Countering PUF Modeling Attacks through Adversarial Machine Learning

Mohammad Ebrahimabadi, Wassila Lalouani, Mohamed Younis, and Naghmeh Karimi
CSEE Department, University of Maryland Baltimore County, Baltimore, MD 21250
Email:{e127, lwassil1, younis, nkarimi}@umbc.edu

***Abstract*— A Physically Unclonable Function (PUF) is an effective option for device authentication, especially for IoT frameworks with resource-constrained devices. However, PUFs are vulnerable to modeling attacks which build a PUF model using a small subset of its Challenge-Response Pairs (CRPs). We propose an effective countermeasure against such an attack by employing adversarial machine learning techniques that introduce errors (poison) to the adversary's model. The approach intermittently provides wrong response for the fed challenges. Coordination among the communicating parties is pursued to prevent the poisoned CRPs from causing the device authentication to fail. The experimental results extracted for a PUF implemented on FPGA demonstrate the efficacy of the proposed approach in thwarting modeling attacks. We also discuss the resiliency of the proposed scheme against impersonation and Sybil attacks.**

I. Introduction

The Internet of Things (IoT) extends the scope of communication and data exchange from servers and personal computers to the objects used in everyday life. However, such increased interconnectivity has magnified the concern about the cyber attacks. Thereby, robust schemes are needed for supporting authentication, data integrity and confidentiality. Given the large number and diversity of devices in an IoT framework, ensuring security is more challenging than in the traditional networks. Security attacks on IoT devices include forcing malicious malfunctions, denial of service, and leaking sensitive information. IoT is a collection of low cost and resource-constrained devices operating in unsupervised environments. To protect IoT frameworks, developing authentication and key management protocols that utilize lightweight cryptography and low-cost tamper-resistant primitives is highly required.

Physically Unclonable Functions (PUFs) generate a unique signature for each IoT device. A PUF's signature corresponds to its input and output pairs, so-called Challenge-Response Pairs (CRPs) [1]. A PUF is embedded in each device during the fabrication process, and a subset of its CRPs are registered once after the device fabrication and during the enrollment phase. These CRPs are then used during operation to authenticate the device [2]. PUFs exploit the inherent physical variations of devices during the manufacturing process to generate a unique signature for the underlying device [1], and avoid storing the device signature in memory.

Although supposed to be unclonable, a PUF behavior may be modeled using Machine Learning (ML) techniques [3], where an adversary opts to devise a behavior model based on a limited subset of the PUF's CRPs. This model is then used to predict the PUF response to any unseen challenges. The attack accuracy depends on the used ML scheme, the target PUF's type, and the number of CRPs the adversary can get access to.

PUFs are classified into strong and weak, based on their CRP space [4]. The former are often used for authentication protocols, while weak PUFs are deemed suitable for generating cryptographic keys [5]. The delay-PUFs and in particular the arbiter-PUFs family (e.g., arbiter-PUF, feed-forward-PUF, XOR-PUF) are deemed the most popular in the strong PUF category. However, recent research studies have shown that even a 64-bit arbiter-PUF can be successfully modeled by ML schemes [6]. The feed-forward and XOR-PUFs that were considered as modeling resistant arbiter-PUF counterparts beforehand, have been also compromised recently using ML [7].

One can benefit from adversarial ML models to increase the resiliency of PUFs against modeling. Conventionally, in adversarial models some of the ML input is poisoned, i.e., intentionally made erroneous, in order to cause inaccuracy. In this paper, we exploit such a concept to make the PUFs secure against modeling attacks, where the communicating nodes follow adversarial models and intermittently send *poisoned CRPs (i.e., the challenge bit-stream along with an incorrect response)* to decrease the accuracy of the PUF modeling attempts made by a malicious eavesdropper. We propose an adversarial ML scheme that deliberately poisons the CRPs transferred between IoT nodes and a server by intermittently toggling the response bit for some challenges before sending to the server. *The adversary will not be able to differentiate the poisoned (i.e., fake) CRPs from the genuine ones, and thereby fails to launch replay and impersonation attacks. On the other hand, the server is made aware of this poisoning so it can still authenticate the device. The agreement between the server and IoT device is made during the IoT node enrolment in the network.* Our contributions are as follows:

- Developing a novel adversarial scheme to prevent ML-based modeling attacks on PUF-based IoT devices;
- Analyzing the resiliency of the IoT framework against Sybil and impersonation attacks when the proposed scheme is deployed;
- Investigating the impact of the poisoned CRPs on the success of the PUF modeling attacks launched via state-of-the-art ML schemes;
- Studying the overhead and resiliency of the proposed scheme for different configurations;
- Evaluating the proposed method using the data extracted from FPGA implementation of the target PUF.

