Attacks on the PUF-Based Authentication Protocols YeHL16 and GaoMAAR17*

Jeroen Delvaux

imec-COSIC, KU Leuven, Belgium, jdelvaux@esat.kuleuven.be

Abstract. A *physically unclonable function* (PUF) is a circuit of which the input–output behavior is designed to be sensitive to the random variations of its manufacturing process. This building block hence facilitates the authentication of any given device in a population of identically laid-out silicon chips, similar to the biometric authentication of a human. The focus and novelty of this work is the development of efficient impersonation attacks on the following two PUF-based authentication protocols: (1) the protocol of Ye, Hu, and Li, as presented at AsianHOST 2016, and (2) the protocol of Gao, Ma, Al-Sarawi, Abbott, and Ranasinghe, as published in the IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems in 2017.

Keywords: physically unclonable function · entity authentication · machine learning

1 Introduction

Since their advent in the early 2000s [LDT00], *physically unclonable functions* (PUFs) have been used as a building block in numerous authentication protocols. The authentication is either unilateral, i.e., one-way, or mutual, i.e., two-way, and usually takes place between a low-cost, resource-constrained device hosting a PUF and a high-cost, resource-rich server storing a selection of the input–output pairs of this PUF. Delvaux [Del17, Chapter 5] analyzed the security and practicality of 21 such protocols, thereby revealing numerous problems to the extent that only six candidates survive. In parallel, Becker [Bec15a, Bec15b] and Tobisch [TB15] pushed the boundaries of machine learning attacks on PUF-based protocols. The previous analyses, however, are not up-to-date with more recently proposed protocols. In this work, we illustrate that the research field of developing new PUF-based authentication protocols remains a minefield. Efficient attacks on the proposals of Ye et al. [YHL16] and Gao et al. [GMA+17] are presented.

The remainder of this paper is organized as follows. Section 2 introduces the notation and provides preliminaries. Section 3 specifies and obliterates the aforementioned authentication protocols. Section 4 concludes this work.

2 Preliminaries

2.1 Notation

Variables are denoted by a character from the Latin alphabet: a, b, c, etc. Constants are denoted by a character from the Greek alphabet: α, β, γ, etc. Vectors are denoted by a bold-faced, lowercase character, e.g., $\mathbf{x} = (x_1\, x_2)$. All vectors are row vectors. The

*If you downloaded this file from any source other than https://eprint.iacr.org/, please check the previous link to ensure that your version is the latest one.

all-zeros vector is denoted by $\mathbf{0}$. Matrices are denoted by a bold-faced, uppercase character, e.g., $\mathbf{X}$. A diagonal matrix is defined by listing the entries on its main diagonal, e.g., $\mathbf{X} = \mathrm{diag}(x_1, x_2)$. A random variable is denoted by an uppercase character, e.g., X. A multivariate normal random variable X with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ is denoted by $X \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. A set, often but not necessarily referring to all possible outcomes of a random variable, is denoted by an uppercase, calligraphic character, e.g., $\mathcal{X}$. The set of all λ-bit vectors is denoted by $\{0,1\}^\lambda$. Custom-defined functions are printed in a sans-serif font, e.g., Hamming distance $\mathsf{HD}(\mathbf{x}_1, \mathbf{x}_2)$.

2.2 Arbiter PUFs and Machine Learning

A PUF maps a binary input, i.e., the so-called challenge $\mathbf{c} \in \{0,1\}^\lambda$, to a binary, device-specific output, i.e., the so-called response $\mathbf{r} \in \{0,1\}^\eta$. Unfortunately, noise sources within the device, as well as changes to its external environment, imply that an initially generated response $\mathbf{r}$ slightly differs from its reproduction $\tilde{\mathbf{r}}$. The value of $\mathsf{HD}(\mathbf{r}, \tilde{\mathbf{r}})$, averaged over numerous challenges $\mathbf{c}$, typically lies between 0.05η and 0.15η. There is a special interest for PUFs that support a large-sized challenge $\mathbf{c}$, e.g., having $\lambda = 128$, because this facilitates the design of an authentication protocol considerably. Even those who are given unrestricted access to such a PUF can neither gather nor tabulate all of its *challenge–response pairs* (CRPs) within the lifetime of its hosting device. Unfortunately, the CRPs are all determined by the variability of a limited number of circuit elements, and are hence correlated. For this reason, machine learning algorithms training on a relatively small set of CRPs, i.e., $\{(\mathbf{c}_1, \mathbf{r}_1), (\mathbf{c}_2, \mathbf{r}_2), \cdots, (\mathbf{c}_\omega, \mathbf{r}_\omega)\}$ where $\omega \ll 2^\lambda$, can produce a model $\hat{\mathbf{m}}$ that allows to accurately predict the unseen response $\mathbf{r}_{\omega+1}$ to any given challenge $\mathbf{c}_{\omega+1}$.

