Design and Analysis of Stability-Guaranteed PUFs

Wei-Che Wang, Yair Yona, Suhas Diggavi and Puneet Gupta

Department of Electrical Engineering, University of California, Los Angeles

{weichewang, suhasdiggavi, yairyo99, puneetg}@ucla.edu

Abstract—The lack of stability is one of the limitations that constrains Physical Unclonable Function (PUF) from being put in widespread practical use. In this paper, we propose a weak PUF and a strong PUF that are both completely stable. These PUFs are called Locally Enhanced Defectivity Physical Unclonable Function (LEDPUF). A LEDPUF is a pure functional PUF that does not require any kinds of correction schemes as conventional parametric PUFs do. The source of randomness of a LEDPUF is extracted from locally enhance defectivity without affecting other parts of the chip. In this work we construct a weak LEDPUF by forming arrays of Directed Self Assembly (DSA) random connections, and the strong LEDPUF is implemented by using the weak LEDPUF as the key of a keyed-hash message authentication code (HMAC). Our simulation and statistical results show that the entropy of the weak LEDPUF bits is close to ideal, and the inter-chip Hamming distances of both weak and strong LEDPUFs are about 50%, which means that these LEDPUFs are not only stable but also unique.

We develop a new unified framework for evaluating the security of PUFs, based on password security, by using information theoretic tools of guesswork. The guesswork model allows to quantitatively compare, with a single unified metric, PUFs with varying levels of stability, bias and available side information. In addition, it generalizes other measures to evaluate the security level such as min-entropy and mutual information. We evaluate guesswork-based security of some measured SRAM and Ring Oscillator PUFs as an example and compare them with LEDPUF to show that stability has a more severe impact on the PUF security than biased responses. Furthermore, we find the guesswork of two new problems: Guesswork under probability of attack failure, and the guesswork of strong PUFs that are used for authentication.

I. Introduction

A Physical Unclonable Function (PUF) is a small piece of circuitry such that its behavior, or Challenge Response Pair (CRP) [2], is uniquely defined and it is hard to be predicted and replicated because of the intrinsic random physical nature and the uncontrollability of process variations. As a security primitive, PUF can enable low overhead hardware identification, tracing, and authentication during the global manufacturing chain. The first PUF was introduced more than a decade ago [3]. Since then, many silicon PUF implementations have been proposed, such as Arbiter PUF [4], Ring Oscillator (RO) PUF [5], SRAM PUF [6], and many other variations.

Since the key commonality between all current silicon PUF implementations is their use of *parametric* manufacturing

W. Wang, S. Diggavi, and P. Gupta are with the Department of Electrical and Computer Engineering, University of California Los Angeles, Los Angeles, CA.

Y. Yona was with the Department of Electrical and Computer Engineering, University of California Los Angeles, Los Angeles, CA. He is now with Intel Labs, Santa Clara, CA (e-mail: yairyo99@ucla.edu).

variations, two of the most challenging design tasks a PUF designer will encounter are :

1) How to make the PUF unique and stable even under extreme conditions without expensive implementation cost?
2) How to evaluate the security level of a PUF given its uniqueness and stability measurements?

A. Limitations of Parametric PUFs

1) Random Local Variation Extraction: One of the major concerns of parametric PUFs is that local variation should be the *only* entropy source for these PUFs [7]. However, from our experiments on a large silicon data set [8], only 13% of total variation is random local variation, which means that most variation is coming from global or spatial variation. Any attempt to use global or spatial variation as the source of randomness can make them vulnerable to a class of *process side channel attacks*. For instance, two PUFs on the same (X,Y) location on different wafers are highly correlated (due to large wafer-level systematics present in most modern fabrication processes). As a result, a few sacrificial wafers can aid in developing a relatively straightforward side channel attack. We tested this side channel attack on silicon RO PUF measurements in 65nm technology across 300 wafers. Figure 1 shows that the inter-distance [7] on the same (X,Y) is much smaller than the inter-distance across all PUFs. Therefore, an adversary with possession of a reference PUF, which is fabricated at the same (X,Y) location as the target PUF, would have a higher probability of guessing the correct answer than random guessing. The radial nature of systematic across wafer variation [8] means that just a few reference PUFs drawn carefully may be sufficient for attackers instead of keeping full sacrificial wafers.
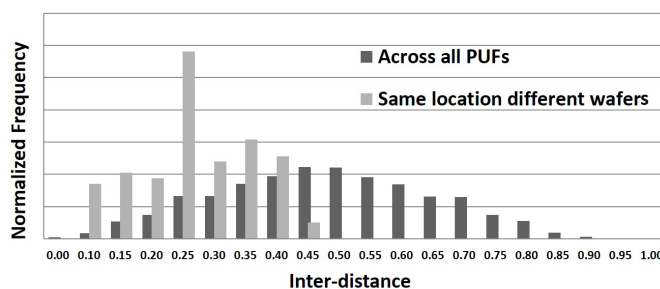

Fig. 1. The inter-distance of PUFs from same (X,Y) location on different wafers is much smaller than that of across all PUFs, which demonstrates a possible side channel attack.

2) Measurement Noise: Measurement noise could be another big issue for parametric PUFs and needs to be carefully compensated. For instance, metastability of the arbiter circuit for Arbiter PUFs and accumulated jitter in RO PUFs can be sources of measurement noises. For weak PUF measurement, we evaluate the intra-distance [7] of SRAM PUFs using fifteen commercial 45nm SOI test chips, where each consists 176kB data memory. The power-up state is measured 10 times during an 8-hour period, and the mean of intra-distance distribution is 2.57%. Since the experiment is done in room temperature with exactly same settings, the difference is essentially contributed by the measurement noise.

3) Environmental fluctuations and wearout: Existing silicon PUFs are in nature susceptible to environmental fluctuations [9] and wearout [10]. To account for the instability issue, techniques such as error correction code (ECC), helper data or fuzzy comparator must be applied. A possible worst case scenario is when the environmental factors change significantly, for instance, the PUF is enrolled at 20°C and is verified at 80°C. In this case, a practical fuzzy extraction process needs to have the ability to still recover the initial PUF response.

B. Techniques to Improve Parametric PUF Quality

A variety of techniques have been intensively studied over the years to extract random local variations or to make a PUF more stable and reliable. A Non-Volatile Memory (NVM) based PUF without helper data is presented in [11]. However, besides its hardware and calibration overhead, the results of uniqueness and entropy analysis are also missing. In [12], the local randomness is distilled by modeling and subtracting the systematic variation. A similar technique is to subtract the averaged frequency from multiple measurements to reveal the true local random variation [13]. However, the calculation and information storage requirement come with the cost of addition latency and hardware. Taking the majority vote [14] or finding stable responses [15] are possible techniques to eliminate the measurement noise, however, at the cost of large latency or reduced number of challenges. Other complex implementations have been proposed to mitigate stability issues that often induce lower hardware efficiency [7] or making the PUF more susceptible to attacks [16]. Also, to protect PUFs from the worst case scenario as described creates overhead as it requires to employ strong ECC [17]. In [18] a reverse fuzzy extraction technique is proposed, which can indeed reduce the implementation cost of error correcting algorithm, however, a hash function is still needed, and to prevent the attacker from immersing the PUF in an stable environment, a True Random Number Generator (TRNG) is required as stated in [19].

The issues of parametric PUFs, such as the described instability, wearout, measurement noise, limited local variation, and limited side channel attack resiliency, clearly motivate the need to design PUFs that do not rely on parametric performance variations as the entropy source.

C. Quantifiable Security Evaluation Model

In order to impersonate the hardware, the PUF attacker needs to respond to a challenge with a correct response (*i.e.,* guess a secret). [1] Comprehensive security models for PUFs are described in [20], including a precise identification of required PUF properties, such as indistinguishably and tamper-resilience. Though this specifies the security requirements, as a "checklist", we believe that a more quantitative assessment of PUF security can be valuable for both PUF designers and PUF users.[2] Inter- and intra- fractional Hamming Distance (FHD) [7], and other statistical tests for randomness [21], have been used for quantifying PUF security. Though it is reasonable that having larger inter-FHD is more secure, it does not tell the PUF designer how much more secure it is. For example, is it worth raising the inter-FHD from 40% to 49% at a cost of extra hardware? In this work we present a more principled way to analyze PUF security by connecting it to how one could evaluate password security, through a guesswork framework. We derive a theoretical framework for PUF security evaluation that brings together two important properties of existing PUFs: predictability and reproducibility [22]. This framework enables a unified security quantification of several effects including bias, noise, and side-channels on PUFs, as well as the security over multiple challenge-response pairs, providing design guidance by quantifying the security level of a PUF.

In the context of the question raised in the previous paragraph, we show that in terms of guesswork the effect of noise is far worse than the effect of bias. Therefore, the tools presented in this paper to evaluate PUFs security, can be used by a PUF designer to determine how to maximize the security level at minimum cost in terms of resources. The actual answer to this question depends on the tradeoff between the security level in terms of guesswork, and the actual cost in resources on the designer end, required to achieve this.

One can think of PUF signatures like passwords, and its breakability should be evaluated by how strong it is, for example, how many attempts (on average) does it take to compromise it. This *guessing* framework has been studied in the information theory literature and has recently been adopted by NIST as a measure of password security [23]. We bring this framework to evaluating the security of PUFs.

D. The Contributions of this Paper

In [1] we have presented (1) the first stability-guaranteed silicon PUF through Locally Enhanced Defectivity (LED) and (2) detailed constructions of the weak and strong LEDPUF using random DSA connections. In this paper, we significantly expand our work in [1] and the contributions of this paper are:

- We present a new unified framework for evaluating the security of PUFs through guesswork analysis. This framework enables us to evaluate and quantify the effect of noise, bias and model attacks on security. We also relate guesswork to other security measures such as min-entropy, and mutual information. The model quantitatively measures the security of various PUFs under

---

[1]In this attacker model the adversary does not have access to the PUF, but rather is trying to impersonate it.

[2]In this paper we do not study "machine learning" attacks, in which the attacker has access to the PUF and is challanging the PUF to learn the underlying randomness.

different scenarios, and by doing so enables us to compare the security level of different sorts of PUFs.

- Security of noisy SRAM PUF and RO PUF implemented on real chips and Field Programmable Gate Arrays (FPGAs) are compared with a stable LEDPUF based on the new model.
- Guesswork is derived for two new problems: Guesswork under probability of attack failure; and the guesswork of strong PUFs under model building attacks.

