Fortified-Edge 4.0: A ML-Based Error Correction Framework for Secure Authentication in Collaborative Edge Computing

Seema G. Aarella
University of North Texas
Denton, Texas, USA
seema.aarella@unt.edu

Venkata P. Yanambaka
Texas Woman's University
Denton, Texas, USA
vyanambaka@twu.edu

Saraju P. Mohanty
University of North Texas
Denton, Texas, USA
saraju.mohanty@unt.edu

Elias Kougianos
University of North Texas
Denton, Texas, USA
elias.kougianos@unt.edu

ABSTRACT

Physical Unclonable Functions (PUFs) are widely researched in the field of security because of their unique, robust, and reliable nature, PUFs are considered device-specific root keys that are hard to duplicate. There are many variants of PUFs that are being studied and implemented including hardware and software PUFs. Though PUFs are believed to be secure and reliable, they are not without challenges of their own. The efficient performance of PUF depends on various environmental factors, which leads to inefficiency. Bit flipping is one such problem that can bring down the reliability of the PUF. Memory-based PUFs are prone to unavoidable bit flips occurring in the hardware, similarly, sensor-based PUFs are prone to bit flips occurring due to temperature variation. The number of errors in the PUF response must be minimized to improve the reliability of the PUF in security applications. In this research we explore the Machine Learning (ML) model based on K-mer sequencing to detect and correct the bit flips in the PUFs, hence fortifying the PUF-based secure authentication system for authentication and authorization of Edge Data Centers (EDC) in a Collaborative Edge Computing (CEC) Environment.

CCS CONCEPTS

• **Security and privacy → Security in hardware**; *Distribute tems security*; • **Computing methodologies → Machine l ing algorithms**.

KEYWORDS

Collaborative Edge Computing (CEC), Cybersecurity, Securit Design (SbD), Hardware Assisted Security (HAS), Physical Un able Functions (PUF), Machine Learning (ML), Error Dete Error Correction

GLSVLSI '24, June 12–14, 2024, Clearwater, FL, USA

ACM ISBN 979-8-4007-0605-9/24/06
https://doi.org/10.1145/3649476.3660384

ACM Reference Format:
Seema G. Aarella, Venkata P. Yanambaka, Saraju P. Mohanty, and Elias Kougianos. 2024. Fortified-Edge 4.0: A ML-Based Error Correction Framework for Secure Authentication in Collaborative Edge Computing. In *Great Lakes Symposium on VLSI 2024 (GLSVLSI '24), June 12–14, 2024, Clearwater, FL, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649476.3660384

1 INTRODUCTION

The hardware security primitive corresponds to the use of different types of PUFs, such as Memory PUF, Ring Oscillator (RO) PUF, Arbiter PUF, Magnetic PUF, Optical PUF, and RF PUF. PUFs are used in security applications where the unique response of PUF to a given challenge is used as a feature for device identification and authentication. PUF response however needs post-processing to improve the reliability as compared to the raw response [15]. A reliable PUF is expected to generate the same unique response for a given identical challenge, however, the instability in the environment or the PUF architecture causes the response to be different. For that reason, the PUF cannot be considered reliable enough to be used for cryptographic key generation and other security applications.

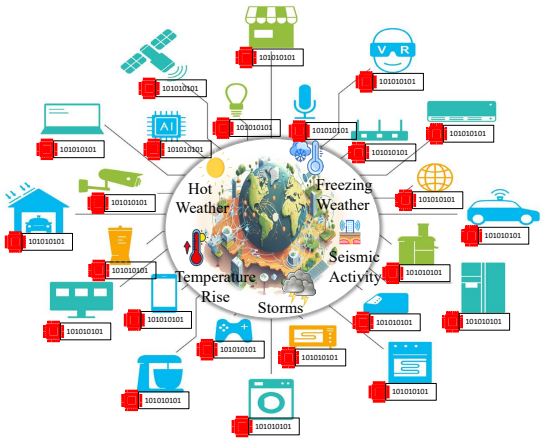

Figure 1: Environmental Effects on PUF enabled IoT Devices

One of the measures taken to ensure the reliability of PUF is to correct the erroneous bits generated by environmental factors like

temperature variations, voltage variations, and aging effects. Based on the geographic location the changes in environment affects the performance of the PUF, as shown in the Figure 1. It has been studied that change in temperature causes the PUF response bits to flip, resulting in erroneous response. Therefore, it is important to have an error correction system in place. An overview of the PUF [illegible]vn in Figur

