Short Paper: XOR Arbiter PUFs have Systematic Response Bias

Nils Wisiol$^{1[0000-0003-2606-614X]}$ and Niklas Pirnay^{1}

Chair for Security in Telecommunications, Technische Universität Berlin, Germany
nils.wisiol@tu-berlin.de, niklas.pirnay@campus.tu-berlin.de

Abstract. We demonstrate that XOR Arbiter PUFs with an even number of arbiter chains have inherently biased responses, even if all arbiter chains are perfectly unbiased. This rebukes the believe that XOR Arbiter PUFs are, like Arbiter PUFs, unbiased when ideally implemented and proves that independently manufactured Arbiter PUFs are not statistically independent.
As an immediate result of this work, we suggest to use XOR Arbiter PUFs with odd numbers of arbiter chains whenever possible. Furthermore, our analysis technique can be applied to future types of PUF designs and can hence be used to identify design weaknesses, in particular when using Arbiter PUFs as building blocks and when developing designs with challenge pre-processing. We support our theoretical findings through simulations of prominent PUF designs. Finally, we discuss consequences for the parameter recommendations of the Interpose PUF.
Investigating the reason of the systematic bias of XOR Arbiter PUF, we exhibit that Arbiter PUFs suffer from a systematic uniqueness weakness.

Keywords: Physically Unclonable Function · Bias · Arbiter PUF · Interpose PUF

1 Introduction and Related Work

Physically Unclonable Functions (PUFs) are "biometrics" for integrated circuits. Like fingerprints for humans, PUFs should expose (somewhat) unique characteristics of a circuit instance that can be used to identify or even authenticate a particular circuit. The specific characteristics of a circuit are usually formalized as input-output ("challenge-response") behavior. *Strong* PUFs have the additional requirement that each circuit has such a large number of features (input-output pairs) that it is impossible for an attacker to copy and imitate all features.

Research in strong PUFs has spent much attention on Arbiter PUFs, which were introduced by Gassend et al. [3], and its countless variations. While the Arbiter PUF does have an exponentially large challenge-space, Gassend et al. noted that its behavior can be characterized by a hyperplane and is thus an easy target for prediction algorithms trained with machine learning on observed examples. This raises legitimate concern, as in many usage scenarios, such training data could be easily obtained by a man-in-the-middle attacker. Sölter and

Rührmair et al. [13,8] demonstrated that prediction is even possible when multiple Arbiter PUFs are used and only the XOR of the responses is returned. Their attack on the *XOR Arbiter PUF* demonstrated that training of a model is feasible, even though the PUF behavior cannot be characterized by a single hyperplane anymore. This also holds true for the Lightweight Secure PUF by Majzoobi et al. [6] that modifies a given challenge before passing it to an underlying XOR Arbiter PUF. Most recently, Nguyen et al. [7] proposed the Interpose PUF, essentially consisting of two XOR Arbiter PUFs. However, using deep neural networks, successful attacks on XOR Arbiter PUF and Interpose PUF have been claimed [11].

Side-channel-based attacks on XOR Arbiter PUFs have also been successfully mounted. In 2013, Delvaux and Verbauwhede [2] modeled a single Arbiter PUF based on the response reliability. In 2014, Tajik et al. [14] were able to extract physical features of the Arbiter PUF circuit using photonic emission analysis, allowing them to deduce a mathematical model and prediction algorithm for the PUF. In 2015, Becker [1] demonstrated an attack against the "4-way" XOR Arbiter PUF, where four variations of the same challenge are fed in the same Arbiter PUF and the parity of the four responses is output to the user. The attack trains a model using an evolution-strategy algorithm based on the reliability of responses rather than their bit-value.

Hardware implementations of Arbiter PUFs have been extensively studied. Katzenbeisser et al. [4] conducted an analysis of Arbiter PUFs implemented in ASIC, evaluating the reliability of responses and sensitivity to temperature change. In a similar study, Maes et al. [5] studied the uniqueness and reliability of Arbiter PUFs in ASIC under the influence of ageing. Sahoo et al. [10] studied the bias inherent to the *implementation* of a PUF design, considering FPGA implementations of the Arbiter PUF.

