{E}[\mathrm{BER}(\Delta_C)] = \frac{1}{\pi}\arctan(\frac{1}{\sqrt{\mathrm{SNR}}}). \tag{5}$$

Proof. As shown in the proof of Lemma 1,

$$\begin{aligned}\widehat{\mathrm{BER}} &= \mathbb{P}[\mathrm{sign}(\Delta_C + Z) \neq \mathrm{sign}(\Delta_C)] \\ &= \mathbb{P}[Z > |\Delta_C|] \\ &= \mathbb{P}[\frac{Z}{\sigma} > |\frac{\Delta_C}{\Sigma}|\sqrt{\mathrm{SNR}}].\end{aligned}$$

Note that this probability is taken jointly over Δ_C, Z and that these are independent Gaussian variables, $\Delta_C \sim \mathcal{N}(0, \Sigma^2), Z \sim \mathcal{N}(0, \sigma^2)$. Therefore, $X = \frac{\Delta_C}{\Sigma}$ and $Y = \frac{Z}{\sigma}$ are independent and follow standard normal distributions, and the formula becomes

$$\widehat{\mathrm{BER}} = \mathbb{P}[Y > |X|\sqrt{\mathrm{SNR}}].$$

Since the probability distribution of (X, Y) is isotropic, it is easily seen that $\widehat{\mathrm{BER}}$ equals the proportion of the hatched area on Fig. 2. This proportion is simply $2\theta/2\pi$, where $\tan(\theta) = 1/\sqrt{\mathrm{SNR}}$ by the geometric definition of the tan function. Thus, we simply have that

$$\widehat{\mathrm{BER}} = \frac{1}{\pi}\arctan(\frac{1}{\sqrt{\mathrm{SNR}}}). \qquad \square$$

The expected BER is represented as a function of the SNR in Fig. 3. Although the expected BER (III-A) vanishes with the noise:

$$\lim_{\mathrm{SNR}\to+\infty} \widehat{\mathrm{BER}} = 0,$$

it is easily seen that the expected BER remains quite high, $> 10^{-3}$, even for large values of SNR (several thousands).

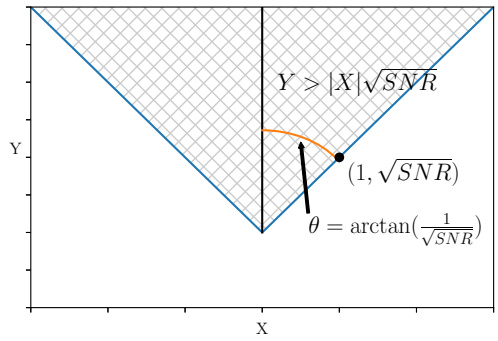

Figure 2. Polar representation of X and Y.

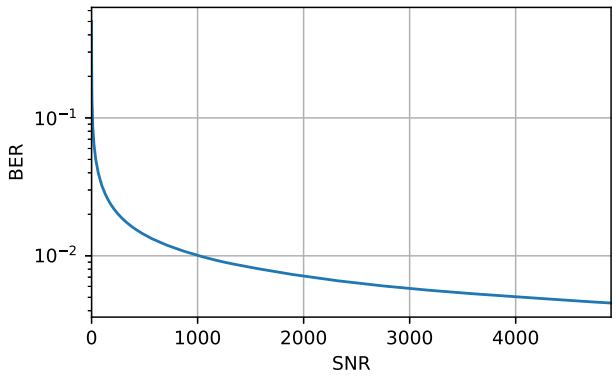

Figure 3. Expected BER as a function of the SNR.

B. Reliability Enhancement by Delay Knowledge

A classical and efficient method to enhance (reduce) the BER is to take advantage of ECCs, like the secure sketch methods presented by Dodis [7] and exploited by reliable architectures like PUFKY [14]. With this method, an enrollment phase takes place once, just after manufacturing, in order to build a public "helper data". The helper data, also called "secure sketch", can be either a n bit code-offset or a $n-k$ bit syndrome. During PUF usage, noise might corrupt the PUF value, but thanks to the secure sketch, the potential errors can be corrected by the ECC decoder.

We will investigate here another method to improve the reliability of the PUF that uses the knowledge of the δ_C values to filter out unreliable bits. Therefore, ECC may not be necessary or at least less complex, which helps to reduce circuit complexity.