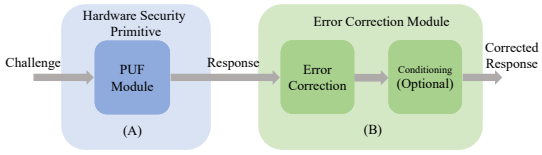

Figure 2: PUF Security Model for Bit Error Correction

Many error-correcting modules can help generate the correct responses by using the priorly obtained correct responses as helper data. However, these modules will need more area overhead and processing power, which is a drawback. Noisy bits, or error bits in the PUF response can be corrected using the helper bits, also called a syndrome. The syndrome or helper data is public information, it is computed based on the PUF response and later sent along with challenges to perform error correction on response bits. To prevent the leakage of secret bits pattern matching algorithms use the PUF function along with helper data for key generation [17]. Error correction schemes that do not involve helper data and uses the initial PUF response as a codeword are studied, it involves coding and decoding algorithms for reconstruction of PUF response, however the practical implications, and application to various use cases are yet to be studied [13].

Edge-based authentication systems where PUFs are employed as a lightweight robust key generation module for authentication of edge devices and data centers are one of the areas that will benefit from a lightweight yet reliable PUF. Fortified Edge proposes a security scheme based on XORArbiter PUF for EDC authentication, the system is further fortified with Certificate Authority based EDC authentication using SRAM PUFs [2]. This research work is towards improving the reliability of the PUFs that can be employed in the CEC environment at the edge.

Machine learning based solutions for edge security that involves authentication and authorization are researched and it is established that they have low area overhead, low computational overhead , have the advantage of learning from new inputs, and the flexibility to adopt suitable ML models based on requirements [1].

The paper is organized as follows: Related prior research in section 2, novel contributions of the current paper is discussed in section 3, problems and solutions proposed in section 4, need for secure authentication and authorization is collaborative edge computing is discussed in section 5, proposed framework in section 6, 7 discusess the experimental setup and section 8 has discussions on results and analysis followed by conclusions in section 9.

2 RELATED PRIOR RESEARCH

A PUF is a digital circuit that possesses an intrinsic randomness due to process variations during manufacturing, making it a unique feature for representing a unique key, thus providing an unclonable identity for each chip. To ensure the reliability of the PUF various error-correcting methods are used to address the PUF errors, such as Replication-based redundancy, Error Correcting Codes (ECC), Fuzzy extractors so on. A PUF-based authentication scheme for Internet-of-Vehicles (IoV) is proposed in the research [18] demostrates PUFs can resist external attacks, eliminate storage requirements, increase security, reduce area overhead and computational overhead.

A novel Slender PUF is proposed in the research [12] that uses a substring matching method to authenticate the responses generated from a strong PUF with minimal information leakage. An automatic self checking and healing (ASCH) system is proposed in research [9] that removes all unstable PUFs, achieves ultra low bit error rate (BER),this method does not need expensive temperature sweeps to find unstable bits and reduces the cost of using ECC while being enery efficient.

The paper [11] focuses on complexities of error correction methods for PUFs, considering requirements like response bits, helper data bits, clock cycles and FPGA slices. A stastical Arbiter PUF model was constructed to generate reliable responses from select challenges independent of environmental condidtions, even if the PUF error rate was high. This research proposes to selectively pick the challenges that generate bit error free responses [8].