Note that *the novelty of our work is in the prevention of modeling attacks on PUF-based authentication schemes, rather than using PUFs for authentication*.

II. Related work

Several authentication protocols proposed in literature for device attestation. However, they may not be suitable for IoT frameworks given the resource constraints and very dynamic network membership of IoT devices [8]. Moreover, storing device identification in its memory is not suitable due to the security vulnerabilities. Hence, PUF-based authentication has

been exploited to mitigate the concern about key storage on vulnerable devices [9], [10]. These PUF-based authentication methods are lightweight, yet suffer from vulnerabilities such as modeling attacks, replay attacks, and impersonation attacks.

To secure data transfer and device authentication, [11] deployed PUFs to generate public and private keys in IoT networks. This method is robust against replay attacks, yet it is computationally intensive. A mutual authentication scheme has been presented in [12] where each challenge bit-stream is a function of the previous one. Although resilient against modeling attacks, leaking even one challenge bit-stream makes the overall process vulnerable to impersonation attacks.

Challenge obfuscation schemes (e.g., [13]) proposed in literature where each challenge bit-stream C is mutated before feeding the PUF; thereby the response is generated for the mutated challenge (i.e., $\hat{C}$) and is not directly related to C that the adversary has intercepted. This scheme misleads the adversary by injecting wrong CRPs into dataset used for training the PUF model. This method decreases the PUF modeling accuracy, yet imposes a significant area overhead. Vatajelu [14] obfuscated a strong PUF with a symmetric encryption algorithm whose key is generated via a weak PUF. This scheme is promising, but suffers from hardware overhead. The Slender [9] and Noise Bifurcation [15] obfuscated PUFs both were compromised by the CMA-ES based attack proposed in [16]. Gu [17] proposed to insert two PUFs in each node (genuine and fake PUFs). The genuine PUF responses are used for authentication while the fake PUF is queried once a while to mislead the adversary who eavesdrops the transmission line and thus prevent PUF modeling. This method suffers from area and power overhead and the increase of traffic related to the exchange of redundant CRPs.

Wang [18] used adversarial models to fool an attacker who eavesdrops CRPs. The approach changes the response of some challenges based on a functions of the challenge bits or in a periodic manner. Although imposes little area overhead, as we show in Sec. VI, it can be easily compromised via ML schmes.

III. Threat Model

We assume that authentication and key management are conducted through a central supervisory node (e.g., server) either as a part of IoT admission control or as a service to enable communication between device pairs. The server is assumed to be trustworthy, i.e., *handling an IoT with an untrusted server is out of scope of this paper.* We assume that a PUF is embedded in each IoT device during the fabrication and is leveraged for its authentication. To authenticate node N_i, the server sends it a challenge bit-pattern. Then N_i applies the challenge to its PUF and sends back the PUF response (although may not send the correct response in our method). By matching the node response to a pre-known value, the server can confirm the identity of N_i. Note that in PUF-based authentication schemes, a subset of CRPs of each device are stored in the server during the device enrolment phase.

In our proposed method, node N_i may send an incorrect response intermittently, instead of the one supposed to, to mislead the adversary who eavesdrops on the wireless link between N_i and the server. This makes the adversary build an incorrect model of the targeted PUF, thus thwarting the modeling attack. Decisions on toggling the response for a challenge received from the server (i.e., sending an incorrect or a correct response from N_i to the server) is made based on our approach discussed in Sec. V. Both the server and N_i apply the same decision process, and thus the server can differentiate correct and fake (incorrect) responses. *This results in imposing no overhead regarding required CRPs as even the challenges with fake responses can be used for authentication.* In this paper, the adversary is unable to differentiate genuine from fake responses. Moreover, we rely on the link- and transport- layer protocols for the reliable exchange of packets between the server and the nodes. We assume that the impact of transmission noise is mitigated via Error Correction Codes.

IV. Preliminaries

A. Arbiter-PUF

An arbiter-PUF consists of a pair of delay chains; when queried, it generates one response bit per challenge [19]. This PUF operates based on the process-variation that induces race between two identical paths (top and bottom paths 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. The sign of this difference (extracted by the arbiter) presents the PUF response. The arbiter can be as simple as an SR-latch. *This paper targets arbiter-PUFs. However, the proposed schemes are applicable to other strong PUFs.*

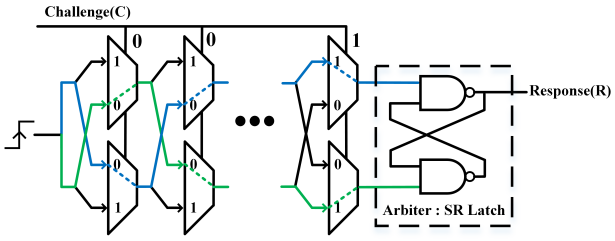


Figure 1. Illustrating the design of an arbiter-PUF.

