A Novel Modeling-Attack Resilient Arbiter-PUF Design

Conference Paper · February 2021

4 authors, including:

Mohammad Ebrahimabadi
University of Maryland, Baltimore County
5 PUBLICATIONS **2** CITATIONS

SEE PROFILE

Mohamed Younis
University of Maryland, Baltimore County
344 PUBLICATIONS **16,147** CITATIONS

SEE PROFILE

Naghmeh Karimi
University of Maryland, Baltimore County
64 PUBLICATIONS **472** CITATIONS

SEE PROFILE

A Novel Modeling-Attack Resilient Arbiter-PUF Design

Mohammad Ebrahimabadi, Mohamed Younis, Wassila Lalouani, and Naghmeh Karimi
CSEE Department, University of Maryland Baltimore County, Baltimore, MD 21250
Email:{ebrahimabadi, younis, lwassil1, nkarimi}@umbc.edu

Abstract— **Physically Unclonable Functions (PUFs) have been considered as promising lightweight primitives for random number generation and device authentication. Thanks to the imperfections occurring during the fabrication process of integrated circuits, each PUF generates a unique signature which can be used for chip identification. Although supposed to be unclonable, PUFs have been shown to be vulnerable to modeling attacks where a set of collected challenge response pairs are used for training a machine learning model to predict the PUF response to unseen challenges. Challenge obfuscation has been proposed to tackle the modeling attacks in recent years. However, knowing the obfuscation algorithm can help the adversary to model the PUF. This paper proposes a modeling-resilient arbiter-PUF architecture that benefits from the randomness provided by PUFs in concealing the obfuscation scheme. The experimental results confirm the effectiveness of the proposed structure in countering PUF modeling attacks.**

I. Introduction

Current era is characterized by the proliferation of miniaturized smart devices that are interconnected at a large scale in order to serve a broad range of applications. Ensuring the security of these devices is of utmost importance. Particularly, device authentication is quite crucial given the scale, heterogeneity, and dynamic interaction. Traditionally authentication was being conducted using Public key Infrastructures (PKI) or Identity-Based Encryption (IBE) [1], [2]. The former uses asymmetric cryptography, e.g., RSA to check the device identity. Although secure, PKI certification is highly costly and not scalable for the devices that have resource constraints such as the devices used in Internet of Things (IoT) frameworks. On the other hand, IBE uses public cryptography for which the public keys are extracted from publicly known information about the device [3]. IBE certifications suffer from power overhead and so are not suitable for power-constrained devices.

Physically Unclonable Functions (PUFs) are deemed a promising solution for authenticating integrated circuits (IC), hardware metering, certified execution, and key generation for cryptographic applications [4]. In practice, PUFs can be deployed as roots of trust for secure authentication and key generation [5]. PUFs benefit from the imperfections that occur during the fabrication process of integrated circuits, so-called process variations, such that each PUF fabricated from the same design constitutes a distinct fingerprint; basically a PUF generates a unique signature referred to as Challenge-Response Pairs (CRP) where the challenge and response denote to the PUF's input and output values, respectively [6]. The PUF circuits tend to be robust and small in size which make them well suited for radio-frequency identifiers (RFIDs), smart cards, and other small low-cost devices.

A PUF is embedded in each device during the fabrication process, and a subset of its CRPs are registered after the device fabrication. These CRPs are then used during operation to authenticate the device [7]. By avoiding storage of signatures in the device memory, PUFs enhance the security of the integrated circuits in which they are embedded. Depending on the number of possible challenge bit patterns, PUFs are divided into two groups, namely, weak and strong PUFs. The former consists of the PUFs that include a limited set of CRPs (e.g., Ring-Oscillator PUFs), and are mainly used for random key generation for cryptographic modules, or for IC metering to counter piracy, overproduction attacks, etc. On the other hand, strong PUFs realize a large set of CRPs, and are suitable for device authentication and integrity checking [8].

A PUF circuitry should be *easy to evaluate yet hard to characterize* [9], i.e., the PUF response to each challenge should be available in a short amount of time upon applying the challenge, and meanwhile, the PUF's response to a challenge should not be predictable based on a limited subset of its CRPs. However, in practice even the so-called strong PUFs may be compromised and their behavior may be modeled using Machine Learning (ML) techniques [10]. In the recent years with the improvement of Artificial Intelligence, machine learning schemes have found their way to several security challenges that would not be otherwise raised, among which modeling of PUFs' behavior using a subset of their CRPs has received the lion's share of attention, and accordingly caused a lot of concern for conducting a secure and reliable authentication of integrated circuits.

Strong PUFs and in particular arbiter-PUFs (one of the most preeminent such PUFs) and its variants, have been shown to be vulnerable to ML based modeling attacks in the recent years [10], [11]. Benefiting from various ML schemes, these attacks model the target PUF based on a subset of its CRPs to be able to predict the PUF response for the unseen challenges. These CRPs are either intercepted through communication when authenticating the device, or applied in a lab if the device is captured. The latter may further factor in the sensitivity of a response to environmental noise caused by temperature or voltage variations in order to model the PUF. Specifically, attacks based on the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) take the reliability information of the target PUF into account to model its behavior [12].

