Identification and Classification of Corrupted PUF Responses via Machine Learning

Reshmi Suragani*, Emiliia Nazarenko*, Nikolaos Athanasios Anagnostopoulos*, Nico Mexis*, Elif Bilge Kavun*
*University of Passau, Faculty of Computer Science and Mathematics, Innstraße 43, 94032 Passau, Germany
Emails: {suraga01, nazare02, anagno02, mexis01, kavun01}@ads.uni-passau.de

2022 IEEE International Symposium on Hardware Oriented Security and Trust (HOST) | 978-1-6654-8532-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/HOST54066.2022.9839919

***Abstract*—A Physical Unclonable Function (PUF) exploits device manufacturing variations to extract a number of unique responses, each of which corresponds to a specific challenge, and use them for secret generation as well as authentication. However, these responses are often excessively noisy for authentication. In this study, in order to avoid using a complicated and costly Error Correction Code (ECC) in the context of a fuzzy extractor, we propose a Machine Learning (ML)-based classification technique that works accurately even in the presence of corrupted responses and we test it on an existing SRAM PUF dataset. In this approach, the SRAM power-up responses are transformed into 2D images in order to extract features from them for classification. With our proposed architecture, an accuracy of 97.34% is achieved for "intact" noisy images. In addition, the presented ML model is successful in classifying responses even when a large part of them was corrupted. With our approach, the model can classify responses accurately with up to a 30% loss in data. Finally, we also propose a proof-of-concept authentication technique using the relevant Convolutional Neural Network (CNN).**

***Index Terms*—Static Random Access Memory Physical Unclonable Function (SRAM PUF), Machine Learning (ML), Transmission error, Corrupted PUF response, Convolutional Neural Network (CNN), Classification, Identification, Authentication**

I. Introduction

With the evolution of single-board computers, the Internet of Things (IoT) is becoming more widespread, and it is most typically employed in smart cities, cars, etc. – in many things that we use in our daily lives. This increased utilization also comes with security and privacy problems. These applications and devices should be protected against all applicable attack types. Especially, hardware security is crucial in preventing device cloning, tampering, and other malicious attacks. Traditional authentication and encryption techniques solve some of these security challenges to a certain extent. These methods rely on a private key, with the assumption that if the secret is securely hidden, the device will be secure. However, such keys are frequently disclosed by attacks [1], [2].

A Physical Unclonable Function (PUF) provides a promising approach in hardware security for secret key generation. Instead of preserving secrets, a PUF derives secret keys from complex features of a physical material that are hard to replicate or clone. Memory-based PUFs are particularly appealing due to the pervasive existence of memory in embedded systems. They also require little (or no) additional hardware, which gives them a substantial advantage over other PUF implementations [3]–[5]. For key generation, PUFs require fewer bits compared to the size of common memories used in the IoT. Because only a fraction of the memory cells is used for key generation, it is hard to break the device's security; identifying those cells will be difficult for the attacker. Memory-based PUFs can thus provide a one-of-a-kind solution for device identification, authentication, and even extraction of cryptographic keys from embedded memory [6].

SRAM PUFs are widely utilized because SRAM memory cells are commonly employed and are easy to manufacture. Because of the mismatch between the transistors, the start-up values of the SRAM memory can be used to construct a digital fingerprint. The SRAM's power-up pattern can be considered as a PUF, with the position of the cells serving as a challenge, and the relevant random values generated during the power-on state serving as the corresponding response. As the SRAM PUF usually has just one CRP, it is an example of a weak PUF. SRAM-based PUFs are also more resistant to temperature changes than many other memory-based PUFs and generally more compact [7].

Employing PUFs to generate keys requires a post-processing method such as an error correction mechanism to convert the raw PUF responses into strong cryptographic keys since PUF responses are naturally noisy and may not be consistently random [8]. In the literature, this post-processing technique is referred to as a fuzzy extractor or helper data algorithm. However, due to its limitations [9], it usually cannot be utilized to address the issue of potential burst errors found in the responses of memory-based PUFs.

In this work, we propose an alternative approach to reduce the complexity of error correction methods, and attempt to determine whether it is possible to authenticate devices even if data are lost owing to transmission errors. Our contributions are twofold:

1) First, we propose a novel technique for authenticating SRAM PUFs based on the classification of their responses using a Convolutional Neural Network (CNN). As the relevant Machine Learning (ML) algorithm is trained using intact, noisy, as well as corrupted data, it has an enhanced capability of dealing with errors.
2) Second, we examine the ability of the proposed method to correctly identify PUF responses corrupted from burst errors (at different rates). To the best of our knowledge, this is the first work to attempt this, hence, providing a novel direction in the relevant field of managing burst error issues in the context of digital biometrics.

