Secure PUF-based Authentication and Key Exchange Protocol using Machine Learning

Amir Ali Pour, Fatemeh Afghah, David Hely, Vincent Beroulle, Giorgio Di Natale

Secure PUF-based Authentication and Key Exchange Protocol using Machine Learning

Amir Ali-pour*, Fatemeh Afghah †, David Hely‡, Vincent Beroulle§, and Giorgio Di Natale¶
(*, †) Electrical and Comptuer Engineering Department, Clemson University, Clemson, SC 29634, USA
(*, §) Univ. Grenoble Alpes, Grenoble INP, LCIS, 26000 Valence, France
(‡) Univ. Grenoble Alpes, CEA, LETI, F-38000 Grenoble, France
¶ Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, 38000 Grenoble, France

(*, †) Email: {aalipou, fafghah}@clemson.edu
(*, ‡, §, ¶) Email: {firstname.lastname}@univ-grenoble-alpes.fr

***Abstract*—Error Correction Codes and Fuzzy Extractors (FE) using publicly available helper data are used to increase the reliability of the secret value generated from noisy sources such as Physically Unclonable Functions (PUFs). Publicly available helper data is, in turn, vulnerable against Helper Data manipulation attacks due to its correlation with the secret value. Instead of using helper data for FE-based error correction, we propose a locally recoverable repetition coding mechanism. Our proposed mechanism is based on sharing only the user's generated challenge values, which is inherently secure against machine learning and PUF cloning attacks. We evaluate the reliability of our method using simulated challenge response pairs (CRP)s captured from various XOR Arbiter PUF structures at different levels of noise embedded in the PUF CRP characteristic. We show for instance that in a scenario of using PUF with 10% error-rate, our method can successfully recover the encryption key with close to zero failure-rate with a repetition code length of 10 or higher.**

***Index Terms*—Physically Unclonable Function, Encryption Key, Repetition Code, Error Correction Codes**

I. Introduction

As concepts such as PUF-based cryptographic methods are emerging, security in internet-based communication, especially in IoT and cyber-physical systems is facing new opportunities and challenges as well. PUF is considered nowadays as security primitive for resource-constraint ecosystems [1], [2]. PUF is characterized as a hardware-bound function which utilizes the unit-specific micro-variations to generate digital fingerprints. The functionality of PUF is based on mapping a bit-vector challenge to a response and generating a so-called Challenge-Response-Pair (CRP). Two variants of PUF exist, the strong PUF, which generates large amount of CRPs [3], and the weak PUF, which generates only few CRPs [4].

Recently, it has become an interesting research idea to utilize Strong PUF for key generation using machine learning (ML). It is proven that ML methods can create a soft model of the PUF which is accessible on the server side and provides access to the full CRP space of the PUF with negligible error rate. Based on such potential, various methods have been proposed to design protocols for authentication and key generation [5], [6]. While the authentication methods can tolerate error rate of the soft model of PUF, key generation protocols on the other hand still use fuzzy extractor (FE) mechanism to increase the reliability of the generated keys. FE mechanism in turn demands publicizing helper data which has correlation the the secret value that is the source for key generation. This in turn makes the protocol susceptible to helper data manipulation attacks [7].

There are several works that practice similar mechanisms for their PUF based key generation/exchange and authentication. For instance, Majzoobi et al. proposed Slender PUF authentication protocol in [5] where TTP server uses a compact model of the PUF that enables the user to generate CRPs randomly with no restriction in the number of CRPs. However the CRP transmission is disclosed over the public communication channels during authentication process, which ultimately renders the mechanism prone to model-building attacks. Idriss et al. also proposed a lightweight highly secure PUF based device authentication method in [8]. This method is based on only exchanging challenge values and disclose no response value during an authentication operation over the communication channels which makes the method inherently secure against model building attacks. However, they propose using a CRP lookup table on the server side rather than a model of the PUF, then the protocol will ultimately face a limit in terms of the number of CRPs it can use. Quadir et al. present a novel key generation mechanism based on strong PUF in [6]. In their mechanism they use a predictive model of the PUF on the server side and propose also to exchange only challenge values. Thus the collective drawbacks of the first two methods are mitigated here. However, they use FE-based ECC mechanism to ensure the reliability of the key values which require publicly exchanging helper data which is prone ultimately to helper data manipulation attacks. In [9] Yan et al. proposes a PUF-based authentication method which requires no ECCs, instead they use a novel fault tolerant authentication scheme which uses ring weight algorithm (RWA) to assure on the reliability of the used response values for authentication. This method also does not require a CRP dataset. Nonetheless, as instated in the work, the method is based on the trade-off between the efficiency in implementation and performance, and the failure rate in authentication, which makes it best suited for IC authentication rather for key generation.