Increasing PUF robustness against modeling attacks and in particular the ML-based scheme is of utmost importance. Accordingly, in this paper we propose a novel PUF architecture (built upon arbiter-PUFs) that is highly resilient to modeling attacks. The basic idea is to obfuscate the PUF challenge bit-stream. Unlike most of the published challenge-obfuscating schemes, e.g. [13], that mainly mutate all (or most) of challenge bits, only a few bits of the challenge bit-stream that are more influential on the PUF's response are obfuscated. Hence, the area overhead of the proposed design is much less than other challenge-obfuscating approach such as [13]. In addition, knowing the obfuscation methodology does not allow the adversary to bypass the anti-modeling protection. We demonstrate the strength of our new design using the data gathered from its FPGA implementation. The contributions of this paper are as follows:

- Proposing a novel PUF architecture that diminishes the vulnerability of an arbiter-PUF to modeling attacks;
- Analyzing the resiliency of the proposed PUF design against state-of-the-art ML-based attacks;
- Studying the robustness of the proposed architecture against reliability-based modeling attacks such as the CMA-ES attack;
- Evaluating the proposed PUF using the data extracted from FPGA implementation.

The rest of this paper is organized as follows. Section II presents related work proposed against PUF modeling attacks. Section III provides the preliminary backgrounds. Section IV describes the proposed PUF design. The validation results are reported in Section V. Finally, Section VI concludes the paper.

II. Related work

Thanks to their low area and the broad range of CRPs, arbiter-PUF and its derivatives (XOR-PUF, Feed-Forward PUF, etc) are one of the most preeminent types of strong PUFs deployed for authentication purposes. However, these PUFs have been shown to be vulnerable to modeling attacks [11]. Thereby, many studies have been conducted to alleviate such vulnerability via enhancing the original arbiter-PUF. The basic idea is to embed some extra logic alongside the PUF to conceal the real PUF response to each challenge. In the so-called controlled PUFs, the PUF responses are not directly revealed but instead only the Hash value of PUF responses is transmitted [9], [12] in order to thwart the modeling attacks. Challenge Obfuscation schemes have been also proposed [13] where each challenge bit-stream C is mutated before feeding the PUF; hence the response is generated for the mutated challenge (i.e., $\hat{C}$) and is not directly related to C that the adversary is aware of. In essence this scheme misleads the adversary by injecting wrong CRPs into the dataset that is being used for training the PUF model. However, both controlled PUFs and challenge obfuscated PUFs suffer from considerable area overhead devoted to the Hash function or the circuitry for the challenge mapping. Our proposed design overcomes such shortcoming by only obfuscating a few challenge bits.

In both controlled and obfuscated PUFs, if the real CRPs can be inferred from the mutated ones via reverse engineering the circuitry, the PUF can be modeled. Therefore, the real challenge or response should not be predictable from the mutated counterpart. Accordingly, Vatajelu et al proposed to incorporate a symmetric encryption circuitry, specifically AES, whose key is generated via a weak PUF embedded in the chip along with the arbiter-PUF [14]. In this method, the arbiter-PUF is fed with the cipher text generated by feeding the challenge bits C to the AES circuit. This scheme imposes a large hardware overhead related to the AES circuit. Gassend et al. [15] proposed a PUF design that consists of one Hash function to mutate the challenge and another to alter the response. However, applying a Hash function imposes area overhead. Moreover, Hash function at the response requires parallel PUFs, something that is being avoided in practice in order to save area; instead of including several PUFs in parallel each generating one bit of response, in many industrial applications a single or a few PUFs are embedded and queried multiple times with different challenges to generate different response bits.

Gu et al. [16] proposed a modeling-attack resilient PUF-based authentication scheme by incorporating two PUFs in each node, specifically a genuine PUF and a fake PUF. The genuine PUF responses are used for authentication while the fake PUF is queried once a while to mislead the adversary who observes (or eavesdrops on) the response bits. This method increases the traffic related to the exchange of redundant CRPs thus not suitable for many applications. Meanwhile, Chauhanet et al proposed a dual-mode PUF to improve the ML resistance [17]. The proposed PUF operates on two modes, namely counting and state stabilization. Although the modeling resistance is improved, the proposed obfuscation increases the sensitivity to environmental noise [18]. On the other hand, Wang et al [19] use adversarial models to fool the adversary by changing the response of some challenges based on a function of the fed challenge bits or in a periodic manner. Although the hardware overhead is little, if the poisoning algorithm is revealed to the adversary (via reverse engineering the hardware) the PUF can be modeled. Our proposed design stays robust even if the adversary knows the obfuscation methodology.