For the well-known Arbiter PUF [Lim04], which quantizes the difference v between the propagation delays of two reconfigurable paths, a large λ can be supported. If $v > 0$, the single-bit response $r = 1$; otherwise, $r = 0$. Numerous authors have experimentally confirmed that the value of v can be accurately described by a dot product: $v = \mathbf{m}\,\mathbf{s}^T$, where variability model $\mathbf{m} \in \mathbb{R}^{\lambda+1}$ aggregates the propagation delays of the logic gates that constitute both paths and where $\mathbf{s} \in \{-1,1\}^{\lambda+1}$ is the result of an invertible challenge transformation $\mathsf{ToSigns}(\mathbf{c})$. To incorporate the effect of both internal noise sources and environmental changes, the latter of which are assumed to be centered around a constant nominal value, the quantization can be extended to $(v + n) \lessgtr 0$, where $N \sim N(0, \sigma^2)$ with respect to the infinite set of evaluations [Mae13]. A crucial insight is that the reproducibility of the response r to a given challenge $\mathbf{c}$ increases monotonically with the absolute value $|v|$.

If pairs $(\mathbf{s}, r)$ instead of pairs $(\mathbf{c}, r)$ are used as training data, the problem of learning $\mathbf{m}$ becomes quasi-linear and hence straightforward to handle for numerous algorithms. This includes the use of *artificial neural networks*, *support vector machines*, and *logistic regression*. Thanks to existing validations with experimental data, it has become a common practice to demonstrate the feasibility of a machine learning attack on randomly generated instances of the mathematical abstraction $\mathbf{m}$. For the abstraction of an ideally manufactured Arbiter PUF, and in disregard of an irrelevant scale factor $a \in \mathbb{R}_0$, it holds that model $M \sim N\big(\mathbf{0}, \mathrm{diag}(1/2, 1, 1, \cdots, 1, 1/2)\big)$ with respect to the infinite set of devices [Del17, Chapter 3]. Noise sources, however, pollute both training and testing data $(\mathbf{s}, r)$, so if omitted from the mathematical abstraction, the reported learning efficiency is usually slightly higher than for experimental data.

To prevent an attacker from successfully modeling a PUF, several authentication protocols either keep the response bits r internal to its hosting device or obfuscate the link between the public challenges $\mathbf{c}$ and the released response bits r. The latter strategy usually entails the use of a *true random number generator* (TRNG). As demonstrated by Becker [Bec15b] and Tobisch [TB15], however, the release of variables that are correlated to r might still enable a modeling attack. For example, if the protocol leaks the error rate p_{error} of a hidden response bit r, an estimate of the absolute value $|v|$ can still be

obtained. Noise sources might hence help rather than hinder an attacker.

2.3 Attacker Model

The analyzed authentication protocols adopt a frequently used attacker model [Del17, Chapter 5]. The enrollment of a PUF-enabled device takes place in a secure environment, and afterwards, an interface for accessing the CRPs might have to be irreversibly disabled. In the field, the protocols should resist both impersonation and denial-of-service attacks. Given that the device comprises a smart card, a *radio-frequency identication* tag, or another mobile entity, it is assumed that an attacker may obtain physical access. The server, however, features both secure computations and secure storage. The communication channel between both parties is assumed to be insecure. This implies that an attacker may not only eavesdrop on a genuine protocol run, but also manipulate, inject, and block messages.

3 Protocols

We first specify and subsequently wipe out each of the authentication protocols. Sections 3.1 and 3.2 may be read in arbitrary order.

3.1 Ye, Hu, and Li

3.1.1 Specification

The unilateral authentication protocol of Ye, Hu, and Li [YHL16] is specified in Figure 1. Although this specification is complete by itself, a visually oriented reader may benefit from the block diagram of a PUF-enabled device in Figure 6(a).

3.1.2 Attack

Analogous to the growth of cracks in solid materials, the mediocre accuracy of $\approx 75\%$ should have been a warning of an imminent failure. Indeed, we now devise an alternative learning strategy that is an order of magnitude more efficient, thereby allowing an attacker to impersonate a PUF-enabled device an unlimited number of times. Given physical access to the device, the attacker can obtain the 2^γ unique responses $\mathbf{r} \in \{0,1\}^\lambda$ to each out of q challenges $\mathbf{c} \in \{0,1\}^\lambda$. There are hence $(2^\gamma!)^q$ possibilities for constructing a combined training and testing set of $2^\gamma \cdot q \cdot \lambda$ transformed CRPs $(\mathbf{s}'', r)$ each. When exhaustively applying a machine learning algorithm to each out of these sets, the one and only correct mapping can be observed to result in the highest accuracy. Alternatively, an attacker who eavesdrops on q genuine protocol runs can iterate over $2^{\gamma \cdot q}$ combined training and testing sets of $q \cdot \lambda$ transformed CRPs $(\mathbf{s}'', r)$ each. Figure 2(a) shows that for either strategy, a relatively limited computational effort corresponds to a relatively large number of CRPs.