In this work, we propose a new encoding and decoding method that enables strong PUF for key generation and exchange using machine learning. Our protocol is capable of recovering an original key using no public helper data. Here we propose to use a strong PUF with increased complexity, such as XOR Arbiter PUF. In order to provide the capability of generating a large number of encryption keys, we propose

This material is based upon the work supported by the French National Research Agency in the framework of the "Investissements d'avenir" program (ANR-15-IDEX-02) and the US National Science Foundation under Grant No. IIP-2204502.

using a machine learning-based equivalent model of the PUF on a Trusted Third Party (TTP) verifying server. The ML model of the PUF in turn gives access to its full CRP space with some negligible misprediction probability. Here we show how our method increases the probability of success in key recovery even in the light of model misprediction and PUF instability while exchanging encryption keys using challenge packs that generate mutual response values, a mechanism similar to repetition coding. In this mechanism only a series of randomized challenge values are transmitted which in turn makes the mechanism inherently secure against modeling and replay attacks. Our contribution is listed as below:

- A repetition code-like error correction technique using the PUF and an equivalent ML model of the PUF.
- A one-to-one and one-to-many authentication and key exchange protocol with locally correctable codes.
- An evaluation of our proposed method using simulated data of various XOR Arbiter PUF structures.

The rest of the paper is as follows. Our proposed method is described in section II. In section III we elaborate on the reliability evaluation of our proposed method. In section IV, we discuss the security analysis of our method against different attacks. The conclusion remarks are presented in section V.

II. Proposed Method

We define three phases that constitute our method, 1) The enrollment phase, 2) The challenge-based synchronization phase, and 3) The authentication and key generation and exchange phase.

The process of enrolling a PUF-enabled device to a TTP server is illustrated in Fig. 1. At this phase the TTP server invokes the PUF-enabled device and transmits a challenge set C comprising n randomly generated challenge vectors and acquires a CRP set comprising n CRPs. The CRP set is then used to train a predictive model M which mimics the characteristic of PUF with prediction accuracy $\epsilon > t$, where t is the minimum acceptable prediction accuracy.

The challenge-based synchronization phase is shown in Fig. 2. This is the initial phase for generating a new encryption key. Thus at first, the device queries the TTP server by exchanging its $Device\_ID$. Once received by the TTP server, the corresponding predictive model of the device's PUF is loaded. Then the server generates a random l-bit binary vector w. The generated random value w and the predictive model M are given to a CRP matchmaking algorithm.

The CRP matchmaking algorithm as shown in Algorithm 1 is responsible for generating a set of randomized challenge

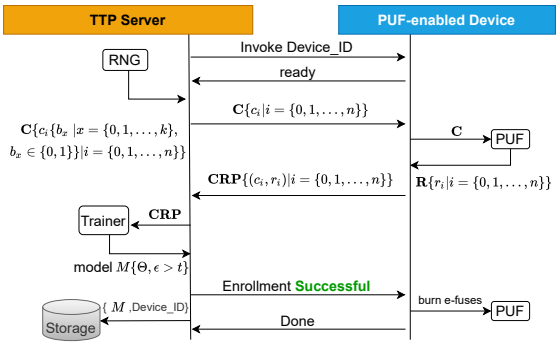

Fig. 1: The PUF enrollment procedure.