III. Preliminaries

A. Arbiter-PUF

An arbiter-PUF consists of a pair of delay chains; when queried, it generates one response bit per challenge [20]. This PUF operates based on variations in the microelectronics manufacturing process that induce race between two identical paths (top and bottom paths shown in Fig. 1). The race corresponds to the difference in signal propagation delay on these two paths, and affects the value latched by the arbiter. Only the sign of this difference, rather than the exact value, is important. The sign, extracted by the arbiter, reflects the response and constitutes the PUF identifier. The arbiter can be realized as a simple SR-latch implemented by two cross-coupled NOR gates.

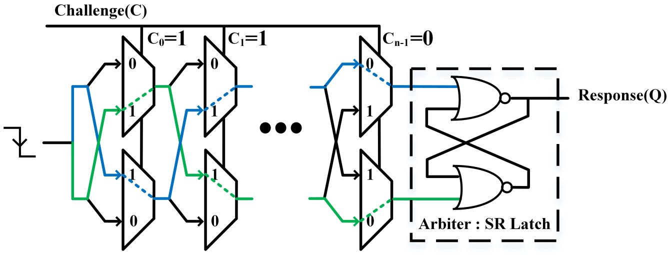


Figure 1. Illustrating the design of an arbiter-PUF.

B. Modeling Arbiter-PUFs

In this paper we assume that the adversary deploys ML to model the target arbiter-PUF where the challenge bit-streams (i.e., $C = c_0, c_1, c_2, \ldots, c_{n-1}$) are used as the training data, and the PUF 1-bit response is considered as the label. Here, n denotes the size of the challenge bit stream. An arbiter PUF can be modeled as an additive linear delay model where the response bit is generated based on the summation of delays in each stage depending on the challenge bits feeding each stage. In particular, the PUF response is extracted based of the sign of the delay differences in the top and bottom paths shown in Fig. 1 (i.e., sign of Δ) evaluated using Equation 1 [11]:

$$\begin{aligned} &\Delta = \overrightarrow{\omega}^T \overrightarrow{\Phi} \\ &\overrightarrow{\omega} = (\omega^0, \omega^1, \ldots, \omega^n) \\ &\overrightarrow{\Phi}(\overrightarrow{C}) = (\overrightarrow{\Phi}^0(\overrightarrow{C}), \overrightarrow{\Phi}^1(\overrightarrow{C}), \ldots, \overrightarrow{\Phi}^{n-1}(\overrightarrow{C}), 1) \end{aligned} \tag{1}$$

In Equation 1, $\overrightarrow{\omega}$ is a vector of the multiplexer delays and Φ is a function of input challenge bit-stream. They are computed using Equation 2 in which ${\delta_0}^i$ and ${\delta_1}^i$ denote the delay of stage i for the uncrossed and crossed signal paths, respectively.

$$\begin{aligned} &\Phi^i = \textstyle\prod_{j=i}^{n-1}(1 - 2c_j) \quad i = 0, 1, 2..., n-1 \\ &\omega^0 = \tfrac{\delta_1^0 - \delta_1^1}{2}, \\ &\omega^i = \tfrac{\delta_{i-1}^0 + \delta_{i-1}^1 + \delta_i^0 - \delta_{i-1}^1}{2}, \ for\ i = 1, 2, 3, ..., n-1 \\ &\omega^n = \tfrac{\delta_{n-1}^0 + \delta_{n-1}^1}{2} \end{aligned} \tag{2}$$

Finally, the output of arbiter-PUF is calculated via Equation 3:

$$Q = \text{sgn}(\overrightarrow{\omega}^T \overrightarrow{\Phi}) \tag{3}$$

In practice, the adversary opts to find an accurate estimation for $\overrightarrow{\omega}$ based on a subset of CRPs and use this vector to predict the response of the unseen challenges. In this paper, we use different ML schemes including Neural Networks (NN), Support Vector Machine (SVM), and Logistic regression [16] to show the efficiency of the proposed PUF design against state-of-the-art attack models that an adversary may deploy. We also evaluate the robustness of our design against the CMA-ES, a well-known reliability based attack against arbiter-PUFs [12].

IV. Proposed PUF Design

Our defense strategy against ML-based modeling attacks is through obfuscating the challenge bit-stream that feeds the arbiter-PUF. We opt to overcome the shortcoming of existing challenge-obfuscating schemes, in terms of hardware overhead, by obfuscating only a few challenge bits. We first categorize the effect of challenge bits on the PUF response and then devise a new PUF design that only obfuscates the most influential challenge bits to conceal the PUF functionality and diminish the accuracy of PUF models formed by attackers. In the balance of this section we discuss our design in detail.