This paper is organized as follows. Sec. 2 introduces the additive delay model and contains the theoretical analysis of XOR Arbiter PUF bias. In Sec. 3, we present simulation results to support our analysis and discuss the results and consequences. We conclude the paper in Sec. 4.

2 Bias Analysis

2.1 Background: Additive Delay Model

An Arbiter PUF consists of two symmetric signal paths going through n *stages* before reaching the *arbiter*. At each stage, the signals may be interchanged, depending on the challenge bit that is assigned to this particular stage. The arbiter will output whether there is a signal first on the top or bottom line of its input. An XOR Arbiter PUF consists of k parallel Arbiter PUFs, but only the parity (XOR) of their responses is output to the user. A schematic representation of a 2-XOR Arbiter PUF can be found in Fig. 1.

To model the behavior of XOR Arbiter PUFs, the additive delay model is widely and successfully used [3,13,8,6]; Delvaux and Verbauwhede [2] give a

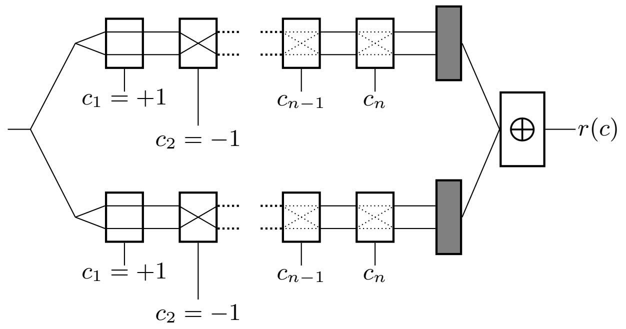


Fig. 1: Schematic representation of an XOR Arbiter PUF with $k = 2$ parallel arbiter chains.

physical motivation. Modeling results above 99% accuracy, as obtained by the modeling attacks mentioned above, show that the additive delay model highly accurately models the physical Arbiter PUF.

Written using -1 and 1 to represent bit values, the additive delay model states that any instance of the Arbiter PUF with n stages can be modeled as an affine hyperplane,[1]

$$r(\boldsymbol{c}) = \operatorname{sgn}\left[\langle \boldsymbol{w}, \boldsymbol{x} \rangle + w_0\right] = \operatorname{sgn}\left[\left(\sum_{i=1}^{n} w_i x_i\right) + w_0\right], \tag{1}$$

where $\boldsymbol{w} \in \mathbb{R}^n$ and $w_0 \in \mathbb{R}$ model the physical properties of the particular instance and $\boldsymbol{x}$ is a function of the given challenge bits $\boldsymbol{c}$ defined by $x_i = c_i \cdot c_{i+1} \cdots c_n$. Note that while the *threshold* w_0 relates to the bias $\mathrm{E}_{\boldsymbol{c}}\left[r(\boldsymbol{c})\right]$ of the Arbiter PUF, the relation of these two values is not linear, as small perturbation of the threshold does not change the bias. We approximate the relation of threshold and bias below.

Building on the Arbiter PUF model, a k-XOR Arbiter PUF can be modeled by the product[2] of k Arbiter PUF models,

$$r(\boldsymbol{c}) = \operatorname{sgn} \prod_{l=1}^{k} \sum_{i=1}^{n} w_{l,i} x_i + w_{l,0}. \tag{2}$$

In the additive delay model, the $k \cdot n$ parameters $w_{l,i}$ and the $w_{l,0}$ are assumed to be normally distributed[3]. XOR Arbiter PUF variants that transform the input challenge before processing it with the arbiter chains, e.g. the Lightweight Secure PUF [6], can be modeled by appropriately augmenting the definition of x.

[1] The *sgn* function returns the sign of the argument. In our setting, $\operatorname{sgn} 0$ will only occur with probability zero; for completeness we define $\operatorname{sgn} 0 = 1$.

[2] Notice that when using -1 and 1 to represent bit values, the standard product of bit values corresponds to the logical XOR operation.