We apply *linear regression* [HTF09, 12th printing, Section 4.2] to each set of transformed CRPs $(\mathbf{s}'', r)$. Although the learning capabilities of this deterministic approach are slightly inferior to several randomized training algorithms, its speed is unparalleled and hence favors exhaustive enumeration. As shown in (1), determining the least-squares solution of a system of linear equations is all what is needed. Although Figure 2(b) demonstrates that a fairly limited brute-force effort already allows for an accuracy of 90%, we suggest adopting a more efficient two-step approach to further improve the accuracy. First, numerous repeated executions of a small-sized exhaustive search, e.g., using $q = 1$ every time, can be used to deobfuscate the mapping between numerous transformed challenges $\mathbf{s}''$ and their corresponding response bits r. Second, a potentially slower training algorithm with superior

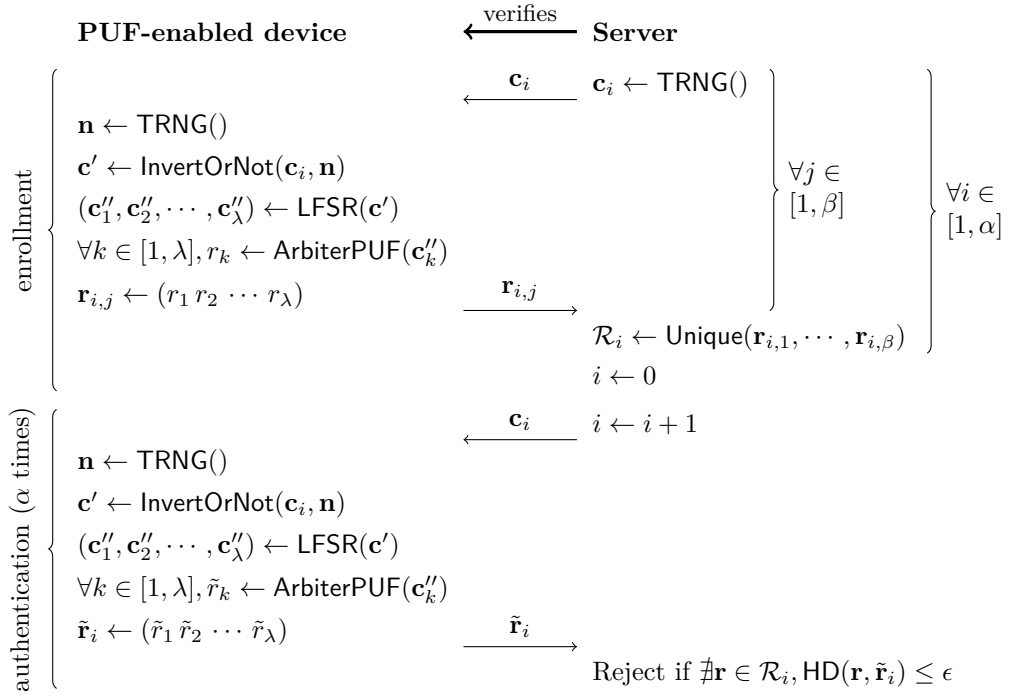


Figure 1: The unilateral authentication protocol of Ye et al. [YHL16]. To prevent the machine learning of its Arbiter PUF, a device either does or does not invert the bits of any received challenge $\mathbf{c} \in \{0,1\}^\lambda$ depending on the value of a nonce $\mathbf{n} \in \{0,1\}^\gamma$. Suggested values for λ are 32, 64, and 128. For $\gamma = 1$, it holds that $\mathbf{c}' \in \{\mathbf{c}, (\bar{c}_1\, \bar{c}_2\, \cdots\, \bar{c}_\lambda)\}$. For $\gamma = 2$, it holds that $\mathbf{c}' \in \{\mathbf{c}, (c_1\, c_2\, \cdots\, c_{\lambda/2}\, \bar{c}_{\lambda/2+1}\, \bar{c}_{\lambda/2+2}\, \cdots\, \bar{c}_\lambda), (\bar{c}_1\, \bar{c}_2\, \cdots\, \bar{c}_{\lambda/2}\, c_{\lambda/2+1}\, c_{\lambda/2+2}\, \cdots\, c_\lambda), (\bar{c}_1\, \bar{c}_2\, \cdots\, \bar{c}_\lambda)\}$. Larger values of γ are not deemed necessary. The randomized challenge $\mathbf{c}'$ is fed into a *linear-feedback shift register* (LFSR) so that the 1-bit responses r to an expanded list of λ challenges $\mathbf{c}''$ can be concatenated into a λ-bit response $\mathbf{r}$. To enroll a device, the server requests the response $\mathbf{r}$ to each out of α randomly generated challenges $\mathbf{c}$ not once but $\beta \gg 2^\gamma$ times and collects the 2^γ unique values. A suggested value for β is 100. Evidently, slightly differing responses $\mathbf{r}$ are attributed to the noisiness of the PUF and are not considered unique. To authenticate a device up to α times, the server checks whether the response $\tilde{\mathbf{r}}$ to a challenge $\mathbf{c}$ is sufficiently close to one out of its 2^γ prerecorded values. The authors emphasize that the nonce N should be uniformly distributed over $\{0,1\}^\gamma$. Otherwise, frequency analysis would allow an attacker to partition the unique responses $\mathbf{r}$ from multiple protocol runs into 2^γ sets that each correspond to a given transformation of the challenge $\mathbf{c}$. The authors collect data from numerous protocol runs and conduct machine learning experiments that do not exceed an accuracy of $\approx 75\%$. They, consequentially, consider their protocol fit for deployment in practical use cases.