The BER can be decreased discarding the challenges which generate unreliable bits. These challenges are recorded during the enrollment phase in the helper data. This helper data is then used during the reconstruction phase of the PUF. This construction resembles the η-out-of-λ scheme by Škoric et al. [18]. However, to make the computations tractable, instead of removing a fixed number of challenges, we remove bits that whose reliability is lower than a given threshold. Below we compute the resulting average reliability in terms of mean BER, and the average remaining entropy after bit-filtering.

From the security point of view, this helper data does not unveil any information of the response bits, since the PUF responses to challenges are assumed independent. However, if an attacker could modify the helper data, she could reconstruct the PUF response, for example using an attack similar to the one described by Hiller et al. [10]. Therefore, we have to assume that the helper data can only be read by an attacker, but cannot be modified. This can, for example, be achieved by storing it on ROM memory on the PUF.

The declaration of "unreliability" is given at enrollment phase when the delay $|\delta_C|$ for a challenge C is below a threshold Th, which has to be chosen to take into account the noise level σ. In what follows, we set $Th = W \cdot \sigma$, where W expresses the capacity to filter the unreliable bits. Increasing W decreases the BER, but reduces the number of bits, hence the entropy.

Figure 4 illustrates the distributions of Δ_C and the noise. It points out the unreliable area in the window $[-Th, +Th]$ of width $2W\sigma$.

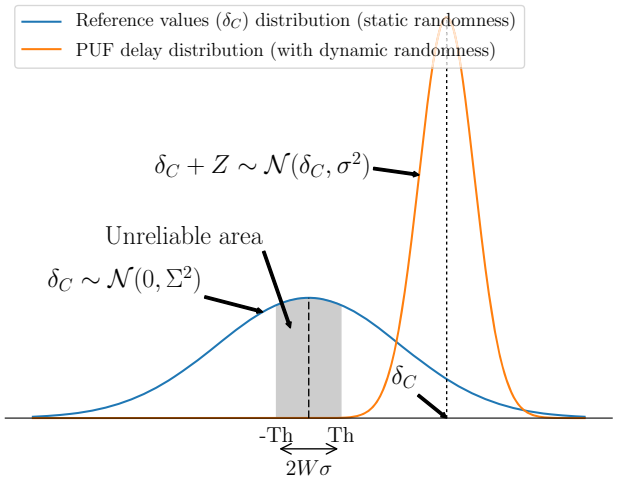

Figure 4. Unreliable area vs distributions of Δ_C and the noise Z.

The average BER reduction after filtering the unreliable bits depends directly on $Th = W\sigma$ and is given by the following

Lemma 3.

$$\widehat{\mathrm{BER}_{filt}} = \frac{2}{\mathrm{erfc}(\frac{W}{\sqrt{2}\sqrt{\mathrm{SNR}}})}\left(T(W, \frac{1}{\sqrt{\mathrm{SNR}}}) + \frac{1}{4}\,\mathrm{erf}(\frac{W}{\sqrt{2}\sqrt{\mathrm{SNR}}})(\mathrm{erf}(\frac{W}{\sqrt{2}}) - 1)\right) \quad (6)$$

where T represents Owen's T function:

$$T(h, a) = \frac{1}{2\pi}\int_0^a \frac{e^{-\frac{1}{2}h^2(1+x^2)}}{1+x^2}\,\mathrm{d}x.$$

Proof. For the sake of simplicity, we will drop the subscript C from the random variable Δ_C.

By definition of the filtered BER, we have that:

$$\widehat{\mathrm{BER}_{filt}} = \int_{-\infty}^{+\infty} p(\Delta \mid |\Delta| > Th) \cdot \mathrm{BER}(\Delta)\,\mathrm{d}\Delta.$$

This generic formulation, or very similar ones, have already been found before, for example by Delvaux [5] (Eq 4.41). However, we will apply it here to a specific PUF, and can therefore derive a more explicit formulation. Indeed, we can find a closed form of $\mathbb{E}(\mathrm{BER}_{filt})$ after filtering the bits as:

$$\begin{aligned}\widehat{\mathrm{BER}_{filt}} &= \int_{-\infty}^{+\infty} p(\Delta \mid |\Delta| > Th)\mathrm{BER}(\Delta)\,\mathrm{d}\Delta \\ &= \int_{-\infty}^{+\infty} \mathbb{1}_{|\Delta|>Th}(\Delta)\frac{p(\Delta)}{\mathbb{P}(|\Delta| > Th)}\mathrm{BER}(\Delta)\,\mathrm{d}\Delta \\ &= \frac{2}{\mathbb{P}(|\Delta| > Th)}\int_{Th}^{+\infty} p(\Delta)\cdot\mathrm{BER}(\Delta)\,\mathrm{d}\Delta \\ &= \frac{2}{\mathbb{P}(|\Delta| > Th)}\frac{1}{2\sqrt{2\pi}\Sigma}\int_{Th}^{+\infty} e^{\frac{-\Delta^2}{2\Sigma^2}}\,\mathrm{erfc}(\frac{\Delta}{\sigma\sqrt{2}})\,\mathrm{d}\Delta.\end{aligned}$$

Using the following integral value for $x, k > 0$:

$$\int e^{-x^2}\,\mathrm{erfc}(kx)\,\mathrm{d}x = -\frac{1}{2}\sqrt{\pi}(4T[\sqrt{2}kx, \frac{1}{k}] + \mathrm{erf}(x)(\mathrm{erf}(kx) - 1) + 1) + \text{constant}$$

(where T is Owen's T function, first introduced by Owen [16]), and using a change of variables, we get that

$$\widehat{\mathrm{BER}_{filt}} = \frac{2}{\mathbb{P}(|\Delta| > Th)}(T(\frac{Th}{\sigma}, \frac{1}{\sqrt{\mathrm{SNR}}}) + \frac{1}{4}\,\mathrm{erf}(\frac{Th}{\sqrt{2}\Sigma})(\mathrm{erf}(\frac{Th}{\sqrt{2}\sigma}) - 1))$$

or, since $\mathbb{P}(|\Delta| > Th) = \mathrm{erfc}(\frac{W}{\sqrt{2}\sqrt{\mathrm{SNR}}})$ and $\frac{Th}{\Sigma} = \frac{W\sigma}{\Sigma} = \frac{W}{\sqrt{\mathrm{SNR}}}$,

$$\widehat{\mathrm{BER}_{filt}} = \frac{2}{\mathrm{erfc}(\frac{W}{\sqrt{2}\sqrt{\mathrm{SNR}}})}\left(T(W, \frac{1}{\sqrt{\mathrm{SNR}}}) + \frac{1}{4}\,\mathrm{erf}(\frac{W}{\sqrt{2}\cdot\sqrt{\mathrm{SNR}}})(\mathrm{erf}(\frac{W}{\sqrt{2}}) - 1)\right).$$

□

C. Entropy After Filtering Out Unreliable Bits

The proportion of unreliable bits is given by

$$\begin{aligned}\mathbb{P}(Bit\ unreliable) &= \mathbb{P}(|\Delta| < Th) = \mathrm{erf}(\frac{Th}{\sqrt{2}\Sigma}) \\ &= \mathrm{erf}(\frac{W}{\sqrt{2\mathrm{SNR}}}).\end{aligned} \quad (7)$$

In other words, the average remaining entropy of a circuit with n elements (thus, of complexity proportional to n) is equal to

$$H(n, W)_{SNR} = n \cdot \mathrm{erfc}(\frac{W}{\sqrt{2\mathrm{SNR}}}). \quad (8)$$

With this method, it is necessary to increase the number of elements to generate a given entropy. The expected number of

elements n, with $n > h$, to consider in order to obtain h bits of entropy is given by:

$$n = \frac{h}{1 - \mathbb{P}(Bit\ unreliable)} = \frac{h}{\mathrm{erfc}(\frac{W}{\sqrt{2\mathrm{SNR}}})}. \quad (9)$$

Figure 5 represents the average remaining entropy for a circuit, depending on the SNR and the target BER. This characterizes the tradeoff between reliability and entropy.

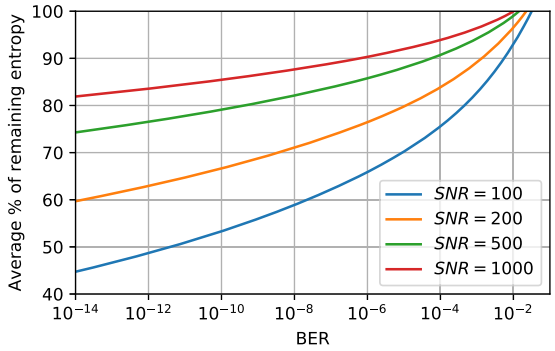

Figure 5. Remaining average entropy after filtering unreliable bits as a function of the BER to reach.