The rest of the paper is organized as follows: Section II presents the motivation and importance of stable randomness through LED. In Section III and Section IV, we summarize the local randomness extraction from DSA and the structure of the stable signal unit (SSU) proposed in [1]. The unified guesswork framework is presented in Section V. In Section VI, the evaluation results of weak LEDPUF and noisy weak PUFs are presented along with a derivation of the guesswork under probability of attack failure. The construction of strong LEDPUF, as well as evaluation of its security are presented in Section VII. Finally, we conclude the paper in Section VIII.

II. THE LEDPUF

When constructing a PUF, stability and sufficient amount of randomness are two primary requirements. Exploiting permanent defects instead of parametric variations can provide stability, but at the same time, these defects are detrimental to other parts of the chip. In addition, it is difficult for a PUF to extract randomness under the condition where process variations are minimized during fabrication. To address such limitations, we propose to exploit locally enhanced permanent random defects to construct stability-guaranteed LEDPUFs.

LEDPUF is stability-guaranteed because its response is extracted through locally manipulated physical layout designs, where permanent defects are intentionally introduced as stable sources of local randomness. To be more specific, LEDPUF leverages intrinsic local randomness in fabrication processes to generate permanent hard defects only at specific areas without 1) affecting other parts of the design and 2) being limited by minimized variations during the fabrication process.

Once a permanent hard defect is formed, the effects of hard defects is then transformed to digital signals, therefore a LEDPUF is not a parametric PUF and it is stability-guaranteed. The Boolean nature of the response without any parametric dependence means that LEDPUF is not only immune to measurement noise and wearout, but also offers a greater level of reliability compared to existing PUFs as the output is resistant to changes in the environmental factors.

Several techniques have been proposed to enhance local variations. In [24] the authors propose to apply PUF-aware Optical Proximity Correction (OPC) at the PUF circuitry to enhance the variability, and [25, 26] propose to extract randomness by utilizing forbidden pitches and poly-Si with random grain boundaries at the PUF associated regions. However, these PUFs are unstable because they are still parametric delay-based PUFs, and they may suffer from systematic variations, for example, between two PUFs printed by the same mask. On the other hand, a LEDPUF is completely stable and

the formations of hard defects are independent to each other. An example of LEDPUF is explained in details in Section III.

The strengths of a LEDPUF over existing PUFs can be summarized as follows:

1) A LEDPUF is stability-guaranteed because its randomness is extracted from permanent hard local defects.
2) A LEDPUF is more secure against side channel attacks because its randomness is extracted efficiently from locally enhanced defectivity with less systematic effects from fabrication process or other parts of the chip.

III. STABLE RANDOMNESS EXTRACTION THROUGH DIRECTED SELF ASSEMBLY

Directed Self Assembly (DSA) is a mechanism that describes block copolymers (BCP) composed of immiscible blocks phase-separate into certain structures [27]. The guiding templates, which are used to *guide* the self-assembly process, can be lithographically-printed trenches (Graphoepitaxy) or chemically-treated surfaces (Chemoepitaxy). During the graphoepitaxy process for contact or via holes, the guiding templates are first lithographically printed, then the surface is spin-coated with the BCP solution. The phase separation occurs during the thermal annealing , and with a particular BCP and surface treatment of substrate [28], cylinders are formed of one block in a matrix of the other block [29].

In case of a diblock copolymer made of two blocks, say A and B: at equilibrium, the microphase separation is established by an energy balance between the stretching energy for the polymer chains and the energy of interactions at the interface between A and B microdomains [30]. Thermal equilibrium is achieved when the free energy is minimized, and the minimum energy state strongly depends on the level of confinement achieved by the layout of guiding templates. In other words, the size, shape, and critical dimension (CD) of the guiding template can greatly affect the DSA defect density [31, 32].

For bigger-sized templates, it becomes energetically less expensive to induce a defect than to achieve a defect-free energy minimization [32–34]. Also, with less confinement forces from the guiding template due to its large size, random interactions from thermal fluctuation [35] or initial kinetics of collective density and state fluctuations [36] begin to dominate the assembly process. Therefore, final assembly results can be random by designing guiding templates that are large enough to cause random assembly errors even if there are no lithographic variations.

Figure 2 shows conceptual energy distributions and simulation results of two guiding templates with an existing DSA simulator [37], where the model of the PS-b-PMMA copolymer has been validated in [38]. In Figure 2 (a), for a proper guiding template shape, only one minimum energy state exists, therefore the final DSA structure is predictable and stable. The three layers inside the polygonal guiding template are the top, middle, and bottom layers of a via. If a cylindrical via hole is formed correctly, the three layers should be three overlapped concentric circles. However, for a large guiding template as shown in Figure 2 (b), many DSA structures have similar minimum energy states, therefore the results of the simulations using the same guiding template are different, where

random arrangement with different orientations begin to occur. In other words, the randomness of DSA is confined within predetermined local areas only by deliberately designing "bad" guiding templates.
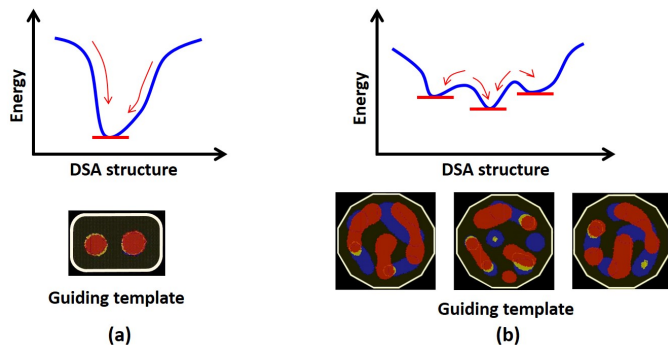

Fig. 2. (a) The guiding template (yellow shape) guides the DSA process to form the only on DSA structure with a clear minimum energy state. (b) The guiding template (yellow shape) does not guide the DSA process to form a specific structure because many DSA structures have similar minimum energy states. The final DSA structure is random and unpredictable.

A. Hard Defective Connection Formation

We leverage the randomness extracted from DSA to form randomly assembled connections, and these connections are then used to fabricate LEDPUF. Though in conventional DSA, the goal of the guiding template design is to achieve high confinement and avoid regions of random phase transitions, we use the same theory but to enhance randomness in assembly. To construct a DSA random hard defective connection, we configure the size of the guiding template so that two vias are formed with a certain probability that they are connected permanently. A DSA hard defective connection is composed of the two vias along with the connection.

In our experiment, each simulation contains three guiding templates with a same shape, and two vias are formed in each of the guiding template. If the via pair in a same guiding template is merged, the DSA hard defective connection is in closed state; otherwise, the connection is in opened state. In our statistical analysis, an open state is represented by a logic "1", and a closed state is represented by a logic "0". 500 simulation were performed in our experiments, so 1500 bits of raw data is obtained from the simulation. Based on our simulated data, the empirical entropy of triples of bits is only 0.0063 bits smaller than the entropy of independent triple of bits, which implies that the states of the three connections in a simulation is independent with each other as expected in real DSA process [38]. Examples of a simulation result in 2D and 3D views are shown in Figure 3 (a) and (b), respectively. In the 2D view, the rectangular shapes with rounding corners are the guiding templates, and the shapes inside of the guiding templates are the vias. If the via pair in a same guiding template is merged, the DSA hard defective connection is formed as shown in Figure 3 (c), and it is in permanent closed state; otherwise, the DSA hard defective connection is in permanent opened state as shown in Figure 3 (d). In Figure

3 (a) and (b), two hard defective connections are in opened state, and one connection is in closed state.
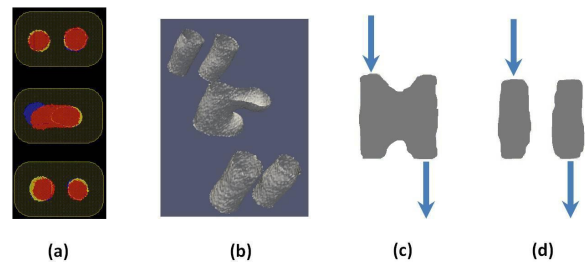

Fig. 3. (a) 2D view of 3 DSA hard defective connections. (b) 3D view of 3 DSA hard defective connections. (c) Vias are partially merged, so the connection is in permanent closed state. (d) Vias are not merged, so the DSA hard defective connection is in permanent opened state.

IV. WEAK LEDPUF CONSTRUCTION

The weak LEDPUF [1] is composed of arrays of SSUs. Each SSU is constructed from a DSA defective connection, which can be considered as random switches with permanent states that determine the unique and stable function of the circuit. Figure 4 (a) shows the implementation of a SSU. Two ends of the DSA connection are connected to VDD and GND through opposite switches. Figure 4 (b) shows the abstraction of a SSU. In standby mode or before the evaluation, the evaluation signal EVA is low and the output is zero. During evaluation mode, EVA becomes high, and the output is either one or zero depending on the permanent state of the DSA connection. If the DSA connection is closed, the output is one; otherwise, the output is zero.
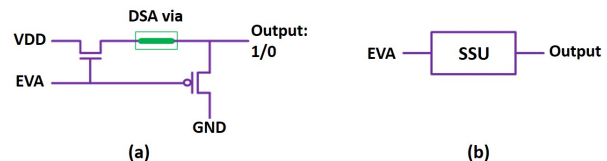

Fig. 4. (a) Stable signal unit implementation. When EVA is high, the output is either one or zero permanently depending on the state of the DSA via. (b) Abstraction of a SSU.

Figure 5 illustrates a weak LEDPUF with n rows and m columns, where the number of SSUs is nm, and the number of CRPs is n. Since only one of the rows is being evaluated at a time, a one-hot decoder is used so that only one bit of the EVA vector is logic 1. The challenge fed into the decoder is a $log(n)$-bit input, and the response is a m-bit output.

Compared with existing weak SRAM PUFs, the weak LEDPUF has several evident advantages:

- It is completely stable, so it has no area or latency overhead. To generate a bit response, the weak LEDPUF requires only one SSU and a transistor, or 3 transistors equivalently, as for a standard SRAM cell, 6 transistors are required. Once the state of the DSA via is determined, the output is fixed permanently, so no additional ECC, fuzzy extraction, or helper data is needed. As stated in [39], for a SRAM PUF to generate a 128-bit response,
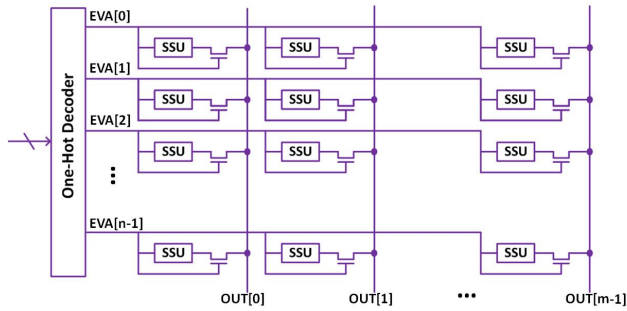
Fig. 5. A weak LEDPUF with n challenges and m-bit response. Only one bit of the EVA vector is logic 1 at a time.

more than 4k SRAM cells are needed under a condition with 15% bit error probability. Therefore, the total number of transistors needed for SRAM PUF to generate a 128-bit response would be 24k, where for the weak LEDPUF, only 384 transistors are needed, which is more than 600x less than a SRAM PUF, thus the area is also much smaller even assuming that the hardware cost of the peripheral circuits are similar.