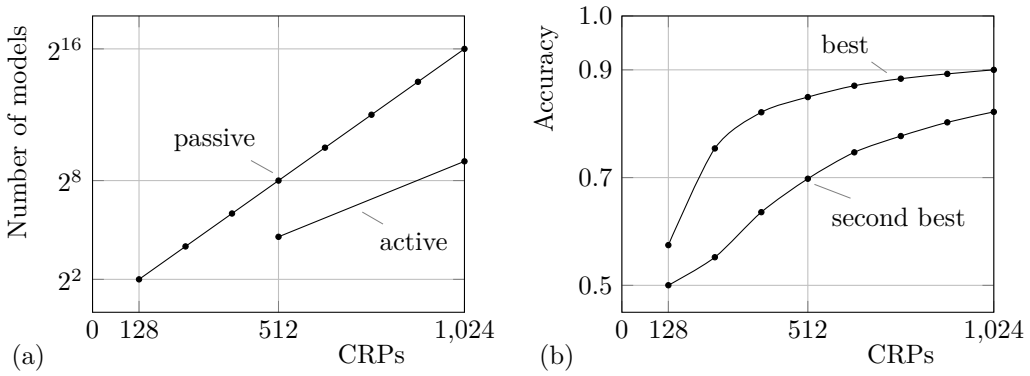


Figure 2: The first phase of an attack on the protocol of Ye et al. [YHL16], where $\lambda = 128$ and $\gamma = 2$. For an either passive or active attacker, subplot (a) shows the number of possible mappings between a given number of transformed challenges $\mathbf{s}''$ and an equal number of response bits r. For each possible mapping, a model is trained and subsequently tested. Subplot (b) shows the accuracy of the best and second-best models, which are obtained through linear regression according to (1). Both accuracies are averaged over 1000 randomly generated and noiseless PUFs $M \sim N\big(\mathbf{0}, \mathrm{diag}(1/2, 1, 1, \cdots, 1, 1/2)\big)$. For any given challenge $\mathbf{c}$, we use $\mathsf{round}(0.8\lambda) = 102$ and $\mathsf{round}(0.2\lambda) = 26$ transformed CRPs $(\mathbf{s}'', r)$ for training and testing purposes respectively.

learning capabilities can be applied to a single large set of deobfuscated pairs $(\mathbf{s}'', r)$. This way, accuracies exceeding 99% can be achieved [RSS+13].

$$\text{Solve} \begin{pmatrix} \mathbf{s}''_1 \\ \mathbf{s}''_2 \\ \vdots \\ \mathbf{s}''_\omega \end{pmatrix} (\hat{\mathbf{m}}_1^T \; \hat{\mathbf{m}}_0^T) = \begin{pmatrix} r_1 & \bar{r}_1 \\ r_2 & \bar{r}_2 \\ \vdots & \vdots \\ r_\omega & \bar{r}_\omega \end{pmatrix} ; \text{ predict } \hat{r}_{\omega+1} = \begin{cases} 1, & \text{if } \mathbf{s}''_{\omega+1}\hat{\mathbf{m}}_1^T > \mathbf{s}''_{\omega+1}\hat{\mathbf{m}}_0^T, \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

We emphasize that the previously elaborated attack cannot simply be mitigated by increasing the value of γ. To enroll a device, the response $\mathbf{r}$ to every challenge $\mathbf{c}$ needs to be evaluated $\beta \gg 2^\gamma$ times. Therefore, the attacker and the server face a similar workload. A final note is that, depending on the non-specified internals of the LFSR, a more straightforward deobfuscation method might exist. It is intuitive to assume that the LFSR has a λ-bit state that is initialized by the randomized challenge $\mathbf{c}' \in \{0,1\}^\lambda$, and that each out of λ^2 state updates generates a single challenge bit c''. This allows an attacker to choose two challenges $\mathbf{c}$ such that for any given value of nonce $\mathbf{n} \in \{0,1\}^\gamma$, the expanded challenge sequences are $(\mathbf{c}''_1, \mathbf{c}''_2, \cdots, \mathbf{c}''_\lambda)$ and $(\mathbf{c}''_{\lambda/2+1}, \mathbf{c}''_{\lambda/2+2}, \cdots, \mathbf{c}''_{3\lambda/2})$ respectively. The respective responses $\mathbf{r}$ hence have an overlap of $\lambda/2$ bits.