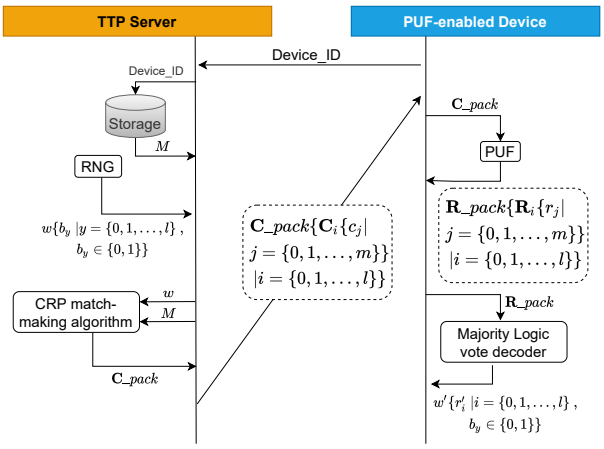

Fig. 2: The challenge-based synchronization procedure.

Algorithm 1 CRP matchmaking algorithm

Require: $w\{b_y|y=\{0,1,...,l\},b_y\in\{0,1\}\}$
Require: $M\{\Theta,\epsilon>t\}$
  $L\leftarrow l$ ▷ size of w
  $M\leftarrow m$ ▷ repetition length
  while $i<L$ **do**
    while $j<M$ **do**
      $ch\leftarrow$ randomly generate challenge vector
      $rp\leftarrow M(ch)$
      if $rp=w[i]$ **then**
        if $ch\notin C\_pack$ **then**
          $j\leftarrow j+1$
          $C\_pack[i]\xleftarrow{+} ch$ ▷ Append ch to C_pack1
    $i\leftarrow i+1$
  return C_pack

vectors we refer to as C_pack, which includes l number of C_i subsets comprising m number of challenges that produce the same response. The CRP matchmaking in turn acts similarly as a repetition code. While the output is not directly the codeword, it is instead the challenge vectors that will lead to the binary response values which then constitute the codeword. This will then allow to securely enable regeneration of the secret value on the device side.

The generated C_pack set is then sent to the device. On the device side, C_pack is passed to the PUF to produce the R_pack set, which comprises l subsets, and each subset comprises m binary response values. Once the R_pack is extracted, each of the subsets is given to a majority voter to vote for the most dominant binary value. The output of the voter for the entire R_pack is w' which comprises l binary values. Here it is expected that w' is equal to w, which is the base condition to issue a successful authentication.

We assume that a request for synchronization initiates a new session. Once completed, the communicating parties can initiate device authentication and key exchange. Fig. 3 shows the process of device authentication and key exchange between the TTP server and the PUF-enabled device. For authentication, both the TTP server and the device create hash values $h1$, and $h1'$, using the $Device\_ID$ and the generated w, and w', respectively. The PUF-enabled device then sends $h1'$ to TTP server. On TTP server, if $h1$ and $h1'$ are equal, it is

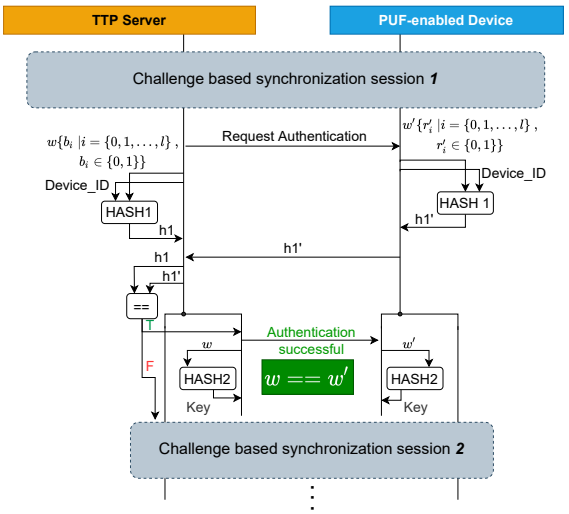

Fig. 3: The authentication and key generation procedure.

assumed the communicating device is authentic and the TTP server sends the authentication successful acknowledgement to the device, meaning that it is known to both parties that their generation w and w' are equal. Both parties then move to creating the encryption key which is a hash value of only the w and w'.