If ever it is not possible to reach the required entropy, the device is discarded. The probability of this happening can also be computed. Since the obtained delays are independent (when choosing a Hadamard matrix for the challenges), the number of unreliable bits is given by a binomial distribution $\mathcal{B}(n, \mathrm{erf}(\frac{W}{\sqrt{2\mathrm{SNR}}}))$ on a PUF with n elements. Thus,

$$\begin{aligned} p_{discard} &= \sum_{i=n-h+1}^{n} \binom{n}{i} \mathrm{erf}(\frac{W}{\sqrt{2\mathrm{SNR}}})^i \,\mathrm{erfc}(\frac{W}{\sqrt{2\mathrm{SNR}}})^{n-i} \\ &= I_{p_d}(n-h+1, h) \end{aligned} \quad (10)$$

where $I_x(a, b)$ is the regularized incomplete beta function and p_d is the probability of discarding a bit, $p_d = \mathrm{erf}(\frac{W}{\sqrt{2\mathrm{SNR}}})$.

IV. Translation for Various PUF Architectures

When applying our results on real PUF architectures, we must suppose that the output bits are independent and non-biased. By restricting the set of possible challenges, we can prove, under the assumption that the theoretical model describing the PUF is accurate, that the remaining output bits are indeed independent and non-biased. However, this is no longer true when the PUF behavior deviates from that predicted by its model. It is possible to ensure that the output bits are not biased and uniform using standard test suites, for instance those provided by NIST [1]. If these tests fail, bias and correlation can be corrected by applying fuzzy extractor techniques [6] prior to bit filtering.

A. RO-PUF

The RO-PUF has been first described by Suh and Devadas [19]. In the general case, it uses a certain number of oscillating loops for which the oscillation frequencies are measured and compared. In the setting that we will analyze, and that had already been described in this seminal work, we will use $2n$ ring oscillators to generate n bits. To describe this PUF in our unified framework, we will define a challenge C as any n-bit string with Hamming weight exactly 1. If C^m is such that $c_i^m = 1$ iff $m = i$, then the delay difference δ_{C^m} will correspond to the frequency difference between oscillators $2m$ and $2m + 1$. Thus, the δ_{C^m} will be mutually independent. Therefore, our framework can be directly applied in order to estimate the reliability-entropy tradeoff in case filtering is used.

B. RO sum PUF

The RO sum PUF, or recombined oscillator, has been proposed by Yu and Devadas [20]. Instead of comparing the oscillator frequencies, they are measured, added or subtracted, before one bit is generated from the sign. More precisely, the $2n$ oscillators are divided into n pairs. Let $C = (c_i)_i$ be a challenge of length n. If d_i is the delay difference for the two oscillators of the i-th pair, then the total delay is obtained as

$$\delta_C = \sum_{i=1}^{n} d_i(-1)^{c_i}.$$

Here, d_i should be modeled as a realization from a normal law, with variance Σ_0^2. Therefore, we will have that $\Delta_C \sim \mathcal{N}(0, n\Sigma_0^2 = \Sigma^2)$. There are 2^n possible challenges, however, the delays for all these challenges will not be independent. It has been shown by Rioul et al.[17], for a different PUF but the same delay model, that the challenges are mutually independent if, when converted to $\{\pm 1\}$ vectors instead of $\{0, 1\}$ vectors, they are orthogonal. We can therefore find a subset of n challenges that are independent if a Hadamard matrix of rank n exists. This is always the case if n is a power of two or a multiple of 4 smaller than 668 [15]. Assuming this is the case, we can choose any such subset of challenges for the n possible challenges. Our framework can then be applied to this PUF.

C. Loop PUF

The Loop PUF, described by Cherif et al. [4], strongly resembles the RO sum PUF, with the exception that one configurable ring oscillator is used, instead of $2n$ simple ROs for the RO sum PUF. For the Loop PUF, each RO comprises n configurable and balanced delay element pairs. During delay measurement, the signal only passes through one half of the delay elements, this half being determined by the input challenge. The same measurement is then done for the complementary challenge, so that the signal passes through the other half of the delay elements, and the delay difference is then computed. The mathematical model is thus very similar to that of the RO sum PUF, with some minor differences. For example, in the RO sum PUF, the delays for the individual ring oscillators are first quantified and then added, which might lead to some rounding errors. This is less the case for the Loop PUF, since a total delay is directly measured. Thus, there are only two delay quantifications for the Loop PUF.