Funded by DFG, under Projects 440182124 and 439892735 of SPP 2253.

Thus, our work not only proposes a proof-of-concept authentication scheme for SRAM PUFs, but also considers data corruption during transmission, while previous works only used ML for correctly classifying intact memory-based (DRAM) PUF responses.

The rest of this work is organized as follows. In Section II, we present a brief overview of the related work. In Section III, we introduce the relevant dataset of SRAM PUF responses, and discuss the introduction of noise into it as well as the application of corruption at different rates. Section IV concerns the implementation of the relevant CNN, while Section V discusses the results of the experiments. Section VI introduces the proposed CNN-based authentication technique that uses our ML approach. Finally, Section VII concludes this work and proposes potential directions for future research.

II. Related Work

Different types of fuzzy extractors were proposed and investigated in various works. For instance, a reverse fuzzy extractor is proposed by Van Herrewege et al. [10]. Maes et al. [11] used a soft decision procedure in the fuzzy extractor. A ternary fuzzy extractor has been implemented and demonstrated by Kazumori et al. [12].

Instead of employing traditional cryptographic methods, some studies have used ML techniques. Karimian et al. [13] were the first to present a DRAM-based PUF authentication protocol that can authenticate the DRAM raw power-up values by employing ML algorithms. Najafi et al. [8] proposed a DRAM-PUF-based authentication technique using Convolutional Neural Networks (CNNs), in order to avoid the need for any additional error-correcting techniques, e.g., in the form of a fuzzy extractor scheme. However, no work seems to have yet examined such ML-based authentication schemes for the most well-known PUF, the SRAM PUF, or to have investigated the ability of such schemes to recognise corrupted PUF responses.

III. The Examined Datasets

The dataset used in our work consists of the second set of SRAM PUF responses examined in [14], which has been published online by Florian Wilde at https://www.ei.tum.de/sec/mitarbeiter/florian-wilde/publication-downloads/.

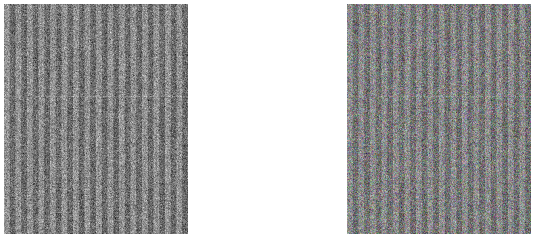
A. Data Pre-processing

In our study, we converted the raw SRAM responses to gray-scale images. The raw data of each binary file are read and stored in a 1D array. The data are converted from the hexadecimal to the binary format, and then the 1D array is reshaped into a 2D array of the following dimensions: $(\text{width} = 1024,\ \text{height} = \frac{\text{length of 1D array}}{\text{width}})$. Finally, the 2D array is converted into a gray-scale image.

Each SRAM PUF response output file contains $163{,}840$ hexadecimal digit pairs, and, thus, the total number of bits, after converting hexadecimal digits to binary ones, is $1{,}310{,}720$. As a result, a gray-scale image is created with a resolution of 1280×1024. However, due to memory limitations, all the images were processed and resized to 512×512 before being fed into the relevant ML algorithms.

B. Gaussian Noise

Noise is constantly present in digital image processing [15]. In order to achieve a high rate of correct classifications even when noisy images are used with our ML model, we created a Gaussian Noise dataset. To this end, an array of (Gaussian) noise is produced, as large as the image array, using the following settings: mode='gaussian', mean = 0, and var = 0.05. Then, each gray-scale image is converted to an array and combined with the array of noise. Finally, the resulting array is converted to an image. The difference in the image before and after adding noise is shown in Figure 1.

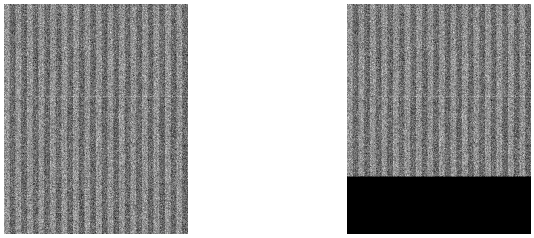


(a) Original Image (b) Noisy Image

Fig. 1: Original image and the relevant noisy image.