A. Characterizing Response Dependence of Challenge Bits

As discussed earlier, the response of an arbiter-PUF is generated based on the race between two paths that seem identical in terms of gates but are in fact different due to process variations that affect the path delays. When the PUF is queried with a challenge bit-stream, a falling transition (rising in case of NAND-based arbiter) is applied as input. If this transition reaches to the upper NOR gate shown in Fig. 1 sooner than the lower one, the response ("Q") gets the value of "1", otherwise it would be "0". The possibility that both transitions reach the arbiter simultaneously is extremely low given the random nature of process variations that all multiplexers residing in the previous stages experience. Let α be the delay difference between the upper and lower transitions when reaching to the last level multiplexer in Fig. 1.

Theorem 1: The Most Significant Bit (MSB), $C[n-1]$, is the most influential challenge bit on the response of an arbiter-PUF.

Proof: Let's consider the transitions when the signals on the two paths reach the last level multiplexers (fed with $C[n-1]$) in Fig. 2. Three scenarios are possible:

1) *α > 0*: Fig. 2(a) depicts this scenario in which the transition reaches to upper multiplexer sooner. In this case, $C[n-1]$ can fully affect the response value, as Q would be $\overline{C[n-1]}$. Indeed, the effect of all other challenge bits resulted in the time difference between the signals shown in green and red, yet finally the value of $C[n-1]$ can determine the response. Note that the relative delay of the last level multiplexers (DM_{n-1}) is not very impactful as for large PUFs (with 32 or more challenges) that are used for authentication, mainly α is greater than DM_{n-1}.
2) *α<0*: Scenario 2 is illustrated in Fig. 2(b). Since α<0, Q get the value as $C[n-1]$.
3) *α=0*: This scenario is shown in Fig. 2(c). This scenario is quite rare considering the random process variations that all previous stages experience. Although the theoretically possible, it is impractical.

Based on these scenarios, the response of the PUF is highly impacted by the MSB of the challenge.

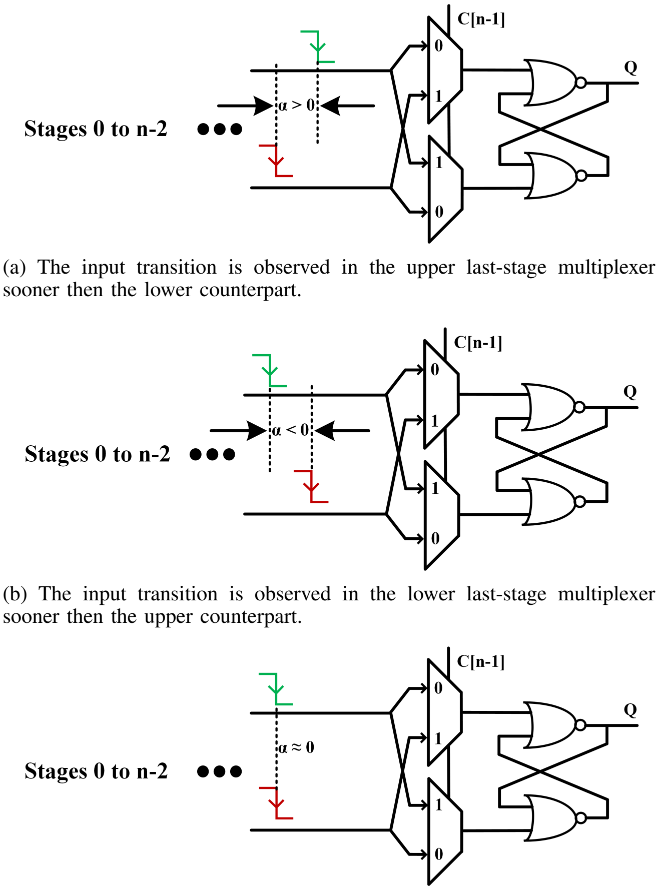
Theorem 1 confirms that for an n-bit arbiter-PUF the challenge bit associated with the multiplexer feeding the arbiter is the most influential bit on determining the PUF response. We can extend the above analysis to conclude the relative importance of the second MSB bit of challenge (i.e., $C[n-2]$).

Corollary 1: The influence of challenge bits on the PUF response grows from least significant bit (LSB) to MSB, i.e., the challenge bits can be ranked according to their influence on the response as $C[n-1] > C[n-2] > ..., C[2] > C[1] > C[0]$

Corollary 1 constitutes the underlying design principle for our new PUF architecture. As we explain in the next subsection, we obfuscate only m out of the n challenge bits in order to increase the PUF resiliency against modeling attacks.

B. Modeling-Resilient PUF Design

To degrade the adversary's ability in modeling the PUF and also limit the overhead, our approach calls for obfuscating only a small subset of the challenge bits. The analysis in the previous section paves the way for selecting what bits to obfuscate. *If we are to select m out of the n challenge bits, Theorem 1 and the subsequent corollary imply that the most significant bits in the challenge bit-stream are the prime choices.* To highlight the effect of obfuscation of a subset of the challenge bits, Fig. 3 shows the accuracy when 61, 62 and



(a) The input transition is observed in the upper last-stage multiplexer sooner then the lower counterpart.

(b) The input transition is observed in the lower last-stage multiplexer sooner then the upper counterpart.

