Robust and Reliable PUF Protocol Exploiting Non-Monotonic Quantization and Neyman-Pearson Lemma

Neelam Nasir[1], Julien Béguinot[1], Wei Cheng[1,2], Ulrich Kühne[1], and Jean-Luc Danger[1]

[1] Télécom Paris, Institut polytechnique de Paris, Palaiseau, France
[2] Secure-IC S.A.S, Paris, France

Abstract. Strong physical unclonable functions (PUFs) provide a cost-effective authentication solution for resource-limited devices. However, they are susceptible to machine learning (ML) attacks. The lightweight defenses against ML rely on adding non-linearity in the PUF behavior (as the XOR-PUF), or limiting the number of challenges at protocol level (as the lockdown protocol) to constrain learning. Another low-cost approach is to use a non-linear quantization of the response when the PUF provides an integer response, like the RO-PUF. This paper studies the non-monotonic quantization (NMQ) which greatly enhances the security when a large number of quantization level is used. Unfortunately, this makes the PUF highly unreliable, rendering it impractical for authentication purposes. In this study, we propose a solution which circumvents the intrinsic PUF unreliability of NMQ to build an effective authentication protocol. It relies on the Neyman-Pearson test which transforms the native dependability of responses into an asset to get a reliable authentication protocol. To validate this approach, we evaluate our solution in FPGA using a loop PUF (ring oscillator-based PUF) which is a multi-bin PUF. The results show that an authentication success of nearly 100% can be obtained with a high resistance as up to 60% accuracy against three types of ML attacks.

Keywords: Hardware security · Physical unclonable functions · ML attacks · Reliability · Non-monotonic quantization · Neyman-Pearson test

1 Introduction

Physical unclonable functions (PUFs) have been proposed as a low-cost security anchor [1, 2], notably for lightweight authentication protocol. This protocol relies on the physical input/output relationship (respectively challenge/response) of the PUF. The PUF called *strong* are well suited for this application as they provide many challenge-responses pairs(CRPs).

In the context of a challenge-response protocol, for a PUF to be *secure*, its output must be hard to predict. Strong PUFs relying on delay chains as entropy source (such as the arbiter PUFs [3] and RO PUFs [1]) have inherently

a linear behavior or a limited entropy. Therefore they are attackable by machine learning (ML) techniques such as linear and/or logistic regressions by collecting challenge-response pairs (CRPs) [4]. In order to prevent such attacks, the PUF must be hard to model. One type of protection is to introduce non-linearity into the current PUF architectures. The composition of PUFs is one way to increase the non-linear behavior. The XOR-PUF [1] which is composed of arbiter PUFs whose outputs are XORed, and the interpose-PUF [5] are the most popular. The use of permutation of multiple delay lines [6] like the Beli PUF [7] is another type of protection. However, even if a deep security analysis of these structures has shown a great increase of robustness against ML attacks, they still remain vulnerable [8, 6].

Another non-linear approach applicable to PUFs where the physical variable can be quantized with more than 1-bit, like the oscillation frequency of the RO-PUF, is to do the *non-monotonic quantization* (NMQ) [9]. The output distribution of these types of PUF is multi-bin PUF as their entropy is more than one bit, hence the values are {0,1,2,3} for a 2-bit linear quantization. A generic multi-bin PUF is the *Alphabet PUF* [10] which takes advantage of a multi-threshold quantization stage and data encoding. The alphabet PUF has been originally studied for secret key generation with Helper data and ECC, but it can also be used as a strong PUF with a CRP protocol. The NMQ quantization principle is to transpose a n-bit PUF to a 1-bit PUF such that the values 1 and 0 values are interleaved, i.e. the values {0,1,2,3} of the 2-bit PUF become {0,1,0,1}, thus generating a 1-bit output in a non-monotonic way. It has been shown in [9] that the resistance against ML grows with the quantization levels, denoted by Q in the sequel, but the reliability becomes so low with high Q that an authentication process becomes impossible. Indeed, Q values involves $Q-1$ thresholds to compare at the quantization stage, and consequently more unsteadiness due the environmental noise around these thresholds.

In this paper, we first study the PUF with NMQ quantization (NMQ-PUF) to better know the impact of the quantization level Q and the environmental noise on both the security against ML and the reliability. Then, we propose an authentication protocol based on the Neyman-Pearson Lemma [11] to compensate for the bad reliability of NMQ with high values of Q. It has to be noted that the security comes from the NMQ quantization, while the protocol helps to mitigate the low reliability of NMQ by exploiting the knowledge of the reliability itself for each challenge. Our simulations and experimental results with a multi-bin PUF – a Loop PUF [12] implemented in FPGA – show the effectiveness of the proposed approach, allowing for high values of Q, while compensating the resulting high bit error rate.

As a summary, the contributions of this paper are the following:

1. Present analyses on the reliability and security against modeling attacks with NMQ at various quantization levels.
2. Propose a new PUF authentication protocol using the Neyman-Pearson test in the presence of high bit error rates.

3. Validate the security and authentication success on a Loop PUF implemented in FPGA.

The remainder of this paper is structured as follows: Related work is discussed in Section 2. The main contributions on reliability, machine learning attacks and our proposed protocol are presented in Sections 3 and 4, respectively. We show experimental results in Section 5. Section 6 discusses about the impact of the environment on the authentication method, and Section 7 concludes.

2 Related Work

In this section we present the context related to the ML attack against PUF when used for authentication. The protections against such attacks are either to build a natively robust PUF or to devise specific PUF protocols, or both. The robustness challenge is all the greater if the PUF has to keep the lightweight property.

2.1 Modeling Attacks on PUF

In the classical PUF Challenge-Response Pair (CRP) protocol, the input challenges and associated output responses can be eavesdropped and replayed by the attacker. A stronger model which is currently used in the literature and in this paper is that the attacker has full access to the device's interface. Thus, she can freely input any challenge and read the associated response to build a CRP dataset and feed Machine Learning (ML) algorithms to get a PUF model. The attack efficiency highly depends on the ML algorithms and the number of collected CRPs to get a high accuracy. The arbiter PUF [13] is one of the first silicon PUF which has been devised and attacked by the Support Vector Machine (SVM) algorithm [14]. More derivatives of Arbiter PUF have been attacked by the Logistic Regression algorithm (LR) in [4]. It takes advantage of the quasi-linear behavior of the arbiter PUF which relies on a delay chain. The progress in ML attack has followed the evolution of PUF. For instance the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [15] uses the information of reliability to increase the attack efficiency of XOR-PUF. The hyperparameters of Neural Networks (NN) have often to be tailored to target a great variety of strong PUFs [16].

2.2 ML Resistant PUFs

Many PUFs have been devised to have a more complex and non-linear behavior, mainly by using a composition of PUFs. One of the most common is the XOR Arbiter PUF [1]. It has been shown that the Neural Network (NN) is an efficient ML algorithm to attack such PUF [17]. The Interpose PUF [5] provides a better security for the same complexity as the XOR PUF. Another variant is the Multiplexer Arbiter PUF where the XOR is replaced by a multiplexer [18]. A

novel architecture relying on permutations of multiple delay lines like the Beli PUF [7] has been recently proposed. All these architectures significantly increase the security, especially with bigger size and composition of delay lines. But all the proposed PUFs can be attacked by learning bigger dataset and optimizing ML algorithms [8, 6]. For instance the attack on the interpose PUF splits the composed PUF in subsets to perpetrate a divide and conquer approach.