B. Adversarial Machine Learning (AML)

In this paper, we assume that the adversary deploys ML to model the target PUF. In the training phase, the model is built utilizing the PUF's CRPs. Then, in the evaluation phase, the response of unseen challenges are predicted based on the built model. We provide the attack outcome when the adversary uses Neural Networks (NN), Support Vector Machine (SVM), or Logistic regression (LR) [17] to model the PUF, and show the efficiency of our proposed countermeasure against all such state-of-the-art modelings. We also show that attacks through the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [19] are not effective against our proposed method.

Meanwhile, AML is a technique to corrupt ML-based models through poisoning or evasion; the former incorrectly manipulates data during the training phase to decrease the effectiveness of the model. Evasion corresponds to the case when fake data is injected at the test time such that the output is classified (i.e., predicted) incorrectly. In this paper, we apply poisoning as a defense mechanism against modeling attacks by flipping the PUF response. The poisoned data is inserted in an unpredictable manner given that the adversary collects CRPs in an incremental way.

V. Proposed Methodology

In this paper, we employ AML to increase the resiliency of the PUF-based security solutions. In particular, we toggle the PUF response intermittently based on some conditions discussed below. Accordingly, the adversary who eavesdrops

on the communication links will intermittently receive poisonous responses (instead of the correct one), will thus fail to accurately model the target PUF. As the poisoning algorithm is known by the server, i.e., decided during enrolment of IoT devices, the poisonous response can also serve in authenticating the device; thereby *zero overhead is imposed by our method to the IoT framework regarding CRPs as no extra CRP is required by the proposed scheme for authenticating a device.*

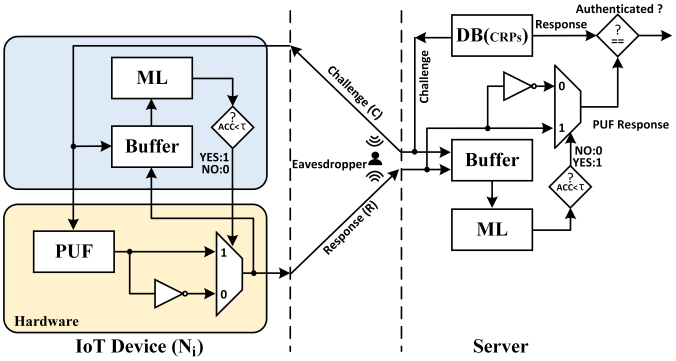


Figure 2. The block diagram of the proposed scheme.

Fig. 2 shows the block diagram of the proposed scheme. To authenticate node N_i the server sends a challenge (say C) to N_i whose response is known by the server. Upon receiving the challenge, it is applied to the N_i's embedded PUF, and either the PUF response or its complement is sent back to the server. The server is aware in case the response has been toggled. The server then checks the response of N_i with a pre-stored value in order to authenticate the node. To decide whether a response needs to be toggled or not, the node runs a ML model, specifically a NN model in our case, based on the last L exchanged CRPs. This is done to mimic what an eavesdropper on the communication link will do to model the PUF behavior and predict the response for challenge bit-streams.

Storing all exchanged CRPs requires a large buffer and is not cost-effective in the node side. Thereby, the ML modeling on the node (and the server) is always performed on the last L exchanged CRPs stored in the "Buffer" (Fig. 2), from which T CRPs are used for training the model and the rest $M = L - T$ CRPs are used for its inference. In practice, we train a model and test it (on the node side) upon receiving the first B challenges (B refers to Bundle Size) and repeat the process iteratively each time after receiving the next B challenges. If the modeling accuracy exceeds a predefined threshold value (e.g., $\tau > 0.6$), we decide to poison (toggle) the PUF response for the next B upcoming challenges. However, if the modeling accuracy is less than τ, the next B responses are sent intact. The threshold τ is a real number in [0,1] and its value is agreed upon the IoT node enrolment between the node and server. Note that the response of each challenge is sent back immediately by the node upon the receipt of that challenge, i.e., the node does not wait to receive a bundle.