As shown by Rioul et al. [17], in order to obtain independent delay differences, and thus independent bits, the challenges need to be orthogonal, in the same sense as before. Thus, an entropy of n bit can be obtained by choosing a $n \times n$ Hadamard matrix for the challenges, if a Hadamard matrix of this size exists.

V. Experiments and Validation with Real Silicon

A. Architecture of the Test Circuit

We used Loop PUFs with $n = 64$ delay cells for our experiment. The cells use 65 nm CMOS technology, and each test chip contains 49 PUFs, embedded in a 7×7 matrix. We performed the delay measurements during $L = 2^{14}$ oscillation periods of the reference clock at $f_{ref} = 100$ MHz. This allows us to simulate:

- 49 Loop PUFs with 64 delay elements, or
- 64 RO-PUFs with 24 delay elements, or
- 64 RO sum PUFs with 48 delay elements.

Following [17], we choose a 64×64 Hadamard matrix as the challenge matrix to control the 49 Loop PUFs. The 64 challenge responses can therefore be considered independent. We perform $T = 1000$ measurements for each challenge and each PUF per chip. This directly yields the responses for the Loop PUF. In order to simulate a RO-PUF, we fix a challenge index m and consider the $24 \times T$ response delays:

$$\left\{ \delta_{m,t}^{2j} - \delta_{m,t}^{2j-1},\ j \in [1, 24] \right\}.$$

In a similar fashion, for the RO sum PUF we choose a 48×48 Hadamard matrix $\widetilde{C}$. For a fixed challenge index m of the Loop PUF, we then obtain $48 \times T$ response delays:

$$\left\{ \widetilde{C} \cdot \begin{pmatrix} \delta_{m,t}^{1} \\ \delta_{m,t}^{2} \\ \vdots \\ \delta_{m,t}^{48} \end{pmatrix} \right\}.$$

B. BER and Entropy Measurement

1) Results: Six test chips have been analyzed, and the measured BER and remaining entropy have been plotted in Fig. 6.

The error bars represent the range of values obtained among the tested chips. Although they do not share the exact same SNR, a middle value has been chosen, so that a simple comparison is possible. The SNR was calculated by estimating the variance of Δ_C and Z from the delay measurements of the test chips. Moreover, the range of measured SNRs is relatively small (between 180 and 250).

2) Discussion: For the remaining entropy, the measured and predicted values match quite closely. This seems to confirm the hypothesis of a Gaussian distribution for the average delay values. For the bit error rate however, the interpretation of the results seems more complicated. Indeed, while the BER for small filtering thresholds, and thus "large" BERs, seems to match our prediction, this is not the case for larger thresholds,