2.3 PUF Protocols

In the survey by Delvaux et al. [19], it is shown than most PUF-based authentication protocols use cryptographic block, True Random Number Generator (TRNG) or Error Correcting Code (ECC). These tools allow to get a higher level of robustness, even if in [20] it is shown that many PUF protocols are attackable by tailored ML. The significant drawback of these protocols result in an heavyweight implementation which is not suitable for low-cost products like RFID or IoT devices. There are very few lightweight protocols which do not require complex blocks at device level. The Lockdown technique [21] is one of the more robust and lightweight protocol as it allows the authentication process to bound the number of CRPs, hence thwarting the ML attacks by limiting the dataset.

2.4 Contribution of the Proposed Method

We consider that the security against ML attack is ensured by the multi-bin PUF architecture which uses NMQ as quantizer. The authentication protocol uses the Neyman-Pearson test to enhance the bad reliability provided by NMQ when the number of quantization levels is high. Hence, from the security point of view, it mainly relies on the PUF rather than the protocol. This latter is to allow the NMQ-PUF to work properly when it has a high level of security. Another strong requirement of the method is its lightweight property. The only addition of complexity is the quantization block which consists of comparators between the PUF internal response and constant threshold values.

The next section is dedicated to the security and reliability of NMQ, notably the impact of the quantization level and noise on these two properties.

3 Non-monotonic Quantization

In this section, we first introduce the basic definitions and notations in order to reason about PUFs and present NMQ and its specific properties of reliability and security.

3.1 Basic Notions

We will denote the set of challenges as $\mathcal{C} = \{C_1, C_2, \ldots, C_N\}$. The response of a PUF is calculated in two steps: First we measure the raw response (in our case

a delay difference) to a given challenge. The raw response is then quantized in a second step to a single bit response. The first step is modeled as a function $\mathcal{P}$ that maps a challenge to a delay difference value:

$$\mathcal{P} : C \mapsto \delta_C. \tag{1}$$

The raw response $\mathcal{P}$ can be modeled as a random [3] variable Δ following a normal distribution: $\Delta \sim \mathcal{N}(\mu, \Sigma^2)$. The standard deviation Σ depends on the underlying circuit technology. Note that the function $\mathcal{P}$ is an idealized model, since it does not take into account measurement noise. This raw response is then mapped to a single bit using some quantization function

$$b : \delta_C \mapsto \{0, 1\}. \tag{2}$$

We summarize both as the PUF's response

$$\mathcal{R}(C) = b(\mathcal{P}(C)). \tag{3}$$

A typical example for b is the sign of the raw response.

For a more realistic model, we need to take into account the noise, which will make the measurements of δ_C vary around their nominal values. Again, we model the noise as a random variable $Z \sim \mathcal{N}(0, \sigma^2)$. By adding the noise to the nominal output, we obtain a probabilistic version of our PUF model:

$$\hat{\mathcal{P}} : C \mapsto \delta_C + Z. \tag{4}$$

An important property of a PUF implementation is the ratio between the variance of the nominal response and the measurement noise. The higher the noise level with respect to the amplitude of the nominal response, the higher will be the probability to get a wrong response. We thus define the signal-to-noise ratio of the PUF as the ratio of the two variances:

$$\mathrm{SNR} = \frac{\Sigma^2}{\sigma^2}. \tag{5}$$

Finally, we define the *bit error rate* (BER) as the probability to obtain a wrong response, i.e. that the response differs from the nominal response:

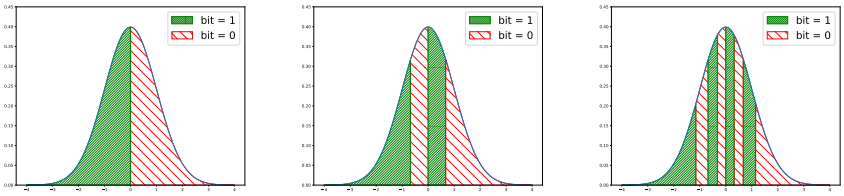
$$\mathrm{BER}(C) = \mathbb{P}\left(b(\mathcal{P}(C)) \neq b(\hat{\mathcal{P}}(C))\right). \tag{6}$$

3.2 NMQ Principle

As described above, in classic delay PUFs like the arbiter PUF, RO PUF, or Loop PUF, the response is based on whether a differential delay Δ is positive or

[3] Note that while the raw outputs δ_C follow a Gaussian distribution, $\mathcal{P}$ is not deterministic but it is considered as such at first, i.e. the environment is steady and the delays are fixed at fabrication time of the PUF. Section 6 will consider the impact of temperature and voltage.

negative, thus equivalent to 1-bit quantization. It has been shown in Section 2 that this simple decision-making process is vulnerable to ML attacks, even for composite PUF which are attackable by ML with a large dataset or a tailored ML. For multi-bin PUFs like RO PUF and Loop PUF, the raw output δ_C can be quantized on more than 1 bit, let us use Q as the quantization level. This allows the user to either increase the entropy of $\log_2(Q)$ bits or to keep a 1-bit entropy by using the non-monotonic quantization (NMQ) method. Figure 1 shows a typical quantization (equivalent to Q=2) and NMQ with Q=4 and 8.



(a) $Q = 2$. (b) $Q = 4$. (c) $Q = 8$.

Fig. 1: Different NMQ quantization levels

Given the raw response δ_C and an even quantization level Q, the non-monotonic quantization can be defined as

$$\mathrm{NMQ}_Q(\delta_C) = \begin{cases} 1, & \text{if } T_{2i-1} < \delta_C \leq T_{2i} \quad \text{for} \quad i \in \{1, \ldots, \frac{Q}{2}\} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

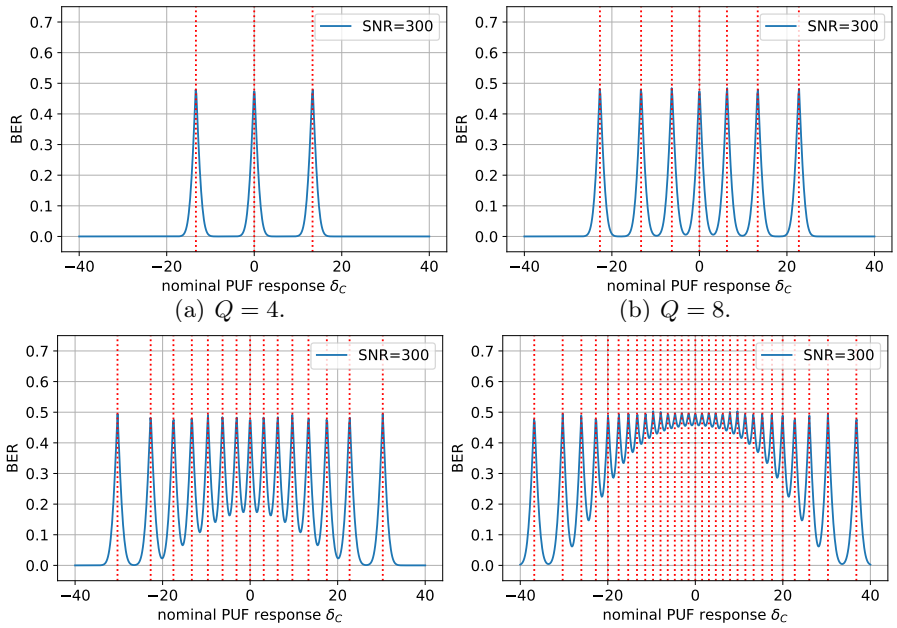
where the T_i are threshold values delimiting the quantiles of the distribution. In order to avoid bias, we want to use equiprobable quantiles. For example, for $Q = 4$, the thresholds can be determined as follows:

$$T_1 = \mu - 0.675\sigma, \quad T_2 = \mu, \quad T_3 = \mu + 0.675\sigma, \quad T_4 = \infty. \tag{8}$$

This new decision metric complicates the mapping between the challenge and the response, enhancing the resilience of PUFs against ML-based attacks.