In the proposed scheme, the buffer embedded in each node is empty upon powering up, and hence the response related to the first B queries are sent intact (not toggled). However, these B CRPs are used to decide about whether poisoning is applied to the next bundle. The process continues iteratively each time a new bundle (i.e., B challenges) is received, yet the ML model uses the whole data residing in the buffer and not just the new bundle. As will be discussed in Sec. VI-A2, the bundle size selection should strike a balance between overhead and PUF modeling prevention. In particular the first B challenges should not be enough to build an accurate PUF model.

Algorithm 1: Proposed AML Scheme

input : Threshold (τ), Bundle-Size (B), New-CRP
output: Flag: (1/0) => select (genuine / poisonous) response

1 $Flag \leftarrow 1$
2 **while** *(true)* **do**
3 $i \leftarrow 0$
4 **while** $(i < B)$ **do**
5 **if** *(New-CRP)* **then**
6 **if** *(buffer is not full)* **then**
7 store new CRP in the buffer
8 **else**
9 replace the oldest CRP with New-CRP
10 $i \leftarrow i + 1$
11 Running ML on the buffer
  if (ACC < τ) **then** $Flag \leftarrow 1$ **else** $Flag \leftarrow 0$

After the buffer becomes full, the whole data is used for the next round of decision making. This buffer is implemented as a FIFO queue. When a bundle is added, the oldest bundle is flushed from the queue and NN is applied. In fact, the modeling is done based on the whole content of the buffer. As mentioned, based on the outcome of each modeling round (i.e., when a new bundle is added to the buffer), a flag is set indicating whether the response of until the next round (bundle) will be sent intact or poisoned. The idea behind this is that the adversary intercepts fake data intermittently which degrades the accuracy of the PUF modeling attack. The data stored in the buffer consists of each challenge along with the related transferred response (it can be genuine or poisonous based on the decision made by the previous ML model assessed accuracy). Thereby, at the node side the model is trained with the same data used by the eavesdropper; the only difference is that node uses the most recent L CRPs while the adversary can consider all exchanged data. Yet, as the experimental results show, this does not affect the effectiveness of the proposed scheme in thwarting modeling attacks.

To authenticate an IoT node, the server needs to follow a similar approach discussed above for the node, i.e., *the server also includes a buffer to store the recent L exchanged CRPs and runs the same ML model (initialized with the exact same weights and seed for random generator agreed upon during the node enrolment process*) upon receiving each B responses and compares the modeling accuracy with the threshold value to find out if the response would be genuine or poisoned. In this case, **the server can also authenticate the device using poisoned ones**. Thus, our method does not require exchanging any extra CRPs. Algorithm 1 depicts the pseudo code of our proposed scheme. As mentioned, each CRP stored in the lines 7 and 9 of this algorithm consist of a challenge with its related genuine or fake response (based on the last poisoning decision made through AML). Since *the adversary is unaware of the ML technique and parameters used at the server and the node, he cannot distinguish between the poisoned and legitimate responses*. Similar to contemporary authentication schemes, we consider the node enrolment phase to be secure. Moreover, being unaware about the poisoning algorithm, the adversary cannot benefit from using the adversarial training

schemes to thwart the poisoning scheme.

VI. Experimental Results and Discussions

We implemented a 64-bit Arbiter-PUF on Xilinx ARTIX7 FPGA. The results provided in this section are based on using NN to build the adversarial model in the node and server, while applying NN, SVM and LR as the representatives of ML techniques that an adversary pursues to conduct a modeling attack against the authentication scheme. Nonetheless, we have alternated among these techniques, e.g., by basing the adversarial model on SVM and pursuing NN, SVM, and LR as a modeling attack scheme; in these alternative scenarios, the results stayed consistent and are excluded from the paper due to space limitations. Our NN consists of one input layer (with 64 neurons reflecting the PUF size), three nonlinear hidden layers (with 5, 10 and 15 neurons) and one output neuron with sigmoid function. Rectified linear unit (ReLU) is used as an activation function in all layers. The learning rate and momentum are 0.01 and 0.99, respectively, and the number of epochs is 1000. This architecture can model our 64-bit arbiter-PUF with 98% accuracy using 2000 CRPs for training in the absence of the proposed poisoning approach.

The adversary is assumed to intercept the exchanged CRPs, but cannot differentiate between the genuine and poisoned ones based on the contents. In all experiments we have used 90% of the data set for training and the rest for the inference unless otherwise mentioned, i.e., $T = 0.9L$ and $M = 0.1L$.

