On the Entropy of Physically Unclonable Functions

Olivier Rioul[1], Patrick Solé[1], Sylvain Guilley[1,2] and Jean-Luc Danger[1,2]
[1]LTCI, CNRS, Télécom ParisTech, Université Paris-Saclay, 75 013 Paris, France.
[2]Secure-IC S.A.S., 35510 Cesson-Sévigné France.
Email: firstname.lastname@telecom-paristech.fr

***Abstract*—A physically unclonable function (PUF) is a hardware device that can generate intrinsic responses from challenges. The responses serve as unique identifiers and it is required that they be as little predictable as possible. A loop-PUF is an architecture where n single-bit delay elements are chained. Each PUF generates one bit response per challenge.**

We model the relationship between responses and challenges in a loop-PUF using Gaussian random variables and give a closed-form expression of the total entropy of the responses. It is shown that n bits of entropy can be obtained with n challenges if and only if the challenges constitute a Hadamard code. Contrary to a previous belief, it is shown that adding more challenges results in an entropy strictly greater than n bits. A greedy code construction is provided for this purpose.

I. Introduction

Having a unique identifier for each electronic chip allows to use them in a secure way. If, for example, the chip is used in a smartphone, the identifier can be used to associate the device with a specific service. The identifier can also be used to thwart overbuilding since it can be recorded at fabrication and can later be checked against a whitelist—in this way, overproduced or counterfeited chips can be detected.

However, for the identifier to be trusted, it must meet some security properties: essentially, it must be *unique* and it must not be *tamperable*. Physically Unclonable Functions (PUFs) are known as technical solutions [1]–[3]. A PUF consists in a hardware design (blueprint) such that every instance behaves differently after fabrication. Indeed, fabrication is not a deterministic process and the elements like transistors are built slightly different each time. In a PUF design, identical elements at blueprint level are compared once fabricated, the comparison results appear to be random. The elements' behavior after fabrication can thus be considered as unique. Moreover, as these elements are fragile, any tampering attempt is doomed to failure (e.g., destruction of the PUF).

The unique behavior after fabrication stems from a *static randomness* due to technological dispersion. It is a well known source of mismatch in electronics circuits design and was characterized by Pelgrom [4] to follow a normal distribution.

The PUF responses are also subject to *dynamic randomness* due to measurement noise, which is detrimental to the reliability of the PUF measurement. For this reason, it is important in practice to increase the signal-to-noise ratio (SNR). In a so-called SRAM-PUF [5] which consists in one SRAM memory bit which boots up at either value 0 or 1, it seems difficult to improve the SNR except by repeating measurements, which demands a power down between each measurement. In a delay-PUF [6], n elements are chained, and the total delay of the chain is measured. The SNR is then increased by a factor n as the signal power grows linearly with n. Because of this property, we focus on delay-PUFs in a *loop-PUF* structure [7], in which the delay chain is looped to form a ring oscillator by means of an inverter in order to measure time delays with high accuracy.

Problem Statement. A unique identification number can be obtained by querying the loop-PUF for M challenges. The quality of randomness of the responses depends on the choice of the challenges. This raises two questions:

- what is the best choice of challenges to get maximum entropy when reading a PUF?
- how many bits of entropy can be expected from one PUF?

Contributions. In this paper, we give a partial answer to both questions. For M challenges, we prove that the maximum possible value of entropy $= M$ bits can be obtained with as few as $n = M$ delay elements provided that the challenges are designed to form a Hadamard code. In addition, given n delays elements, we show that with $M > n$ challenges, an entropy strictly greater than n bits (albeit $< M$) can be obtained. We give a greedy algorithm to increase the entropy as much as possible beyond n bits as M increases.

Outline. The remainder of this paper is organized as follows. Section II provides a probabilistic model for the loop-PUF responses, which allows to express the amount of entropy of the responses as a function of the various challenges. Section III lists as a preliminary some known results on "orthant" probabilities. In Section IV, the least number of elements required to get an entropy of M bits is proved to be $n = M$, and the optimal set of the $n = M$ challenges is determined. Section V shows how to make the entropy grow beyond n bits by choosing more than n challenges. Section VI concludes and raises new perspectives.