3.3 Reliability of NMQ Implementation

In a PUF equipped with NMQ, its reliability tends to decrease as the quantization level Q increases. Figure 2 illustrates the BER of δ_C for different values of Q. The BER shown in the Figure has been calculated based on a simulated PUF using a SNR of 300, which corresponds to the SNR that we have measured for an FPGA implementation of a Loop PUF, and which is consistent with values from the literature [22]. We can see that the BER greatly increases for δ_C being close to the quantization thresholds. At higher numbers of Q, as the number of thresholds increases in $Q - 1$, the boundaries between quantization intervals become finer. Figure 3 shows the significant impact of SNR on the BER. The increased granularity associated with low SNR makes the PUF more sensitive



(a) $Q = 4$. (b) $Q = 8$.

(c) $Q = 16$. (d) $Q = 32$.

Fig. 2: BER estimation for different quantization levels and SNR = 300.

to noise when Q is high. This leads to an higher average BER, and thus lower reliability.

The reduced reliability at higher Q poses a challenge for the authentication process. With a higher BER, it becomes more difficult to distinguish between a legitimate device and an adversary, thus creating a significant rate of false negatives. Although increasing the number of quantization level can enhance security by introducing more complex response patterns, this comes at the cost of reliability. Therefore, there is a trade-off between achieving higher security and reliability.

3.4 Security of NMQ against Modeling Attacks

The question is to know if PUFs equipped with NMQ can really resist against the modeling attacks. The first study of NMQ [9] has shown that it is attackable by Convolutional Neural Network (CNN) with a low level of quantization. Increasing the quantization level increases the resistance but the PUF becomes unpracticable because of its very poor reliability. In order to assess the modeling attack resistance, we utilize Logistic regression (LR) [4], CNN, and multi-layer perceptron (MLP) as three attacking means[4].

[4] In this work, we do not differentiate the conventional modeling attacks (like LR, support vector machine, etc) with the deep learning-based attacks, since they typically act similarly in evaluating the security of PUFs.

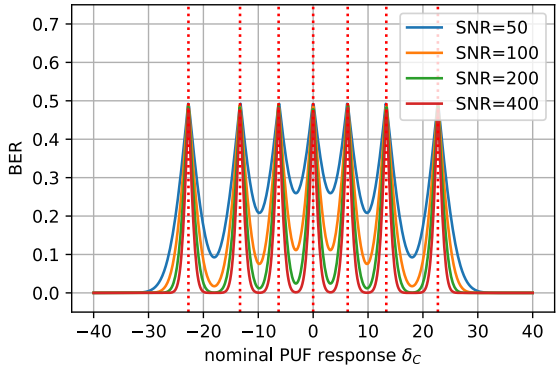


Fig. 3: BER estimation for $Q = 8$ and different noise levels

In the sequel, we consider NMQ running on a Loop PUF as an exemplary target with various NMQ parameter. We launch experiments on both simulated CRP dataset and real-world CRP dataset acquired on an FPGA implementation.

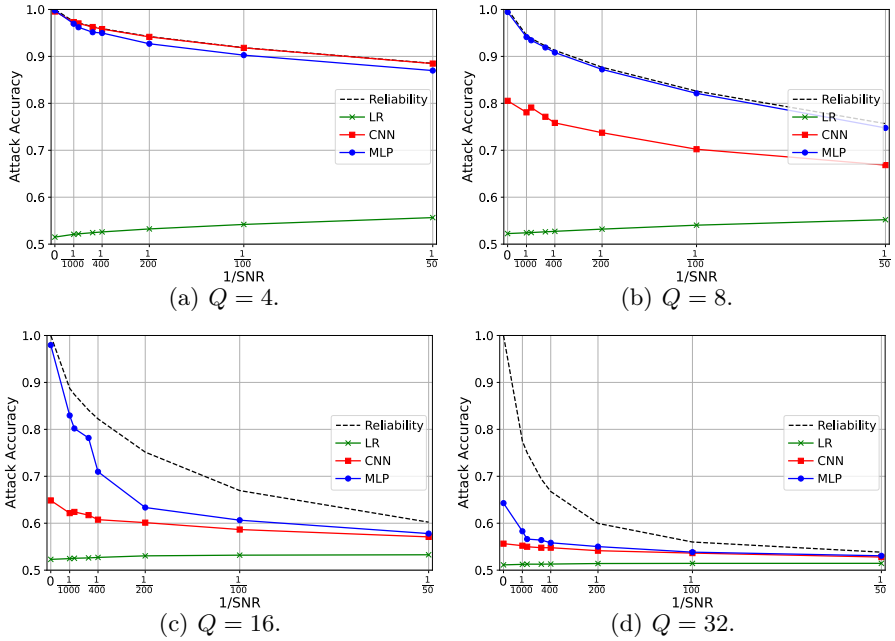
Modeling Attacks on Simulated NMQ-LPUFs

We first simulate RO-PUFs equipped with NMQ as introduced in Sec. 3.2. In addition, we also consider the PUF process is perturbed by the environmental and electronic factors by adding extra noise as in Equation 4. The main impact of this extra noise is the reliability of the PUF instance, which shall affect the learnable feature in CRPs and thus the attack accuracy of modeling attacks.

In this work, we have selected several noise levels to show their impact. In particular, we choose $\frac{1}{\mathrm{SNR}} \in \{0, \frac{1}{1000}, \frac{1}{800}, \frac{1}{500}, \frac{1}{400}, \frac{1}{200}, \frac{1}{100}, \frac{1}{50}\}$ (namely, SNR ranges from noiseless to 50). This range of SNR includes typical SNR of FPGA-based and ASIC PUF implementations that have an SNR between 50 and 300. For instance, SNR = 220 has been measured on a PUF implementation using 65nm CMOS technology [22], or SNR ≈ 50 when using a 28nm CMOS FD-SOI technology [23]. In our FPGA experiments, we measured a value of around 300. We set the total number of CRPs for the training phase to $400,000$ [5], which is enough to get stable results, and use another $100,000$ CRPs for the attack phase. Each experiment is repeated 10 times to reduce the numerical biases.

The results are shown in Fig. 4. We show the attack accuracy of the three modeling attacks compared to the corresponding reliability results at each noise level. The main takeaway is that with increasing of the NMQ parameter, the modeling attack resistance is largely increased, especially when $Q = 32$. Secondly, the noise can have a distinct impact on the learning ability of modeling attack methods, which are consistent for both CNN and MLP results. Notably, the best attack accuracy is, as expected, upper-bounded by the reliability of the corresponding noise level.

[5] We have tested for other dataset size for verification, while obtained similar results starting from around $200,000$ CRPs.



(a) $Q = 4$. (b) $Q = 8$.