(c) Transition is observed simultaneously by both multiplexers in last stage.

Figure 2. Various race scenarios in the last stage of an arbiter-PUF.

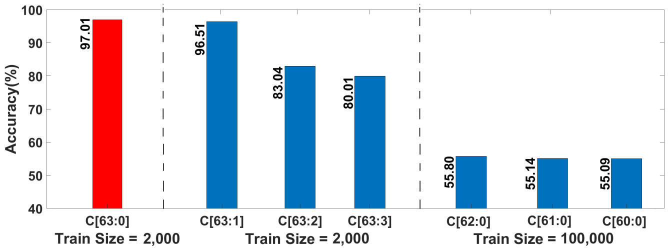


Figure 3. Modeling accuracy when a subset of the challenge bits are used in modeling the PUF.

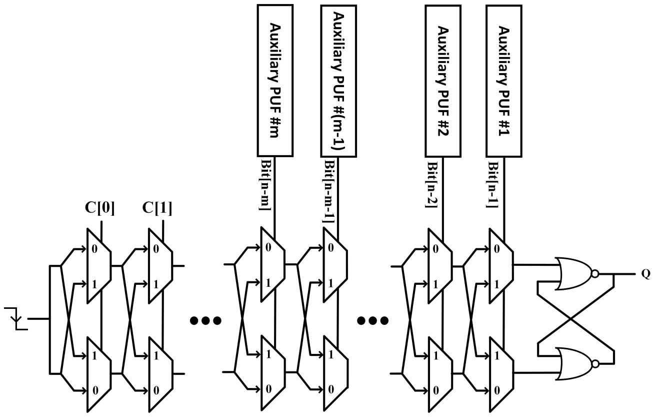


Figure 4. The overall view of the proposed obfuscated PUF. The auxiliary PUFs as realised as arbiter-PUFs.

63 out of the 64 challenge bits are used in modeling the PUF. The data is collected based on a 64-bit PUF implementation on a Xilinx ARTIX7 FPGA. The figure reports what happens when the excluded bits are MSB and LSB. The model accuracy when all 64 bits are used is also shown as a baseline. The results confirm that focusing on the MSB bits would yield sufficient protection. It is important to note the size of the training dataset. For LSB and the baseline, with as little as 2,000 CRPs, the accuracy exceeds 80%. Meanwhile for the MSB, accuracy stays below 55% even when using 100,000 CRPs for training the model.

The next logical questions are how many bits are to be obfuscated and how the picked MSB bits are mutated. Fig. 3 provides hints on how to determine the number of MSB bits to be obfuscated. The results in this figure clearly implies that 1-3 MSB bits suffice for a 64-bit arbiter-PUF. Note that an accuracy of 50% reflects inconclusiveness given the binary nature of the response, i.e., the adversary predicts "1" or "0" with the same probability. Meanwhile, to prevent revealing the challenge obfuscating hardware via reverse engineering the chip, and to make this piece of circuit unpredictable as well, we propose to obfuscate a few of MSB challenge bits with the assist of other PUFs (called auxiliary PUFs in Fig. 4) whose responses are used as the most significant challenge bits for the main PUF.

Fig. 4 shows the proposed architecture. It is noteworthy to mention that in our design *the auxiliary PUFs are realized as arbiter-PUFs. Moreover, the number of auxiliary PUFs and their size is decided such that the whole structure is resilient against modeling attacks.* More details are given in Section V-B. As we pointed out and will be supported by additional results, securing a 64-bit arbiter-PUF would need no more than obfuscating the 3 MSBs. In addition, as will be shown, when the auxiliary PUFs are diverse in size the design becomes even more resilient to modeling attacks. We note that we add a delay element (not shown) before feeding the transition to the main PUF in order to stabilize the response of auxiliary PUFs before the main PUF is queried. The area overhead of the proposed design is much less than other challenge-obfuscating approaches such as [13]. Another advantage of our approach is that knowing the obfuscation algorithm (or reverse engineering the obfuscating hardware) does not enable the adversary to defeat our design as the effect of obfuscation on the challenge would vary from one device to another, i.e., the obfuscation itself is unpredictable and unclonable as it is performed via other PUFs.

V. Experimental Results and Discussions

A. Experimental Setup

To demonstrate the resiliency of the designed PUF against modeling attacks, we have implemented a number of PUFs with the proposed architecture on Xilinx ARTIX7. The main PUF is 64 bit in all setups. We have tried auxiliary PUFs with various configurations; we show results when different numbers of auxiliary PUFs are used and when they have similar or diverse sizes. SVM, NN, LR, and CMA-ES have been used to apply modeling attacks against our proposed design. Our NN-based model is a 5-layer fully connected architecture. The adversary is assumed to know a subset of challenge-response pairs and opts to model the PUF using that subset to be able to predict the response for other possible challenges. We show the results for varying sizes of the training dataset (CRPs) and different configurations of our design, and use modeling accuracy as a metric to gauge the resilience against ML assisted modeling attacks.