A comprehensive study on Helper Data Algorithms (HDAs) is done in the research [7], where it discusses the various HDAs, their efficiency, and open problems in the HDA-based error correction scheme implementation. The research emphasizes global optimization, secure testing, security against physical attacks, leakage reduction, and so on. It also reveals various new threats regarding helper data leakage and manipulation.

A Configurable RO PUF (CRO PUF) is proposed in the research [6], that has a bit error rate as low as $1x10^{-6}$, which eliminates the need for ECC during key generation. This is achieved by improving the existing CRO design to increase the number of possible configurations to 2^{40} configurations per CLB tile. Eliminating the ECC module will reduce the area overhead and power consumption, the method achieved 100% reliability within the absolute maximum rated voltage ranges of an FPGA.

A novel dynamic soft-decision fuzzy extractor is proposed in the research [16], which utilizes the Gaussian error function to dynamically derive the reliability of each PUF response bits. The model was proposed for reducing post-processing overhead in CMOS image sensor PUFs and enabling cryptographic security of camera output images. The use of machine learning-based error detection and correction methods gives more flexibility and deployability options to the security system. Some of the research that uses ML for improving the reliability of PUFs through various methods of error detection, correction, and authentication process are listed in Table 1.

3 NOVEL CONTRIBUTIONS OF THE CURRENT PAPER

The proposed research is based on ML for PUF bit error detection and correction, the model can be deployed at the device end to

Table 1: Comparative Table for State-of-the-Art Literature.

Research	Year	ML Algorithm	Application
Upadhyaya et. al. [20]	2019	Natural Redundancy decoders based on Machine Learning	Error Correction
Suragani et. al. [19]	2022	Proof-of-Concept using CNN	Classification of corrupted PUF responses
Chatterjee et. al. [5]	2020	Random Forest based PUF Calibration scheme	Validate sensor data
Najafi et. al. [14]	2021	Deep CNN	Recognize PUF responses under error conditions
Wen et. al. [21]	2017	Fuzzy Extractor	PUF reliability
Current Research Fortified-Edge 4.0	2024	K-mer Sequence	PUF bit error correction

improve the reliability of the PUF. The following are the novel contributions of the research:

- Novel machine learning method for bit error detection and correction
- Data preprocessing done through visualization, data cleaning
- Sequencing methods used in DNA sequencing and Natural Language Processing (NLP) applied to split PUF response into chunks of sequences
- K-mer function used for vectorization of the sequences
- MultinomialNB classifier used for classification
- A deployable working model that can predict the correct response from the response with error

4 PROBLEMS ADDRESSED AND SOLUTIONS PROPOSED

Bit error correction in the response bits generated by the PUF is an important feature in generating secure and reliable PUF responses in cryptographic key generation and authentication applications that use PUF as a lightweight hardware security primitive. However, some of the problems associated with bit error or noisy bit correction are listed as follows:

- Area overhead added by the bit error correction module
- Computational overhead
- Extensive error correction schemes do not suit the lightweight aspect of the security system
- Data leakage issues related to helper data in schemes that use helper bits
- Secure storage of the helper data, an added feature making the design complex

To overcome the said disadvantages, a novel machine learning-based bit error correction scheme is proposed in this research that has the following features.

- The area overhead and computational overhead is low as the trained model is used at the device end
- There is no need to store the helper data
- Helper data leakage is not an issue as the trained model is deployed at the device end
- The ML model is highly accurate in correcting the erroneous response bits

5 FORTIFIED-EDGE ECOSYSTEM

Authentication and Authorization attacks accounts for critical security threat at the edge computing layer that can affect the user devices, edge devices and edge computing resources. CEC is an emerging paradigm where edge resources are utilized for faster and local processing by resource sharing or task offloading through a process called load balancing. During Load balancing the EDCs in the environment off load tasks to other available EDC, but first the EDCs must authenticate each other for security purpose. Fortified Edge is a scheme that is independent of cloud and authentication process happens at the edge. The use of PUF is to keep the computational requirements low, while keeping up the robustness.