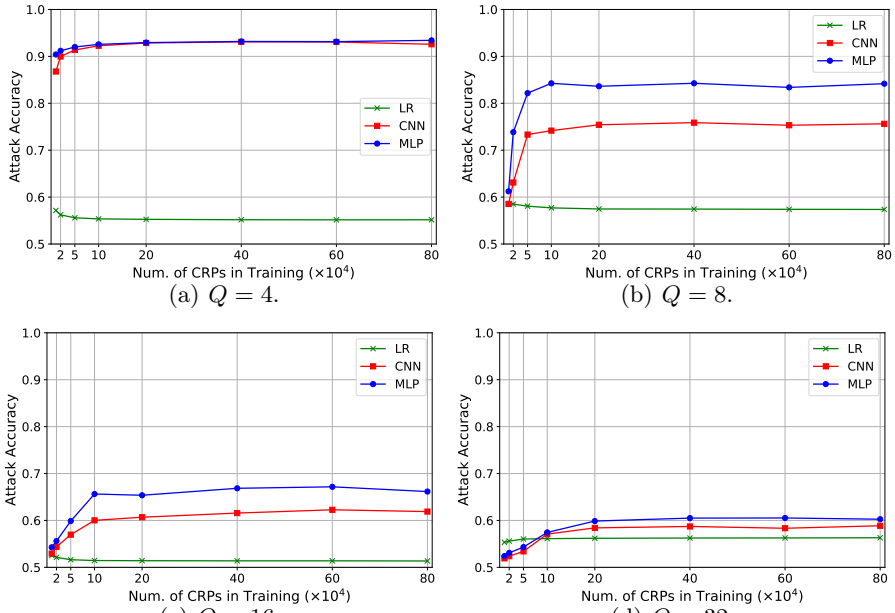
(c) $Q = 16$. (d) $Q = 32$.

Fig. 4: Modeling attack results against NMQ-LPUF with $Q \in \{4, 8, 16, 32\}$ in simulated scenario.

Modeling Attacks on FPGA-based NMQ-LPUFs

We next present the results on real-world FPGA-based PUF implementation. We implement NMQ-LPUFs on the Basys3 FPGA boards, which are based on the Xilinx Artix-7 28nm technology. It is worth noting that SNR $\approx$ 300 on the measured dataset containing 1,000,000 CRPs. The attack results are shown in Fig. 5 for $Q \in \{2, 4, 8, 16, 32\}$ where $Q = 2$ is selected as the baseline to demonstrate the security gains against the modeling attacks. Specifically, for each of LR, CNN and MLP, we vary the size of the training set from 10,000 to 800,000 CRPs and train the model with 50 epochs to show the impact of dataset size; while the attacking (test) set contains 100,000 CRPs (without overlapping with training set). We repeat each experiments by 10 times and then take the average to have more stable results.

Notably, the results from the real-world devices are overall consistent with the simulated data as shown in Fig. 4. From Fig. 5, there are two main takeaways: Firstly, NMQ can resist against LR-based modeling attacks even with quite small Q (e.g., $Q = 4$). Secondly, NMQ seems to show significantly enhanced resistance against CNN-based and MLP-based modeling attacks when taking large values of Q (e.g., $Q \geq 16$). The underlying reason is that NMQ discloses less significant information on the delay difference in deciding the output rather than the sign function in many ideal PUFs [4]. However, NMQ with smaller Q is still ineffective, which the designers should be careful to avoid for potential vulnerability against CNN-based and MLP-based modeling attacks.



(a) $Q = 4$. (b) $Q = 8$.

(c) $Q = 16$. (d) $Q = 32$.

Fig. 5: Modeling attack results against NMQ-LPUF with $Q \in \{4, 8, 16, 32\}$ implemented on an FPGA board.

4 Neyman-Pearson Test for PUF Authentication

To address the issue of higher BER at higher Q, we propose the application of the Neyman-Pearson test in the PUF protocol.

4.1 Principle

The Neyman-Pearson framework provides a statistically rigorous method to differentiate between a legitimate device and an adversarial device. It takes advantage of the BER knowledge for each challenge. By the Neyman-Pearson Lemma [24, Theorem 11.7.1], this test is uniformly most powerful at level α. In other words, for fixed level of security (probability to accept an adversary) this test will minimize the probability to reject a legitimate device.

The proposed protocol works by using a fixed number of challenges, say n, to authenticate a device and by using their associated BER. Leveraging the BER knowledge helps in a sense to discard the least reliable response and consider reliable responses in the test outcome while not disclosing to the adversary which challenge response pair is unreliable.

The Neyman-Pearson test aims to distinguish between two hypotheses: the null hypothesis H_0 and the alternative hypothesis H_1, while focusing on controlling the probabilities of two types of errors. This test maximizes the probability of correctly rejecting H_1 (i.e., the *power* of the test) for a given threshold, ensuring that false positives do not exceed. In the context of device authentication,

the Neyman-Pearson test can be used to distinguish between a legitimate device H_0 and an adversary device H_1. The attack efficiency is characterized by:

- The probability of rejecting the hypothesis that the device is legitimate (H_0) when it is actually legitimate: **false negative** or **missed authentication**).
- The probability of accepting the hypothesis that the device is illegitimate (H_1) when it is actually an adversary: **false positive** or **false authentication**.

4.2 Steps of the Neyman-Pearson Test

Four steps are required to carry out the test:

1. **Formulate the hypotheses**:

$$H_0 : \text{The device is legitimate,}$$
$$H_1 : \text{The device is an adversary.}$$

2. **Choose a threshold**: The threshold k defines the boundary for authentication process, it is chosen to yield a separation between the two hypotheses.
3. **Define the rejection region**: The test is based on the likelihood ratio:

$$\alpha = \frac{L(R|H_1)}{L(R|H_0)} = \frac{\prod_{i=0}^{n} \mathrm{BER}(C_i)^{e_i} \left(1 - \mathrm{BER}(C_i)\right)^{(1-e_i)}}{\left(\frac{1}{2}\right)^n} \tag{9}$$

   where $L(R|H_0)$ and $L(R|H_1)$ are the likelihood functions under the hypotheses H_0 (legitimate device) and H_1 (adversary device), respectively. Here, R represents the n responses from the device, and the likelihood ratio α is computed to compare the hypotheses.
   In the formula, $\mathrm{BER}(C_i)$ represents the BER for the i-th challenge, and e_i denotes the observed error for the i-th challenge. The test rejects H_0 if the likelihood ratio α is less than threshold k. In particular, errors on reliable challenge will be heavily penalized while errors on unreliable challenges are less significant (in the extreme case where a bit error rate is maximal equals to $\frac{1}{2}$ an error is completely ignored by the test).
4. **Make a decision**: Based on the observed data R, if $\alpha < k$, reject H_0 (consider the device as adversarial); otherwise, do not reject H_0 (consider the device as legitimate).

Overall, this procedure mitigates the high bit error rates of the challenges as long as some challenges are reliable.

4.3 Design of the PUF Authentication Protocol

The protocol is represented in Fig. 6. The enrollment phase is to prepare the Neyman-Pearson test by modeling the PUF and BER. The authentication phase is to check the two hypotheses of the Neyman-Pearson test.

The two phases are described below in a detailed manner:

Server		**Device**
Enrollment Phase		
Generate C_H	$\xrightarrow{C_H, i}$	
$D_i = C_{Hi}^{-1}.\delta_{di}$ Store D_i	$\xleftarrow[\delta_{di}]{}$	$\delta_d = PUF(C_H)$
Generate C_{ref}	$\xrightarrow{C_{ref}, j}$	
$std(noise) = \sigma(\delta_{ref})$ Store $std(noise)$	$\xleftarrow[\delta_{ref}]{}$	$\delta_{ref} = PUF(C_{ref})$
Authentication Phase		
Generates C_{random} and select Q	$\xrightarrow{C_{random}, Q}$	
$\delta_C = PUF_{model}(C_{random})$ $R_s = NMQ_Q(\delta_C)$		
		$\delta_c = PUF(C_{random})$ $R_d = NMQ_Q(\delta_C)$
$e_i = \begin{cases} 0, & \text{if } R_d = R_s \\ 1, & \text{if } R_d \neq R_s \end{cases}$	$\xleftarrow[R_d]{}$	
where $i = 1, 2, 3, \ldots, N$ $BER_i(\delta_C) = \mathbf{P}_{err}(C_i)$ $\alpha = NP_{test}(BER, e, N)$		
if $\alpha < k$, then $\begin{cases} \text{True, Authenticated,} \\ \text{False, Unauthenticated.} \end{cases}$		
where $k = Threshold$	$\xrightarrow{Auth/Unauth}$	

Fig. 6: Neyman-Pearson PUF authentication protocol: the enrollment and the authentication.