III. Evaluation of Reliability

To evaluate the reliability of our method, we considered using XOR Arbiter PUF which is family of strong PUFs with increased and expandable complexity. The structure of XOR Arbiter PUF is based on multiple Arbiter PUFs whose input (challenge) are of the same size, and triggered by a global input. The output of the XOR Arbiter PUF is also the XOR of the output of each Arbiter PUF. Fig. 4 shows the structure of an n-stage k-XOR XOR Arbiter PUF.

We used a Python-based XOR Arbiter PUF simulator to generate 10 instances of each 4,5,6, and 7 XOR arbiter PUF variations with a 64-bit challenge size. The simulator code is developed by Ruhrmair et al. and presented initially in [10], and is available online in [11]. To represent a realistic characteristic, we added artificial bit flipping characteristic to the captured CRP datasets from the simulated PUF instances. Overall, we considered different noise levels, including 0% 2%, 5% and 10% of CRP population for each dataset to be affected by bit flipping. We assessed the modeling of the

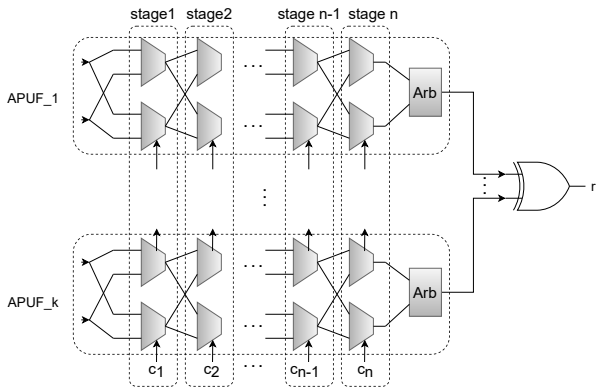

Fig. 4: The structure of an n-stage k-XOR Arbiter PUF.

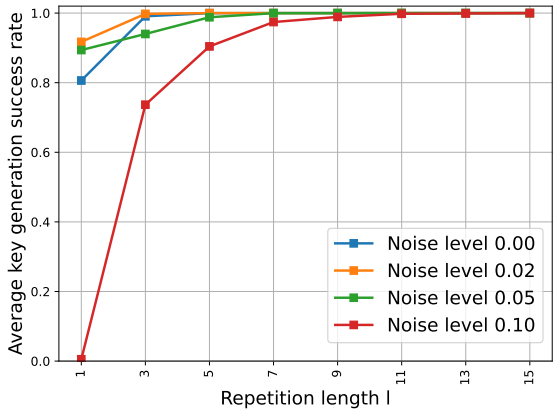

Fig. 5: Key generation Success with respect to code length l.

PUF instances, and the performance of our proposed method separately for each noise level.

Using Pytorch, we created the corresponding predictive models of the PUF instances using a novel Multi-Layer Perceptron (MLP) proposed by Mursi et al. in [12]. We implemented the training procedure as proposed in [13] and we used the initial hyper parameters and a novel transfer learning technique as proposed in [14] to reduce the number of CRPs required for training an accurate model.

We measured the accuracy of the models we trained for each PUF instance. Ideally we consider a model accurate if it has prediction accuracy above 80%. For all XOR sizes, and for each noise level 0%, 2%, 5%, and 10%, we could achieve 99%, 97%, 94%, and 89% prediction accuracy, respectively.

For our measurement purposes, we implemented the enrollment phase, the challenge-based synchronization phase and the key generation and exchange phase on python, by developing the class of a TTP server and a PUF-enabled device, and the exchange mechanism interface between the two classes. We measured the success-rate of key generation for each noise level, with respect to increasing repetition length l, testing the key generation also for 1000 iterations for each case. Fig. 5 shows the success-rate progress with respect to increasing repetition length for various XOR sizes. As expected, we can see that for all XOR sizes, as the noise levels increase, the success rate degrades relatively. Given however, that with increasing l, the success-rate increases as well to compensate for both the noisy PUF source as well as the mis-prediction error-rate in the equivalent MLP model. We can see that for the worst case of having 10% of noisiness in the PUF characteristic, as well 10% mis-prediction error-rate, with repetition length above 10, we can achieve a success-rate in key exchange reaching and stabilizing at 1.0.