Fortified-Edge model using PUF for authentication was implemented and test for effectiveness at the edge, Initial algorithm was devloped using XORArbiter PUF, 64 bit CRP was used to authenticate the EDCs. Arbiter PUF utilizes the property of intrinsic delay variations of each device to generate a unique identity. It is best suited for lightweight hardware security, it is a delay based model consisting of a path-swapping switch block and an arbiter circuit [10].

Further the model was improved in Fortified-Edge 1.0 to use SRAM PUF based certificate authority scheme to reduce the data security issue due to storing of CRP dataset locally. To enhance the secure authentication scheme and monitoring the EDCs, ML based monitoring and authentication verification scheme was introduced in Fortified-Edge 2.0. Fortified-Edge 3.0 tested various lightweight ML algorithms suitable for processing at the edge. Progressive research done on fortifying the EDCs are listed in the Table 2. The current research focuses on integration of hardware and software to improve the reliability of Fortified-Edge providing added security features without adding extra area overhead.

Current proposed work focuses on improving the reliability of Fortified-Edge model by integrating the error detection and correction scheme.

6 PROPOSED FORTIFIED-EDGE 4.0

The proposed framework for PUF response bit error detection and correction is shown in Figure 4. The PUF module along with the error correction system is part of the hardware security primitive, the CRP dataset consists of pre-obtained responses for a set of

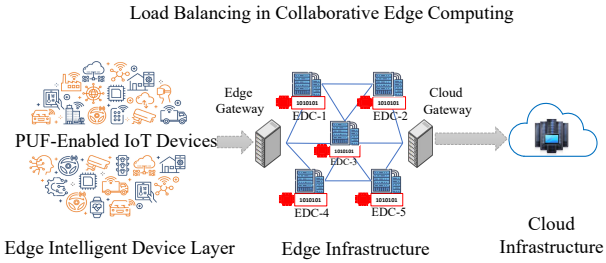

Figure 3: Secure authentication of EDC in collaborative edge computing

Table 2: Comparative Table for Fortified-Edge Research.

Research	Algorithm	Application	Accuracy(%)
Fortified-Edge 1.0 [4]	SRAM PUF-based Certificate	EDC Authentication	NA
Fortified-Edge 2.0[3]	SVM	ML-based Authentication & Monitoring	100
Fortified-Edge 3.0[1]	Lightweight ML models	Anomaly & Intrusion detection	99.33
Current Research Fortified-Edge 4.0	K-mer Sequence	PUF Response Bit Error Correction	99.74

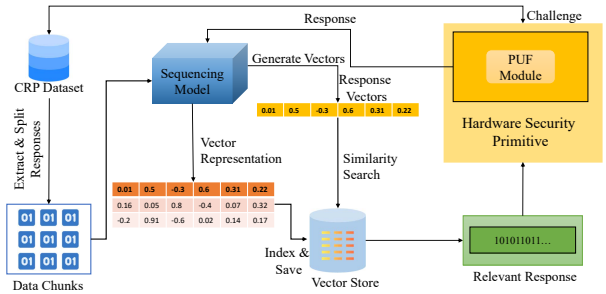

Figure 4: Proposed framework for PUF bit error detection and correction

challenges. The data that is the responses are divided into smaller chunks using K-mers, K-mers is used largely in studies of DNA sequencing, where a substring is extracted from a larger string of data. For applications like error correction a shorter K-mer is preferred, the K-mer considered in this sequencing model is of the size 6.

The extracted words are vectorized, where each row in the matrix represents a document and each column represents a unique word (n-gram), the collection of the vectors is visually represented as the Vector store in the Figure 4. The ML model is trained with a 100K dataset, consisting of 100 responses generated for each challenge, for a total of 1000 challenges.

The words are vectorized using the CountVectorizer class from the scikit-learn library in Python, the vector-matrix columns represent the unique word or n-gram, rows represent the document, and the values in the matrix represent how many times each word appears in each document. multinomialNB of the Naive Bayes is used as the classifier and the model is fit, it is a probabilistic classification algorithm based on Bayes' theorem, particularly used for classification tasks where features represent the frequency of events.