1. Enrollment Phase During the enrollment phase, the server collects the responses of Hadamard challenges. The Hadamard challenges are used because their response are theoretically uncorrelated [25, 26] and the delay model D of the PUF can be easily reconstructed by a matrix inversion. This approach allows

the server to calculate responses for any future challenges without storing a big set of CRPs. Furthermore, the Hadamard challenges can be used multiple times to reduce the effect of noise and create a more accurate model by averaging the raw responses δ_{c_H}.

To account for the impact of noise during the enrollment phase, the responses are sampled multiple (j) times during the enrollment phase[6]. Additionally, the BER model is created which is essential for the following Neyman-Pearson test. This BER model estimates the error rates for all challenges, quantifying the likelihood of errors due to noise, environmental variations, or quantization effects. The noise's standard deviation σ is calculated using a reference challenge and stored for use during the authentication phase. It is used to estimate the BER for each challenge at a given quantization level.

2. Authentication Phase The server sends a set of n random challenges at a given quantification Q to the device. The device generates responses R_d using the PUF. The server, utilizing its stored PUF model, computes the expected raw responses δ_C and quantized responses R_s for the same set of random challenges. By comparing R_d and R_s, it identifies the error e_i for each challenge, creating an error vector e. The BER model is then used to estimate the probability of error $\mathbf{P}_{err}(C_i)$. The server subsequently applies the Neyman-Pearson test using the calculated BER for each challenge and the observed errors e from the comparison between the device's PUF responses and the expected responses from the server's PUF model. The likelihood ration α is computed and compared with the predefined threshold k. If $\alpha < k$, the device is authenticated as legitimate; otherwise, the device is flagged as an adversary.

This PUF authentication protocol offers the following properties:

- The application of the Neyman-Pearson Lemma reduces the likelihood of false positives and false negatives by carefully choosing the threshold k and the number of challenges n. k guarantees the best trade-off between the false positives and false negatives, whereas n decreases both.
- The protocol increases the security for a high number of quantization level, thanks to NMQ.

4.4 PUF Delay and BER Models

The protocol requires at server side an accurate delay model and BER model. The build of the models is described below.

Delay modeling

[6] We assume that the enrollment of the PUF takes place in a controlled environment, minimizing temperature and voltage variations as well as external electro-magnetic radiation.

The responses obtained from the PUF for the Hadamard challenges are then used to calculate the delays of the physical elements in the PUF. For the delay PUF like the Loop PUF, the delay chain consists of M delay elements and the challenge is M bits long. Hence M Hadamard challenges and their corresponding responses are used. The delay model of the M elements can be computed using matrix multiplication, where the inverse of the matrix of M Hadamard challenges is multiplied by the vector of recorded responses. The delay for the i-th element is given by:

$$D_i = C_i^{-1} \cdot R_i. \tag{10}$$

where C_i^{-1} is the inverse of the Hadamard challenge matrix, and R_i is the vector of responses for the i-th challenge. This method allows the server to model the delay characteristics of the PUF accurately and use this model for future challenge-response calculations without needing to store all CRPs.

BER Modeling

As shown in Fig. 7, the BER requires first the delay model and the noise model of the device. Then the quantization is executed according to the quantization level Q.

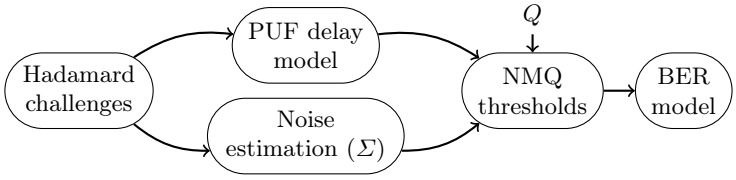


Fig. 7: BER estimation for Neyman-Pearson PUF Authentication Protocol

Noise model: At the enrollment phase, the server gathers several raw responses δ_{ref} for reference challenge C_{ref} to estimate the noise affecting the PUF. The standard deviation of the noise, namely σ(noise), is calculated based on these repeated observations.

More formally, we have

$$\sigma(\text{noise}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\delta_{ref_i} - \mu_{ref}\right)^2} \quad . \tag{11}$$

where:

- δ_{ref_i} is the raw response to a reference challenge in ith iterations,
- μ_{ref} is the mean response over N iterations,
- N is the number of iterations or repeated raw responses collected.

By storing σ(noise) at the enrollment phase, the system can later predict how much variation is expected in the responses during the authentication phase.

BER model: During the authentication phase, the BER is estimated by analyzing the probability of error in a response derived from the server-side PUF model, denoted as δ_s, when subjected to noise characterized by a standard deviation σ(noise).

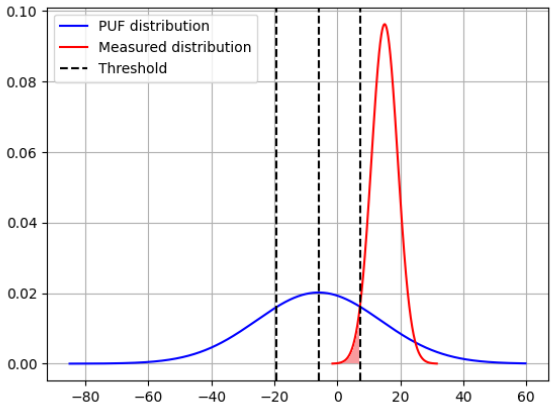


Fig. 8: BER estimation for a single challenge with $Q = 4$.

Figure 8 visualizes the relationship between the distributions of the PUF response values (in blue) and the measured distribution at δ_s (in red). The blue curve represents the distribution of the responses from the server-side PUF model. These values can vary slightly around a central value due to natural variations in the PUF's physical characteristics. The distribution is modeled as a normal distribution as follows.

$$pdf(\delta) = \mathcal{N}(0, \Sigma^2). \tag{12}$$

where Σ^2 represents the variance of the PUF response values. The red curve represents the measured response, which is the actual PUF response corrupted by noise. The noise is modeled as by a normal distribution as $\mathcal{N}(0, \sigma^2)$.

The error probability (shaded region) refers to the likelihood that the measured response is different from the actual PUF response. If the response is close to a threshold (the vertical dotted line), this likelihood increases. If the measured response lies in the overlapping region between the two curves, there is a higher chance of an error. The larger the overlap between the PUF response distribution and the measurement distribution, the higher the BER.