IV. Security Analysis

Below we also evaluate the security of our method for various known attacks against PUF-based security protocols.

A. Machine Learning Modeling Attacks

Modeling attacks on strong PUF require eavesdropping on the communicating channel to capture PUF CRPs which is a necessity to build a model of a strong PUF [10], [15]–[17]. It is proven that with enough number of CRPs, it is possible to model any PUF using machine learning methods. If an attacker successfully models a PUF, he will be able to impersonate the

target device and query the server for important information and be able to decrypt them using the model of the PUF. In our key exchange mechanism however, this way of attacking PUF is not possible since only randomly generated challenge values are exchanged, and without the response value for each challenge, modeling the PUF is not possible.

B. Helper Data manipulation attacks

The vulnerability against helper data manipulation (HDM) attacks exists mainly due to the publicity of the helper data. However, in our proposed method we can recreate the code-words using the PUF itself and only publicize challenge values which are randomized and have no correlation to the secret value without any knowledge of (e.g, the predictive model of the PUF) or physical access to the PUF. This inherently makes the method secure against HDM attacks.

C. Side Channel Attacks

PUFs are prone to side channel attacks as well [18], leading to leaking CRP values that enables the attacker to build a model of the PUF. In our proposed method, we suggest one can implement a mechanism that adds correlative noise to the leaking power traces. Here, the designer can employ two PUFs, where one PUF PUF_a is the source of key generation and the other one, PUF_b, is the source of a dummy key generation. We then propose that the TTP server transfers two challenge sets, C_pack_a and C_pack_b, where R_pack_a as the offspring of C_pack_a is logically the opposite of R_pack_b as the offspring of C_pack_b. In other words, we expect:

$$r_i \neq r'_i, r_i \in R\_pack_a, r'_i \in R\_pack_b, i = \{1, 2, ..., (l \times m)\} \quad (1)$$

Once C_pack_a and C_pack_b sets are received on the device side, they are fed to PUF_a and PUF_b, respectively. Once the responses are generated and passed through the majority logic decoder, only the recovered code from PUF_a is used for key generation. This mechanism in turn confuses the captured power traces which are recorded during the decoding process, since for each SET operation performed in generating the secret value w, there is a RESET operation performed as well.

D. Replay Attacks

Replay attacks can practically compromise our authentication and key exchange mechanism if there is no watchdog method that assures no challenge value is being used twice [19]. However, we propose using predictive models that are trained on the server side based on the already generated and transferred challenge values. By keeping a record of already used challenge vectors, we can recognize if a generated challenge value is fresh or not. Such solution can be compact and sit aside to the main PUF enrollment solution we proposed.

V. Conclusion

In this paper, we proposed a secure strong PUF-based lightweight authentication and encryption key generation and exchange mechanism where the secret values generated from the PUF source are recovered using a local decoding mechanism.We showed that our repetition code-like mechanism does not require any helper data, and can be decoded locally using the PUF itself. We later evaluated the performance of our method for various XOR Arbiter PUFs with different levels of noisiness, and showed that for repetition code length of 10 or above, our decoding mechanism can recover the full key value with 100% probability.

References

[1] C. Herder, M.-D. Yu, F. Koushanfar, and S. Devadas, "Physical unclonable functions and applications: A tutorial," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1126–1141, 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6823677/

[2] A. Alipour, V. Beroulle, B. Cambou, J. Danger, G. D. Natale, D. Hely, S. Guilley, and N. Karimi, "Puf enrollment and life cycle management: Solutions and perspectives for the test community," in *2020 IEEE European Test Symposium (ETS)*, 2020, pp. 1–10, ISSN: 1558-1780.