[3] In fact, some parameters have different variances [2], but this is immaterial to the discussion in this paper.

2.2 Analysis

XOR Arbiter PUF bias can be analyzed by expanding the product of the additive delay model (see (2)) to observe the resulting threshold term. In order to focus on the *systematic* bias of XOR Arbiter PUF designs, we assume each Arbiter PUF to be independently chosen and *unbiased.*

The term $\prod_{l=1}^{k} \sum_{i=1}^{n} w_{l,i} x_i$ in (2) is a polynomial of degree k over variables x_i that take values in $\{-1, 1\}$. Hence, $r(\boldsymbol{c})$ is a polynomial threshold function of degree at most k, including some monomials of the form $x_i^k \cdot \prod_{l=1}^{k} w_{l,i}$. If (and only if) k is even, these monomials contribute to the threshold term. When k is odd, no product will degenerate into a constant term, i.e. perfectly unbiased Arbiter PUFs will yield a perfectly unbiased XOR Arbiter PUF. As an example, a 2-XOR Arbiter PUF can then be modeled as

$$r(\boldsymbol{c}) = r_1(\boldsymbol{c}) \cdot r_2(\boldsymbol{c}) = \operatorname{sgn}\left[\sum_{\substack{i,j\\ i \neq j}} w_{1,i} w_{2,j} x_i x_j + \sum_{\substack{i,j\\ i=j}} w_{1,i} w_{2,j}\right]. \tag{3}$$

It can be seen that even assuming unbiased building blocks, we obtain a non-zero threshold term of $\sum_{i=1}^{n} w_{1,i} w_{2,i}$. While the expectation of this value in the manufacturing process is zero, a high variance causes the 2-XOR Arbiter PUF to likely have significant bias. In other words, any 2-XOR Arbiter PUF consisting of two *unbiased* arbiter chains is biased with probability 1.

Theorem 1. *The responses of independently chosen unbiased Arbiter PUFs queried on the same challenge are not statistically independent.*

Proof. Let r_1, r_2 be models of unbiased Arbiter PUFs with parameters $w_i^{(1)}$ and $w_i^{(2)}$ for $1 \leq i \leq n$ chosen independently at random and $w_0^{(1)} = w_0^{(2)} = 0$ as defined in (1). As demonstrated in (3), the threshold of $r_1(\boldsymbol{c}) \cdot r_2(\boldsymbol{c})$ is non-zero and hence[4] $\Pr\left[r_1(\boldsymbol{c}) \cdot r_2(\boldsymbol{c}) = 1\right] \neq 1/2$. However, assuming statistical independence we have $\Pr\left[r_1(\boldsymbol{c}) \cdot r_2(\boldsymbol{c}) = 1\right] = \Pr\left[r_1(\boldsymbol{c}) = r_2(\boldsymbol{c})\right] = 1/2$. $\square$

These results are relevant for novel designs based on several XOR Arbiter PUFs, like the Interpose PUF, as well as for designs based on a single XOR Arbiter PUF, but with novel transformation of the challenge. As an example,

[4] An approximation of the bias $\mathrm{E}_{\boldsymbol{c}}\left[r(\boldsymbol{c})\right]$ in dependence of the threshold value can be obtained using the Berry-Esseen-Theorem to approximate $\sum_{i,j} w_{1,i} w_{2,j} x_1 x_2$ for $i \neq j$ as a Gaussian random variable with variance σ^2 over uniformly chosen random challenges, resulting in $\mathrm{E}_{\boldsymbol{c}}\left[r(\boldsymbol{c})\right] \approx \operatorname{erf}\left(\frac{\sum_{i=1}^{n} w_{1,i} w_{2,i}}{\sigma\sqrt{2}}\right)$;the value $\sum_{i=1}^{n} w_{1,i} w_{2,i}$ in turn follows (in the manufacturing random process) a distribution composed of the sum of product-normal distributions, which has increasing variance for increasing n. Extending the setting, for higher (but even) k the distribution narrows as the variance of the product-normal distribution narrows. The later effect can be observed in our simulations, cf. Fig. 2.