II. Probabilistic Model for the Loop-PUF

Fig. 1 illustrates one delay element in the chain of n elements. For each $i = 1, 2, \dots, n$, element i can have two delays (theoretically equal at blueprint level), chosen according to one challenge bit[1] $c_i \in \{-1, 1\}$. Let $d(c_i)$ be the corresponding

[1] It is convenient to consider *signed* bits equal to ± 1 (instead of the usual $0, 1$) throughout this paper.

978-1-5090-1805-5/16/$31.00 ©2016 IEEE

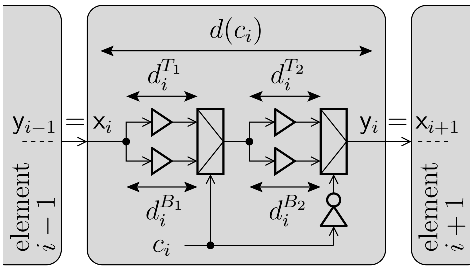


Fig. 1. Element i in a delay-PUF. In this blueprint, the triangle elements (▷) represent buffers, the triangle with circle elements (▷○) represent inverters, and the 2 → 1 rectangle (⧄) represent multiplexers. The output y_i is equal to the input x_i, but occurs after a delay $d(c_i)$, conditioned by challenge bit $c_i \in \{-1, 1\}$.

delay. As time is an extensive physical quantity, we have

$$d(c_i) = \begin{cases} d_i^{T_1} + d_i^{B_2} = d_i^{TB} & \text{if } c_i = -1, \\ d_i^{B_1} + d_i^{T_2} = d_i^{BT} & \text{if } c_i = +1. \end{cases}$$

The delays d_i^{TB} and d_i^{BT} are modeled as i.i.d. normal random variables selected at fabrication [4]. Fig. 2 illustrates the Gaussian nature of the propagation delay distribution.

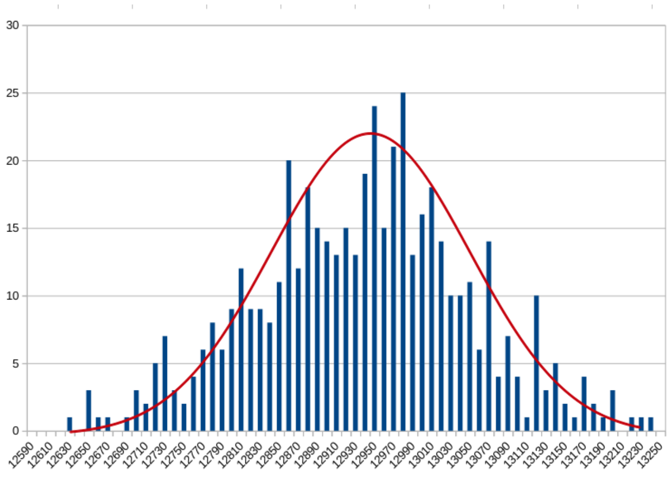


Fig. 2. Monte-Carlo simulation (with 500 runs) of the delays in a chain of 60 basic buffers implemented in a 55 nm CMOS technology. The mean value is 12938 ps, and the standard deviation is 115 ps.

The n delay elements are *chained* by connecting y_i to x_{i+1}, for $i = 1, \ldots, n-1$. The principle of the loop-PUF is to measure the difference Δ_c of cumulative delays $d(c) = \sum_{i=1}^{n} d(c_i)$ for a challenge $c = (c_1, \ldots, c_n)$ and its complementary value $-c = (-c_1, \ldots, -c_n)$:

$$\Delta_c = \sum_{i=1}^{n} d(c_i) - d(-c_i) = \sum_{i=1}^{n} c_i(d_i^{TB} - d_i^{BT}). \tag{1}$$

Since d_i^{TB} and d_i^{BT} are i.i.d. normal, the random variables

$$\Delta_i = d_i^{TB} - d_i^{BT} \qquad (i = 1, 2, \ldots, n) \tag{2}$$

are themselves i.i.d. normal and have zero mean. Each Δ_i represents the delay difference from x_i to y_i in the path through first top/second bottom and first bottom/second top buffers.