3.2 Gao, Ma, Al-Sarawi, Abbott, and Ranasinghe

3.2.1 Specification

The mutual authentication protocol of Gao, Ma, Al-Sarawi, Abbott, and Ranasinghe [GMA+17] is specified in Figure 3. Although this specification is complete by itself, a visually oriented reader may benefit from the block diagram of a PUF-enabled device in Figure 6(b).

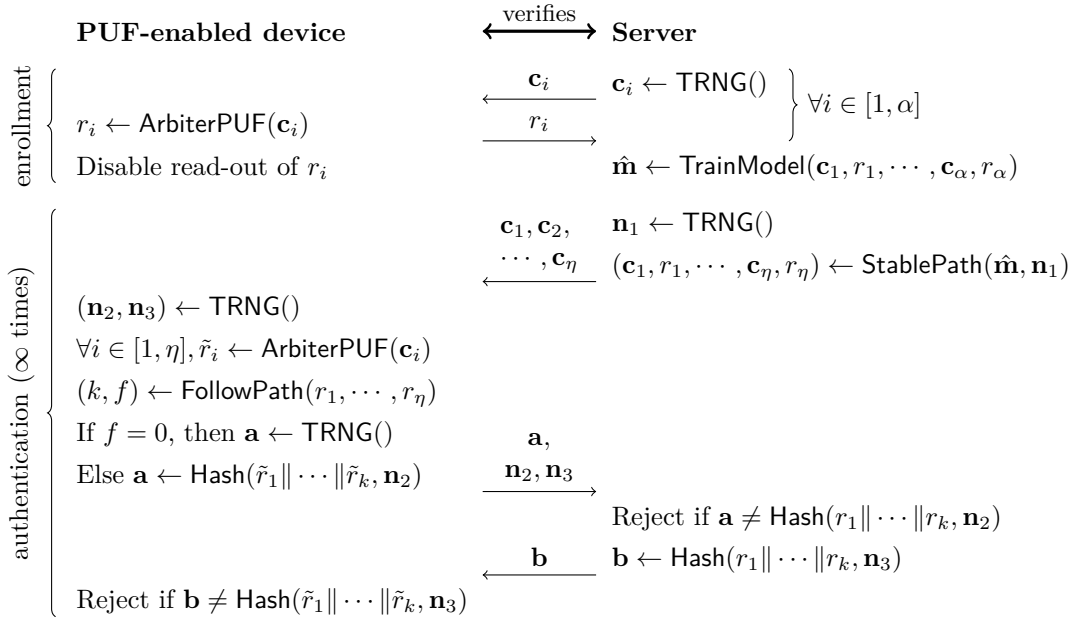


Figure 3: The mutual authentication protocol of Gao et al. [GMA+17]. To enroll any given device, each hosting an Arbiter PUF with $\lambda = 64$ challenge bits, the server collects $\alpha = 10^4$ CRPs $(\mathbf{c}, r)$ so that an accurate predictive model $\hat{\mathbf{m}}$ can be trained. Both response bits r, which are the result of a comparison $v \lessgtr 0$, and their respective error rates p_{error}, which decrease monotonically with $|v|$, can be predicted. After the enrollment, the interface for reading out response bits r is irreversibly disabled. During any out of a virtually unlimited number of protocol runs, the server is restricted to using the CRPs $(\mathbf{c}, r)$ that have the lowest error rates p_{error}, which is a fairly common technique to obtain a low failure rate [Del17, Chapter 4]. Considering the noisiness of their implemented Arbiter PUFs, the authors opt to maintain $1.8 \cdot 10^{17}$ out of 2^{64} CRPs, which corresponds to a retention rate $\rho_{\text{ret}} \approx 1\%$. For a hardwired *finite state machine* (FSM), having one start state and one end state as further specified in Figure 4, the server randomly selects one out of a large number of paths from start to finish. The corresponding sequence of state transitions defines a sequence of η response bits $(r_1, r_2, \cdots, r_\eta)$, where η is a constant and where a variable number of $k \leq \eta$ bits suffices to reach the end state. The server randomly selects a corresponding sequence of η challenges $\mathbf{c}$ that is subsequently transmitted to the device. The latter party then reconstructs the path from newly generated response bits $\tilde{r}_i$. If the end state is successfully reached, i.e., flag $f = 1$, the first k response bits are used to establish a shared secret with the server. This secret, in addition to nonce $\mathbf{n}_2$ or $\mathbf{n}_3$, is then fed into a cryptographic hash function to perform the authentication. To preserve the secrecy of flag f, an attacker is not allowed to observe whether or not the authentication succeeds. Otherwise, an attacker would be able to replace a server-determined challenge $\mathbf{c}_i$ by an arbitrary challenge $\mathbf{c}_j$ with $\mathbf{c}_j \neq \mathbf{c}_i$ and determine whether or not $r_i = r_j$.