In this section, we use the following notation: $M64_{S1,S2,S3}$ where $M64$ shows the size of the Main PUF (i.e., 64 in our experiments) and $S1, S2, S3$ denotes the size of the auxiliary PUF that drives the first, second, and third most significant bit of the main PUF's challenge, respectively. For example, $M64_{64,32,16}$ denotes a 64-bit main arbiter-PUF for which the most significant bit of challenge is provided by another 64-bit auxiliary arbiter-PUF, its 2^{nd} MSB is fed by a 32-bit PUF and its 3^{rd} MSB is driven by a 16-bit PUF. Note that when the size of auxiliary PUFs is less than 64 bits, they are fed with a subset of the 64-bit challenge of the main arbiter-PUF. Our experiments shows that the selection of such a subset does not have a significant impact on the modeling accuracy.

B. Validation Results

1) Impact of obfuscating the MSB bits of the main arbiter-PUF on the modeling attack success: This set of results assesses the resilience of the unprotected 64-bit arbiter-PUF and its protected variants when 1, 2, or 3 MSB bits of the main PUF are fed with a distinct 64-bit auxiliary PUF. Fig. 5 depicts the results for different training size (different number of CRPs). As expected, the unprotected PUF can be easily modeled; even with as low as 500 CRPs.

The results shown in Fig. 5 confirm that embedding the auxiliary PUFs on the most significant bits of the main PUF highly reduces the modeling accuracy when a small set of CRPs are used for training. As expected, the more the number of auxiliary PUFs, the higher the resiliency of the PUF against modeling. However, when training the model with 10,000 CRPs, only $M64_{64,64,64}$ is effective. Increasing the training size beyond 10,000 CRPs makes this circuitry also vulnerable to modeling attacks, where with 40,000 CRPs $M64_{64,64,64}$ can be modeled with an accuracy of 87.04%.

Table I depicts the modeling accuracy when the adversary applies SVM. As indicated by the results, unlike NN, SVM is not very promising in attacking the designed PUFs. Thereby, we focus on NN modeling for the next set of results.

The takeaway point from these observations is that incorporating auxiliary PUFs diminishes the modeling accuracy, but still not providing sufficient protection when the training set size grows. The next set of results gives more clues on how to configure our design for better resiliency to modeling attacks.

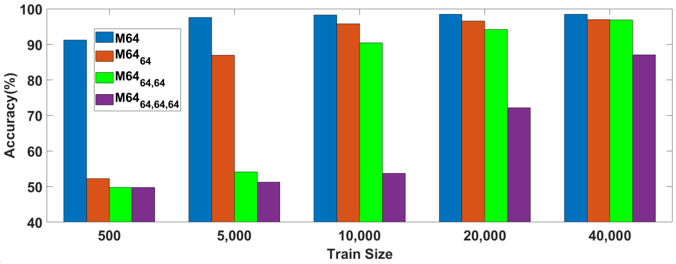


Figure 5. The PUF modeling accuracy using NN for different training dataset sizes. The main PUF is 64-bit, while 1, 2, or 3 64-bit auxiliary arbiter-PUFs feed the 1^{st}, 2^{nd}, and 3^{rd} MSBs of the main PUF. M64 denotes an unprotected configuration.

2) The effect of using diverse auxiliary PUF sizes: Here we investigate the impact of using auxiliary PUFs with different sizes on the accuracy of the ML-based modeling attacks. As the results in Fig. 5 indicate, increasing the training dataset size negatively affects the contribution of the auxiliary PUFs. We note that using the same PUF architecture for the main and all auxiliary PUFs facilitate the modeling process since all inputs to the main PUF remain to be a function of the entire challenge bit-stream. Thereby, we designed three diverse configurations and exposed them to NN-based modeling attacks. The first configuration has one 32-bit auxiliary PUF feeding the MSB bit of the main 64-bit PUF; the second configuration includes a 32-bit and a 16-bit PUF driving the 2 MSB bits of the main PUF; finally, the third structure includes a 32- a 16- and an 8-bit PUF feeding the 3 most significant bits of the main PUF. The results in Fig. 6 show that the third structure ($M64_{32,16,8}$) is very robust even when 200,000 CRPs are used for training the model. The results imply that the auxiliary PUFs need to be diverse in order for our design to be effective.

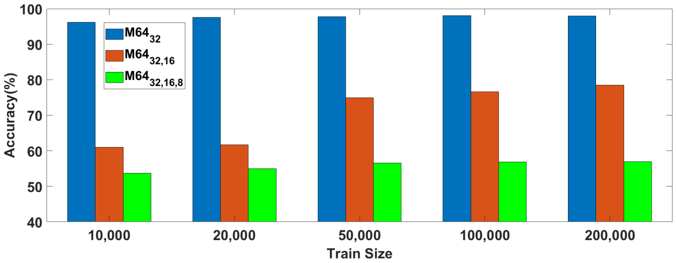


Figure 6. The PUF modeling accuracy using NN with different sizes of the training set. The main PUF is 64-bit and 1, 2, or 3 auxiliary PUFs with different sizes feed the 1^{st}, 2^{nd}, and 3^{rd} MSB bits of the main PUF.