As the delay measurement requires a great accuracy, the delay chain is looped to form a ring oscillator and the delay $\Delta_c = \sum_{i=1}^{n} c_i \Delta_i$ is obtained by inverting the measured frequency of the ring oscillator, hence the name “loop-PUF”. It is the *sign* of the cumulative delay difference Δ_c that yields one bit of unique identifier. The overall loop-PUF function is summarized in Fig. 3. A unique identification number can be obtained by querying the loop-PUF for M different challenges $c = (c_1, c_2, \ldots, c_n)$.

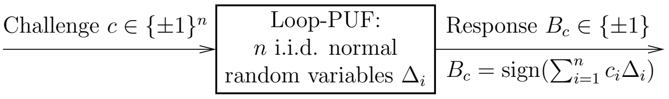

Challenge $c \in \{\pm 1\}^n$
Loop-PUF:
n i.i.d. normal
random variables Δ_i
Response $B_c \in \{\pm 1\}$
$B_c = \text{sign}(\sum_{i=1}^{n} c_i \Delta_i)$

Fig. 3. Operation of a loop-PUF

Definition 1. A *challenge* c is a vector of n control bits $c = (c_1, c_2, \ldots, c_n) \in \{\pm 1\}^n$. Let $\Delta_1, \Delta_2, \ldots, \Delta_n$ be i.i.d. zero-mean normal (Gaussian) variables characterizing the technological dispersion. A *bit response* to challenge c is defined as

$$B_c = \text{sign}(\Delta_c) \in \{\pm 1\} \tag{3}$$

where

$$\Delta_c = c_1\Delta_1 + c_2\Delta_2 + \cdots + c_n\Delta_n. \tag{4}$$

Definition 2. A challenge *code* $\mathcal{C}$ is a set of M n-bit challenges that form a (n, M) binary code. We shall identify $\mathcal{C}$ with the $M \times n$ matrix of ± 1’s whose lines are the challenges.

The M codewords and their complements are used to challenge the PUF elements. The corresponding identifier is the M-bit vector

$$B = (B_c)_{c \in \mathcal{C}}. \tag{5}$$

The *entropy* of the PUF responses is denoted by $H = H(B)$.

To increase uniqueness of the PUF response—the fact that it should be intrinsic for a given device—we aim at finding the best code $\mathcal{C}$ such that the resulting entropy $H = H(B)$ of the different responses over different devices is *maximal*.

III. Orthant Probability of a Multivariate Normal

The determination of the entropy $H(B)$ requires that of the joint probabilities of signs of Gaussian variables. Let $X_1, X_2, \ldots, X_n$ be zero-mean, jointly Gaussian (not necessarily independent) and identically distributed. As a prerequisite to the derivations that follow, we wish to compute the *orthant probability*

$$\mathbb{P}(X_1 > 0, X_2 > 0, \ldots, X_n > 0).$$

The probabilities associated to other sign combinations can easily be deduced from it using the symmetry properties of the Gaussian distribution.

Since the value of the orthant probability does not depend on the common variance of the random variables we may assume without loss of generality that each X_i has unit variance:

$X_i \sim \mathcal{N}(0,1)$. The orthant probability will depend only on the correlation coefficients

$$\rho_{i,j} = \mathbb{E}(X_i X_j) \qquad (i \neq j). \tag{6}$$

For $n = 1$ we obviously have $\mathbb{P}(X_1 > 0) = \frac{1}{2}$. For $n = 2$, we have the following known result [8] which dates back at least to Hermite [9].

Lemma 1 (Quadrant probability of a bivariate normal)**.**

$$\mathbb{P}(X_1 > 0, X_2 > 0) = \frac{1}{4} + \frac{\arcsin \rho_{1,2}}{2\pi}. \tag{7}$$

Neither [8] nor [9] contains a proof. For completeness we provide a simple argument.