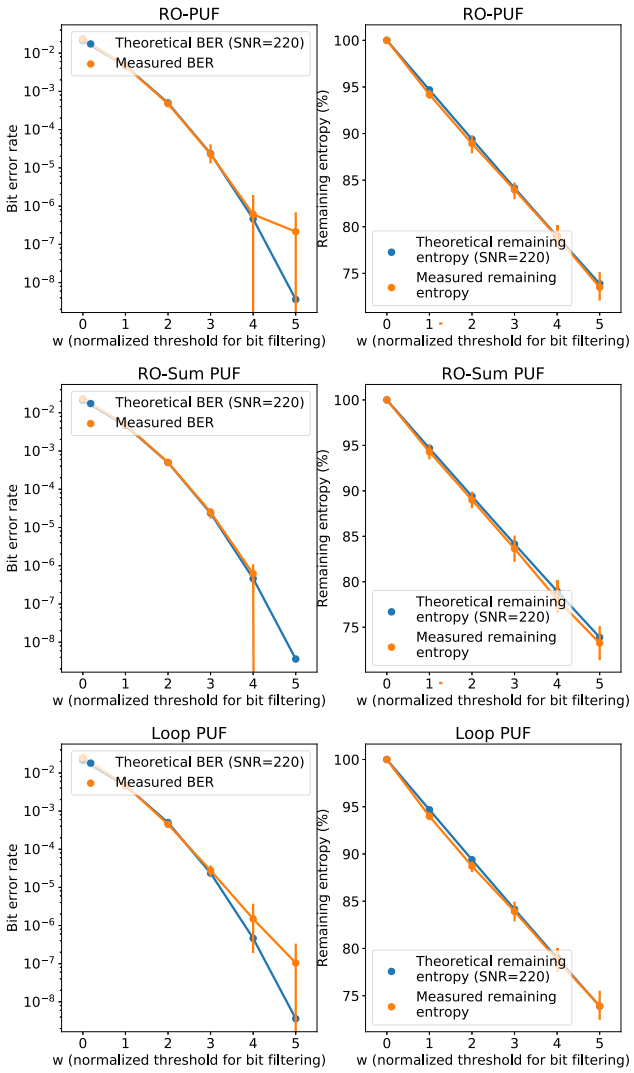

Figure 6. Experimental validation of the SNR and remaining entropy.

at least for the RO-PUF simulation and the Loop PUF. We can see two explanations for this:

First, the sample size is probably not large enough to reliably estimate probabilities around 10^{-8}. Indeed, for each chip, we record about 3 million samples, and thus, even one bit error would yield a BER, for that circuit, of more than $3 \cdot 10^{-7}$. Therefore, the BERs for parameters $W \geq 4$ come with a fairly large uncertainty.

The small sample size does not explain everything, however. When further analyzing the delay measurements, we notice that the noise distribution does not perfectly follow a Gaussian distribution. Indeed, on some chips, we observe multiple measurements that are more than 7σ away from the computed mean delay value, as taken over 1000 measurements. This should not happen more than once in about 500 million measurements, if the noise was truly Gaussian. Thus we must admit that the noise is not exactly Gaussian. More exactly, it

seems to be more heavy-tailed than a Gaussian noise. This could be an artifact of our experimental setup. Indeed, it forces us to wait a relatively long time span between measures, and the outliers could be explained, for example, with voltage fluctuations (the Loop PUF is relatively sensible to supply voltages changes). On the other hand, it should not come to a great surprise that a physical phenomenon does not exactly follow a Gaussian distribution. In order to derive a more precise model, other types of noise distributions need to be considered.

One can further notice that the divergence from the expected BER is almost absent from the RO sum PUF simulation. This can be easily explained. A simulated delay measurement for the RO sum PUF corresponds to the sum of 48 independent Loop PUF delay measurements. If only one Loop PUF measure is an outlier relative to the expected Gaussian noise distribution, this will less affect the whole sum. This also explains why the RO-PUF exhibits less divergent behavior than the Loop PUF, as any outlier will be summed with another delay measurement. These results, however, are possibly artifacts of our experimental setup, if we suppose that external factors cause these outlier measures. Indeed, in a real RO-PUF or RO sum PUF, all measures would certainly be done in parallel, and might be affected by the same glitch at the same time. Therefore, this does not say anything about the intrinsic robustness of these three PUF types.

VI. Effect of environmental changes: Temperature

PUFs are not necessarily used in the same environmental conditions they were enrolled at. Mainly two factors seem to be able to affect their behavior: temperature and input voltage [8]. We will assume that the input voltage can be controlled, via an voltage regulator for instance, and not further investigate in this direction. However, it is more complicated to control the temperature at which the PUF will be used, and it would therefore be helpful if it was possible to model the PUF-response dependency on temperature. In this section, we propose and test such a model. A similar model has been proposed by Maes [13], but for delay PUFs, it is possible to more directly test the model and make more straightforward predictions.

A. Assumptions

Given our experiments, we think that it is safe to make the following assumption: The delay response of a given oscillator is linearly dependent on the temperature, but the proportional constant might vary among ring oscillators. Testing on the Loop PUF circuits yielded a linear regression R^2 score above 0.999 for every oscillator. In addition, we will assume that this proportional constant follows a normal law. The curve for different oscillators in Figure 7 seems to validate this kind of distribution. Since in general, only the differences between ring oscillators are being considered (for the RO-PUF as well as the RO sum PUF), we can suppose that the probability distribution is centered. More formally, let's denote the temperature by θ, and the linear dependency coefficient by ℓ, where ℓ is a