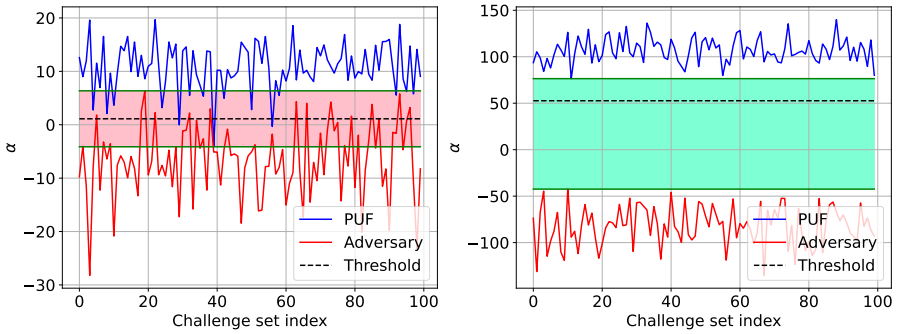
5 Results and Observations

5.1 Experimental Setup

The experiments were conducted on sixteen Basys3 FPGA boards, which are based on the Xilinx Artix-7 28nm technology. As a multi-bin delay PUF, we used a Loop PUF [12] with a delay chain of 64 elements and a δ_C response of 16 bits.

5.2 Neyman-Pearson PUF Authentication Protocol: Safety Window for Threshold

The safety window in the Neyman-Pearson PUF authentication protocol refers to the gap between the α-PUF and α-adversary values. This window plays a crucial role in distinguishing between legitimate devices and adversaries during authentication. A larger safety window indicates a significant difference between the responses of the PUF and the adversary, reducing the likelihood of wrong authentication. When the safety window is wide, the system can more reliably differentiate between legitimate responses and random guesses made by adversaries, even in the presence of noise or environmental variations. Conversely, a narrow safety window increases the risk of false positives or false negatives, as the adversary's response values may be too close to the legitimate PUF values.



(a) $n = 50$ (b) $n = 500$

Fig. 9: Neyman-Pearson PUF authentication protocol with $Q = 16$: the legitimate PUF vs the adversary.

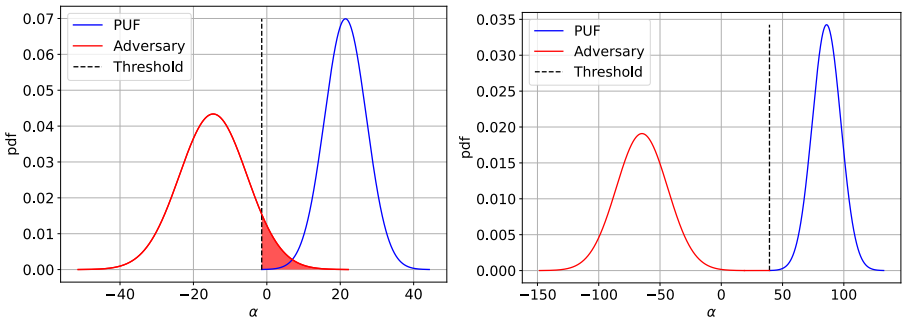
Figure 9 shows the fluctuation in α-PUF and α-adversary due to system noise and physical factors, and predicts a region of safety to define a threshold. The green region in each graph represents the safety window, within which a threshold can be defined based on security requirements. The threshold is set closer to α-PUF to minimize the risk of false positives, which occur when an adversary is mistakenly accepted as a legitimate PUF. It can be observed that with a small number of challenges, it is difficult to distinguish between the PUF and the adversary as shown in Fig. 9(a). Therefore, it is crucial to choose a sufficient number of challenges for secure and reliable authentication. Fig. 9(b) shows that a safety window (in green) is obtained with more challenges.

However, the safety window decreases with an increase in the quantization level, requiring more challenges to maintain the protocol's reliability.

5.3 False Authentication Probabilities

Figure 10 represents two probability distributions in the context of an authentication protocol. The safety window depends on the proximity of these two distributions. Below is a breakdown of the components and what they represent:

- **Blue Curve (PUF)**: This curve represents the probability distribution of responses from a legitimate PUF. In an authentication scenario, these responses are from a trusted device or system.
- **Red Curve (Adversary)**: This curve represents the probability distribution of random responses from an adversary.
- **Threshold (Black Dashed Line)**: The vertical line represents the decision boundary or threshold. Responses to the right of this line are classified as legitimate, while those to the left are classified as adversarial.
- **Shaded Region (Red)**: The shaded region represents the *False Positive Rate (FPR)*. It corresponds to the portion of the adversary's responses (red curve) that are incorrectly classified as legitimate responses.



(a) Using less challenges (b) Using more challenges

Fig. 10: Neyman-Pearson PUF Authentication Protocol: False Authentication Probability

In Fig. 10(a), the false positive region is the area under the red curve to the right of the threshold, indicated by the black dashed line. The threshold is chosen such that there is no false negative for all the challenge sets. Thus the false negative rate is less than the inverse of the number of sets. This shaded region highlights the probability that the adversary's response, despite originating from a non-legitimate source, is incorrectly identified as a valid response by the server. The larger the shaded region, the higher the likelihood that the adversary will be able to deceive the authentication server. In particular, Fig. 10(b) shows a very small false positive region, which implies a very low probability of incorrectly accepting the adversary as legitimate.

Table 1 provides the range false authentication probability for different numbers of challenges and various quantization levels Q in an authentication protocol using 16 PUF devices. The rows represent different *numbers of challenges* (e.g., n equals 50, 100, 150, etc.), while the columns represent different *quantization levels* $Q \in \{4, 8, 16, 32\}$. Table 1 shows that the false authentication probability generally decreases as the number of challenges increases, while as the quantization level increases, the false authentication probability increases, highlighting the trade-off between security and complexity.

Table 1: Range of false authentication probability for 16 different PUFs.

Number of challenges	Q = 4 (Avg Rel: 93.7%)	Q = 8 (Avg Rel: 86.6%)	Q = 16 (Avg Rel: 72.5%)	Q = 32 (Avg Rel: 60.5%)
50	0.008 - 0.055	0.052 - 0.119	0.115 - 0.268	0.220 - 0.361
100	0.000 - 0.011	0.006 - 0.035	0.046 - 0.140	0.159 - 0.315
150	$3.605 \cdot 10^{-5}$ - 0.001	0.002 - 0.012	0.016 - 0.104	0.073 - 0.255
200	$1.345 \cdot 10^{-6}$ - $3.134 \cdot 10^{-4}$	0.001 - 0.004	0.010 - 0.068	0.050 - 0.211
250	$1.220 \cdot 10^{-5}$ - $2.509 \cdot 10^{-4}$	$2.527 \cdot 10^{-4}$ - 0.002	0.006 - 0.044	0.034 - 0.168
300	$1.090 \cdot 10^{-9}$ - $2.285 \cdot 10^{-7}$	$1.006 \cdot 10^{-5}$ - 0.001	0.002 - 0.034	0.036 - 0.134
350	$2.125 \cdot 10^{-10}$ - $2.484 \cdot 10^{-7}$	$1.250 \cdot 10^{-5}$ - $1.291 \cdot 10^{-4}$	0.001 - 0.029	0.024 - 0.129
400	$1.221 \cdot 10^{-6}$ - $1.781 \cdot 10^{-5}$	$3.727 \cdot 10^{-6}$ - $8.169 \cdot 10^{-4}$	0.001 - 0.015	0.005 - 0.105
450	$1.607 \cdot 10^{-9}$ - $3.337 \cdot 10^{-7}$	$1.478 \cdot 10^{-6}$ - $1.986 \cdot 10^{-4}$	$2.252 \cdot 10^{-4}$ - 0.007	0.006 - 0.079
500	$1.004 \cdot 10^{-10}$ - $1.195 \cdot 10^{-7}$	$4.542 \cdot 10^{-6}$ - $8.002 \cdot 10^{-6}$	$2.105 \cdot 10^{-4}$ - 0.004	0.004 - 0.076
550	$1.021 \cdot 10^{-13}$ - $4.862 \cdot 10^{-10}$	$1.579 \cdot 10^{-8}$ - $2.908 \cdot 10^{-7}$	$1.812 \cdot 10^{-5}$ - 0.001	0.004 - 0.065
600	0.0 - $1.519 \cdot 10^{-12}$	$1.014 \cdot 10^{-8}$ - $1.576 \cdot 10^{-5}$	$2.435 \cdot 10^{-5}$ - 0.001	0.004 - 0.052

For each number of challenges, the false authentication probability generally *increases* as the quantization level Q increases. This suggests that higher values of Q require higher number of challenges for authentication but it provides better security against ML attacks. For small numbers of challenges (e.g., 50 challenges), the false authentication probabilities are higher for higher values of Q, meaning the authentication server is more vulnerable to attacks. As the number of challenges increases, the false authentication probability tends to decrease across all quantization levels. For example, with 400 challenges, the false authentication probability is as low as 2.4-4.3% for $Q = 32$, and with 600 challenges, it decreases further to 0.1-1.6%.