Proof: Let $\theta = \arcsin \rho_{1,2} \in [-\pi/2, \pi/2]$ and let $\tilde{X}_1$ be an independent copy of X_1. Then $\tilde{X}_2 = X_1 \sin\theta - \tilde{X}_1 \cos\theta$ is also normal $\mathcal{N}(0,1)$ with $\mathbb{E}(X_1\tilde{X}_2) = \sin\theta = \rho_{1,2}$. Therefore, (X_1, X_2) has the same distribution as $(X_1, \tilde{X}_2)$ and

$$\begin{aligned}\mathbb{P}(X_1 > 0, X_2 > 0) &= \mathbb{P}(X_1 > 0, \tilde{X}_2 > 0)\\ &= \mathbb{P}(X_1 > 0, \tilde{X}_1 < X_1 \tan\theta).\end{aligned} \tag{8}$$

Since $(X_1, \tilde{X}_1)$ is isotropic, by the Box-Muller transformation in polar coordinates [10] we can write $X_1 = R\cos U$, $\tilde{X}_1 = R \sin U$ where U is uniformly distributed in $[-\pi, \pi]$. Therefore,

$$\begin{aligned}\mathbb{P}(X_1 > 0, X_2 > 0) &= \mathbb{P}(\cos U > 0, \tan U < \tan\theta)\\ &= \mathbb{P}(|U| < \pi/2, U < \theta) = \frac{\theta + \pi/2}{2\pi}\end{aligned} \tag{9}$$

which is the announced formula. ■

For $n = 3$, we have the lesser known extension [11].

Lemma 2 (Orthant probability of a trivariate normal)**.**

$$\begin{aligned}&\mathbb{P}(X_1 > 0, X_2 > 0, X_3 > 0)\\ &\quad = \frac{1}{8} + \frac{\arcsin \rho_{1,2} + \arcsin \rho_{2,3} + \arcsin \rho_{1,3}}{4\pi}.\end{aligned} \tag{10}$$

Again for completeness we provide a simple proof.

Proof: The complementary probability is

$$\begin{aligned}1 - &\mathbb{P}(X_1 > 0, X_2 > 0, X_3 > 0)\\ &= \mathbb{P}(X_1 < 0 \text{ or } X_2 < 0 \text{ or } X_3 < 0)\\ &= \mathbb{P}(X_1 > 0 \text{ or } X_2 > 0 \text{ or } X_3 > 0)\end{aligned} \tag{11}$$

by symmetry of the Gaussian. By the inclusion-exclusion principle, this equals

$$\begin{aligned}&\mathbb{P}(X_1 > 0) + \mathbb{P}(X_2 > 0) + \mathbb{P}(X_3 > 0) - \mathbb{P}(X_1 > 0, X_2 > 0)\\ &\quad - \mathbb{P}(X_2 > 0, X_3 > 0) - \mathbb{P}(X_3 > 0, X_1 > 0)\\ &\quad + \mathbb{P}(X_1 > 0, X_2 > 0, X_3 > 0)\end{aligned}$$

hence

$$\begin{aligned}&1 - 2\mathbb{P}(X_1 > 0, X_2 > 0, X_3 > 0)\\ &\quad = \frac{3}{2} - \frac{3}{4} - \frac{\arcsin \rho_{1,2}}{2\pi} - \frac{\arcsin \rho_{2,3}}{2\pi} - \frac{\arcsin \rho_{1,3}}{2\pi}\end{aligned} \tag{12}$$

which yields the announced formula. ■

Unfortunately, no such closed-form formula seems to exist for $n \geq 4$ and one has then recourse to numerical computation [12].

IV. Minimum Number of Elements Reaching an Entropy of M Bits

In this section we find the optimal (n, M) code of length n such that the entropy $H(B)$ corresponding M-bit vector B attains its maximum possible value $H(B) = M$ bits.