A. Experimental Result

1) Impact of AML on the PUF modeling attack success: The first set of results shows the effectiveness of the proposed AML-based scheme in preventing the PUF modeling attack. These results were extracted for buffering 200 CRPs (at both the IoT node and the server) where each bundle includes 20 CRPs and the threshold accuracy (τ) is set to 0.6. Fig. 3 presents the accuracy of the PUF model, when our poisoning scheme is or is not used. As expected, the more CRPs the adversary intercepts, the higher the accuracy of the PUF model. Without our scheme (i.e., unprotected), the modeling accuracy is 96% even with 2000 CRPs. When applying our AML-based protection, building a model based on intercepting 2000, 5000, 10,000, and 50,000 CRPs results in 56%, 60%, 61%, and 64% accuracy, respectively. Even with intercepting 100,000 CRPs, the PUF modeling accuracy is 64.2%.

The first take away point from the above observations is that our data poisoning strategy is highly effective in thwarting the PUF modeling attack, regardless of the training size (i.e., the number of the intercepted CRPs). As shown, in all cases the modeling accuracy is around 60%. This relates to the considered τ value (=0.6) and confirms that the assessed modeling accuracy using a limited number of CRPs stored in the buffer (200 in our case) highly corresponds to the adversary achieved modeling-accuracy even despite using much more CRPs. The results also show that limiting the buffer size (to 200 CRPs in our experiments) and making decisions on poisoning the responses based on such limited training data, does not negatively affect our proposed countermeasure as long as an appropriate buffer size (L) is chosen. In fact, selecting an appropriate buffer size is of utmost importance, and its impact on the modeling accuracy will be discussed in Sec. VI-A2.

Fig. 4 depicts the fluctuation of the accuracy around τ=0.6 for the model built in the node. The results are

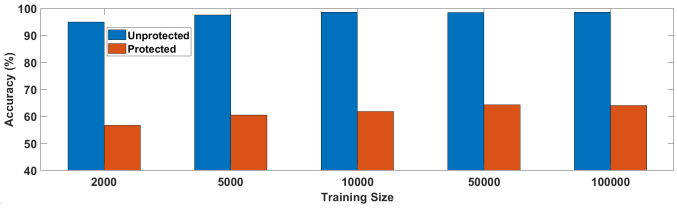


Figure 3. The adversary's achieved modeling accuracy for different training-set size with and without using our poisoning scheme. A NN was used for modeling, L=200, and B=20.

shown for 100 consecutive bundles each includes 20 CRPs. We show the accuracy that the node achieves while running its own model based on the whole data stored in its buffer at each point of time. As discussed, whenever the accuracy exceeds τ, poisoning is activated and the responses for the next B challenges are toggled before being sent. This pulls the accuracy down and forces it to a value below τ, as confirmed by the results.

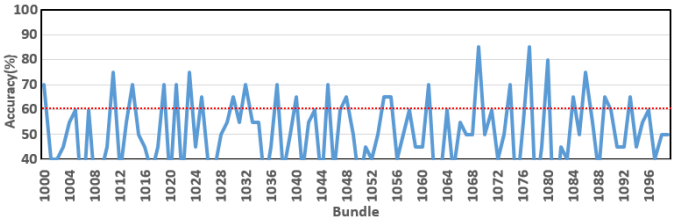
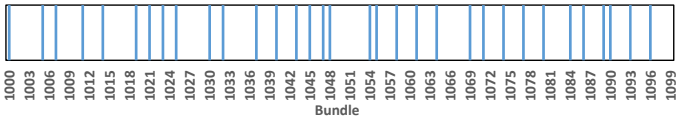
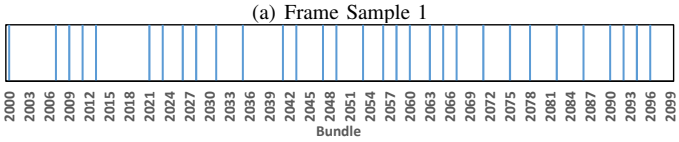


Figure 4. The modeling accuracy in the IoT node using all stored 200 CRPs in the buffer when the poisoning decision is made upon receiving each of the shown 100 consecutive bundles of 20 CRPs.This window was randomly chosen.

As will be discussed below if the CRPs are poisoned in a periodic fashion (similar to [18]), the PUF can be modeled. Accordingly, to ensure that the proposed scheme does not follow a periodic poisoning pattern, Fig. 5(a) demonstrates the location of poisonous data in the sequence of PUF responses. Each blue vertical bar represents a bundle of poisonous responses. We only capture a frame size of 100 bundles for the sake of readability. As shown the poisoning strategy is not periodic, thus more resilient against the modeling attacks discussed below. Fig. 5(b) depicts the poisoning pattern in another point of time during the CRP exchange. This pattern is very different from the one in Fig. 5(a) confirming the adhoc nature of our poisoning strategy.