The ML model is trained on the PUF CRP dataset, the model groups the responses into classes based on the sequences, the model is trained to predict the correct response for each challenge by identifying the class of the new response and comparing it with the original classes of the errorless response bits.

7 EXPERIMENTAL SETUP

This research uses the 64-bit Arbiter PUF architecture. PUFs. PYNQ™ Z2 FPGA which is based on Xilinx Zynq C7Z020 SoC was used for PUF implementation. Also, Xilinx BASYS3 FPGA was used to build the PUF. The key metrics used to verify the performance of PUFs are Uniqueness, Randomness and Hamming Distance (HD). The uniqueness of the PUF is a measure of the average interchip HD of the response, and ideal PUF should have 50% uniqueness. Randomness is the measure of the balance between the number of "1"s and "0"s in the reponse. The performance of the arbiter PUF used in this research was measured, and it showed 49.52% Uniqueness, 86.85% Randomness, and 45.67% inter-HD.

The process flow of the implemented ML model for PUF bit error detection and correction using the K-mers function for vectorization of the bit sequences and classification using multinomial Naive Bayes method is as shown in Figure 5. The simple representation of the PUF bit error correction method is represented in Figure 6, where the model reads in a PUF response with error and predicts the corrected/actual response for a given challenge. 1000 challenges were given to the PUF 100 times each and responses were generated for dataset. 80% was used for training and 20% was used for testing from tl

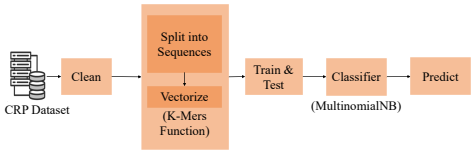

Figure 5: Process flow of the implemented machine learning model

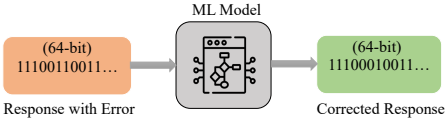

Figure 6: Error correction process using the machine learning model

The 100K response data is initialized as string data and each set of 100 responses is grouped into a class, thus giving us 1000 classes. The top 10 classes with 100 responses in each class as identified by the algorithm are shown in Figure 8. This classification is the proof that the algorithm can accurately group 100 responses of 1 challenge into a unique class, which helps to retrieve the original response from a response which has bit error, by identifying the class to which it belongs. K-mers are applied on the response strings to generate sequences and converted to human texts called words, a total of 511 features are generated from 100000 human texts.

The K-mer of size 6 generates 51 unique sequences as shown in Figure 7. The y-axis represents the "K-mer Index" ranging from 0 to the total number of K-mers minus 1. The x-axis presents "Bit Position", indicating the position of each bit (0 or 1) within the binary sequence. The visual representation of the K-mer sequences helps to study the pattern distribution and examine bit positions for diversity, consistency, and other structural information. The top 10 actual classes and the predicted classes are shown in the confusion matrix in Figure 9.

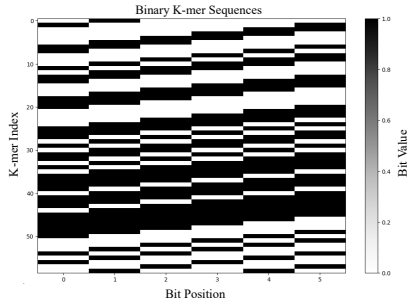

Figure 7: Binary K-mer sequences generated from 64-bit responses

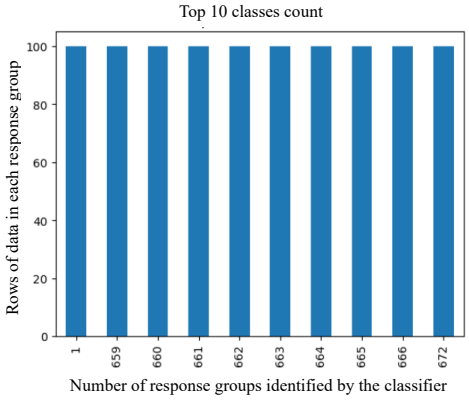

Figure 8: Classification of sequences into classes

8 RESULTS AND ANALYSIS

The model is trained on a 100K dataset, with 80% data used for training and 20% data used for testing. The model can predict the