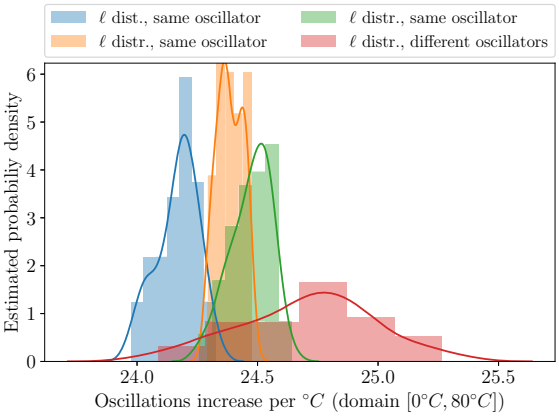

Figure 7. Distribution of temperature dependency coefficients (for 49 distinct oscillators, as well as 64 challenges of the same oscillator, for three different oscillators)

realization of a random variable $L \sim \mathcal{N}(0, \sigma_\theta)$. We therefore have the model for the temperature dependent PUF:

$$\mathcal{P}_\theta : C \mapsto \delta_C + Z + \ell\theta, \; b = \text{sign}(\delta_C + Z + \ell\theta) \qquad (11)$$

B. Average BER

We can now try to compute the average bit error rate over all average delays δ_C and dependency coefficients ℓ. As a reminder, the BER is defined here as

$$\widehat{BER}_\theta = P[\text{sign}(\Delta + Z + L\theta) \neq \text{sign}(\Delta)] \qquad (12)$$

Since Z and $L\theta$ are two centered independent Gaussian random variables, with variance respectively σ^2 and $\theta^2\sigma_\theta^2$, the sum is also a Gaussian random variable with variance $\sigma^2 + \theta^2\sigma_\theta^2$. Therefore, the result for the average BER obtained in III-A can be directly applied, by replacing σ with $\sqrt{\sigma^2 + \theta^2\sigma_\theta^2}$:

$$\widehat{BER}_\theta = \frac{1}{\pi}\arctan\left(\frac{\sqrt{\sigma^2 + \theta^2\sigma_\theta^2}}{\Sigma}\right) \qquad (13)$$

Thus, for the average BER, using the PUF at a temperature that is different from the enrollment temperature is equivalent to a loss of SNR. Of course, for individual delay measurements, this is not true, as the BER can exceed 0.5 if an inversion of the average sign happens due to the temperature difference, but it remains true for the average BER.

C. Effect on delay PUFs

The RO-PUF and RO sum PUF are equally affected by the temperature dependency of the ring oscillators on the temperature. Indeed, for the RO-PUF, the delay difference is simply the difference of delay among two oscillators, and the model can be directly applied as is. For the RO sum PUF, the total delay difference is actually the sum of a larger number of ring oscillator-pair delays. However, since the sum of Gaussian random variables still follows a Gaussian distribution, the same formula applies for the RO sum PUF, where σ, σ_θ and Σ are simply multiplied by the square root of the number of ring

oscillator pairs. Since the average BER only depends on the ratio between these quantities, the BER formula is unchanged.

The case of the Loop PUF is a little different. Indeed, the delay differences are measured on the same oscillator, and different challenges should have a similar temperature dependency. However, as Figure 7 shows, this is not exactly the case. While the temperature dependency coefficients vary less between challenges of the same oscillator than between oscillators, the variance is not zero. The model seems also applicable to the Loop PUF, albeit with a lower standard deviation σ_θ.

VII. Conclusion

This paper first presents the formalism to express the entropy and reliability of multiple delay PUFs: the RO-PUF, the RO sum PUF and the Loop PUF. We obtained a closed-form expression of the reliability which shows that the BER cannot go lower than about 10^{-3} even with large SNRs. The gain provided by the bit-filtering method that discards unreliable bits at enrollment phase has been formalized, giving a BER which can go to less than 10^{-10}.

The tradeoff between BER, entropy and complexity has been characterized. The resulting parameter selection for a given application is quite straightforward and simple. Practical experiments on few hundred PUFs designed in 65 nm CMOS process validate the theory. Testing the effect of temperature on the different types of PUFs is difficult when simulating with Loop PUFs. Tests with "native" PUFs might be necessary for a more thorough validation.

The Gaussian model for process and noise variables are validated by these experiments up to a certain threshold. Beyond, the Gaussian model may not be valid at the far tail of the noise distribution, and an adequate model for the noise distribution is a subject for future work. Such a model would allow more efficiently designs of PUFs with very low error rates. In particular, this model for reliability (also sometimes termed steadiness) is a suitable metric for stochastic models being developped in ISO/IEC 20897 project [9].