(a) Frame Sample 1



(b) Frame Sample 2

Figure 5. Depicting the time intervals between poisoning each two bundles. The blue bars show the cases where the data got poisoned. Each window includes 100 consecutive bundles.

The conclusion drawn from the results in Fig. 3 - Fig. 5 is that the adversary who intercepts all exchanged CRPs cannot distinguish the poisoned data from the genuine one, hence fails to model the PUF behavior. This weakness is exacerbated due to the adhoc (non-periodic) nature of our poisoning scheme.

The next set of results compares the efficacy of the proposed method with the AML-based scheme presented in [18] that introduced different poisoning schemes. In [18], the response toggling decision is made either based on a combinational function applied on each challenge bit-stream (when satisfied, the PUF response is poisoned) or the poisoning is performed in a periodic fashion. Although claimed to be resilient against modeling, we were able to attack these schemes with a high success. Table I shows the results for the combinational case. In each case, the authors toggle the PUF response when the trigger function gets the value of "1". As shown, we could attack the models with as low as 5000 CRPs with > 95% accuracy while [18] failed to attack the model using 10^6 CRPs.

The sequential-trigger poisoning scheme in [18] toggles the response for every other N coming challenges while sends the other responses intact. The weakness of this method is that to be resilient, the poisoning period P cannot be too high (e.g. > 10 based on our investigation) as the higher the P the less the rate of poisoning and so less resiliency. Meanwhile, the attacker can try P$\in \{1, .., 10\}$ to model the PUF otherwise. This shows the superiority of our proposed scheme over the previous AML-based ones; our poisoning scheme does not follow a periodic pattern thus cannot be tackled in a similar way.

Table I
DEFEATING [18] WITH OUR NN MODELING

Poison Trigger Function	**Accuracy reported in [18]** 10^6 **CRPs**	**Our Accuracy (using NN) 5000 CRPs**
$C_0 \cdot C_1 + C_2 \cdot C_3$	59.21%	**95.70%**
C_0	56.60%	**96.20%**
$C_0 \oplus C_1 \cdots \oplus C_7$	55.37%	**96.70%**

2) The effect of buffer and bundle size, and threshold value on the attack success: The buffer size (L), bundle size (B), and τ should be selected such that the adversary fails to model the PUF. Obviously, the larger τ gets, the higher the probability of the adversary's success becomes. Given the binary nature of the PUF's response, $\tau = 0.5$ is ideal to prevent modeling attack. However, $\tau = 0.5$ implies a periodic poisoning pattern. Setting $\tau = 0.6$ strikes a balance by degrading the adversary's ability for modeling the PUF while introducing randomness in the poisoning pattern.

A large-size buffer can impose unacceptable area overhead. On the other hand, making poisoning decisions based on few data points (in case of very small buffers) is not justified as in this case the adversary's model (with access to the whole exchanged CRPs in the best case) would diverge from the model built in the node (based on few data points). Moreover, a very large bundle size results in transfer of excessive consecutive CRPs that either *all* are genuine or *all* are toggled, thus facilitates the modeling attack. Moreover, small bundle size increases the effort as results in more learning iterations.

Fig. 6 shows the cases where the bundle size is 10%, 30% and 50% of the buffer size. As expected, the lower the $\frac{B}{L}$ value is, the more resilient the target PUF would be against modeling its behavior. Regardless of the buffer size, the adversary cannot achieve more than 60% success rate when $B = 0.1 \times L$, and thus fails to model the PUF. By increasing $\frac{B}{L}$ to 0.3, the PUF can be modeled easily (ACC. > 80%) for all the buffer sizes shown in this figure. Note that the situation is different for $\frac{B}{L}$=0.5 where increasing the buffer size to 2000 prevents the adversary from modeling the PUF accurately (ACC.≈65%). This is because in this case B=1000, hence either 1000 genuine or 1000 fake responses are sent consecutively each time. This number is large enough to push the accuracy > 60% and thus results in making the opposite decision in the next round. Thus, the decision would be one toggling among each 3 consecutive decisions, resulting in around 33% toggling and consequently low accuracy. This can also be explained based on Fig. 7.