[1000 rows x 1000 columns]

Predicted	554	958	47	164	309	397	498	883	24	118
Actual										
554	33	0	0	0	0	0	0	0	0	0
958	0	32	0	0	0	0	0	0	0	0
47	0	0	31	0	0	0	0	0	0	0
164	0	0	0	31	0	0	0	0	0	0
309	0	0	0	0	31	0	0	0	0	0
397	0	0	0	0	0	31	0	0	0	0
498	0	0	0	0	0	0	31	0	0	0
883	0	0	0	0	0	0	0	31	0	0
24	0	0	0	0	0	0	0	0	30	0
118	0	0	0	0	0	0	0	0	0	30

Figure 9: Confusion matrix showing the top 10 classes

class of the response accurately, which is then compared to the actual class of the response, as shown in Figure 10. In case any error is found in the response bit, the corrected response is displayed. The trained model is deployed on Raspberry Pi 4 and both training and prediction analysis is done. The algorithm is evaluated for accuracy, precision, recall, and F1-score, for each of the metrics it gives 100% results. The ML model is tested on the 10K dataset first and later tested on the 100K dataset, both giving the same results on the evaluation metrics.

To analyze the coverage of K-mers, the coverage rate was calculated, the coverage rate gives insights into the percentage of unique K-mers in the dataset, it was estimated to be 2.28% for the 6-mers used, suggesting that, on average, 2.28% of all possible unique 6-mers are present in the dataset. This could be due to the highly conserved nature of the dataset, the size of the dataset causing this issue can be ruled out as the dataset used is larger. By increasing the size of the K-mers it was observed that the coverage rate increases significantly.

To test for overfitting of the model, a KFold cross-validation was done, to assess the model's performance on multiple folds of data, the cross-validation scores obtained are 99.78%, 99.74%, 99.70%, 99.75%, 99.73%, with a mean accuracy of 99.74%. The cross-validation scores are consistently high indicating the model is performing well across different folds of data. The mean accuracy of 99.74% suggests that the model is effective in making accurate predictions and can generalize well to unseen data.

The algorithm is also modified to look up the unique challenges in a dataset and retrieve the corresponding challenge for a given response for further verification of the accuracy of the predicted response for the given input challenge as represented in Figure 11.

The process of training and prediction of new data is analysed for time and power consumed. The total training time is 30.63 seconds, and the power consumed for training on a Raspberry Pi 4 is 4.6-4.7 Watts, in actuality a trained model will be deployed to an edge device, the training is done on Raspberry Pi for analysis purpose only. The total execution time to predict bit error and correct the error is 0.08 seconds, with processing speed of 13.15 sequences per second and processing power of 0.28 sequences per character.The idle power of Raspberry Pi 4 was in range of 3.4-3.6 Watts, the total power consumed for bit error detection and correction is an average of 4.1 Watts.

Table 3 shows the results of various ML algorithms used for improving the reliability of PUFs using methods like error detection, correction, classification, and authentication processes that exclude the need for large area overhead ECC modules.

```
Input Response:      11001001100000101011110000111010001101111100001110001O
Predicted Class: 293
************************************************************
Actual Class:  293
Corrected Response:  110010011000001010111100001110100011011111000011100010
```

Figure 10: Result of the algorithm showing the input response and corrected response

```
Predicted ID:  293
    ID                                                Challenge  Corresponding_Response
0  293  1001000111011010111111000011111101001001100...  1100100110000010101111000011101000110111110000...
```

Figure 11: Result showing the corresponding challenge to the corrected response

Table 3: Comparative Table for State-of-the-Art Literature.