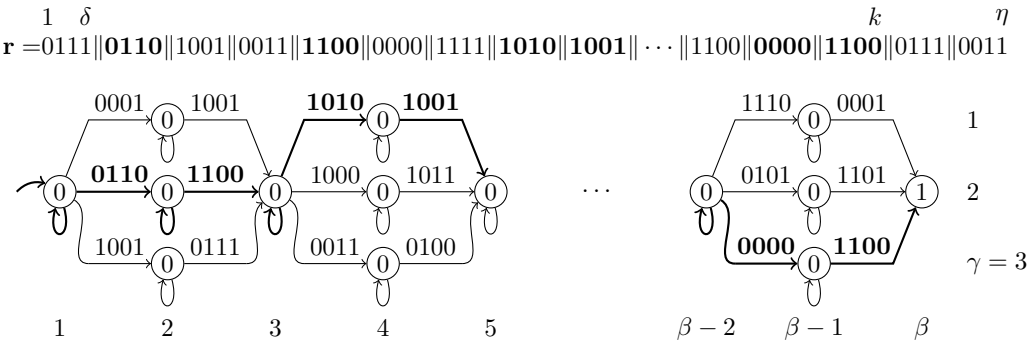


Figure 4: The FSM of Gao et al. [GMA+17] consists of β stages, where constant β is odd. Odd- and even-numbered stages, in turn, consist of 1 and γ states respectively. The authors suggest using $\beta = 41$ and $\gamma = 3$. A path from stage 1 to stage β is traversed using non-overlapping δ-bit substrings of response $\mathbf{x} \in \{0,1\}^\eta$ as a sequence of inputs. A total of $k \in [(\beta-1)\delta, \eta]$ response bits suffices for this purpose. A flag f indicating whether or not the finish is reached is 1 and 0 for stage β and stages 1 to $\beta - 1$ respectively. A value for η has not been suggested.

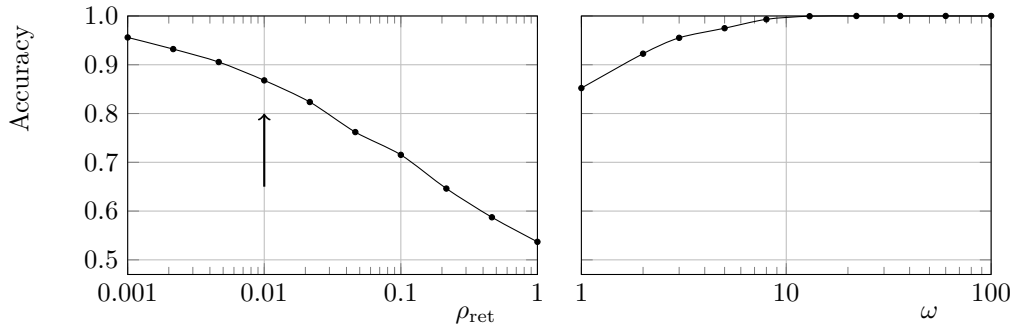
3.2.2 Attack

It suffices for an attacker to eavesdrop on a single genuine protocol run in order to train an accurate predictive model $\hat{\mathbf{m}}$ of the corresponding PUF. This model $\hat{\mathbf{m}}$, in turn, allows the attacker to impersonate either the device or the server a virtually unlimited number of times. Although the authors are aware of Becker's work [Bec15a] and designed their protocol such that not only the response bits r but also their corresponding error rates p_{error} remain internal to the device, it is overlooked that other variables that are correlated to the delay difference v are released. Most notably, for each server-determined challenge $\mathbf{c}$, it is known that the absolute value $|v|$ is relatively high. Given $\omega = 1$ challenge $\mathbf{c} \in \{0,1\}^\lambda$, having transformed version $\mathbf{s} \in \{-1,1\}^{\lambda+1}$, the two best guesses for a model are hence $\hat{\mathbf{m}} = ({}^{s_1}/_2\ s_2\ s_3\ \cdots\ s_\lambda\ {}^{s_{\lambda+1}}/_2)$ and $\hat{\mathbf{m}} = -({}^{s_1}/_2\ s_2\ s_3\ \cdots\ s_\lambda\ {}^{s_{\lambda+1}}/_2)$. The choice between these two models $\hat{\mathbf{m}}$ corresponds to one bit of entropy and is hence irrelevant from a security perspective. Figure 5(a) shows that for a retention ratio $\rho_{\text{ret}} = 1\%$, the best out of two models already exceeds an accuracy of 85%, which suffices to consider the protocol broken.