C. Data Corruption

Data corruption refers to errors that occur during the writing, reading, transmission, or processing of the SRAM PUF responses, resulting in unintended alteration of the original data. In our study, we manually tampered with the data to see if the model can still identify such data after a certain amount of data loss. To this end, a particular segment, e.g. the last 25%, of the hexadecimal data contained in each of the original raw SRAM PUF response data files is converted to zeros. Then, these hexadecimal data are transformed into a 2D binary array and a gray-scale image, as explained in Section III-A. This process is repeated for different percentages of data loss, ranging from 10% to 60%. The difference between an original (intact) image and the corresponding corrupted image created with a 25% data loss at its end, can be seen in Figure 2. For reasons of brevity, the results presented in this work concern only data corruption occurring at the end of the PUF response.



(a) Original Image (b) Corrupted Image

Fig. 2: Original image and the relevant corrupted image.

IV. Implementation and Experimental Results

In this section, we examine the CNN model used in this research more closely. Initially, we used a CNN architecture

identical to the one used in [13], utilising only four convolutional layers and then pooling layers. In this case, the model was trained with a learning rate of 0.01 and achieved a 97.23% accuracy on the original (intact) images. However, when data augmentation was applied to this CNN architecture, the model performance was poor, with less than 20% accuracy. Therefore, we have modified the original CNN architecture as shown in Figure 3, in order to improve the model's accuracy.

In particular, layers were applied and removed using a trial-and-error method. Batch normalization was employed after each convolutional layer, to normalize the activation function, improve accuracy, and speed up training. After the batch normalization layer, pooling is utilized to down-sample, and the process is repeated four times with the same number of filters each time. However, instead of being applied after the first convolutional layer, the first pooling layer is applied after two convolutional layers.

We trained the dataset on Google Colab Pro using a single NVIDIA Tesla K80 GPU with 25 GB of RAM. With a batch size of 200 and a total number of 3200 images in the training set, one epoch takes 16 iterations. To prevent the model from overfitting and to reduce the learning pace, the learning rate is lowered from 0.01 to 0.001.

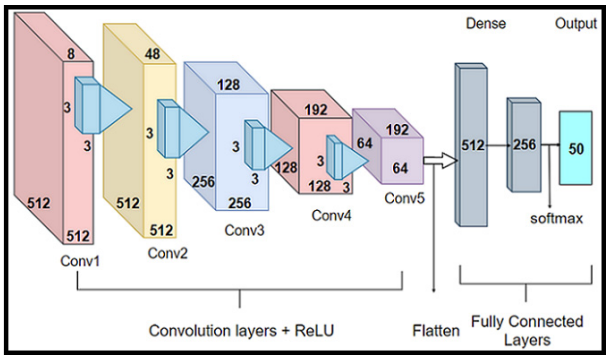


Fig. 3: The proposed CNN architecture.

V. Model Results

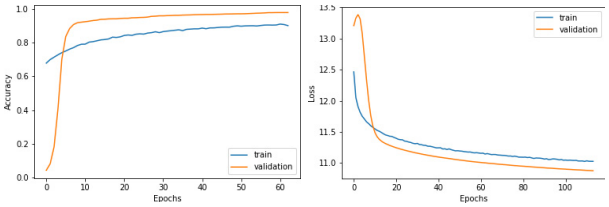
Our CNN model is trained with sets of data corresponding to intact, noisy, augmented, and corrupted SRAM PUF responses, and tested with the relevant test datasets, as explained in this section, in order to evaluate the model's overall performance. The accuracy of our model is determined by dividing the number of correct predictions by the total number of predictions. The accuracy results for different cases are as follows.

A. Intact Dataset Results

The first dataset used to train the CNN model is the original (intact) dataset. Figure 4a shows the accuracy and Figure 4b depicts the loss curve for the model with a learning rate of 0.001. The model's accuracy on an unseen dataset is 97.34%, after 60 epochs.