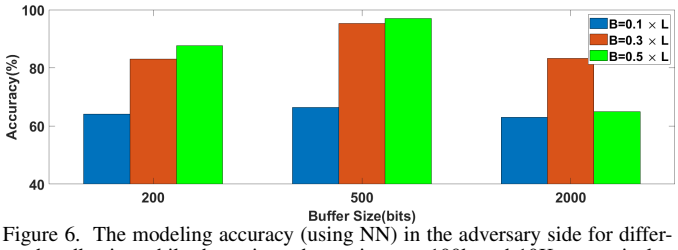
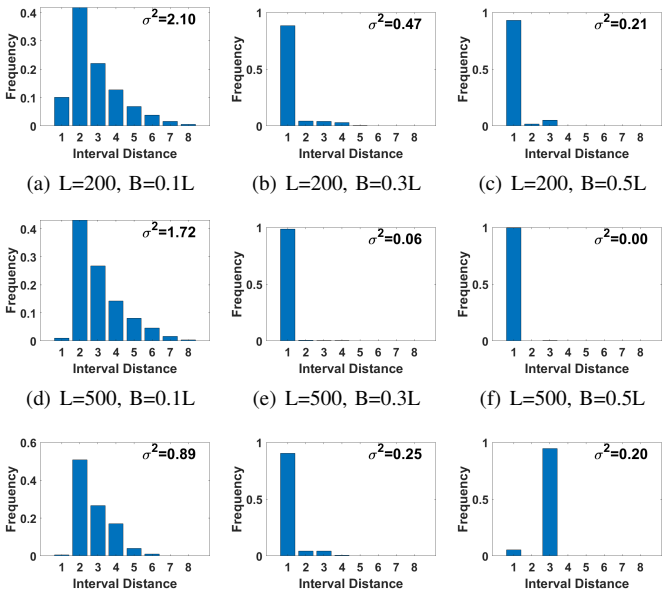


Figure 6. The modeling accuracy (using NN) in the adversary side for different bundle size while the train and test sizes are 100k and 10K respectively.

Another factor that needs to be taken into account when selecting proper sizing (B and L) is whether the toggling pattern is random or periodic. As mentioned above, periodic poisoning of CRPs can be exploited by the adversary to model the PUF. Fig. 7 depicts the histogram for the distribution of time intervals between two poisoned bundles while varying the bundle and buffer sizes. For each case, the variance (σ^2) has been shown in the figure. As expected, by increasing the $\frac{B}{L}$, the poisoning tends to be periodic (i.e., low variance). This figure along with Fig. 6 confirm that with a $\frac{B}{L} \approx 10\%$ our proposed scheme is highly efficient in terms of both low modeling accuracy and less periodicity. Note that for the $\frac{B}{L}$ less than 0.1 (not shown) the result are promising but with a higher overhead as discussed earlier.

3) Resistance against PUF modeling mechanisms: SVM, LR, and CMA-ES have been shown to be effective in attacking the arbiter-PUF and its derivatives (XOR-PUF, Feed-Forward PUF, etc) with high accuracy [19], [20]. Accordingly, this set of results investigates the resiliency of our proposed method against these modeling mechanisms. As Fig. 8 depicts, neither



(a) L=200, B=0.1L (b) L=200, B=0.3L (c) L=200, B=0.5L

(d) L=500, B=0.1L (e) L=500, B=0.3L (f) L=500, B=0.5L

(g) L=2000, B=0.1L (h) L=2000, B=0.3L (i) L=2000, B=0.5L

Figure 7. Distribution of the time intervals between poisoning two bundles for different buffer & bundle sizes.

LR nor SVM is successful when our AML scheme is deployed, where in both cases the modeling accuracy is ≈60% even with intercepting 100,000 CRPs. Meanwhile the unprotected PUF can be modeled with an accuracy of 98% with as low as 2000 CRPs using either SVM or LR. *We again note that we have repeated the experiment where NN is used for modeling attack while the poisoning decision is made by SVM or LR. The observed results were consistent with Fig. 8.*

Becker [19] uses the CMA-ES scheme to model the PUF based on the sensitivity of its response to environmental noise, e.g. temperature or voltage variations. *Such CMA-ES based attack is not applicable when our AML scheme is used* since firstly based on our threat model the adversary doesn't have physical access to the PUF to repeat the same query multiple times to benefit from the measurement noise in PUF modeling, instead intercepting the exchanged messages is the only means. Secondly, even if the PUF is queried with the same challenge once a while, the node's response to the same query may change due to poisoning. This confuses the adversary as CMA-ES may consider the poisoned data as measurement noise.

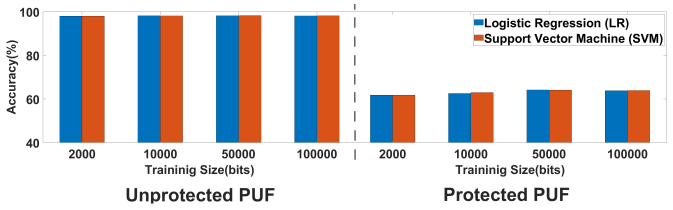


Figure 8. The accuracy that the adversary can achieve (using SVM and LR). Bundle size = 20, buffer size = 200 while the train size is increasing to 100K.