VIII. Acknowledgements

This work was partly supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2016-0-00399, Study on secure key hiding technology for IoT devices [KeyHAS Project]) and the ANR CHIST-ERA project SECODE (*Secure Codes to thwart Cyber-physical Attacks*). Alexander Schaub acknowledges the financial support from the *Direction Générale de l'Armement* (DGA).

References

[1] Lawrence E Bassham III, Andrew L Rukhin, Juan Soto, James R Nechvatal, Miles E Smid, Elaine B Barker, Stefan D Leigh, Mark Levenson, Mark Vangel, David L Banks, et al. Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications. 2010.

[2] Mudit Bhargava and Ken Mai. An efficient reliable PUF-based cryptographic key generator in 65nm CMOS. In *Proceedings of the conference on Design, Automation & Test in Europe*, page 70. European Design and Automation Association, 2014.

[3] Lilian Bossuet, Xuan Thuy Ngo, Zhoua Cherif, and Viktor Fischer. 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.

[4] Zouha Cherif, Jean-Luc Danger, Sylvain Guilley, and Lilian 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*, pages 156–162. IEEE Computer Society, 2012.

[5] Jeroen Delvaux. *Security Analysis of PUF-Based Key Generation and Entity Authentication*. PhD thesis, Shanghai Jiao Tong University, China, 2017.

[6] Jeroen Delvaux, Dawu Gu, Ingrid Verbauwhede, Matthias Hiller, and Meng-Day Mandel Yu. Efficient fuzzy extraction of puf-induced secrets: Theory and applications. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 412–431. Springer, 2016.

[7] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[8] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Silicon physical random functions. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, pages 148–160, New York, NY, USA, 2002. ACM.

[9] Sylvain Guilley, Soshi Hamaguchi, and Yousung 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.

[10] Matthias Hiller, Michael Weiner, Leandro Rodrigues Lima, Maximilian Birkner, and Georg Sigl. Breaking Through Fixed PUF Block Limitations with Differential Sequence Coding and Convolutional Codes. In *Proceedings of the 3rd International Workshop on Trustworthy Embedded Devices*, TrustED '13, pages 43–54, New York, NY, USA, 2013. ACM.

[11] Stefan Katzenbeisser, Ünal Kocabaş, Vladimir Rožić, Ahmad-Reza Sadeghi, Ingrid Verbauwhede, and Christian Wachsmann. PUFs: Myth, Fact or Busted? A Security Evaluation of Physically Unclonable Functions (PUFs) Cast in Silicon. In *CHES 2012*, volume 7428 of *Lecture Notes in Computer Science*, pages 283–301. Springer Berlin Heidelberg, 2012.

[12] Daihyun Lim, J.W. Lee, B. Gassend, G.E. Suh, M. van Dijk, and S. Devadas. Extracting secret keys from integrated circuits. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 13(10):1200 –1205, oct. 2005.

[13] Roel Maes. An accurate probabilistic reliability model for silicon PUFs. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 73–89. Springer, 2013.

[14] Roel Maes, Anthony Van Herrewege, and Ingrid Verbauwhede. Pufky: A fully functional puf-based cryptographic key generator. In *Proceedings of the 14th International Conference on Cryptographic Hardware and Embedded Systems*, CHES'12, pages 302–319, Berlin, Heidelberg, 2012. Springer-Verlag.

[15] Dragomir Ž Đoković. Hadamard matrices of order 764 exist. *Combinatorica*, 28(4):487–489, 2008.

[16] Donald B Owen. Tables for computing bivariate normal probabilities. *The Annals of Mathematical Statistics*, 27(4):1075–1090, 1956.

[17] Olivier Rioul, Patrick Solé, Sylvain Guilley, and Jean-Luc Danger. On the Entropy of Physically Unclonable Functions. In *ISIT, IEEE International Symposium on Information Theory*, July 2016. Barcelona, Spain.

[18] Boris Škoric, Pim Tuyls, and Wil Ophey. Robust key extraction from physical uncloneable functions. In *Applied Cryptography and Network Security*, volume 3531, pages 407–422. Springer, 2005.

[19] G. Edward Suh and Srinivas Devadas. 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*, pages 9–14. IEEE, 2007.

[20] Meng-Day Mandel Yu and Srinivas Devadas. Recombination of physical unclonable functions. *35th Annual GOMACTech Conference*, March 2010. Reno, NV, USA.