B. Data Augmentation

Even if the ML model has a high level of accuracy, it is likely to be prone to various types of errors. Despite the



(a) Accuracy Plot (b) Loss Plot

Fig. 4: Accuracy and loss curve plots for our model.

fact that our model could recognise the intact test data with 97.34% accuracy, the relevant confusion matrix demonstrated that it was unable to anticipate certain classes of data. These errors were resolved by applying different data augmentation techniques. The model was trained using such data augmentation techniques as rotation, horizontal flip, zoom range, shear range, width shift, and height shift. However, as the model was unable to correctly identify the intact images after horizontal flip was applied, horizontal flip is not a label-preserving transformation for our dataset, and was ultimately not used.

C. Noisy Dataset Results

The CNN model that was trained using normal noise-free images, was only able to classify noisy images with an accuracy of 52.31%. Our "Gaussian noise" model accurately classified both noisy and normal images with 95.46% and 96.85% accuracy, respectively. However, the model's performance on images that had been 20% corrupted dropped from 74.92% to 49.54% when compared to the original model. When evaluated with data corrupted 20%, 25%, or 50%, the model accuracy is less than 10% in all cases.

D. Corrupted Dataset Results

In order to improve the model's performance on both corrupted and intact images, we utilized a combination of the original (intact) dataset, the noisy dataset, and a 25% corrupted dataset as the training dataset. In this case, the accuracy with intact images, 20% corrupted images, and 25% corrupted images is 98.91%, 97.31%, and 96.10%, respectively. Thus, even when 25% of the data in the image is somehow corrupted, the combined dataset model successfully recognises the images. However, none of the models examined can adequately recognise images that have been 40% corrupted. We also note that using noisy images for training only increases the model's accuracy in recognising noisy images, and has no effect on the model's performance in the case of corrupted ones.

E. Comparison of Results on a Range of Corrupted Datasets

In order to determine the performance of our method, we examine models that have been trained using various types of datasets and test their ability to recognise images with varying levels of corruption, as shown in Table I. We note that introducing corrupted images to the training dataset helps the model to better classify such images. We also observe that the models trained with one of the combined datasets under test, are capable of correctly classifying images that have been corrupted up to 30%, as well as intact and noisy images.

TABLE I: Accuracy Results Among Examined Datasets. (**C-x:** x% Corrupted Images, **I:** Intact Images, **N:** Noisy Images, **Aug:** Augmented Images)

Training / Testing	Original (Intact)	C-25	I + N + C-25	I + N + C-25 + Aug
Intact	97.34%	5.23%	98.15%	98.91%
Noisy	52.31%	4.77%	91.46%	96.23%
C-10	94.85%	39.69%	98.69%	91.15%
C-20	74.92%	93.69%	97.31%	92.77%
C-25	67.33%	98.00%	96.10%	94.29%
C-30	47.46%	92.62%	89.15%	90.23%
C-40	17.69%	64.46%	61.38%	53.23%
C-50	14.08%	33.31%	28.91%	24.31%
C-60	1.54%	5.85%	1.77%	2.31%

VI. Authentication Using Machine Learning

In this section, we propose a proof-of-concept authentication technique using our ML model. This approach consists of two steps. The enrollment phase is the initial step, during which SRAM PUF Challenge-Response Pairs (CRPs) are collected in order to construct a database. The following stage is the authentication phase, in which the device sends an authentication request to the server. A challenge is then sent to the device by the server. The device responds with the image corresponding to the response relevant to the received challenge. This image is fed into the trained CNN model, which can correctly predict the challenge, even if the received image is up to 30% corrupted. If the predicted class matches the challenge given by the server, the device is authenticated. In this way, device authentication is accomplished without the need for complex error-correcting techniques, even when burst errors occur. Our proposed ML-based authentication scheme can be seen in Figure 5.

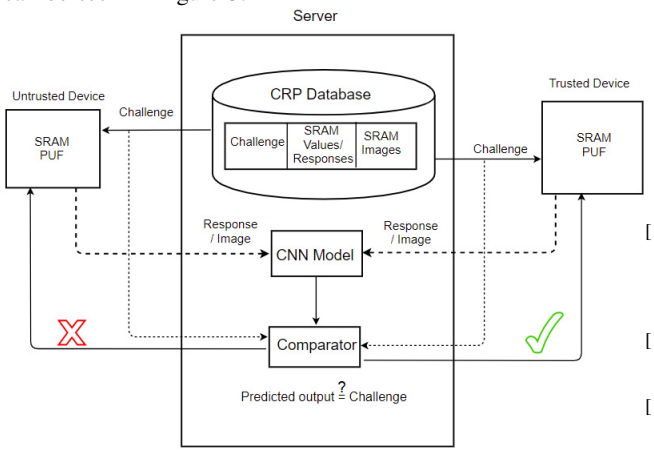


Fig. 5: Authentication scheme (authentication phase).