Research	Year	Algorithm	Accuracy(%)
Upadhyaya et. al. [20]	2019	Natural Redundancy decoders based on Machine Learning	NA
Suragani et. al. [19]	2022	Proof-of-Concept using CNN	97.34
Chatterjee et. al. [5]	2020	Random Forest based PUF Calibration scheme	90.00
Najafi et. al. [14]	2021	Deep CNN	94.90
Wen et. al. [21]	2017	Fuzzy Extractor	98.00
Current Research Fortified-Edge 4.0	2024	K-mer Sequence	99.74

9 CONCLUSIONS

This research proposes a novel K-mer sequence-based bit error detection and correction algorithm for correcting the PUF responses. The stability of the PUF response increases the reliability of the PUF when employing it in security and cryptographic applications. The power and time analysis proves that the ML model is low power consuming, faster in processing, making it suitable for EDC Authentication in resource constrained environment at the edge. The multinomialNB classifier used is fast and computationally efficient as well.

For future research, we are considering using this reliable PUF architecture for deepfake detection or prevention. This PUF module can be used as a device authenticator if installed in the camera module to identify the device. The machine learning model can be used as a verifier for the images generated from the authorized device.

REFERENCES

[1] Seema G. Aarella, Saraju P Mohanty, and Elias Kougianos. 2023. Fortified Edge 3.0: A Lightweight Machine Learning Based Approach for Security in Collaborative Edge Computing. In *21st OITS International Conference on Information Technology (OCIT)*. Raipur, India, 6.

[2] Seema G. Aarella, Saraju P. Mohanty, Elias Kougianos, and Deepak Puthal. 2022. PUF-based Authentication Scheme for Edge Data Centers in Collaborative Edge Computing. In *IEEE International Symposium on Smart Electronic Systems (iSES)*. 433–438. https://doi.org/10.1109/iSES54909.2022.00094

[3] Seema G. Aarella, Saraju P. Mohanty, Elias Kougianos, and Deepak Puthal. 2023. Fortified-Edge 2.0: Machine Learning based Monitoring and Authentication of PUF-Integrated Secure Edge Data Center. In *2023 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. 1–6. https://doi.org/10.1109/ISVLSI59464.2023.10238517

[4] Seema G. Aarella, Saraju P. Mohanty, Elias Kougianos, and Deepak Puthal. 2023. Fortified-Edge: Secure PUF Certificate Authentication Mechanism for Edge Data Centers in Collaborative Edge Computing. In *Proceedings of the Great Lakes Symposium on VLSI 2023* (Knoxville, TN, USA) *(GLSVLSI '23)*. Association for Computing Machinery, New York, NY, USA, 249–254. https://doi.org/10.1145/3583781.3590249

[5] Urbi Chatterjee, Soumi Chatterjee, Debdeep Mukhopadhyay, and Rajat Subhra Chakraborty. 2020. Machine Learning Assisted PUF Calibration for Trustworthy Proof of Sensor Data in IoT. *ACM Trans. Des. Autom. Electron. Syst.* 25, 4, Article 32 (jun 2020), 21 pages. https://doi.org/10.1145/3393628

[6] Hayden Cook, Zephram Tripp, Brad Hutchings, and Jeffrey Goeders. 2023. Improving the Reliability of FPGA CRO PUFs. In *33rd International Conference on Field-Programmable Logic and Applications (FPL)*. 311–316. https://doi.org/10.1109/FPL60245.2023.00053

[7] Jeroen Delvaux, Dawu Gu, Dries Schellekens, and Ingrid Verbauwhede. 2015. Helper Data Algorithms for PUF-Based Key Generation: Overview and Analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 34, 6 (2015), 889–902. https://doi.org/10.1109/TCAD.2014.2370531

[8] Yansong Gao, Hua Ma, Gefei Li, Shaza Zeitouni, Said F. Al-Sarawi, Derek Abbott, Ahmad-Reza Sadeghi, and Damith Chinthana Ranasinghe. 2017. Exploiting PUF Models for Error Free Response Generation. *ArXiv* abs/1701.08241 (2017). https://api.semanticscholar.org/CorpusID:13979390