[3] M. S. Alkatheiri, Y. Zhuang, M. Korobkov, and A. R. Sangi, "An experimental study of the state-of-the-art pufs implemented on fpgas," in *2017 IEEE Conference on Dependable and Secure Computing*. IEEE, 2017, pp. 174–180. [Online]. Available: http://ieeexplore.ieee.org/document/8073844/

[4] A. Shamsoshoara, A. Korenda, F. Afghah, and S. Zeadally, "A survey on physical unclonable function (puf)-based security solutions for internet of things," *Computer Networks*, vol. 183, p. 107593, 2020.

[5] M. Majzoobi, M. Rostami, F. Koushanfar, D. S. Wallach, and S. Devadas, "Slender puf protocol: A lightweight, robust, and secure authentication by substring matching," in *2012 IEEE Symposium on Security and Privacy Workshops*, 2012, pp. 33–44.

[6] M. S. E. Quadir and J. A. Chandy, "Embedded systems authentication and encryption using strong puf modeling," in *2020 IEEE International Conference on Consumer Electronics (ICCE)*, 2020, pp. 1–6.

[7] G. T. Becker, "Robust fuzzy extractors and helper data manipulation attacks revisited: Theory versus practice," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 5, pp. 783–795, 2017.

[8] T. Idriss and M. Bayoumi, "Lightweight highly secure puf protocol for mutual authentication and secret message exchange," in *2017 IEEE International Conference on RFID Technology Application (RFID-TA)*, 2017, pp. 214–219.

[9] W. Yan, F. Tehranipoor, and J. A. Chandy, "Puf-based fuzzy authentication without error correcting codes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 9, pp. 1445–1457, 2016.

[10] U. Ruhrmair, F. Sehnke, J. S olter, G. Dror, S. Devadas, and J. u. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proceedings of the 17th ACM conference on Computer and communications security - CCS '10*. ACM Press, 2010, p. 237. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1866307.1866335

[11] http://www.pcp.in.tum.de/code/lr.zip.

[12] K. T. Mursi, B. Thapaliya, Y. Zhuang, A. O. Aseeri, and M. S. Alkatheiri, "A fast deep learning method for security vulnerability study of XOR PUFs," *Multidisciplinary Digital Publishing Institute (MDPI) Electronics*, vol. 9, no. 10, p. 1715, 2020, number: 10 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2079-9292/9/10/1715

[13] A. Ali-Pour, D. Hely, V. Beroulle, and G. Di Natale, "Strong puf enrollment with machine learning: A methodical approach," *Electronics*, vol. 11, no. 4, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/4/653

[14] A. Ali Pour, D. Hely, V. Beroulle, and G. Di Natale, "An Efficient Approach to Model Strong PUF with Multi-Layer Perceptron using Transfer Learning," in *International Symposium on Quality Electronic Design (ISQED 2022)*, IEEE, Ed. Virtual event, United States: IEEE, Apr. 2022. [Online]. Available: https://hal.archives-ouvertes.fr/hal-03599336

[15] J.-Q. Huang, M. Zhu, B. Liu, and W. Ge, "Deep learning modeling attack analysis for multiple fpga-based apuf protection structures," in *2018 14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT)*. IEEE, 2018, pp. 1–3. [Online]. Available: https://ieeexplore.ieee.org/document/8565728/

[16] M. Khalafalla and C. Gebotys, "PUFs deep attacks: Enhanced modeling attacks using deep learning techniques to break the security of double arbiter PUFs," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019, pp. 204–209. [Online]. Available: https://ieeexplore.ieee.org/document/8714862/

[17] K. T. Mursi, Y. Zhuang, M. S. Alkatheiri, and A. O. Aseeri, "Extensive examination of XOR arbiter PUFs as security primitives for resource-constrained IoT devices," in *2019 17th International Conference on Privacy, Security and Trust (PST)*. IEEE, 2019, pp. 1–9. [Online]. Available: https://ieeexplore.ieee.org/document/8949070/

[18] X. Xi, A. Aysu, and M. Orshansky, "Fresh re-keying with strong pufs: A new approach to side-channel security," in *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2018, pp. 118–125.

[19] C. Gu, C.-H. Chang, W. Liu, S. Yu, Y. Wang, and M. O'Neill, "A modeling attack resistant deception technique for securing lightweight-puf-based authentication," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 6, pp. 1183–1196, 2021.