VII. Conclusion and Future Work

Based on our results, we can conclude that by utilizing ML classification techniques, we can eliminate the need for the employment of ECC-based methods in the context of PUF-based authentication, even when burst errors occur due to transmission or other issues. In particular, the ML model examined in this work is successful in correctly classifying the relevant images, even with a 30% loss of consecutive data. Nevertheless, once the data loss is higher than 30%, the model's performance deteriorates. Similar results occur when the data corruption occurs at the end and at the beginning of the PUF response. Future research will include testing the ML model with randomly corrupted responses and with responses containing random noise of different levels, as well as providing a detailed security analysis, and implementing the proof-of-concept authentication scheme described in Section VI.

Acknowledgment

The authors would like to thank Florian Wilde for clarifying the structure and contents of the datasets examined in [14].

References

[1] B. M. S. Bahar Talukder, F. Ferdaus, and M. T. Rahman, "Memory-based PUFs are vulnerable as well: A non-invasive attack against SRAM PUFs," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4035–4049, 2021.

[2] S. Sutar, A. Raha, and V. Raghunathan, "Memory-based combination PUFs for device authentication in embedded systems," *IEEE Trans. on Multi-Scale Computing Systems*, vol. 4, no. 4, pp. 793–810, 2018.

[3] D. E. Holcomb, W. P. Burleson, and K. Fu, "Power-up SRAM state as an identifying fingerprint and source of true random numbers," *IEEE Transactions on Computers*, vol. 58, no. 9, pp. 1198–1210, 2008.

[4] F. Zhang, S. Yang, J. Plusquellic, and S. Bhunia, "Current based PUF exploiting random variations in SRAM cells," in *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2016, pp. 277–280.

[5] C. Böhm, M. Hofer, and W. Pribyl, "A microcontroller SRAM-PUF," in *2011 5th International Conference on Network and System Security*. IEEE, 2011, pp. 269–273.

[6] 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.

[7] S. Katzenbeisser, Ü. Kocabaş, V. Rožić, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann, "Pufs: Myth, fact or busted? a security evaluation of physically unclonable functions (pufs) cast in silicon," in *Cryptographic Hardware and Embedded Systems – CHES 2012*, E. Prouff and P. Schaumont, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 283–301.

[8] F. Najafi, M. Kaveh, D. Martín, and M. Reza Mosavi, "Deep PUF: A highly reliable DRAM PUF-based authentication for IoT networks using deep convolutional neural networks," *Sensors*, vol. 21 (6), p. 2009, 2021.

[9] P. Koeberl, J. Li, A. Rajan, and W. Wu, "Entropy loss in PUF-based key generation schemes: The repetition code pitfall," in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*. IEEE, 2014, pp. 44–49.

[10] A. Van Herrewege, S. Katzenbeisser, R. Maes, R. Peeters, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann, "Reverse fuzzy extractors: Enabling lightweight mutual authentication for puf-enabled rfids," in *Financial Cryptography and Data Security*, A. D. Keromytis, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 374–389.

[11] R. Maes, P. Tuyls, and I. Verbauwhede, "A soft decision helper data algorithm for SRAM PUFs," in *2009 IEEE international symposium on information theory*. IEEE, 2009, pp. 2101–2105.

[12] K. Kazumori, R. Ueno, and N. Homma, "A ternary fuzzy extractor for efficient cryptographic key generation," in *2019 IEEE 49th International Symposium on Multiple-Valued Logic (ISMVL)*, 2019, pp. 49–54.

[13] N. Karimian, F. Tehranipoor, N. Anagnostopoulos, and W. Yan, "DRAMNet: Authentication based on physical unique features of DRAM using deep convolutional neural networks," *arXiv preprint arXiv:1902.09094*, 2019.

[14] F. Wilde, "Large scale characterization of SRAM on Infineon XMC microcontrollers as PUF," in *Proceedings of the Fourth Workshop on Cryptography and Security in Computing Systems*, ser. CS2 '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 13–18. [Online]. Available: https://doi.org/10.1145/3031836.3031839

[15] A. K. Boyat and B. K. Joshi, "A review paper: Noise models in digital image processing," *arXiv preprint arXiv:1505.03489*, 2015.