6 Discussions

6.1 Comparison with Other Protocols

Contrary to protocols of the literature which are to increase the security against ML attacks, the proposed protocol relying on the Neyman-Pearson test is to increase the reliability, as the security is ensured by NMQ. As far as we know, the only lightweight security protocol against ML used in the literature is the *Lockdown protocol* [21]. This countermeasure has a relatively low complexity (PUF + PRNG). The countermeasure in our study is the PUF which takes advantage of the NMQ quantization. The complexity added by NMQ is very low (comparators for quantization stage). This NMQ-PUF is particularly unreliable when the security is high. The goal of the proposed protocol is to enhance its reliability. Hence, the Lockdown protocol can be used jointly with the Neyman-Pearson test. Table 2 summarized the main properties of 3 scenarios according to the criteria of security, reliability and hardware complexity. Concerning the reliability, it is indicated if it is exploited in order to strenghten the authentication process.

Table 2: Comparison of 3 different scenarios.

	Lockdown	This work: NMQ + Neyman-Pearson	Lockdown + Neyman-Pearson
Security	proven	high with high Q	proven
Reliability	not exploited	exploited	exploited
Complexity	PUF + PRNG	multi-bin PUF + quantizer	PUF + PRNG

6.2 Impact of Environmental Changes

The delays δ_C of a delay PUF are greatly impacted by the temperature, the voltage and the device aging [27]. Consequently, the quantization thresholds of NMQ have to be adapted as their values is the standard deviation Σ of the δ_C multiplied by constant numbers. A first scenario is to use a specific process between the PUF and the server to assess the new environment and consequently update the thresholds. A simpler solution when using a multi-bin PUF based on ring oscillators (ROs), is to use a reference clock which follows the same environmental changes. The PUF response corresponds to a difference of oscillations between two configurations of the RO (Loop PUF) or two different ROs (RO PUF). Let's call T_{osc1} and T_{osc2} the oscillation period of each RO, respectively. The measurement window lasts K oscillations of each RO configuration, K is a constant. During this time window, a counter is incremented with a clock having a period T. At the end of the measurement the counter reaches respectively n_1 and n_2. Hence the PUF response before quantization is $n_1 - n_2$ and can be written:

$$K \cdot T_{osc1} = n_1 \cdot T, \qquad K \cdot T_{osc2} = n_2 \cdot T \tag{13}$$

$$\Rightarrow \quad n_1 - n_2 = \frac{K}{T} \cdot (T_{osc1} - T_{osc2}) \tag{14}$$

If there is an environmental change with a factor α on T_{osc1} and T_{osc2}, the response is changed to $n_1 - n_2 = \alpha \cdot \frac{K}{T}(T_{osc1} - T_{osc2})$. Consequently the thresholds have to be updated. Now, if the clock period T comes from another ring oscillator which is also impacted by a factor α, the clock period becomes αT and the response stays the same whatever the environment. This solution assumes that all the elements of the PUF are impacted in the same way and that the noise remains stable over the measurement time. More experiments will be carried out to verify this hypothesis in a future study.

6.3 Scalability

The Neyman-Pearson test requires the knowledge of the BER for every challenge in order to be fully scalable. Thanks to the Hadamard challenges used during the enrollment phase, the BER model build on the server is fully generic. During the authentication phase, the BER for a given challenge is computed from this model. Therefore, the server does not need to store the BER for every challenge, thus ensuring scalability. On the client side (the PUF), no additional computation is needed.

6.4 ML Techniques with Reliability as Feature

If the considered adversary model is such that the PUF interface is open and the attacker can replay the CRPs or try new challenges, the reliability of the CRPs can be measured and the ML could use it as a feature. It has been shown that the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [15] is particularly efficient to attack the XOR-PUF by using the reliability information. More recently the gradient-based ML attack [28] has shown to be even more powerful against the interpose-PUF. In case of the NMQ-PUF, the reliability-based attacks should definitely be considered but they are not straightforward to implement in a multi-threshold PUF. They are parts of future works.

6.5 Feasibility for IoT applications

The main properties to meet for the IoT domain are the cost, real-time and low-power. The cost, or hardware complexity, remain low for the NMQ-PUF as the only add-on is the quantizer with multiple thresholds. The real-time constraint may be more difficult to respect as a multi-bin PUF generally relies on a ring oscillator (RO-PUF) and could require at most $1ms$ to output a response. Hence the authentification process can last a few $100ms$. The multi-bin PUF relying on Ring oscillators is not optimal in terms of power consumption compared to arbiter PUF. However, in addition to the low-power RO-PUF proposed in the literature as [29, 30], the authentication process remains a relatively short operation with a limited energy consumption when the PUF is in standby mode after authentication.

7 Conclusion

Non-Monotonic Quantization strengthens the resistance of multi-bin delay PUFs such as RO-PUFs against ML attacks by introducing non-linearity at the quantization stage. However, the price to pay for this security increase is a high drop of reliability. By using the Neyman-Pearson Lemma in the PUF protocol, we show that the native unreliability caused by the NMQ quantization can be tackled. More precisely, the unreliability is exploited to help distinguishing between legitimate and adversarial devices. We validated the efficiency of an authentication protocol relying on both an NMQ-PUF and the Neyman-Pearson test. The results obtained with a very high number of quantization levels display a perfect resistance against ML, thanks to NMQ and noise, and an authentication success without failure, thanks to the Neyman-Pearson test. Future works will be to validate the efficiency of the protocol to face environmental changes and to consider new approaches of reliability based attacks.

References

1. G. Edward Suh and Srinivas Devadas. Physical unclonable functions for device authentication and secret key generation. In *Proceedings of the 44th annual Design*

Automation Conference, DAC '07, pages 9–14, New York, NY, USA, June 2007. Association for Computing Machinery.