[9] Yan He, Dai Li, Zhanghao Yu, and Kaiyuan Yang. 2023. ASCH-PUF: A "Zero" Bit Error Rate CMOS Physically Unclonable Function With Dual-Mode Low-Cost Stabilization. *IEEE Journal of Solid-State Circuits* 58, 7 (2023), 2087–2097. https://doi.org/10.1109/JSSC.2022.3233373

[10] S. Hemavathy and V. S. Kanchana Bhaaskaran. 2023. Arbiter PUF—A Review of Design, Composition, and Security Aspects. *IEEE Access* 11 (2023), 33979–34004. https://doi.org/10.1109/ACCESS.2023.3264016

[11] Matthias Hiller, Ludwig Kürzinger, and Georg Sigl. 2020. Review of error correction for PUFs and evaluation on state-of-the-art FPGAs. *Journal of Cryptographic Engineering* 10, 3 (01 Sep 2020), 229–247. https://doi.org/10.1007/s13389-020-00223-w

[12] Mehrdad Majzoobi, Masoud Rostami, Farinaz Koushanfar, Dan S. Wallach, and Srinivas Devadas. 2012. Slender PUF Protocol: A Lightweight, Robust, and Secure Authentication by Substring Matching. In *IEEE Symposium on Security and Privacy Workshops*. 33–44. https://doi.org/10.1109/SPW.2012.30

[13] Sven Müelich and Martin Bossert. 2016. A New Error Correction Scheme for Physical Unclonable Functions. *ArXiv* abs/1611.01960 (2016). https://api.semanticscholar.org/CorpusID:14597000

[14] Fatemeh Najafi, Masoud Kaveh, Diego Martín, and Mohammad Reza Mosavi. 2021. Deep PUF: A Highly Reliable DRAM PUF-Based Authentication for IoT Networks Using Deep Convolutional Neural Networks. *Sensors* 21, 6 (2021). https://doi.org/10.3390/s21062009

[15] Mi-Kyung Oh, Sangjae Lee, Yousung Kang, and Dooho Choi. 2023. Implementation and characterization of flash-based hardware security primitives for cryptographic key generation. *ETRI Journal* 45, 2 (2023), 346–357.

[16] Shunsuke Okura, Ryota Ishiki, Masayoshi Shirahata, Takaya Kubota, Mitsuru Shiozaki, Kenichiro Ishikawa, Isao Takayanagi, and Takeshi Fujino. 2018. A Dynamic Soft-Decision Fuzzy Extractor for a CMOS Image Sensor PUF. In *International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*. 54–59. https://doi.org/10.1109/ISPACS.2018.8923459

[17] Zdenek Paral and Srinivas Devadas. 2011. Reliable and efficient PUF-based key generation using pattern matching. In *IEEE International Symposium on Hardware-Oriented Security and Trust*. 128–133. https://doi.org/10.1109/HST.2011.5955010

[18] Pintu Kumar Sadhu, Venkata P. Yanambaka, Saraju P. Mohanty, and Elias Kougianos. 2022. Easy-Sec: PUF-Based Rapid and Robust Authentication Framework for the Internet of Vehicles. arXiv:2204.07709 [cs.CR]

[19] Reshmi Suragani, Emiliia Nazarenko, Nikolaos Athanasios Anagnostopoulos, Nico Mexis, and Elif Bilge Kavun. 2022. Identification and Classification of Corrupted PUF Responses via Machine Learning. In *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. 137–140. https://doi.org/10.1109/HOST54066.2022.9839919

[20] Pulakesh Upadhyaya and Anxiao Jiang. 2019. Machine learning for error correction with natural redundancy. *arXiv preprint arXiv:1910.07420* (2019).

[21] Yuejiang Wen and Yingjie Lao. 2017. Efficient fuzzy extractor implementations for PUF based authentication. In *12th International Conference on Malicious and Unwanted Software (MALWARE)*. 119–125. https://doi.org/10.1109/MALWARE.2017.8323964