Lemma 3. *The signs of two jointly Gaussian identically distributed zero-mean variables are independent if and only if the two variables are independent.*

Proof: One direction is obvious. For the other, let X_1, X_2 be zero-mean Gaussian with correlation factor $\rho_{1,2}$ and assume that $\operatorname{sign}(X_1)$ and $\operatorname{sign}(X_2)$ are independent. Since X_1, X_2 have zero mean, $\operatorname{sign}(X_1), \operatorname{sign}(X_2)$ are each equiprobable ± 1 with probability $1/2$, hence

$$\mathbb{P}(X_1 > 0, X_2 > 0) = \mathbb{P}(X_1 > 0)\mathbb{P}(X_2 > 0) = \frac{1}{4}. \tag{13}$$

From Lemma 1 this implies $\arcsin \rho_{1,2} = 0$ hence $\rho_{1,2} = 0$. Since X_1, X_2 are uncorrelated and jointly Gaussian, they are independent. ■

As an example, when $n = 2$, the code $\mathcal{C}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ generates responses

$$\begin{cases} B_1 = \operatorname{sign}(\Delta_1 + \Delta_2),\\ B_2 = \operatorname{sign}(\Delta_1 - \Delta_2).\end{cases}$$

Since Δ_1 and Δ_2 are i.i.d. normal, $\Delta_1 + \Delta_2$ and $\Delta_1 - \Delta_2$ are also i.i.d. normal, and the entropy of $B = (B_1, B_2)$ equals 2 bits. By lemma 3, it follows that up to equivalence (trivial permutations), $\mathcal{C}_2$ is the only code achieving $H(B) = 2$ bits. The general case is given by the following

Theorem 1. *Up to equivalence, the only code of minimum length n such that the corresponding M-bit vector B has maximum entropy $H(B) = M$ bits is the Hadamard (n, n) code of length $n = M$.*

Proof: Let $\Delta \in \mathbb{R}^n$ be the zero-mean Gaussian (column) vector $(\Delta_1, \Delta_2, \ldots, \Delta_n)^t$. Without loss of generality, we assume that its components are i.i.d. $\sim \mathcal{N}(0,1)$. Let $\mathcal{C}$ be the $M \times n$ matrix of ± 1's whose lines are the challenges (codewords). The product $\mathcal{C}\Delta$ is a Gaussian vector with covariance matrix $\mathcal{C}\mathcal{C}^t$, and the M-bit vector B is the vector of signs of the components of $\mathcal{C}\Delta$. Each of these signs B_c is an equiprobable binary variable $\in \{\pm 1\}$ with one bit of entropy. Therefore,

$$H(B) \leq \sum_{c \in \mathcal{C}} H(B_c) = M \text{ bits} \tag{14}$$

with equality if and only if the B_c's are independent. Now suppose that $H(B) = M$. Since the B_c's are pairwise independent, by Lemma 3 it follows that the components of $\mathcal{C}\Delta$ are (pairwise) uncorrelated. Thus $\mathcal{C}\mathcal{C}^t = n\mathbf{I}$. Because $\mathcal{C}\mathcal{C}^t$ has full rank, it follows that $n \geq M$, hence the minimal value of n is $n = M$. In this case all lines of the $n \times n$ matrix $\mathcal{C}$ of ± 1s are mutually orthogonal, hence $\mathcal{C}$ is an Hadamard matrix whose lines form the (n, n) Hadamard code. ■

Remark 1. Any Hadamard matrix (not necessarily of Sylvester type) can be used for the (generalized) Hadamard code. Thus in Theorem 1, n is necessarily one of $n = 1, 2$, or a multiple of 4 (up to at least 662 in 2005—cf. Hadamard's conjecture [13]). For instance, for $n = 12$, the unique (up to equivalence) Hadamard matrix which can be found in Magma `HadamardDatabase` list is

$$\mathcal{C}_{12} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & 1 & -1 \\ 1 & -1 & 1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \end{pmatrix}$$

which gives $H = 12$ bits.

Remark 2. The elementary delay-PUF structure in Fig. 1 consists of two stages. Thus, irrespective of the challenge bit c_i, the path from x_i to y_i travels through the same number of top and bottom buffers. In a previous work [14], [15], the considered structure was simpler as it had only one stage. However, the counterpart is that challenges had a fixed Hamming weight (globally, or by subsets). Now considering the Hadamard code where the all-one codeword is removed, all remaining codewords have Hamming weight $n/2$. Therefore, in this specific case, the complexity of the PUF can be halved by cutting one stage for all elements of Fig. 1.