4) Performance and security analysis: Below, we discuss the security as well as overhead of the proposed framework.

Preventing Impersonation and Sybil Attacks: An impersonation attack is realized when an adversary tries to identify itself as a legitimate IoT node and sends erroneous messages or leaks secret information upon authentication. Due to the uniqueness of PUFs, the adversary does not have access to the same PUF and has to pursue ML-based modeling. Our poisoning method thwarts such a modeling attempt. In addition, intermittent poisoning of the CRPs may result in sending different responses to the same challenge at different points of time by the node. This is known by the server and a legitimate node would be authenticated; yet the adversary cannot distinguish genuine and poisoned responses, and thus cannot decide whether to send the response or its complement even if he intercepted the response beforehand and is aware that some responses are poisonous. A Sybil attack reflects the case in which the adversary impersonates multiple nodes. This is also prevented in a similar way as impersonation.

Overhead and Noise: The hardware overhead includes an inverter and a 2-bit multiplexer per response. The ML is performed in the IoT device's software based on which a response or its complement is selected by the multiplexer. Moreover, similar to other PUF-based authentication schemes (e.g. [11]), the transmission noise is mitigated via error correction mechanisms used in the transport layer of IoT frameworks (out of scope of this paper). Also the rate of measurement noise in PUFs, related to the environmental condition change such as temperature and voltage is low (0.2% in our experiments) and for a delay-PUF, the measurement noise can be handled by multiple measurements and majority voting in the PUF side to reduce the signal-to-noise ratio [11]. Since the server and node ML models are trained with the same data collected by the node's PUF, the measurement noise doesn't affect the synchronization process; In the rare case of receiving noisy data, it can be discarded by the server due to failed authentication.

While ML techniques vary in their computational complexity, our results show that there is no significant difference in robustness when NN, SVM or LL are used by the node and server. Moreover, the frequency of conducting authentication is typically not high for most practical applications. Given such flexibility, we do not see the computational overhead of our approach to be an obstacle for wide adoption.

VII. Conclusion

Physically Unclonable Functions offer a promising solution for the authentication of IoT devices due to their low cost, unique signature, and easy implementation. Although deemed to be unclonable, a PUF behavior may be modeled by an adversary who has access to a subset of its challenge response pairs. In this paper, we proposed a robust modeling-resilient PUF-based authentication scheme that poisons the transmitted challenge response pairs intermittently to diminish the accuracy in the adversary's model. The validation results confirm the efficacy of the proposed method against PUF modeling as well as the conventional attacks launched in IoT frameworks such as impersonation and Sybil attacks. Future research directions include developing PUF-based data integrity solutions.

References

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

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

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

[4] M. Ebrahimabadi *et al.*, "Hardware assisted smart grid authentication," in *IEEE Int'l Conf. on Communications*, 2021.

[5] ——, "A novel modeling-attack resilient Arbiter-PUF design," in *VLSID*, 2021, pp. 123–128.

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

[7] M. S. Alkatheiri et al, "Towards fast and accurate machine learning attacks of feed-forward arbiter PUFs," in *DSC*, 2017, pp. 181–187.

[8] T. Xu *et al.*, "Security of IoT systems: Design challenges and opportunities," in *ICCAD*, 2014, pp. 417–423.

[9] M. Majzoobi et al., "Slender PUF Protocol: a Lightweight, Robust and Secure Authentication by Substring Matching," in *S&P*, 2012, pp. 33– 44.

[10] Y. Lao et al., "Reliable PUF-Based Local Authentication with Self-Correction," *TCAD*, vol. 36, no. 2, pp. 201–213, 2016.

[11] 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.

[12] M. N. Aman *et al.*, "Physical unclonable functions for IoT security," in *Int'l Workshop on IoT privacy, trust, and security*, 2016, pp. 10–13.

[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] M.-D. Yu et al, "A noise bifurcation architecture for linear additive physical functions," in *HOST*, 2014, pp. 124–129.

[16] G. T. Becker, "On the pitfalls of using Arbiter-PUFs as building blocks," *TCAD*, vol. 34, no. 8, pp. 1295–1307, 2015.

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

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

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

[20] P. H. Nguyen *et al.*, "The interpose PUF: Secure PUF design against state-of-the-art machine learning attacks," *CHES*, pp. 243–290, 2019.