- In addition to model building attacks [40], the weak LED-PUF is also more resistant to existing attacks to SRAM PUFs, such as laser stimulation [41] or Photonic Emission Analysis (PEA) [42]. The laser stimulation attack focuses on retrieving the on/off state of transistors, but for weak LEDPUF, the states of the transistors, which depend on the EVA signal, do not reveal secret information. The PEA attack detects the photon emission, which only occurs in saturation region for high drain-source voltages. It can be effective to SRAM PUF because the loading of the addressed SRAM cell can lead to a strong voltage degradation of SRAM transistors and increase the time spent in saturation region. However, PEA attack does not work effectively for the LEDPUF because for each SSU, the source voltage (VDD) of the NMOS is always higher than the drain voltage, and the PMOS at the output will not stay in saturation region after evaluation (EVA goes from high to low) since the output will be pulled down regardless of the DSA connection.

V. GUESSWORK AS A UNIFIED FRAMEWORK FOR EVALUATING THE SECURITY LEVEL

A. Why Consider Guesswork?

Consider the following game: Bob draws a sample x from a random variable X, and an attacker Alice who does not know x but knows the probability mass function $P_X(\cdot)$, tries to guess it. An oracle tells Alice whether her guess is right or wrong. This is the situation where an attacker tries different passwords to access an account.

If Alice has only one guess, then the optimal strategy that maximizes her chance of guessing x successfully is choosing the most probable x. In this case the chance of guessing x is $\max_{x' \in X} P_X\left(x'\right)$ and the *predictability* of X is given by its

min-entropy [43]:

$$H_{\infty}\left(P_X\right) = -\log_2\left(\max_{x' \in X} P_X\left(x'\right)\right). \quad (1)$$

If Alice is allowed to make as many guesses as required until she finds x, then the optimal strategy is guessing elements in X based on their probabilities in ascending order [44]. It has been shown that the average number of guesses it takes Alice to find x (denoted by $G(X)$ and termed guesswork) is not given by the traditional Shannon entropy [44]. For example, when drawing a random vector of length m, $\underline{X}$, which is independent and identically distributed (i.i.d.) with distribution $P = [p_1, \ldots, p_L]$, the exponential growth rate of the guesswork scales according to the Renyi entropy $H_\alpha(X)$ with parameter $\alpha = 1/2$ [44]:

$$\lim_{m \to \infty} \frac{1}{m} \log_2\left(E\left(G\left(X\right)\right)\right) = H_{1/2}\left(P\right) = 2 \cdot \log_2\left(\sum_i p_i^{1/2}\right) \quad (2)$$

where $H_{\frac{1}{2}}\left(P\right) \geq H\left(P\right)$ with equality only for the uniform probability mass function (in the context of Figure 5 m is the length of the PUF response and $L = 2^m$).

The security of a PUF is predicated by the inherent random signature in the hardware. An attacker wants to either impersonate a hardware by guessing its random signature, or to learn it by eavesdropping.[3] In order to impersonate the hardware, the attacker needs to respond to a challenge with a correct response. In order to evaluate the security of a PUF we connect to the framework for password security [23]. For a dictionary attack, a guessing framework quantifies security through the number of guesses the impersonator has to make in order to identify the password (or inherent randomness) and therefore respond correctly to all possible challenges. Therefore, we quantify the security level of a PUF through the number of guesses required to break it.

In subsection V-C we show that guesswork can serve as a unified framework for evaluating and quantifying the security of PUFs. Essentially, other measures of evaluating the security level such as min-entropy and mutual information are special cases of guesswork (as shown in Subsection V-C min-entropy is the probability of correctly guessing the PUF response in a single guess, that is, the probability that the guesswork is equal to 1, and so it does not capture the entire probability mass function of the number of guesses; in terms of guessing, guesswork is a more general security criterion than min-entropy). Furthermore, guesswork allows to quantify the security level under more elaborate scenarios such as the security level when key stretching mechanism is used [45] as well as when allowing an attack failure probability (this problem is presented in Subsection V-C).

Characterizing the security of a PUF through guesswork reveals an interesting interplay between the bias of a PUF response, and the noise (due to instability) which is incorporated in each sample. Guesswork is very sensitive to the presence of instability, but yet is not very susceptible to bias. These properties are discussed in subsection V-E.

---

[3] In this attacker model the adversary does not have access to the PUF, but rather is trying to impersonate it.

Therefore, guesswork highlights the advantage of stable PUFs over unstable PUFs, when evaluated through the number of guesses required to break a PUF.

Moreover, we present a formal evaluation methodology for PUFs security, while identifying the impact of bias, noise and side-channels.

Note that the interplay between noise and bias, as well as the advantage of stable PUFs have been reported in the literature. For example, [46] discusses the advantage of stable PUFs, and [47] considers the interplay between noise and the PUF response in terms of efficient post-processing methods. However, our results provide methods to analytically evaluate the security level of PUFs, based on their fundamental properties (i.e., noise level and bias).

B. Background

The guesswork $G(X)$ is a random variable that represents the number of guesses required to guess a random variable x. Therefore, the probability of having $G(x)$ guesses is $P_X(x)$. The ρth moment of guesswork is

$$E\left(G\left(X\right)^\rho\right) = \sum_x G\left(x\right)^\rho \cdot p_X\left(x\right). \quad (3)$$

The definition of guesswork can be extended to the case where the attacker has a side information Y available. In this case the average guesswork for $Y = y$ is defined as $G(X|Y=y)$, and the ρth moment of $G(X|Y)$ is

$$E\left(G\left(X|Y\right)^\rho\right) = \sum_y E\left(G\left(X|Y=y\right)^\rho\right) \cdot p_Y\left(y\right). \quad (4)$$

Massey [48] noted that a dictionary attack minimizes the expected number of guesses (i.e., guessing the values in the decreasing order of $P_X(x)$). Arikan [44] has bounded the ρth moment of the optimal guesswork, $G^*(X|Y)$, by

$$\left(1 + \ln\left(M\right)\right)^{-\rho} \sum_y \left(\sum_x P_{X,Y}\left(x,y\right)^{\frac{1}{1+\rho}}\right)^{1+\rho} \leq$$

$$E\left(G^*\left(X|Y\right)^\rho\right) \leq \sum_y \left(\sum_x P_{X,Y}\left(x,y\right)^{\frac{1}{1+\rho}}\right)^{1+\rho} \quad (5)$$

where $M = |X|$ is the cardinality of X. Furthermore, in [44] it has been shown that when X and Y are strings of length m, where the pairs (X_i, Y_i) are drawn i.i.d. and $1 \leq i \leq m$, the exponential growth rate of the optimal guesswork is

$$\lim_{m\to\infty} \frac{1}{m} \log_2\left(E\left(G^*\left(X|Y\right)^\rho\right)\right) = \rho \cdot H_{\frac{1}{1+\rho}}\left(P_{X,Y}\left(x,y\right)\right) \quad (6)$$

where m is the size of X and Y, and

$$H_{\frac{1}{1+\rho}}\left(P_{X,Y}\left(x,y\right)\right) = \frac{1}{\rho} \log_2\left(\sum_y \left(\sum_x P\left(x,y\right)^{\frac{1}{1+\rho}}\right)^{1+\rho}\right) \quad (7)$$

is Renyi's conditional entropy of order $\frac{1}{1+\rho}$ [44].

Two remarks are in order regarding why considering the growth rate is meaningful.

Remark 1 (The non-asymptotic behavior is also dictated by growth rate). *Note that although (6) is an asymptotic result,*

TABLE I
THE AVERAGE GUESSWORK AS A FUNCTION OF m WHEN $p = 1/2$.

$\rho = 1$, $p = 1/2$	m=64	m=128	m=256	m=1024
Lower bound	$1.4 \cdot 2^{58}$	$1.4 \cdot 2^{121}$	$1.4 \cdot 2^{248}$	$1.4 \cdot 2^{1014}$
Upper bound	2^{64}	2^{128}	2^{256}	2^{1024}

it converges very quickly. This is because in [44] it was shown that the guesswork of any moment is lower bounded by $\left(1 + m \cdot \ln\left(2\right)\right)^{-\rho} \cdot 2^{m \cdot \rho \cdot H_{1/(1+\rho)}(p)}$ and upper bounded by $2^{m \cdot \rho \cdot H_{1/(1+\rho)}(p)}$, when X is of size m and is drawn i.i.d. Bernoulli(p). Table I presents the lower and upper bounds for various values of m.

Remark 2 (the operational meaning of growth rate). *Based on the bounds presented in Table I it can be shown that even for finite values of m, a decrease in growth rate can have a tremendous impact on the security level. For example, when $p = 0.0015$, which leads to $H_{1/2}\left(0.0015\right) = 0.1$, the average guesswork for $m = 256$ is lower bounded by $1.4 \times 2^{17.6}$ and upper bounded by $2^{25.6}$ which is far smaller than 2^{256} (or a lower bound of 1.4×2^{248} as presented in Table I) in the case when $p = 1/2$.*

Furthermore, growth and decrease rates are commonly used when evaluating the security level. For instance the min-entropy is a measure of the decrease-rate of the probability of guessing a password.

Another extension of guesswork [49] considers a game in which it is sufficient to guess x up to a certain level of distortion D, according to some distance metric $d(x, \hat{x}(i)) = \sum_{i=1}^m d(x_i, \hat{x}(i))$. Essentially, when $G(x) = i$, the word $\hat{x}(i)$ is guessed such that $d(x, \hat{x}(i)) \leq m \cdot D$, that is, when the attacker guesses a word which is within a Hamming distance $m \cdot D$ of x the game is over. The authors in [49] have solved this problem for the general case; more specifically, for a binary source which is drawn i.i.d. Bernoulli(p) and Hamming distortion

$$d\left(x_j, \hat{x}_j\right) = \begin{cases} 1 & x_j = \hat{x}_j \\ 0 & x_j \neq \hat{x}_j \end{cases} \quad (8)$$

where $1 \leq j \leq m$, they have shown that the exponential growth rate of the guesswork equals

$$\lim_{m\to\infty} \frac{1}{m} \log_2\left(E\left(G^*\left(X, D\right)^\rho\right)\right) = \rho \cdot E\left(D, p\right) =$$

$$\max\left(\rho H_{\frac{1}{1+\rho}}\left(p\right) - \rho \cdot H\left(D\right), 0\right) \quad (9)$$

where $H(D) = -D \log_2(D) - (1-D) \log_2(1-D)$ is the binary Shannon entropy [50].