3) Resistance against state-of-the-art modeling attacks: This set of results assesses the resiliency of our design with $M64_{32,16,8}$ configuration when LR, CMA-ES, and NN are applied. Fig. 7 shows the modeling accuracy when these schemes are used for training data sizes up to 1,000,000 CRPs. As shown neither of these schemes is successful in modeling our PUF design, where in all cases the accuracy is below 58% even when 1,000,000 CRPs are used for modeling. Note that the unprotected PUF can be modeled with an accuracy of 98% with as low as 2000, as shown earlier in Fig. 5 and Table I.

The CMA-ES based attack takes the reliability information obtained from the repeated measurements of challenge-response pairs into account for PUF modeling [21]. Such noise-induced reliability information is used as a side channel to assess the relative delay of multiplexers employed at different stages in arbiter-PUF families, and in turn to model the

Table I
THE PUF MODELING ATTACK ACCURACY USING SVM WITH DIFFERENT SIZES OF TRAINING DATASET.

PUF Circuitry	**500 CRPs**	**5K CRPs**	**10K CRPs**	**20K CRPs**	**40K CRPs**
$M64$	93.4%	98.42%	98.69%	98.625%	98.56%
$M64_{64}$	51.8%	58.44%	57.93%	57.48%	57.94%
$M64_{64,64}$	48.8%	53.14%	53.38%	53.36%	53.52%
$M64_{64,64,64}$	52.6%	56.82%	57.3%	57.405%	57.68%

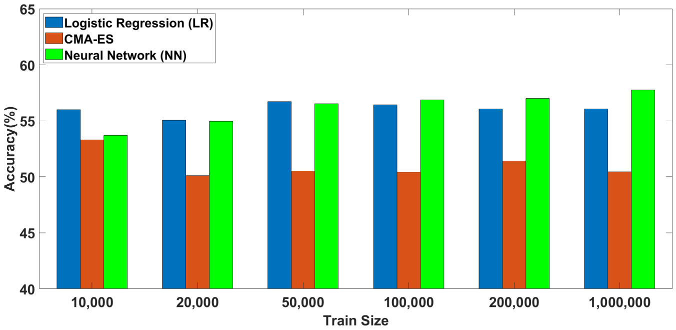


Figure 7. Modeling accuracy of the $M64_{32,16,8}$ configuration for LR, CMA-ES, and NN with varying training dataset sizes.

PUF behavior. To launch the CMA-ES attack we repeated each response measurement 5 times. As shown in Fig. 7, CMA-ES is also unsuccessful in modeling the $M64_{32,16,8}$ configuration.

4) PUF metrics: Uniqueness, randomness, uniformity, and reliability are important metrics based on which the PUFs are evaluated. The randomness denotes the unpredictability of PUF responses, uniqueness shows how well a single PUF is differentiated from other PUFs based on its CRPs, uniformity reflects the distribution of zeros and ones in the PUF response, and reliability shows how stable the PUF response is in different environmental conditions (e.g., change in temperature).We have implemented eight $M64_{32,16,8}$ PUFs (each with an 8-bit response) in our FPGA and evaluated the randomness, uniformity, and uniqueness of each PUF via 4,000,000 randomly chosen challenges. It has been observed that on average, the uniformity is about 50.02%, and the uniqueness among the 8 samples is 45.66%. Note that the ideal values for uniformity and uniqueness is 50%.

To evaluate the reliability of the proposed architecture in different temperatures, we applied 4,000,000 randomly generated challenges to our $M64_{32,16,8}$ PUF and measured the hamming distance of the responses when a similar challenge is applied. We considered the base temperature as 30°C and repeated the experiments in 0°C, 60°C and 90°C where on average the discrepancy was 2%, 2.1% and 4% in these temperatures, respectively. This demonstrates the reliability of our design. Moreover, the noise effect in the same temperature resulted in a negligible (0.5%) discrepancy in response, which confirms the viability of our design for PUF-based authentication schemes. The power consumption, measured by the Xilinx Power Estimator (XPE), estimated as 0.002W. The area overhead is much less than the obfuscated PUF in [13].