V. Increasing Entropy with Additional Challenges

Theorem 1 states that for n delay elements, only the first n challenges can bring up to n bits of entropy. However, one can always use $M > n$ challenges to further increase the entropy. Of course each additional challenge will only bring strictly less than one bit of information so that $n < H(B) < M$. This is to be contrasted with the state-of-the-art for other kinds of delay-PUFs (e.g., [5] for an SRAM-PUF and [16] for a delay-PUF based on ring oscillators) for which the obtained entropy was always smaller than n bits irrespective of the number of challenges.

A. Case $n = 3$

In this case there is no $(3, 3)$ Hadamard code. To compute the maximal possible entropy for M challenges, we proceed as follows.

For $M = 1$, we consider without loss of generality the codeword $c = (1, 1, 1)$. The entropy of B_c is 1 bit.

For $M = 2$, appending the opposite codeword $(-1, -1, -1)$ does not increase entropy. However, all other non trivial solutions with one or two -1 are equivalent (they yield the same code up to permutation). Consider for example

$$\mathcal{C}_2 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \end{pmatrix}.$$

Using Lemma 1, it is easily found that the corresponding orthant probability is

$$p = \frac{1}{4} + \frac{\arcsin \frac{1}{3}}{2\pi}$$

so that the resulting entropy is

$$H(B) = -2p \log p - (1 - 2p) \log(\frac{1}{2} - p) \approx 1.966 \text{ bits.}$$

For $M = 3$, we can add one of the two codewords $(-1, 1, 1)$ or $(-1, 1, -1)$, yielding a code equivalent to

$$\mathcal{C}_3 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ -1 & 1 & 1 \end{pmatrix}.$$

It is easily checked using Lemma 2 that the entropy of $B = (B_1, B_2, B_3)$ is then

$$\begin{aligned} H(B) = &-6 \left(\frac{1}{8} + \frac{\arcsin \frac{1}{3}}{4\pi}\right) \log \left(\frac{1}{8} + \frac{\arcsin \frac{1}{3}}{4\pi}\right) \\ &- 2 \left(\frac{1}{8} - 3\frac{\arcsin \frac{1}{3}}{4\pi}\right) \log \left(\frac{1}{8} - 3\frac{\arcsin \frac{1}{3}}{4\pi}\right) \\ &\approx 2.875 \text{ bits.} \end{aligned}$$

For $M = 4$, the entropy can still be increased by appending $(-1, 1, -1)$, yielding

$$\mathcal{C}_4 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ -1 & 1 & 1 \\ -1 & 1 & -1 \end{pmatrix}.$$

As there is no closed-form expression we found by Monte-Carlo simulation that

$$H(B) \approx 3.666 \text{ bits,}$$

which is strictly greater that $n = 3$. It is easily seen that adding more codewords does not increase the entropy anymore, hence this value is the maximum possible entropy.

B. Cases $n = 4$ and $n = 8$

In these cases there is an (n, n) Hadamard code. It is clear that for any $M \leq n$, the expurgated Hadamard code of parameter (n, M) yields $H(B) = M$ bits. The interesting question is the computation of $H(B)$ for $M > n$.

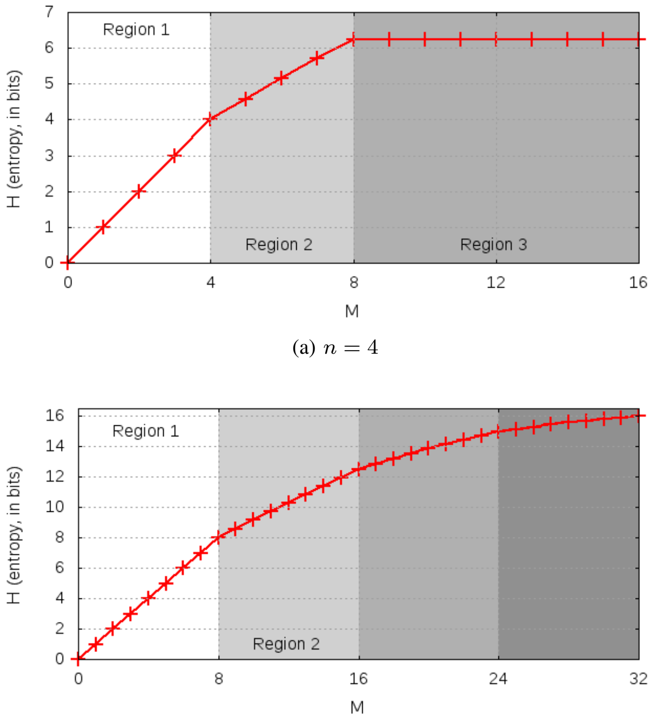
For this purpose, we have carried out a greedy search, detailed in Algorithm 1. Obviously, in the cases $n = 4, 8$, the algorithm can easily be adapted to start from the (n, n) Walsh-Hadamard code.