Guesswork has been analyzed in many other scenarios such as guessing under source uncertainty with and without side information [51], [52], using guesswork to lower bound the complexity of sequential decoding [44], guesswork for Markov chains [53], and guesswork for multi-user systems [54].

C. Extending Guesswork

In this subsection we extend the definition of guesswork and show that it can serve as a unified framework for evaluating the security level of PUFs by incorporating noise and bias.

In addition, we relate guesswork to other measures such as mutual information and min-entropy.

We begin by finding the moments of the guesswork of a noisy weak PUF.

Theorem 1. *When the response of a weak PUF is noisy such that the noise is additive and drawn i.i.d. Bernoulli(D), and the original response is i.i.d. Bernoulli(p) , the ρth moment of the guesswork increases at rate $\rho \cdot E(D, p)$ as defined in (9).*

Proof. The idea behind the proof is that guessing within Hamming distance $m \cdot D$ of the original response, enables the attacker to find the original response by using the helper-data.

Essentially there are two options. The first possibility is to construct a code to guess the original response up to Hamming distance $m \cdot D$ as is done in [49], and then use the helper-data in order to find the original response, in which case the rate of the ρth moment is $\rho \cdot E(D, p)$. The second possibility is to use the helper data (e.g., the coset of an ECC) to guess over a subset. In this case, since the helper-data breaks up the space into subspaces of size $2^{(1-H(D)) \cdot m}$ [50], guessing through the subspace can only bring the rate of the ρth moment down to $\rho \cdot E(D, p)$. Therefore, the minimal rate is $\rho \cdot E(D, \rho)$. □

Remark 3. *$\rho \cdot E(D, p)$ is the maximum rate at which the ρth moment of the guesswork of a noisy PUF can increase. This can be achieved by employing an ECC that operates very close to the channel capacity under the statistical profile of the noise [50]. However, the hardware size required to implement this ECC may be large, and so PUF designers may resort to other ECCs that can be implemented more efficiently in terms of their hardware size but yet cannot achieve the channel capacity. In this case, the decrease in rate is expressed in the amount of information revealed by the helper data, W, whose entropy is larger than the noise entropy, that is, $H(W) \geq H(D)$. This in turn brings the guesswork down to $\max\left(\rho H_{\frac{1}{1+\rho}}(p) - \rho \cdot H(W), 0\right) \geq \rho \cdot E(D, p)$.*

Remark 4. *A meaningful way to compare the efficiency of PUFs is by fixing their security level as well as the probability of error of the ECC to certain values and then compare the hardware size required to achieve these by different PUFs, as presented in [46]. In [46] the security level is evaluated through the Shannon entropy. However, Guesswork can also be used in the method presented in [46] to evaluate the security level. This in turn provides a measure that has a wider operational meaning in terms of security than the Shannon entropy. For example, the average guesswork is directly related to the average number of guesses required to guess the PUF response (this is highlighted in Theorem 2 and Remark 8).*

Remark 5. *In [47] and [55] the noise of a PUF is not distributed Bernoulli(D), but rather is asymmetric and affected by the bias level such that the conditional transition probability is different when the PUF response is equal to 1 and 0, and the combined transition probability is D (see subsection V-E for more details). In this asymmetric case the rate at which the ρth moment of the maximum guesswork increases is lower bounded by $\rho \cdot E(D, p)$ due to the convexity of of $E(D, p)$ in the noise distribution.*

Before we present a new game that extends the traditional definition of guesswork, let us define the type of a vector.

Definition 1. *Consider a binary vector x of size m and assume that $N(x|1)$ is the number of elements of this vector that are equal to 1. In this case when $N(x|1)/m = q$ the vector x is of type q.*

We now define a new guessing game that captures different measures for evaluating the security level.

Definition 2 (Guesswork under attack failure constraint)**.** *Consider the following game: Bob draws a vector x of size m i.i.d. Bernoulli(p). The attacker Alice has to guess x up to Hamming distance $m \cdot D$ as defined in subsection V-B, under the constraint that the probability of attack failure is smaller than or equal to $2^{-\alpha \cdot m}$ where $\alpha \geq 0$, that is, Alice may decrease the number of guesses by guessing only a subset of all possible words, which leads in turn to a certain probability of attack failure. We define the optimal guesswork for this game as $G^*(X; D, \alpha)$. Furthermore, we define the guesswork in the case where the probability of attack failure is zero as $G^*(X; D, \infty) = G^*(X; D)$.*

Remark 6. *The relation between $G^*(X; D, \alpha)$ and previous works is as follows:*

- *$Pr(G^*(X; 0, \infty) = 1) = 2^{-m \cdot H_\infty(p)}$, that is, the min-entropy.*
- *$\lim_{m \to \infty} \frac{1}{m} \log(E(G^*(X; D, \infty)^\rho)) = \rho \cdot E(D, p)$ as defined in (9).*

The following theorem characterizes a lower bound for $G^*(X; D, \alpha)$ for any moment $\rho > 0$ in the case where the attacker is allowed not to guess certain types.

Theorem 2.

$$\lim_{m \to \infty} \frac{1}{m} \log\left(E\left(G^*\left(X; D, \alpha = D(s\|p)\right)^\rho\right)\right)$$
$$\leq \begin{cases} \rho \cdot \left(H_{\frac{1}{1+\rho}}(p) - H(D)\right) & s^* \leq s \leq 1 \\ \rho \cdot (H(s) - H(D)) - D(s\|p) & p < s \leq s^* \end{cases}$$

$0 \leq D \leq p \leq 1/2$, and

$$\lim_{m \to \infty} \frac{1}{m} \log\left(E\left(G^*\left(X; D, \infty\right)^\rho\right)\right)$$
$$= \rho \cdot \left(H_{\frac{1}{1+\rho}}(p) - H(D)\right)$$

where $s^* = \frac{p^{(1+\rho)^{-1}}}{p^{(1+\rho)^{-1}} + (1-p)^{(1+\rho)^{-1}}}$, the probability of attack failure decreases like $2^{-m \cdot \alpha}$,

$$D(s\|p) = s \cdot \log_2(s/p) + (1-s) \cdot \log_2((1-s)/(1-p))$$

is the Kullback-Leibler divergence [50], and Alice chooses a set $A = \{q_1, \ldots, q_L\}$ of types whose vectors are not guessed, such that the probability that $N(x|1)/m \in A$ is smaller than or equal to $2^{-\alpha \cdot m}$, that is, Alice guesses words in A^C.

Proof. The proof is in Appendix A □

Note that $\rho \cdot H(s^*) - D(s^*\|p) = \rho \cdot H_{\frac{1}{1+\rho}}(p)$

The next three remarks point out a few properties of $G^*(X; D, \alpha)$.

Remark 7. *When an attacker attempts to break a very large number of independent PUF responses (or passwords), where the probability of attack failure is $2^{-m \cdot \alpha}$, he is very likely to break a fraction of $1 - 2^{-m \cdot \alpha}$ of the PUF responses (passwords), and this in turn leads to ρth moment of guesswork across PUF responses (passwords) that increases at a rate $\lim_{m \to \infty} \frac{1}{m} \log \left(E \left(G^* \left(X; D, \alpha \right)^\rho \right) \right) \leq \rho \cdot \left(H_{\frac{1}{1+\rho}} \left(p \right) - H \left(D \right) \right) = \rho \cdot E \left(D, p \right)$, when $D \leq p$, that is, the moments of guesswork decrease as the probability of attack failure increases.*

Remark 8. *When $s = p + \epsilon$ the average guesswork is approximately $H \left(p \right) - H \left(D \right)$, which is the rate distortion function of Hamming distortion [50].*

Remark 9. *Note that when $p = 1/2$ also $s^* = 1/2$ and the upper bound of Theorem 2 is equal to $\lim_{m \to \infty} \frac{1}{m} \log \left(E \left(G^* \left(X; D, \infty \right)^\rho \right) \right)$, that is, when guessing according to the method presented in Theorem 2 Alice does not gain anything from having a failure probability larger than zero.*

We now derive an expression for the min-entropy when the attacker has to guess a word that is within Hamming distance $m \cdot D$ of the password. In this case the min-entropy of a binary i.i.d. source subject to Hamming distortion D is equivalent to choosing a word which is in a ball of radius $m \cdot D$ around the most likely word (i.e., the probability of guessing a word which is in the most likely ball). The asymptotic value of the min-entropy subject to distortion D is given by the following lemma.

Lemma 1. *Consider a binary word of length m for which each element is drawn i.i.d. from Bernoulli(p). The min-entropy subject to Hamming distortion D converges to*

$$- \lim_{m \to \infty} \frac{1}{m} \log_2 \left(P_X^{ball} \right) = \begin{cases} D \left(D \| p \right) & 0 \leq D \leq p \\ 0 & p < D \leq 1 \end{cases} \quad (10)$$

where $p \leq 1/2$, $P_X^{ball} = \sum_{i=0}^{m \cdot D} \binom{m}{i} p^i \left(1 - p \right)^{m-i}$.

Proof. The proof is in Appendix B. □

Remark 10. *Note that the result of Lemma 1 can also be interpreted as $Pr \left(G^* \left(X; 0, \infty \right) \leq 2^{m \cdot H(D)} \right)$ when a password of length m is drawn i.i.d. Bernoulli(p), where $0 \leq D \leq 1/2$.*

D. Examples for Quantifying the Security of PUFs

In this subsection we present a few examples that illustrate how to use guesswork in order to quantify the security level of PUFs. We address evaluations for unstable PUFs as well as for stable PUFs. We incorporate into the expressions noise, bias, and side information coming from other PUFs or from side channel/model attacks.

The first step in calculating the guesswork of a PUF is evaluating the probability function according to which it is drawn, as well as the noise level. In this subsection we assume that the bits are i.i.d. for which case the first step is evaluating the bias of the stable bits and then estimating the noise level of the unstable bits; evaluating the bias of the stable bits enables

us to state that the PUF response is drawn i.i.d. from the probability function

$$P_0 = p \quad P_1 = 1 - p \quad (11)$$

whereas the probability of transition of a bit when re-sampling a PUF is q, such that

$$x^{(2)} = x^{(1)} \oplus e \quad (12)$$

where $x^{(1)}$, $x^{(2)}$ are the first and second samples of the unstable (noisy) PUF, and $Pr \left(e_j = 1 \right) = q$, $1 \leq j \leq m$.