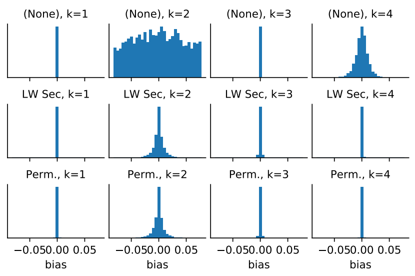
Thm. 1 can easily be extended to cover all input transformations that result in the same challenge for each arbiter chain.

Finally, we emphasize again that the analytical results hold regardless of any implementation weakness and thus are a systematic weakness of the Arbiter PUF design.

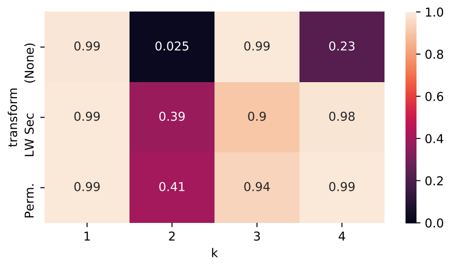
3 Discussion

3.1 Simulation Results

We confirmed the systematic XOR Arbiter PUF bias in simulations[5] for different XOR Arbiter PUF sizes and input transformations, including the Interpose PUF. All simulations are based on the additive delay model with standard Gaussian weights and were conducted using *unbiased* arbiter chains. The distribution of the systematic bias is based on sampling 100 instances each; the bias of each instance is estimated using 1,000,000 responses to uniformly random challenges.



(a) Histogram of bias estimates. A bias value of zero represents perfectly unbiased responses.



(b) Proportion of instances that passed the NIST frequency test at significance level 1%.

Fig. 2: Analysis results for simulated 64-bit k-XOR Arbiter PUFs, k-Lightweight Secure PUFs and k-Permutation XOR Arbiter PUFs build from unbiased Arbiter PUFs. For each type and size, 5000 instances were sampled and queried with one million uniformly random challenges each.

In Fig. 2 we show the estimated bias distribution for XOR Arbiter PUFs and Lightweight Secure PUFs, which confirm our theoretical findings. As expected, the systematic bias is only present for PUFs with an even number of arbiter chains, while PUFs with an odd number of arbiter chains remain (systematically) unbiased. The bias variance becomes smaller as k increases. The statistical significance of these findings can be confirmed by applying a bias test like the one specified in NIST's SP-800-22 test suite [9]: while our simulation

[5] The software used for simulation and analysis publicly available as free software at `https://github.com/nils-wisiol/pypuf/tree/2020-systematic-bias`.

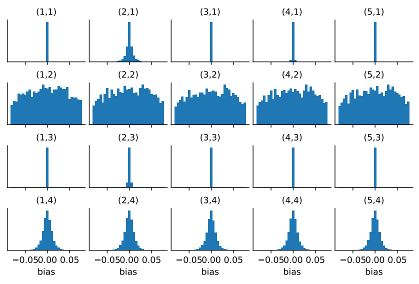
passes the tests on 99% of XOR Arbiter PUF instances whenever k is odd; it fails for almost all instances when $k = 2$, and fails for the majority of instances when $k = 4$ (see Fig. 2). This is also in line with our theoretical findings and simulation results, as we expect the effect to become smaller as k increases.

We hence recommend using an odd number of arbiter chains to avoid potential additional attack surface and especially discourage the use of two or four parallel chains. These recommendations also apply whenever (XOR) Arbiter PUFs are used as building blocks for larger PUFs, such as is the case in the Interpose PUF, as bias in intermediate values can result in increased predictability.

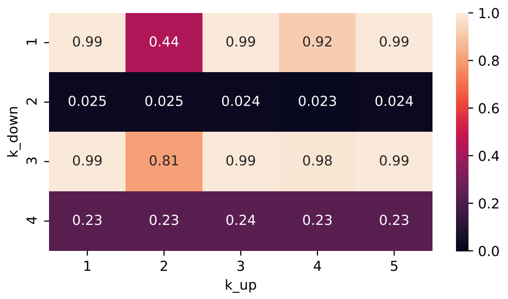
The bias distribution also suggests that the input transformation as done by the Lightweight Secure PUF [6] compensates the systematic bias to some extend, which may be a contributing factor to the increased machine learning resistance [8,15] of the Lightweight Secure PUF. On the other hand, the Lightweight Secure PUF and Permutation XOR Arbiter PUF [15] seems to introduce bias for the case $k = 3$. Such effects should be considered when designing novel input transformations.