The result of Algorithm 1 is plotted in Fig. 4. One observes several regimes. For $M \leq n$, the entropy grows linearly as $H(B) = M$ as proved in Theorem 1. For $n < M \leqslant 2n$, the entropy continues to grow linearly with a smaller slope. For $n = 4$, the appended codewords at iterations $m = 5, 6, 7$ and 8 are respectively $(-1, 1, 1, 1)$, $(1, -1, 1, 1)$, $(1, 1, -1, 1)$, and $(1, 1, 1, -1)$. Finally, when $n = 4$, for $M \geqslant 2n$, the entropy remains unchanged, at a maximal value ≈ 6.251 bits.

For $n = 8$, we observe more regimes. The slopes are 1, 0.558, 0.311 and 0.143 bit per new challenge, when M is in the range $[0, 8]$, $[8, 16]$, $[16, 24]$, and $[24, 32]$ respectively. We observed that in each region, the challenges added to the code are pairwise orthogonal.

```
input  : Length $n$, number of challenges $M$
output : Code $\mathcal{C}$ of parameters $(n, M)$ and entropy $H(B)$
$\mathcal{C} \leftarrow \emptyset$
for $m \in \{1, \dots, M\}$ do
    $c_{\text{best}} \leftarrow (1, \dots, 1)$ ; $H_{\text{best}} \leftarrow 0$
    for $c \in \{\pm 1\}^n$ do
        if $c \notin \mathcal{C}$ and $-c \notin \mathcal{C}$ then // Optimization
            $\mathcal{C}' \leftarrow \mathcal{C} \cup \{c\}$
            $H = H(B)$ where $B = (B_c)_{c \in \mathcal{C}'}$
            if $H > H_{\text{best}}$ then
                $c_{\text{best}} \leftarrow c$ ; $H_{\text{best}} \leftarrow H$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{c_{\text{best}}\}$ // Saving best codeword
return $(\mathcal{C}, H_{\text{best}})$
```

Algorithm 1: Greedy algorithm for increasing the entropy as the number of challenges increases.



(a) $n = 4$

(b) $n = 8$

Fig. 4. Entropy given by Alg. 1 for n elements as a function of the number M of challenges.

VI. Conclusion

It was previously believed [16] that a PUF made up of n elements can only provide less than n bits of entropy (the entropy of the whole cannot be greater than the sum of the entropies of its parts). This would be the case for the SRAM-PUF where each element is independent from the others. In contrast, we have shown that by aggregating n PUF elements together as in delay-PUFs, it is possible to attain higher values of entropy $> n$. This is especially important as the state-of-the-art attacks against PUFs [17], [18] were set up to predict the response of a PUF only from a few pairs of known challenges and responses.

Acknowledgments

The authors are grateful to Florent Lozach and Thibault Porteboeuf (Secure-IC S.A.S.) for the Monte-Carlo simulations of propagation delays in delay-PUFs. Inspiring discussions with Michèle Wigger are also gratefully acknowledged. Sylvain Guilley, Soshi Hamaguchi and Yousung Kang are co-editors of ISO 20897, and they wish to thank working group 3 of subcommittee ISO/IEC JTC1/SC 27 for supporting this standard.

References

[1] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Controlled physical random functions," in *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*, 2002, pp. 149 – 160.