For stable PUFs such as the one presented in this paper, it is sufficient to calculate the bias and assign the probability function to $\rho \cdot H_{\frac{1}{1+\rho}} \left(P \right)$ in equation (6), in order to get the ρth moment of guesswork. For example, when the stable PUF is drawn i.i.d. according to Bernoulli(0.47) the average guesswork of a PUF of large enough size ($m = 256$, say) is proportional to

$$2^{H_{1/2}(0.47) \cdot m} = 2^{0.9987 \cdot m} \quad (13)$$

whereas the largest guesswork that we can expect for is achieved by an unbiased PUF for which each bit is drawn i.i.d. Bernoulli(0.5), and is proportional to

$$2^{H_{1/2}(0.5) \cdot m} = 2^m. \quad (14)$$

For unstable PUFs, re-sampling the PUF yields a noisy version of the original response as presented in equation (12). When the probability of transition is q, Theorem 1 shows us that it is sufficient to guess the original response $x^{(1)}$ up to Hamming distance $m \cdot q$. The intra distance can be used to evaluate the noise level. For example, when considering an unbiased unstable PUF with a transition probability $q = 0.1$, we get that the guesswork is proportional to

$$2^{m \cdot (1 - H(0.1))} = 2^{0.531 \cdot m} \quad (15)$$

which means that noise decreases the average number of guesses significantly.

The conditional guesswork (4) enables us to quantify the effect of side information on the security level of both stable and unstable PUFs. In order to evaluate the conditional guesswork we first need to characterize the conditional probability. The conditional probability depends on the type of attack which is being carried; in some cases characterizing its effect on the randomness of the response requires some effort. A simple example for a side information attack is one in which an attacker has another PUF which is correlated with the original one. For example, consider an unbiased stable PUF x for which each element is drawn i.i.d. Bernoulli(0.5), and assume that an attacker has another unbiased stable PUF, y, which is correlated with x such that

$$P \left(y | x \right) = P \left(e \right) \quad (16)$$

where e is drawn i.i.d. Bernoulli(0.2). In this case the unconditional guesswork $G \left(X \right)$ is proportional to

$$2^{H_{1/2}(1/2) \cdot m} = 2^m \quad (17)$$

whereas the conditional guesswork $G \left(X | Y \right)$ is proportional to

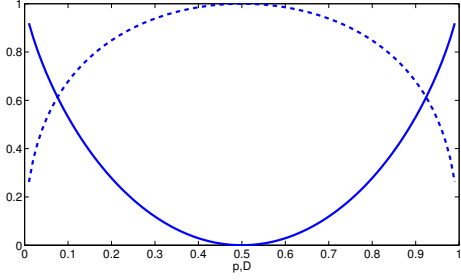$$2^{H_{1/2}(0.2)m} = 2^{0.848 \cdot m} \quad (18)$$

8

Fig. 6. The solid line presents the average guesswork of an unstable unbiased PUF $1 - H(D)$, whereas the dotted line is the average guesswork of a stable biased PUF $H_{1/2}(p)$.

because of the fact that in this case x given y is also drawn i.i.d. Bernoulli(0.2). In general, the correlation between PUFs can be evaluated through the inter distance.

Conditional guesswork subject to distortion enables to evaluate the guesswork of an unstable PUF when side information is available. The method of evaluating the guesswork is similar to the previously mentioned methods for evaluating conditional guesswork and guesswork subject to distortion.

E. The Effect of Noise Vs. The Effect of a Bias

In this subsection we analyze the expressions for guesswork as well as min-entropy subject to distortion, and quantify the impact that noise and bias have on PUFs. Furthermore, we show that the effect of noise is far worse than the effect of bias in terms of average guesswork.

First, let us focus on the effect of noise and bias on the expected value of the guesswork (i.e., the case where $\rho = 1$), when the noise is Bernoulli(D). From Theorem 1 we get that when the transition probability is D, the asymptotic growth rate of the expected value of the guesswork is

$$H_{1/2}(P_X) - H(D). \tag{19}$$

On the other hand the asymptotic growth rate of the expected value of the guesswork of a stable PUF whose bits are drawn i.i.d. from Bernoulli(p) is

$$H_{1/2}(p) = 2 \cdot \log_2 \left(\sqrt{p} + \sqrt{1-p} \right). \tag{20}$$

The first derivative of equation (19) equals

$$\log_2(D) - \log_2(1 - D) \tag{21}$$

which diverges as D approaches 0. Therefore, even when the noise level (and D) is very small, it decreases the expected value significantly. On the other hand the first derivative of (20) is equal to zero at $p = 1/2$ (i.e., when there is no bias). The first derivative around $p = 1/2$ is very small and therefore bias does not affect the guesswork as much as noise. Figure 6 presents the guesswork of an unstable unbiased PUF and the guesswork of a stable biased PUF.

For example, the asymptotic exponential growth rate of the guesswork of an unbiased ($p = 1/2$) unstable PUF with transition probability $D = 0.1$ (i.e., a 10% noise) is equal

THE AVERAGE GUESSWORK WHEN THE TRANSITION PROBABILITY OF THE NOISE HAS STATES VERSUS THE CASE WHEN THE NOISE IS BERNOULI(D) (IN PARENTHESES)

p/D	$D = 0.1$	$D = 0.2$
$p = 0.49$	0.53 (0.53)	0.2781 (0.2779)
$p = 0.4$	0.5197 (0.516)	0.2707 (0.2634)

to 0.53 which is the guesswork of a stable biased PUF with $p = 0.05$ (i.e., a 95% bias).

In terms of min-entropy as presented in Lemma 1, the divergence $D(D||p) = -H(D) - D\log_2(p) - (1-D)\log_2(1-p)$ and therefore its first derivative also diverge as D goes to zero. Therefore, min-entropy is also very sensitive to the presence of noise. On the other hand, the min-entropy of a stable PUF is equal to

$$-m \log_2(1 - p). \tag{22}$$

The first derivative of (22) equals $\frac{m}{1-p}$ when $0 \le p \le 1/2$ and therefore it does not diverge. Hence, the effect of bias on the min-entropy is also less significant than the effect of noise.

For example, the asymptotic min-entropy of an unbiased ($p = 1/2$) unstable PUF with transition probability $D = 0.1$ is equal to $1 - H(0.1) = 0.53$ which is the min-entropy of a stable biased PUF with $p = 0.31$ (i.e., a bias level of 69%).

Note that in general the first derivative of the min-entropy does not equal to zero at $p = 1/2$, and therefore bias has a stronger effect on min-entropy than on average guesswork.

Figure 7 presents the behavior of the min-entropy as a function of p. It shows that it is more sensitive to bias than $H_{1/2}(p)$.

So far we have discussed the case where the noise is Bernoulli(D). However in [47] and [55] it was shown that the conditional probability of the noise can be as follows

$$P(e = 1|X = 0) = \frac{D}{2 \cdot (1 - p)}$$

and

$$P(e = 1|X = 1) = \frac{D}{2p}$$

which satisfies both $P(e = 1) = D$ and $P(X = 1) = p$. In this case as stated in Remark 5 $\rho \cdot E(D, p)$ is a lower bound on the rate at which the ρth moment of the guesswork increases, due to the concavity of the Shannon entropy. However, the actual rate at which the ρth moment increases can be related to the amount of information revealed by the helper data, that is, $H(D|X)$ such that the rate at which the average guesswork increases is equal to $H_{1/2}(P) - H(D|X) \ge H_{1/2}(P) - H(D)$. Table II presents $H_{1/2}(P) - H(D|X)$ for various values of p and D versus $H_{1/2}(P) - H(D)$; these results show that the behavior in both cases is very similar and that the lower bound is very tight.

VI. EVALUATING THE SECURITY LEVEL OF WEAK PUFS THROUGH GUESSWORK

A. Evaluation of Weak LEDPUF

We evaluate the probability mass function of a bit generated by a weak LEDPUF based on simulation results for the
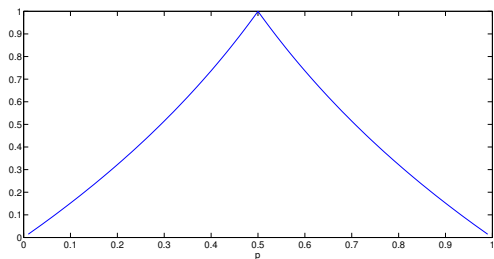
9

Fig. 7. The min-entropy as a function of p. Note that the first derivative is always larger than zero.

TABLE III
NOISY PUF MEASUREMENTS. ALL NUMBERS ARE PERCENTAGES.

	intra-FHD	inter-FHD	Stability	Bias Level	
				One	Zero
SRAM PUF	2.26	48.33	93.42	49.13	50.87
RO PUF	2.48	47.13	91.19	51.38	48.62

formation of connections

$$p_X(1) = 0.4626 \quad p_X(0) = 0.5374. \tag{23}$$

The uniqueness is evaluated by calculating the fractional inter-distance [7] of 1000 weak LEDPUFs, each producing 512 bits of response. The distribution is with mean=0.503 and standard deviation=0.02. Since the variance value is proportional to the inverse of the length of the response, as the length of the response increases the variance value goes to zero while the mean value goes to 0.505.

B. Measurements of Noisy Weak PUFs

Two noisy silicon PUFs: SRAM PUF and RO PUF, are measured at 20°C in our experiments. For the SRAM PUF, responses from 10 commercial 45nm SOI test chips with 176k byte of SRAM cells each are obtained. Every SRAM PUF is measured 10 times. The RO PUF is implemented on 15 Altera DE2-115 FPGA boards. To avoid correlated CRPs, 90 CRPs are generated from the 91 ROs in each RO PUF. Each RO PUF is measured 10 times. No error correcting techniques are applied on these PUFs.

The measurement results of noisy PUFs are summarized in Table III. The intra-FHD and inter-FHD are given in the second and third columns, respectively. Both PUFs show good results of small intra-FHD and close to 50% inter-FHD. The stability shown in the forth column gives the percentage of stable bits through all 10 measurements, where a stable bit is a bit that remains the same during all measurements. A 93% stability for the SRAM PUF, for example, means that 7% of the bits flip at least once during the 10 measurements. For LEDPUF, the intra-FHD is 0% and the stability is 100%. The bias level (percentages of ones and zeros) are given in the last two columns.