2. Roel Maes and Ingrid Verbauwhede. Physically unclonable functions: A study on the state of the art and future research directions. *Towards Hardware-Intrinsic Security: Foundations and Practice*, pages 3–37, 2010.
3. Blaise Gassend, Daihyun Lim, Dwaine Clarke, Marten Van Dijk, and Srinivas Devadas. Identification and authentication of integrated circuits. *Concurrency and Computation: Practice and Experience*, 16(11):1077–1098, 2004.
4. Ulrich Rührmair, Frank Sehnke, Jan Sölter, Gideon Dror, Srinivas Devadas, and Jürgen Schmidhuber. Modeling attacks on physical unclonable functions. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 237–249, Chicago Illinois USA, October 2010. ACM.
5. Phuong Ha Nguyen, Durga Prasad Sahoo, Chenglu Jin, Kaleel Mahmood, Ulrich Rührmair, and Marten Van Dijk. The interpose puf: Secure puf design against state-of-the-art machine learning attacks. *Cryptology ePrint Archive*, 2018.
6. Anita Aghaie, Amir Moradi, Johannes Tobisch, and Nils Wisiol. Security analysis of delay-based strong pufs with multiple delay lines. In *2022 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 125–128. IEEE, 2022.
7. Nils Wisiol. Beli puf. In *Modeling Attack Security of Physical Unclonable Functions based on Arbiter PUFs*, pages 79–87. Springer, 2023.
8. Nils Wisiol, Christopher Mühl, Niklas Pirnay, Phuong Ha Nguyen, Marian Margraf, Jean-Pierre Seifert, Marten Van Dijk, and Ulrich Rührmair. Splitting the interpose puf: A novel modeling attack strategy. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 97–120, 2020.
9. Kleber Stangherlin, Zhuanhao Wu, Hiren Patel, and Manoj Sachdev. Enhancing Strong PUF Security With Nonmonotonic Response Quantization. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 31(1):55–64, January 2023. Conference Name: IEEE Transactions on Very Large Scale Integration (VLSI) Systems.
10. Vincent Immler, Matthias Hiller, Qinzhi Liu, Andreas Lenz, and Antonia Wachter-Zeh. Variable-length bit mapping and error-correcting codes for higher-order alphabet pufs—extended version. *Journal of Hardware and Systems Security*, 3:78–93, 2019.
11. EL Lehmann. Introduction to neyman and pearson (1933) on the problem of the most efficient tests of statistical hypotheses. *Breakthroughs in Statistics: Foundations and Basic Theory*, pages 67–72, 1992.
12. Zouha Cherif, Jean-Luc Danger, Sylvain Guilley, and Lilian Bossuet. An Easy-to-Design PUF Based on a Single Oscillator: The Loop PUF. In *2012 15th Euromicro Conference on Digital System Design*, pages 156–162, Cesme, Izmir, Turkey, September 2012. IEEE.
13. Jae W Lee, Daihyun Lim, Blaise Gassend, G Edward Suh, Marten Van Dijk, and Srinivas Devadas. A technique to build a secret key in integrated circuits for identification and authentication applications. In *2004 Symposium on VLSI Circuits. Digest of Technical Papers (IEEE Cat. No. 04CH37525)*, pages 176–179. IEEE, 2004.
14. Daihyun Lim, Jae W Lee, Blaise Gassend, G Edward Suh, Marten Van Dijk, and Srinivas Devadas. Extracting secret keys from integrated circuits. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 13(10):1200–1205, 2005.

15. Georg T Becker. The gap between promise and reality: On the insecurity of xor arbiter pufs. In *Cryptographic Hardware and Embedded Systems–CHES 2015: 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings 17*, pages 535–555. Springer, 2015.
16. Pranesh Santikellur, Aritra Bhattacharyay, and Rajat Subhra Chakraborty. Deep learning based model building attacks on arbiter puf compositions. *Cryptology ePrint Archive*, 2019.
17. Gabriel Hospodar, Roel Maes, and Ingrid Verbauwhede. Machine learning attacks on 65nm arbiter pufs: Accurate modeling poses strict bounds on usability. In *2012 IEEE international workshop on Information forensics and security (WIFS)*, pages 37–42. IEEE, 2012.
18. Durga Prasad Sahoo, Debdeep Mukhopadhyay, Rajat Subhra Chakraborty, and Phuong Ha Nguyen. A multiplexer-based arbiter puf composition with enhanced reliability and security. *IEEE Transactions on Computers*, 67(3):403–417, 2017.
19. Jeroen Delvaux, Roel Peeters, Dawu Gu, and Ingrid Verbauwhede. A survey on lightweight entity authentication with strong pufs. *ACM Computing Surveys (CSUR)*, 48(2):1–42, 2015.
20. Jeroen Delvaux. Machine-learning attacks on polypufs, ob-pufs, rpufs, lhs-pufs, and puf–fsms. *IEEE Transactions on Information Forensics and Security*, 14(8):2043–2058, 2019.
21. Meng-Day Yu, Matthias Hiller, Jeroen Delvaux, Richard Sowell, Srinivas Devadas, and Ingrid Verbauwhede. A lockdown technique to prevent machine learning on pufs for lightweight authentication. *IEEE Transactions on Multi-Scale Computing Systems*, 2(3):146–159, 2016.
22. Alexander Schaub, Jean-Luc Danger, Sylvain Guilley, and Olivier Rioul. An Improved Analysis of Reliability and Entropy for Delay PUFs. In *2018 21st Euromicro Conference on Digital System Design (DSD)*, pages 553–560, August 2018.
23. Jean-Luc Danger, Risa Yashiro, Tarik Graba, Yves Mathieu, Abdelmalek Si-Merabet, Kazuo Sakiyama, Noriyuki Miura, and Makoto Nagata. Analysis of mixed puf-trng circuit based on sr-latches in fd-soi technology. In *2018 21st Euromicro Conference on Digital System Design (DSD)*, pages 508–515, 2018.
24. Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006.
25. Olivier Rioul, Patrick Solé, Sylvain Guilley, and Jean-Luc Danger. On the entropy of physically unclonable functions. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 2928–2932. IEEE, 2016.
26. Patrick Solé, Wei Cheng, Sylvain Guilley, and Olivier Rioul. Bent sequences over hadamard codes for physically unclonable functions. In *IEEE International Symposium on Information Theory, ISIT 2021, Melbourne, Australia, July 12-20, 2021*, pages 801–806. IEEE, 2021.
27. Trevor Kroeger, Wei Cheng, Sylvain Guilley, Jean-Luc Danger, and Naghmeh Karimi. Effect of aging on PUF modeling attacks based on power side-channel observations. In *2020 Design, Automation & Test in Europe Conference & Exhibition, DATE 2020, Grenoble, France, March 9-13, 2020*, pages 454–459. IEEE, 2020.
28. Johannes Tobisch, Anita Aghaie, and Georg T Becker. Combining optimization objectives: New modeling attacks on strong pufs. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 357–389, 2021.
29. Yuan Cao, Le Zhang, Chip-Hong Chang, and Shoushun Chen. A low-power hybrid ro puf with improved thermal stability for lightweight applications. *IEEE Transac-*

tions on computer-aided design of integrated circuits and systems, 34(7):1143–1147, 2015.
30. Sauvagya Ranjan Sahoo, Sudeendra Kumar, Kamalakanta Mahapatra, and Ayaskanta Swain. A novel aging tolerant ro-puf for low power application. In *2016 IEEE international symposium on nanoelectronic and information systems (iNIS)*, pages 187–192. IEEE, 2016.

A Neural Network Structure and Parameters

In Tab. 3, we detail the network structure of our CNN and MLP models against NMQ-LPUF in Sec. 3.4. In addition, we implemented an adaptive loss function that is halved by every 10 epochs starting from 0.01 (which shows better attack performances among several settings).

Table 3: Network architecture of CNN/MLP models, and training parameters.

Network	Architecture	Epochs	Batch Size
CNN	Conv1D(16, 64) BatchNormalization Dropout(0.2) Conv1D(10, 64) BatchNormalization Dropout(0.2) Flatten() Dense(32) Dropout(0.2) Dense(2): Output layer	50	1000
MLP	Flatten() Dense(64) Dropout(0.2) Dense(32) Dropout(0.2) Dense(2): Output layer	50	1000