During a single protocol run, however, the server releases not one but $\eta \gg 1$ challenges $\mathbf{c}_i$. There is hence plenty of margin to improve the accuracy of model $\hat{\mathbf{m}}$. We adopt a *covariance matrix adaptation* (CMA) variant of an *evolution strategy* (ES) [Han06] and perform minimal changes to its open-source implementation in MATLAB. Similar to Darwin's theory on biological evolution, the fittest candidates in a population of prospective models $\hat{\mathbf{m}}$ recombine and mutate into a new and presumably fitter population. It is crucial to define an appropriate fitness function, i.e., $\mathsf{fitness} : \{0,1\}^{\lambda+1} \to \mathbb{R}$. We instantiate the fitness function as shown in (2).

$$\mathsf{fitness}(\hat{\mathbf{m}}) = \frac{1}{\omega}\sum_{i=1}^{\omega}\left|\mathbf{s}_i\,\hat{\mathbf{m}}^T\right| \Bigg/ \frac{1}{q}\sum_{i=1}^{q}\left|\mathbf{s}_{\text{ref},i}\,\hat{\mathbf{m}}^T\right| . \tag{2}$$

The ω transformed challenges $\mathbf{s}_i$ in its numerator originate from a genuine protocol run and are hence known to have stable response bits r_i. The q transformed challenges $\mathbf{s}_{\text{ref},i}$ in the denominator are chosen uniformly at random from the set of all 2^λ transformed challenges and hence have response bits r that cover the full spectrum of error rates p_{error}.



(a) $\omega = 1$ (b) $\rho_{\text{ret}} = 0.01$

Figure 5: The accuracy of modeling an Arbiter PUF with $\lambda = 64$ challenge bits that is used in the protocol of Gao et al. [GMA+17]. For each dot, we generate 100 PUFs $M \sim N\big(\mathbf{0}, \text{diag}(1/2, 1, 1, \cdots, 1, 1/2)\big)$ and average the best accuracies P_{acc} for each out of two reproduced models $\hat{\mathbf{m}}$. Stated otherwise, we show an estimate of $\mathbb{E}[\max(P_{\text{acc}}, 1-P_{\text{acc}})]$, where P_{acc} is the accuracy for one out of two possible models $\hat{\mathbf{m}}$. For each individual modeling experiment, we select $\omega \in [1, 100]$ training and 1000 test challenges $\mathbf{c}$ uniformly at random from the subset $\mathcal{C}_{stab} \subseteq \mathcal{C}$ that contains the challenges with the most stable responses r, where $|\mathcal{C}_{\text{stab}}|/|\mathcal{C}| = \rho_{\text{ret}}$. We emphasize that for impersonation purposes, an attacker is only required to predict stable response bits r. In subplot (a), models $\hat{\mathbf{m}}$ are directly derived from $\omega = 1$ transformed challenge $\mathbf{s}$. In subplot (b), we use CMA-ES. Because its randomized training algorithm does not always converge to an accurate model $\hat{\mathbf{m}}$, we only retain the best out of five trials.

We use the same $q = 1000$ transformed challenges $\mathbf{s}_{\text{ref},i}$ for each evaluation of the fitness function. The scale invariance, i.e., $\forall a \in \mathbb{R}_0, \mathsf{fitness}(a\,\hat{\boldsymbol{\delta}}) = \mathsf{fitness}(\hat{\boldsymbol{\delta}})$, is desired for positive factors $a \in \mathbb{R}_0^+$, but the inclusion of negative factors $a \in \mathbb{R}_0^-$ once again implies that one bit of entropy always remains present. Default values suffice for all parameters of the adopted CMA-ES algorithm, e.g., the population size is $4 + \lfloor 3 \ln(\lambda + 1) \rfloor = 16$. For a retention ratio $\rho_{\text{ret}} = 1\%$ and $\omega = 10$ server-defined challenges $\mathbf{c}_i$, Figure 5(b) shows that the best out of two models approaches the ideal accuracy of 100%.

The previously presented modeling techniques are successful despite disregarding the internal specifics of the FSM. For the sake of completeness, we briefly discuss how this disregarded knowledge could facilitate CMA-ES. For a given model $\hat{\mathbf{m}}$ and a given protocol run, the prospective η-bit response $\mathbf{r}$ could be computed. For this sequence of state transitions, the fitness of the best possible match with an available path can then be computed. Numerous path-matching metrics could be devised but, given that our main objective has already been achieved, we abstain from further exploration.

4 Conclusion

A fairly conservative approach to craft a PUF-based authentication protocol is to convert a long response $\mathbf{r} \in \{0,1\}^\eta$ into a secret key $\mathbf{k} \in \{0,1\}^\kappa$ through a fuzzy extractor [DORS08] and then use a keyed cryptographic algorithm to perform the authentication. Protocol designers frequently aim to save resources by avoiding the use of an error-correcting code and/or the cryptographic logic, but this often turns out to be fatal for the system security. Through the use of custom-tailored machine learning techniques, we were able to construct an accurate model of the Arbiter PUF that is used in the protocols of Ye et al. [YHL16] and Gao et al. [GMA+17] and hence enable an impersonation attack.