For the RO PUF, in addition to the intra-FHD at 20°C, we also compare the intra-FHD between 20°C and 60°C, which is the reliability of the PUF if it is enrolled at 20°C but verified at 60°C. The results are presented in Figure 8. We can see that for

TABLE IV
GROWTH RATE OF THE EXPECTED VALUE OF THE GUESSWORK. WHEN THE KEY SIZE OF THE PUF IS 32, THE AVERAGE GUESSWORK OF THE SRAM PUF IS PROPORTIONAL TO $2^{32 \times 0.8442}$, AND THE AVERAGE GUESSWORK OF THE LEDPUF IS PROPORTIONAL TO $2^{32 \times 0.9980}$.

PUF Type	SRAM	RO at 20°C	RO at 60°C	LEDPUF
Growth rate	0.8442	0.8323	0.4706	0.9980

most PUFs, the averaged intra-FHD at the extreme temperature is about 12%, which implies that conventional ECC margin with error reduction techniques for the PUF would be required.
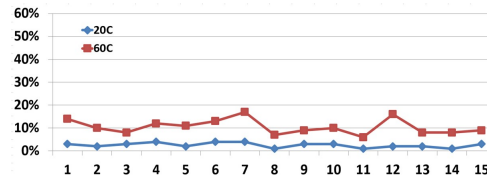

Fig. 8. Intra-FHD at extreme temperature variation for 15 RO PUFs.

For noisy SRAM PUF and RO PUF, the expected growth rate is calculated by plugging the intra-FHD to equation (19). The expected growth rate of weak LEDPUF is obtained by applying the bit probabilities given in (23) to equation (20). The results are summarized in Table IV. We can see that even though the weak LEDPUF is more biased than the noisy PUFs, its guesswork growth rate is still higher than noisy PUFs. For RO PUF at 60°C, the guesswork growth rate becomes much worse compared with RO PUF at 20°C, which implies quantitatively how insecure a PUF can become under environmental variations.

To give an estimated area comparison if error correcting techniques are applied, for a SRAM PUF to generate a secure 128-bit response with Equal Error Rate (EER) $< 10^{-9}$, the area required is about $1630\mu m^2$ using 65nm technology [46]. Using the same technology, for the weak LEDPUF to generate a bit, the area of a SSU as shown in Figure 4 is $3.24\mu m^2$. From Table IV we know that for weak LEDPUF to generate a secure 128-bit response, we need $\frac{128}{0.998}$ bits (SSUs), which roughly translates to an area of $415\mu m^2$. This shows that *the area of the SRAM PUF with EER $< 10^{-9}$ is more than 3X larger than the area of a weak LEDPUF*. For the SRAM PUF with ERR $< 10^{-6}$, the area to generate 128 bits is about $604\mu m^2$, which is still about 1.4X larger than the area of a weak LEDPUF.

VII. STRONG LEDPUF CONSTRUCTION AND GUESSWORK ANALYSIS

A. The Construction of A Strong LEDPUF

One of the shortcomings of using memory-based PUFs for CRPs, is the scaling of the hardware size as a function of the number of CRPs [22]. In general, each channel response pair requires a different set of circuits, and as a result the hardware size is proportional to the number of CRPs. On the other hand for strong PUFs the hardware size scales logarithmically as a function of the number of CRPs.

In order to create a strong LEDPUF we consider a keyed hash function along with a weak LEDPUF. The weak LEDPUF
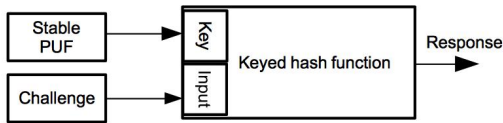
10

Fig. 9. A strong LEDPUF based on a keyed hash function (HMAC or NMAC) and a weak LEDPUF.

response is used as a key for the keyed hash function. The challenges serve as the input to the hash function, whereas the response is the output of the keyed hash function. Figure 9 presents a strong LEFPUF based on a keyed hash function and on a weak LEDPUF with N bits of input and L bits of output.

It is important that the keyed hash function uses the key in such a way that does not enable the attacker to predict responses to unobserved challenges based on the observed ones. Therefore, concatenating the key directly to the challenge, which is vulnerable to extension or collision attacks, is not a good realization of the strong LEDPUF.

We create strong LEDPUF by using a weak LEDPUF as a key for a keyed-hash message authentication code (HMAC) [56]. Any cryptographic hash function, such as SHA-3 can be used for HMAC. To give a rough estimation of the hardware implementation cost of the strong LEDPUF, a lightweight SHA-3 implementation requires about 5k gates [57]. It is worth mentioning that in [58] the authors also propose the use of PUFs with an HMAC in a somewhat similar manner; however, they do not take into consideration the overhead incurred by the instability of parametric PUFs.

The security level of a strong LEDPUF depends on the underlying hash function and the quality of the weak LEDPUF that serves as a key, whereas weak LEDPUFs rely solely on the randomness in the manufacturing process.

For the simulation results, each strong LEDPUF consists of a weak LEDPUF that provides 2x256 bits for the two initial vectors (IV) of the nested hash, and each response is a 256-bit stream because SHA-256 is used in the construction. The same challenge is given to 1000 strong LEDPUFs, and the inter-distance of the responses is a distribution with mean=0.500 and standard deviation=0.03.

To construct a strong LEDPUF, only the weak LEDPUF can be used because of its 0% intra-distance. If other existing weak PUFs with even small intra-distance are used, the intra-distance of the strong LEDPUF would be increased dramatically due to the avalanche effect. In other words, even a single bit flip of the weak PUF can completely change the response of the strong LEDPUF. Figure 10 (a) shows that the intra-distance of the strong LEDPUF jumps from 0% to 50% as the number of bit flips increases from zero to one.

Figure 10 (b) shows how the intra-distance of the strong LEDPUF rises as the intra-distance of the weak PUF increases in logarithmic scale. Since 2x256 bits of the IVs are from the weak PUF, for a weak PUF with 0.1% intra-distance, the probability that it generates a same 512-bit response twice is about 60%, which translates to a roughly 20% intra-distance of the strong LEDPUF. Therefore, only the weak LEDPUF

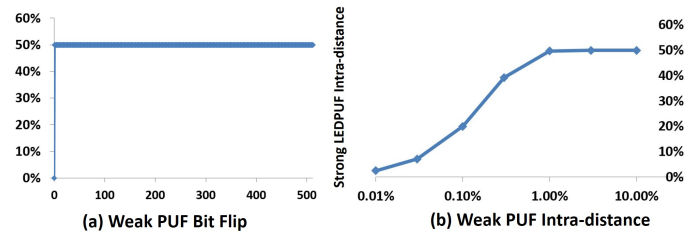with a guaranteed 0% intra-distance can be used for the IV generation.

Fig. 10. (a) A single bit flip from the weak PUF can induce a completely different response of the strong LEDPUF due to the avalanche effect of the hash function. (b) Intra-distance of the strong LEDPUF rises dramatically if other weak PUFs with small intra-distances are used in the strong LEDPUF construction.

B. The Guesswork of any Strong PUF

In this subsection we quantify the security of strong PUFs in terms of the number of guesses required to break them. Our results quantify the number of secure authentications for which any strong PUF is good for. Furthermore, we compare the guesswork of our proposed strong LEDPUF to the guesswork of other strong PUFs that have been introduced in the literature. Finally, to demonstrate the importance of stability of a strong PUF, we show that the guesswork of a stable XOR arbiter PUF is larger than the guesswork of noisy ones, for the same number of observed CRPs.

We begin by defining the following game.

Definition 3. *Consider a strong PUF, which is used by an authentication scheme to authenticate n unique challenges through observing their responses. The authentication problem is defined as follows:*
1) *For each challenge the attacker has to guess with a single response.*
2) *When the attacker does not guess correctly, it can mask itself to receive a new challenge.*
3) *Once the attacker makes a correct guess it is authenticated.*

Remark 11. *Note that when authenticating a strong PUF through CRPs each challenge can be used only once. Furthermore, the problem defined above captures a strict security requirement that the system is compromised once the attacker manages to deceive the verifier.*

Remark 12. *When the attacker fails to guess any response correctly the attack fails. When the number of challenges is large we show that this event decays exponentially fast.*

We now find the average guesswork of the game presented in Definition 3 as well the probability that the number of guesses is smaller than or equal to a certain number.

Theorem 3. *The average guesswork of the authentication problem presented in Definition 3 is*

$$E\left(G\right) = 2^{-H_\infty(1)} + \sum_{i=2}^{n} i \cdot 2^{-H_\infty(i)} \cdot \prod_{k=1}^{i-1} \left(1 - 2^{-H_\infty(k)}\right) \quad (24)$$

11

where $2^{-H_\infty(k)}$ *is the most probable response to the kth challenge either given that the guesses for challenges* $1, \ldots, k-1$ *were incorrect or given the previous* $k-1$ *CRPs (these two scenarios can lead to different min-entropy). Furthermore, the probability that the number of guesses is smaller than or equal to* l *is*

$$Pr\left(G \in \{1, \ldots, l\}\right) =$$
$$2^{-H_\infty(1)} + \sum_{i=2}^{l} 2^{-H_\infty(i)} \prod_{j=1}^{i-1}\left(1 - 2^{-H_\infty(j)}\right)$$

Finally, the probability of attack failure is $\prod_{i=1}^{n}\left(1 - 2^{-H_\infty(i)}\right)$.

Proof. Each response has a certain statistical profile based on the previous CRPs. When the attacker knows this profile the optimal strategy to minimize the number of guesses is to guess the most probable one. This in turn leads to $2^{-H_\infty(i)}$ where $H_\infty(i)$ is the min-entropy given that either the previous $i-1$ guesses were not correct or that the previous $i-1$ CRPs were revealed to the attacker. The results of the theorem follow directly from this argument. □

Corollary 1. *When the statistical profile does not change across challenges we get that*

$$E\left(G\right) = 2^{H_\infty} - \left(1 - 2^{-H_\infty}\right)^n \cdot \left(n + 2^{H_\infty}\right) \quad (25)$$

and

$$Pr\left(G \in \{1, \ldots, l\}\right) = 1 - \left(1 - 2^{-H_\infty}\right)^l. \quad (26)$$

Remark 13. *Note that the average guesswork* (25) *is equal to*

$$E\left(G\right) = 2^{H_\infty} - \epsilon \quad (27)$$

when $n \gg 1$, *where* $\epsilon \ll 1$ *decays exponentially fast.*

Remark 14. *When the attacker does not know the statistical profile of the response (for instance when the structure is too complex for him to infer it), all he can do is to guess a response uniformly, which leads in turn to* $H_\infty = 1$.