Our findings also extend to the recently proposed Interpose PUF [7], which is a combination of two XOR Arbiter PUFs and was designed to be resilient against all state-of-the-art modeling attacks, while being CMOS-compatible. Consisting of two interposed XOR Arbiter PUFs, our simulation shows that the "down" XOR Arbiter PUF plays an important role for the systematic bias, while the "up" XOR Arbiter PUF only has minor influence on it (see Fig. 3).

Given these findings, we provide additional evidence for the original author's advice to use the Interpose PUFs with an odd number of arbiter chains in the lower layer. Furthermore, as our findings are applicable any XOR Arbiter PUF, we extend the parameter recommendation to include the upper layer as well.



(a) Histogram of bias estimates. A bias value of zero represents perfectly unbiased responses.



k_down \ k_up	1	2	3	4	5
1	0.99	0.44	0.99	0.92	0.99
2	0.025	0.025	0.024	0.023	0.024
3	0.99	0.81	0.99	0.98	0.99
4	0.23	0.23	0.24	0.23	0.23

(b) Proportion of instances that passed the NIST frequency test at significance level 1%.

Fig. 3: Analysis results for simulated 64 bit (k_{up}, k_{down})-iPUF instances build from unbiased Arbiter PUFs. For each size, 5000 instances were sampled and queried with one million uniformly random challenges each.

3.2 XOR Arbiter PUF Bias and Arbiter PUF Uniqueness

Our results above, stated in terms of the bias of XOR Arbiter PUFs, are closely related to the uniqueness of Arbiter PUFs. The theoretical and simulation results show that any 64 bit 2-XOR Arbiter PUF has significant bias with high probability, even when implemented ideally (i.e., composed of unbiased Arbiter PUFs). In terms of uniqueness this means that any independently chosen pair of (ideal) Arbiter PUFs has, with high probability, low uniqueness, as perfectly unique Arbiter PUFs are statistically independent and hence their XOR Arbiter PUF would not have any bias. It must be noted though, that, as per the properties of the parity, the uniqueness will only play out in a systematic bias whenever k is even, hence our recommendation to use XOR Arbiter PUFs with odd k.

The inherent low uniqueness of Arbiter PUFs, independent of their implementation, may relate to findings by Schaub et al. [12] that claim an upper bound to the entropy of Arbiter PUFs at $O(n^2)$.

4 Conclusion

In this paper, we exhibited that XOR Arbiter PUFs with an even number of arbiter chains have systematic bias, independently of implementation issues. As bias is an inherent weakness to any PUF, parameter recommendations for XOR Arbiter PUFs, Lightweight Secure PUFs, and Interpose PUFs should no longer include an even number of arbiter chains to remove any additional attack surface.

For future designs, our findings mandate additional testing: it is not sufficient to choose building blocks independently of each other; also their uniqueness must be studied implementation-independent. For designs based on the Arbiter PUF design, our methodology based on the additive delay model may be applicable and facilitate theoretical study of threshold and bias values. For other designs, a different model or empirical testing based on simulation may be necessary.

In future work, we will investigate if other strong PUFs also suffer from implementation-independent uniqueness weaknesses. It should also be investigated if the bias of an XOR Arbiter PUF can assist a modeling attack.

The authors like to thank Ulrich Rührmair for encouraging this research.