Table II
NIST Randomness Test Result

Test Description	**Passed (Total)**	**P**	**Test Description**	**Passed (Total)**	**P**
Frequency	80(80)	0.56	Frequency Block	79(80)	0.57
Runs	80(80)	0.49	The Longest Run	78(80)	0.45
FFT	79(80)	0.49	Non-overlap. Temp.	80(80)	0.48
Universal	4(4)	0.99	Linear Complexity	3(3)	0.60
Serial	79(80)	0.50	Binary Matrix Rank	26(26)	0.55
Entropy	80(80)	0.51	Cumulative Sums	80(80)	0.58
Random exc.	2(2)	0.51	Random exc. var.	2(2)	0.52

The $M64_{32,16,8}$ PUF randomness was evaluated using the statistical tests offered by NIST for assessing the randomness of true random generators [22]. We divided 4,000,000 responses to 80 blocks each including 50,000 responses, applied the NIST tests to each block. Table II shows the results for $M64_{32,16,8}$ PUF (our pick). Some of the tests, e.g., Universal, needs larger blocks so we partitioned our responses accordingly. As shown our PUF structure passed almost all tests. This confirms the randomness of our PUF structure.

VI. Conclusion

Physically unclonable functions provide an effective solution for authenticating integrated circuits. Delay-PUFs and in particular arbiter-PUFs have received a lot of attention in this regard due to their lightweight implementation and broad range of challenge-response pairs. However, machine learning schemes can be used to model the PUF behavior even with a small subset of CRPs. To alleviate such vulnerability, this paper has presented a novel arbiter-PUF design in which some of the challenge bits are obfuscated using other arbiter-PUFs. The new design not only diminishes the modeling accuracy but also conceals the challenge obfuscation scheme itself. The experimental results have demonstrated the resilience of the new design to state-of-the-art modeling attacks.

References

[1] X. Liu et al., "A Security Framework for the Internet of Things in the Future Internet Architecture," *Future Internet*, vol. 9, p. 27, 06 2017.

[2] X. Zhang *et al.*, "Towards name-based trust and security for content-centric network," in *Int'l Conf. on Network Protocols*, 2011, pp. 1–6.

[3] U. Chatterjee et al., "Building PUF Based Authentication and Key Exchange Protocol for IoT Without Explicit CRPs in Verifier Database," *IEEE TDSC*, vol. 16, no. 3, pp. 424–437, 2019.

[4] M. T. Rahman *et al.*, "An aging-resistant RO-PUF for reliable key generation," *IEEE TETC*, vol. 4, no. 3, pp. 335–348, 2015.

[5] M. N. Islam *et al.*, "On enhancing reliability of weak PUFs via intelligent post-silicon accelerated aging," *IEEE TCAS*, vol. 65, pp. 960–969, 2017.

[6] C. Herder *et al.*, "Physical unclonable functions and applications: A tutorial," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1126–1141, 2014.

[7] A. Aysu *et al.*, "End-to-end design of a PUF-based privacy preserving authentication protocol," in *CHES*, 2015, pp. 556–576.

[8] T. Kroeger *et al.*, "Effect of aging on puf modeling attacks based on power side-channel observations," in *DATE*, 2020, pp. 454–459.

[9] B. Gassend *et al.*, "Controlled physical random functions," in *Computer Security Applications Conf.*, 2002, pp. 149–160.

[10] U. Ruhrmair and J. Solter, "PUF modeling attacks: An introduction and overview," in *DATE*, 2014.

[11] U. Rührmair *et al.*, "Modeling attacks on physical unclonable functions," in *CCS*, 2010, pp. 237–249.

[12] G. T. Becker, "The gap between promise and reality: On the insecurity of xor arbiter PUFs," in *CHES*, 2015, pp. 535–555.

[13] S. S. Zalivaka *et al.*, "Reliable and modeling attack resistant authentication of arbiter PUF in FPGA implementation with trinary quadruple response," *IEEE TIFS*, vol. 14, no. 4, pp. 1109–1123, 2018.

[14] E. I. Vatajelu *et al.*, "On the encryption of the challenge in physically unclonable functions," in *IOLTS*, 2019, pp. 115–120.

[15] B. Gassend et al, "Controlled physical random functions and applications," *Trans. on Info. and Sys. Security*, vol. 10, no. 4, pp. 1–22, 2008.

[16] C. Gu *et al.*, "A modeling attack resistant deception technique for securing PUF based authentication," in *AsianHOST*, 2019, pp. 1–6.

[17] Q. Wang et al, "A machine learning attack resistant dual-mode PUF," in *GLSVLSI*, 2018, pp. 177–182.

[18] A. S. Chauhan *et al.*, "Novel randomized placement for FPGA based robust ROPUF with improved uniqueness," *JETTA*, vol. 35, no. 5, pp. 581–601, 2019.

[19] S.-J. Wang *et al.*, "Adversarial attack against modeling attack on PUF," in *DAC*, 2019, pp. 1–6.

[20] N. Karimi *et al.*, "Impact of aging on the reliability of delay pufs," *JETTA*, vol. 34, no. 5, pp. 571–586, 2018.

[21] J. Delvaux et al, "Side channel modeling attacks on 65nm arbiter PUFs exploiting cmos device noise," in *HOST*. IEEE, 2013, pp. 137–142.

[22] L. E. Bassham et al, *NIST SP 800-22: A statistical test suite for random & pseudorandom number generators for cryptographic applications*. NIST, 2010.