Furthermore, in some cases the attacker can infer the statistical profile based on the structure of a strong PUF and a set of CRPs that have been revealed to him (see [59, 60] *for attacks on arbiter PUFs, XOR arbiter PUFs, etc.).*

C. Quantifying the Security of Specific Strong PUFs

In this subsection we quantify the security level of various strong PUFs in terms of their guesswork.

the result in (27) also applies to the case when an attacker can observe multiple CRPs. For example model attacks over strong PUFs [59, 60] enable attackers to accurately guess responses based on previously observed CRPs. In terms of guesswork it means that once an attacker observes a certain set of CRPs, conditioned on the CRPs that have been revealed so far, the most likely conditional probability can be very high.

The average guesswork in (27) allows us to quantify how secure strong LEDPUF is compared to other strong PUFs from the literature that are susceptible to model building attacks.

Strong LEDPUFs are based on HMAC, and so ideally an attacker can not infer anything from observing CRPs in this case. Therefore, when the number of bits at the output of a strong LEDPUF is m, the average guesswork converges to 2^m after observing any amount of CRPs. Even when the key is biased such that each bit is drawn Bernoulli(0.53) [1], the average guesswork when HMAC is a strongly universal set of hash functions [61], is $2^{-\log_2(0.53) \cdot m} = 2^{0.91 \cdot m}$ based on (27).

On the other hand, in [59] it has been shown for various noise-free PUF simulations such as arbiter PUFs, XOR arbiter PUFs and Feed-Forward Arbiter PUFs that the prediction rate varies between 97% and 99%, after observing a few hundreds of thousands of CRPs and implementing a model building attack. Essentially, this type of attacks achieve a prediction rate, which is an estimation of the probability mass function of the next response, conditioned on a certain set of CRPs. In [59] it leads to conditional probability of at least $2^{\log_2(0.97) \cdot m} = 2^{-0.04 \cdot m}$ when the prediction rate is 97%, and so the average guesswork under this model building attack is achieved by assigning this probability to the average guesswork in (27) in which case we get $2^{0.04 \cdot m}$. Note that when the prediction rate is 99% the average guesswork is $2^{-\log_2(0.99) \cdot m} = 2^{0.014 \cdot m}$; therefore, when $m = 256$ we get that at 97% the PUF is $2^{0.026 \times 256} = 100$ times more secure than at 99%.

Therefore, guesswork provides a unified framework for comparing the security level of different PUFs under model building attacks. It can also be used as a means of understanding what is the desired prediction rate for a model based attack, and as a result how many challenge response pairs should be observed.

Next, we compare the security of stable and noisy XOR arbiter PUFs [59] under model based attacks in terms of the number of guesses for which the probability of guessing the correct response is 99%. We use equation (26) to derive the results of this subsection.

The expression in (26) depends on the min-entropy, and so for noisy PUFs we need to incorporate the effect of the noise into the min-entropy. For this we use Lemma 1 in which the min-entropy is extended to the noisy case. In Table V we use guesswork to compare the security of stable XOR arbiter PUFs to the one of noisy XOR arbiter PUFs with the same number of XORs, under model based attacks. We assign the prediction rates and noise levels reported in [59] to (26), and find the number of guesses under model based attacks in which the verifier has to take into account the noise as the PUF owner observes noisy responses. The table shows how much more secure stable XOR arbiter PUFs are when compared to the noisy versions under model based attacks. Essentially, it shows how susceptible such arbiter-based strong PUF is after observing a certain number of CRPs. In fact, when the noise level is over 5%, the probability of guessing the correct response up to the noise level is very close to one (about $1 - 10^{-10}$), which means that this PUF is completely broken. This is because the prediction rate of each bit as reported in [59] is 97.34%, whereas the noise level is 5% and so the chance that the guessed response is not within the noise level of the original response is extremely small for reasonable values of m. Therefore, guesswork enables us to incorporate the effect of noise and model based attacks into one framework that allows

CRPs ($\times 10^3$)	D=0%	D=2%	D=5%	D=10%
200	41	10	1	1
500	22	10	1	1

comparison between different PUFs and various scenarios.

VIII. CONCLUSION

In this paper we describe the stability-guaranteed LEDPUFs that requires no stability enhancement techniques, where the source of randomness is extracted from locally enhanced DSA process. Inter-distance measurements on the LEDPUFs show that both weak and strong LEDPUFs are ideally unique. The area and latency of the weak LEDPUF is much smaller than existing weak PUFs because no error correcting schemes are needed. The strong LEDPUF provides large CRP space because of its cryptographic hash based structure. The weak LEDPUF used in the strong LEDPUF construction cannot be replaced by existing weak PUFs because an absolute 0% intra-distance is required for the weak PUF to avoid the avalanche effect of the strong LEDPUF. We quantify the security level provided by weak LEDPUF by calculating the expected guesswork resulting from weak LEDPUFs empirical probability function; the loss compared to a fair coin toss is negligible.

Furthermore, we develop a unified guesswork-based analyses for PUFs. We show through guesswork analysis that stability has a more severe impact on the PUF security than biased responses. In addition, we analyze guesswork for two new problems: Guesswork under probability of attack failure, and the guesswork of strong PUFs.

Our ongoing work includes exploring other sources of LEDPUF that are more resilient to invasive attacks, such as Scan Electron Microscopy (SEM) or Transmission Electron Microscopy (TEM).

APPENDIX A
PROOF OF THEOREM 2

The proof relies on the method of types [50] as well as on the following assumptions.

1) When $\alpha = D(s||p)$ and $s > p$ the types for which $D(q||p) \geq D(s||p)$ are $p < s \leq q \leq 1$ as well as any $0 \leq q < p$ for which $D(q||p) \geq D(s||p)$.
2) Since the probability that a type is in A decays like $2^{-m \cdot D(s||p)}$, the conditional probability of drawing a certain vector of type q given that q is not in A decreases like $2^{-(H(q)+D(q||p)) \cdot m}$ [50] (i.e., it decays at the same rate as the original probability mass function).

3) When there are no constraints, the rate at which the average guesswork of any moment increases, is the solution to the following optimization problem

$$\max_{0 \leq q \leq 1} \rho \cdot H(q) - D(q||p) = \rho \cdot H_{1/(1+\rho)}(p)$$

where $\rho \cdot H(q) - D(q||p)$ is a concave function whose maximum is at s^*.

First let us consider the case where $D = 0$. Since the rate at which the conditional probability mass function decreases does not change (as stated above), the average guesswork of any moment is the solution to the following optimization problem.

$$\max_{q \notin A} \rho \cdot H(q) - D(q||p). \tag{28}$$

When $s^* \notin A$ (i.e., when the attacker can guess s^*) which occurs in the range $s* \leq s \leq 1$, the average guesswork does not change although the probability of attack failure is no longer zero. However, when $p < s < s^*$, $s^* \in A$ and since the function is concave the solution to the optimization problem is s, which is the closest element to s^* among the elements that are not in A.

In the case when $D > 0$ the proof follows along the same lines as Theorem 1 in [49]. This is because the conditional probability remains the same and therefore for the binary case with Hamming distance, the problem can be reduced to the following optimization problem

$$\max_{q \notin A} (\rho \cdot H(q) - D(q||p)) - \rho \cdot H(D). \tag{29}$$

APPENDIX B
PROOF OF LEMMA 1

The proof is based on the method of types [50]. We need to find the exponential rate at which the sum

$$\sum_{i=0}^{m \cdot D} \binom{m}{i} p^i (1-p)^{m-i} \tag{30}$$

decreases. Let us define $i = \alpha \cdot m$. From the method of types we know that the exponential growth rate of $\binom{m}{\alpha \cdot m}$ is $H(\alpha)$. On the other hand the rate at which $p^{\alpha m} (1-p)^{(1-\alpha)m}$ decreases is $H(\alpha) + D(\alpha||p)$. We are interested in the most dominant term in (30), and therefore we wish to find

$$\min_{0 \leq \alpha \leq D} D(\alpha||p). \tag{31}$$

The solution to this optimization problem is the statement of the lemma.

REFERENCES

[1] W. Wang, Y. Yona, S. Diggavi, and P. Gupta. LEDPUF: Stability-Guaranteed Physical Unclonable Functions through Locally Enhanced Defectivity. In *IEEE HOST*, May 2016.
[2] R. Maes and I. Verbauwhede. Physically Unclonable Functions: A Study on the State of the Art and Future Research Directions. In *Towards Hardware-Intrinsic Security*. 2010.
[3] B. Gassend, D. Clarke, M.V. Dijk, and S. Devadas. Silicon physical random functions. In *Proc. CCSC*, 2002.
[4] G.E. Suh and S. Devadas. Physical Unclonable Functions for Device Authentication and Secret Key Generation. In *Proc. DAC*, 2007.

[5] C. Yin, G. Qu, and Q. Zhou. Design and implementation of a group-based RO PUF. In *Proc. DATE*, 2013.

[6] D.E. Holcomb, W.P. Burleson, and K. Fu. Power-Up SRAM State as an Identifying Fingerprint and Source of True Random Numbers. *IEEE Transactions on Computers*, 2009.

[7] A. Maiti and P. Schaumont. Improving the quality of a Physical Unclonable Function using configurable Ring Oscillators. In *International Conference on FPL*, Aug 2009.

[8] Lerong Cheng et al. Physically Justifiable Die-Level Modeling of Spatial Variation in View of Systematic Across Wafer Variability. *IEEE TCAD*, 2011.

[9] Lang Lin, S. Srivathsa, D.K. Krishnappa, P. Shabadi, and W. Burleson. Design and Validation of Arbiter-Based PUFs for Sub-45-nm Low-Power Security Applications. *IEEE TIFS*, 2012.

[10] D. Ganta and L. Nazhandali. Study of IC Aging on Ring Oscillator Physical Unclonable Functions. In *IEEE ISQED*, 2014.

[11] W. Che, J. Plusquellic, and S. Bhunia. A Non-Volatile Memory Based Physically Unclonable Function without Helper Data. In *Proc. ICCAD*, 2014.

[12] C. Yin and G. Qu. Improving PUF security with regression-based distiller. In *Proc. DAC*, 2013.

[13] L. Feiten, T. Martin, M. Sauer, and B. Becker. Improving RO-PUF Quality on FPGAs by Incorporating Design-Dependent Frequency Biases. In *IEEE ETS*, 2015.

[14] M. Majzoobi, F. Koushanfar, and S. Devadas. FPGA PUF using programmable delay lines. In *IEEE International Workshop on WIFS*, Dec 2010.

[15] T. Xu and M. Potkonjak. Robust and flexible FPGA-based digital PUF. In *International Conference on FPL*, 2014.

[16] J. Delvaux and I. Verbauwhede. Key-recovery attacks on various RO PUF constructions via helper data manipulation. In *Proc. DATE*, March 2014.