References

1. Becker, G.T.: The Gap Between Promise and Reality: On the Insecurity of XOR Arbiter PUFs. In: Güneysu, T., Handschuh, H. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2015. pp. 535–555. Lecture Notes in Computer Science, Springer Berlin Heidelberg (2015)
2. Delvaux, J., Verbauwhede, I.: Side channel modeling attacks on 65nm arbiter PUFs exploiting CMOS device noise. In: Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium On. pp. 137–142. IEEE (2013)
3. Gassend, B., Lim, D., Clarke, D., van Dijk, M., Devadas, S.: Identification and authentication of integrated circuits. Concurrency and Computation: Practice and Experience **16**(11), 1077–1098 (Sep 2004), `https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.805`

4. Katzenbeisser, S., Kocabaş, Ü., Rožić, V., Sadeghi, A.R., Verbauwhede, I., Wachsmann, C.: PUFs: Myth, Fact or Busted? A Security Evaluation of Physically Unclonable Functions (PUFs) Cast in Silicon. In: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Prouff, E., Schaumont, P. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2012, vol. 7428, pp. 283–301. Springer Berlin Heidelberg, Berlin, Heidelberg (2012), http://link.springer.com/10.1007/978-3-642-33027-8_17
5. Maes, R., Rozic, V., Verbauwhede, I., Koeberl, P., van der Sluis, E., van der Leest, V.: Experimental evaluation of Physically Unclonable Functions in 65 nm CMOS. In: 2012 Proceedings of the ESSCIRC (ESSCIRC). pp. 486–489. IEEE, Bordeaux, France (Sep 2012), http://ieeexplore.ieee.org/document/6341361/
6. Majzoobi, M., Koushanfar, F., Potkonjak, M.: Lightweight Secure PUFs. In: Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design. pp. 670–673. ICCAD '08, IEEE Press, Piscataway, NJ, USA (2008), http://dl.acm.org/citation.cfm?id=1509456.1509603
7. Nguyen, P.H., Sahoo, D.P., Jin, C., Mahmood, K., Rührmair, U.: The Interpose PUF: Secure PUF Design against State-of-the-art Machine Learning Attacks p. 48 (2018)
8. Rührmair, U., Busch, H., Katzenbeisser, S.: Strong PUFs: Models, Constructions, and Security Proofs. In: Sadeghi, A.R., Naccache, D. (eds.) Towards Hardware-Intrinsic Security: Foundations and Practice, pp. 79–96. Information Security and Cryptography, Springer Berlin Heidelberg, Berlin, Heidelberg (2010), https://doi.org/10.1007/978-3-642-14452-3_4
9. Rukhin, A., Soto, J., Nechvatal, J., Barker, E., Leigh, S., Levenson, M., Banks, D., Heckert, A., Dray, J.: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. NIST Special Publication 800-22 p. 131 (2010)
10. Sahoo, D.P., Nguyen, P.H., Chakraborty, R.S., Mukhopadhyay, D.: Architectural Bias: A Novel Statistical Metric to Evaluate Arbiter PUF Variants p. 14 (2016), https://eprint.iacr.org/2016/057
11. Santikellur, P., Bhattacharyay, A., Chakraborty, R.S.: Deep Learning based Model Building Attacks on Arbiter PUF Compositions p. 10 (2019)
12. Schaub, A., Rioul, O., Joseph, Boutros, J.J.: Entropy Estimation of Physically Unclonable Functions via Chow Parameters. arXiv:1907.05494 [cs, math] (Jul 2019), http://arxiv.org/abs/1907.05494
13. Sölter, J.: Cryptanalysis of electrical PUFs via machine learning algorithms p. 52 (2009)
14. Tajik, S., Dietz, E., Frohmann, S., Seifert, J.P., Nedospasov, D., Helfmeier, C., Boit, C., Dittrich, H.: Physical Characterization of Arbiter PUFs. In: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Salinesi, C., Norrie, M.C., Pastor, Ó. (eds.) Advanced Information Systems Engineering, vol. 7908, pp. 493–509. Springer Berlin Heidelberg, Berlin, Heidelberg (2014), http://link.springer.com/10.1007/978-3-662-44709-3_27
15. Wisiol, N., Becker, G.T., Margraf, M., Soroceanu, T.A.A., Tobisch, J., Zengin, B.: Breaking the Lightweight Secure PUF: Understanding the Relation of Input Transformations and Machine Learning Resistance p. 9 (2019), https://eprint.iacr.org/2019/799