Future Work

It is not unlikely that, in the near future, an additional protocol might be analyzed and appended to this manuscript.

Acknowledgement

This work is partially funded by the Research Council of KU Leuven through C16/15/058, the Hercules Foundation through AKUL/11/19, and the European Union's Horizon 2020 research and innovation programme under grant number 644052 (HECTOR) and the ERC Advanced Grant 695305.

References

[Bec15a] Georg T. Becker. The gap between promise and reality: On the insecurity of XOR arbiter PUFs. In Tim Güneysu and Helena Handschuh, editors, *17th Workshop on Cryptographic Hardware and Embedded Systems (CHES 2015)*, volume 9293 of *Lecture Notes in Computer Science*, pages 535–555. Springer, September 2015.

[Bec15b] Georg T. Becker. On the pitfalls of using arbiter-PUFs as building blocks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(8):1295–1307, August 2015.

[Del17] Jeroen Delvaux. *Security Analysis of PUF-Based Key Generation and Entity Authentication*. PhD thesis, KU Leuven and Shanghai Jiao Tong University, June 2017.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, March 2008.

[GMA+17] Yansong Gao, Hua Ma, Said F. Al-Sarawi, Derek Abbott, and Damith C. Ranasinghe. PUF-FSM: A controlled strong PUF. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 99(99):99, 2017.

[Han06] Nikolaus Hansen. *The CMA Evolution Strategy: A Comparing Review*, volume 192 of *Studies in Fuzziness and Soft Computing*, pages 75–102. Springer, 2006.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2009.

[LDT00] Keith Lofstrom, W. Robert Daasch, and Donald Taylor. IC identification circuit using device mismatch. In *2000 International Solid-State Circuits Conference (ISSCC)*, pages 372–373. IEEE, February 2000.

[Lim04] Daihyun Lim. Extracting secret keys from integrated circuits. Master's thesis, Massachusetts Institute of Technology, May 2004.

[Mae13] Roel Maes. An accurate probabilistic reliability model for silicon PUFs. In Guido Bertoni and Jean-Sébastien Coron, editors, *15th Workshop on Cryptographic Hardware and Embedded Systems (CHES 2013)*, volume 8086 of *Lecture Notes in Computer Science*, pages 73–89. Springer, August 2013.

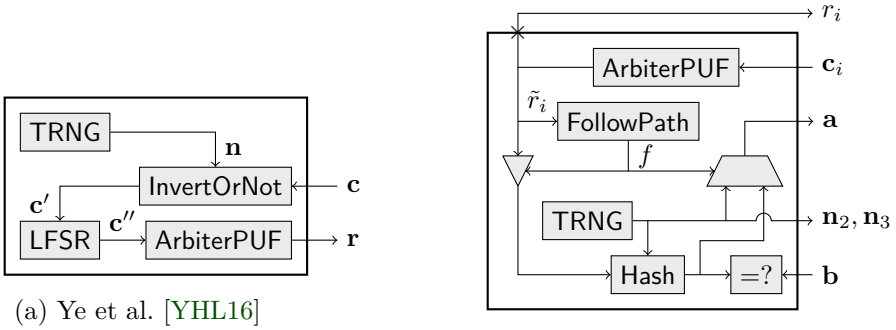
[RSS+13] Ulrich Rührmair, Jan Sölter, Frank Sehnke, Xiaolin Xu, Ahmed Mahmoud, Vera Stoyanova, Gideon Dror, Jürgen Schmidhuber, Wayne Burleson, and Srinivas Devadas. PUF modeling attacks on simulated and silicon data. *IEEE Transactions on Information Forensics and Security*, 8(11):1876–1891, November 2013.

[TB15] Johannes Tobisch and Georg T. Becker. On the scaling of machine learning attacks on PUFs with application to noise bifurcation. In Stefan Mangard and Patrick Schaumont, editors, *RFIDSec 2015: Radio Frequency Identification*, volume 9440 of *Lecture Notes in Computer Science*, pages 17–31. Springer, June 2015.

[YHL16] Jing Ye, Yu Hu, and Xiaowei Li. RPUF: Physical unclonable function with randomized challenge to resist modeling attack. In *1st Asian Hardware Oriented Security and Trust Symposium (AsianHOST 2016)*, pages 1–6. IEEE, December 2016.

A Block Diagrams of a PUF-Enabled Device

To facilitate the understanding of the analyzed authentication protocols for a visually oriented reader, Figure 6 shows the hardware of a PUF-enabled device. The implementation efficiency is reflected but is irrelevant in light of the newly revealed security issues.



(a) Ye et al. [YHL16]

(b) Gao et al. [GMA+17]

Figure 6: The hardware of a PUF-enabled device for the analyzed authentication protocols. Intermediary registers and control logic are not drawn. The symbol × on the boundary of an IC denotes a one-time interface that is irreversibly disabled after its enrollment.