[17] X. Zhang. VLSI Architectures for Modern Error-Correcting Codes. 2015.

[18] Van Herrewege et al. *Reverse Fuzzy Extractors: Enabling Lightweight Mutual Authentication for PUF-Enabled RFIDs*. March 2012.

[19] J. Delvaux, R. Peeters, D. Gu, and I. Verbauwhede. A Survey on Lightweight Entity Authentication with Strong PUFs. In *ACM Computing Surveys*, Nov 2015.

[20] F. Armknecht et al. Towards a Unified Security Model for Physically Unclonable Functions. In *Topics in Cryptology*. 2016.

[21] A. Rukhin et al. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. 2010.

[22] R. Maes and I. Verbauwhede. Physically Unclonable Functions: a Study on the State of the Art and Future Research Directions. *Towards Hardware-Intrinsic Security*, 2010.

[23] K. Scarfone and M. Souppaya. Guide to enterprise password management. *Recommendations of the NIST*.

[24] Domenic Forte and Ankur Srivastava. On Improving the Uniqueness of Silicon-based Physically Unclonable Functions via Optical Proximity Correction. In *Proc. DAC*, June 2012.

[25] R. Kumar, S. N. Dhanuskodi, and S. Kundu. On Manufacturing Aware Physical Design to Improve the Uniqueness of Silicon-Based Physically Unclonable Functions. In *VLSID*, Jan 2014.

[26] H. T. Shen et al. Selective Enhancement of Randomness at the Materials Level: Poly-Si Based Physical Unclonable Functions (PUFs). In *ISVLSI*, July 2016.

[27] B. Xu et al. Self-assembly of liquid crystal block copolymer PEG-b-smectic polymer in pure state and in dilute aqueous solution. *Faraday discussions*, 2009.

[28] M. Kim et al. Interplay of surface chemical composition and film thickness on graphoepitaxial assembly of asymmetric block copolymers. *Soft Matter*, 2013.

[29] Y. Badr, A. J. Torres, and P. Gupta. Mask assignment and synthesis of DSA-MP hybrid lithography for sub-7nm contacts/vias. In *Proc. DAC*, June 2015.

[30] H. Kim, S. Park, and W. D Hinsberg. Block copolymer based nanostructures: materials, processes, and applications to electronics. *Chemical reviews*, 2009.

[31] H. Pathangi et al. Defect mitigation and root cause studies in IMEC's 14 nm half-pitch chemo-epitaxy DSA flow. *Proc. SPIE*, 2015.

[32] D. Sundrani, S. Darling, and S. Sibener. Hierarchical assembly and compliance of aligned nanoscale polymer cylinders in confinement. *Langmuir*, 2004.

[33] C. T. Black. Polymer Self-Assembly as a Novel Extension to Optical Lithography. *ACS Nano*, October 2007.

[34] H. Kang et al. Degree of perfection and pattern uniformity in the directed assembly of cylinder-forming block copolymer on chemically patterned surfaces. *Macromolecules*, 2011.

[35] K. Yoshimotoa and T. Taniguchi. Large-Scale Dynamics of Directed Self-Assembly Defects on Chemically Pre-Patterned Surface. *Proc. SPIE*, 2013.

[36] M. Mller et al. Kinetics of directed self-assembly of block copolymers on chemically patterned substrates. *Journal of Physics: Conference Series*, 2015.

[37] B. L. Peters et al. Graphoepitaxial Assembly of Cylinder-Forming Block Compolymers in Cylindrical Holes. In *Journal of Polymer Science*, Dec 2014.

[38] F. Detcheverry et al. Monte Carlo simulations of a coarse grain model for block copolymers and nanocomposites. In *Macromolecules*, 2008.

[39] J. Guajardo et al. FPGA Intrinsic PUFs and Their Use for IP Protections. In *CHES*, Sep 2007.

[40] C. Herder, M. Yu, F. Koushanfar, and S. Devadas. Physical Unclonable Functions and Applications: A Tutorial. *Proc. of the IEEE*, Aug 2014.

[41] D. Nedospasov, J.-P. Seifert, C. Helfmeier, and C. Boit. Invasive PUF Analysis. In *Workshop on FDTC*, Aug 2013.

[42] C. Helfmeier, C. Boit, D. Nedospasov, and J.-P. Seifert. Cloning Physically Unclonable Functions. In *IEEE HOST*, June 2013.

[43] Y. Dodis, A. Reyzin, and A. Smith. Fuzzy extractor, A brief survey of results from 2004 to 2006. In *Security with Noisy Data*. 2007.

[44] E. Arikan. An inequality on guessing and its application to sequential decoding. *IEEE Tran. on Inf. Th.*, 1996.

[45] J. Kelsey et al. Secure applications of low entropy keys. *Proc. of ISW*, Sep 1998.

[46] Roel Maes. *PUF-Based Entity Identification and Authentication*, pages 117–141. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[47] A. Aysu, Y. Wang, P. Schaumont, and M. Orshansky. A new maskless debiasing method for lightweight physical unclonable functions. In *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 134–139, May 2017.

[48] J.L. Massey. Guessing and entropy. In *ISIT*, 1994.

[49] E. Arikan and N. Merhav. Guessing subject to distortion. *IEEE Transactions on Information Theory*, May 1998.

[50] T.A. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley& Sons, 2006.

[51] R. Sundaresan. Guessing under source uncertainty. *IEEE Transactions on Information Theory*, 53(1):269–287, Jan 2007.

[52] R. Sundaresan. Guessing Under Source Uncertainty With Side Information. In *IEEE ISIT*, July 2006.

[53] D. Malone and W. G. Sullivan. Guesswork and entropy. *IEEE Transactions on Information Theory*, March 2004.

[54] M. M. Christiansen, K. R. Duffy, F. du Pin Calmon, and M. Medard. Multi-User Guesswork and Brute Force Security. *IEEE Transactions on Information Theory*, Dec 2015.

[55] Roel Maes, Vincent van der Leest, Erik van der Sluis, and Frans Willems. *Secure Key Generation from Biased PUFs*, pages 517–534. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[56] M. Bellare, R. Canetti, and H. Krawczyk. Keyed Hash Functions

14

and Message Authentication. In *Crypto*, 1996.

[57] Peter Pessl and Michael Hutter. Pushing the limits of SHA-3 hardware implementations to fit on RFID. In *CHES*, Aug. 2013.

[58] S.W. Jung and S. Jung. HRP: A HMAC-based RFID mutual authentication protocol using PUF. In *ICOIN 2013*.

[59] U. Ruhrmair, , et al. PUF Modeling Attacks on Simulated and Silicon Data. *IEEE TIFS*, 2013.

[60] G. T. Becker. The Gap Between Promise and Reality: On the Insecurity of XOR Arbiter PUFs. In *IEEE HOST*, Sep. 2015.

[61] J. L. Carter and M. N. Wegman. Universal Classes of Hash Functions (Extended Abstract). In *Proc. ACM STOC*, 1977.

Puneet Gupta is currently a faculty member of the Electrical and Computer Engineering Department at UCLA. He received the B.Tech degree in Electrical Engineering from Indian Institute of Technology, Delhi in 2000 and Ph.D. in 2007 from University of California, San Diego. He cofounded Blaze DFM Inc. (acquired by Tela Inc.) in 2004 and served as its product architect till 2007. He has authored over 100 papers, 16 U.S. patents, a book and a book chapter. He is a recipient of NSF CAREER award, ACM/SIGDA Outstanding New Faculty Award, SRC Inventor Recognition Award and IBM Faculty Award. He currently leads the C-DEN center (http://cden.ucsd.edu/index.php) which focuses on future semiconductor technologies. Dr. Gupta's research has focused on building high-value bridges across application-architecture-implementation-fabrication interfaces for lowered cost and power, increased yield and improved predictability of integrated circuits and systems.

Wei-Che Wang received the B.S. degree in Computer Science from National Taiwan University, Taiwan in 2007 and the M.S. degree in Electronics Engineering from National Taiwan University, Taiwan, in 2009. He was with Taiwan Semiconductor Manufacturing Company (TSMC) from 2009 to 2013 as an engineer in standard cell library department. He is currently a Ph.D. student in the NanoCad Lab led by Dr. Puneet Gupta at University of California, Los Angeles (UCLA) Electrical and Computer Engineering. His research interests include hardware security and the development of computational techniques for optimizing semiconductor technologies.

Yair Yona received a B.Sc. in electrical engineering (magna cum laude), an M.Sc. in electrical engineering (magna cum laude), and a Ph.D. in electrical engineering, all from Tel-Aviv University, Israel, in 2005, 2009 and 2014, respectively. He is currently a research scientist at Intel Labs, Santa Clara, CA. Between 2015 and 2017 he was a postdoctoral scholar in the Department of Electrical and Computer Engineering at the University of California Los Angeles, Los Angeles, CA. Between 2003-2008 he served as a DSP and algorithms engineer at several companies, including Intel Corporation, Marvell Ltd., and Amimon Ltd. Dr. Yona was a recipient of the Intel award for excellence in academic studies and research (2009), a Motorola Scholarship in the field of advanced communication (2009), and the Weinstein Prize (2010, 2014) for research in the area of signal processing.

Suhas N. Diggavi received the B. Tech. degree in electrical engineering from the Indian Institute of Technology, Delhi, India, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1998. After completing his Ph.D., he was a Principal Member Technical Staff in the Information Sciences Center, AT&T Shannon Laboratories, Florham Park, NJ. After that he was on the faculty of the School of Computer and Communication Sciences, EPFL, where he directed the Laboratory for Information and Communication Systems (LICOS). He is currently a Professor, in the Department of Electrical and Computer Engineering, at the University of California, Los Angeles, where he directs the Information Theory and Systems laboratory.

His research interests include wireless network information theory, wireless networking systems, network data compression and network algorithms; more information can be found at http://licos.ee.ucla.edu. He has received several recognitions for his research including the 2013 IEEE Information Theory Society & Communications Society Joint Paper Award, the 2013 ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc) best paper award, the 2006 IEEE Donald Fink prize paper award. He is currently a Distinguished Lecturer and also serves on board of governors for the IEEE Information theory society. He is a Fellow of the IEEE.

He has been an associate editor for IEEE Transactions on Information Theory, ACM/IEEE Transactions on Networking, IEEE Communication Letters, a guest editor for IEEE Selected Topics in Signal Processing and in the program committees of several IEEE conferences. He has also helped organize IEEE and ACM conferences including serving as the Technical Program Co-Chair for 2012 IEEE Information Theory Workshop (ITW), the Technical Program Co-Chair for the 2015 IEEE International Symposium on Information Theory (ISIT) and General co-chair for Mobihoc 2018. He has 8 issued patents.