[2] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *DAC*, 2007, pp. 9–14.

[3] J. Guajardo, S. S. Kumar, G. J. Schrijen, and P. Tuyls, "FPGA Intrinsic PUFs and Their Use for IP Protection," in *CHES*, ser. Lecture Notes in Computer Science (LNCS). Springer, 2007, pp. 63–80.

[4] M. J. Pelgrom, A. C. Duinmaijer, and A. P. Welbers, "Matching properties of MOS transistors," *IEEE Journal of Solid State Circuits*, vol. 24, no. 5, pp. 1433–1439, 1989, dOI: 10.1109/JSSC.1989.572629.

[5] D. E. Holcomb, W. P. Burleson, and K. Fu, "Power-up SRAM state as an identifying fingerprint and source of true random numbers," *IEEE Trans. Computers*, vol. 58, no. 9, pp. 1198–1210, 2009.

[6] S. Morozov, A. Maiti, and P. Schaumont, "An analysis of delay based PUF implementations on FPGA," in *ARC 2010, Bangkok, Thailand, March 17-19, 2010. Proceedings*, ser. Lecture Notes in Computer Science, P. Sirisuk, F. Morgan, T. A. El-Ghazawi, and H. Amano, Eds., vol. 5992. Springer, 2010, pp. 382–387.

[7] 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.

[8] S. S. Wolff and J. L. Gastwirth, "The effect of autoregressive dependence on a nonparametric test," *IEEE Transactions on Information Theory*, pp. 311–313, April 1967.

[9] T. J. Stieltjes, "Extrait d'une lettre à M. Hermite sur l'intégrale $\int_0^\infty \int_0^\infty e^{-(ax^2+2bxy+cy^2)}dxdy$," *Bulletin des Sciences Mathématiques, 2e Série, rédigé par M. Darboux. Paris.*, vol. 13, pp. 170–172, 1889.

[10] G. E. P. Box and M. E. Muller, "A note on the generation of random normal deviates," *Ann. Math. Statist.*, vol. 29, no. 2, pp. 610–611, 06 1958. [Online]. Available: http://dx.doi.org/10.1214/aoms/1177706645

[11] C. Rose and M. Smith, *Mathematical Statistics with Mathematica*. New York: Springer-Verlag, 2002.

[12] I. G. Abrahamson, "Orthant probabilities for the quadrivariate normal distribution," *The Annals of Mathematical Statistics*, vol. 35, no. 4, pp. 1685–1703, 1964.

[13] A. S. Hedayat and W. D. Wallis, "Hadamard matrices and their applications," *Ann. Statist.*, vol. 6, no. 6, pp. 1184–1238, 11 1978.

[14] Z. Cherif, J.-L. Danger, S. Guilley, J.-L. Kim, and P. Solé, "Multiply constant weight codes," in *ISIT*. IEEE, 2013, pp. 306–310, Turkey.

[15] Y. M. Chee, Z. Cherif, J. Danger, S. Guilley, H. M. Kiah, J. Kim, P. Solé, and X. Zhang, "Multiply Constant-Weight Codes and the Reliability of Loop Physically Unclonable Functions," *IEEE Transactions on Information Theory*, vol. 60, no. 11, pp. 7026–7034, 2014.

[16] R. Maes, A. Van Herrewege, and I. Verbauwhede, "PUFKY: A Fully Functional PUF-based Cryptographic Key Generator," in *Proceedings of CHES '12*, ser. LNCS, vol. 7428. Springer-Verlag, 2012, pp. 302–319.

[17] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proceedings of the 17th ACM conference on Computer and communications security*, ser. CCS '10. New York, NY, USA: ACM, 2010, pp. 237–249.

[18] F. Ganji, S. Tajik, and J. Seifert, "Why Attackers Win: On the Learnability of XOR Arbiter PUFs," in *TRUST 2015, Heraklion, Greece, August 24-26, 2015, Proceedings*, ser. Lecture Notes in Computer Science, M. Conti, M. Schunter, and I. G. Askoxylakis, Eds., vol. 9229. Springer, 2015, pp. 22–39.