iPUF: A Novel Security-by-Design Paradigm to Mitigate Data Manipulation and External Attacks in Cyber-Physical Systems

Seema G. Aarella
Dept. of Computer Science and Engineering
University of North Texas, USA
Email: seemaaarella@my.unt.edu

Sameer Agarwal
Texas Academy of Math and Science
University of North Texas, USA
Email: sameeragarwal@my.unt.edu

Saraju P. Mohanty
Dept. of Computer Science and Engineering
University of North Texas, USA
Email: saraju.mohanty@unt.edu

Elias Kougianos
Dept. of Electrical Engineering
University of North Texas, USA
Email: elias.kougianos@unt.edu

Vasanth Iyer
Computer Science and Digital Technologies
Grambling State University, USA
Email: iyerv@gram.edu

Bibhudutta Rout
Dept. of Physics
University of North Texas, USA
Email: bibhudutta.rout@unt.edu

***Abstract*—Fault Injection attack is a type of side-channel attack on the Physical Unclonable Function (PUF) module that can induce faults in the PUF response by manipulating the PUF circuit behavior through voltage glitches, laser attacks, temperature manipulations, or any other attacks potentially leading to information loss or security system failure. This type of attack exposes the physical characteristics of PUFs that can be analyzed to predict or compromise the unique challenge-response pairs (CRPs) reducing the security and reliability of the PUF. Mitigation strategies against such attacks typically include adding noise to the PUF output, using error-correcting codes, or enhanced cryptographic protocols that obscure physical side-channel attacks. In this research, we propose a Generative Adversarial Network (GAN) based security model, that monitors the PUF behavior and detects the variations in PUF response. The model can detect glitches in the PUF response and generate alerts to take mitigation measures.**

***Index Terms*—Security-by-Design (SbD), Physical Unclonable Function (PUF), Generative Adverserial Networks (GAN), Edge Computing, Cybersecurity, Data Security, Reliability**

I. Introduction

Cyber-Physical Systems (CPS) are widely employed in Internet-of-Things (IoT) applications, they are increasingly becoming vulnerable to data manipulation and external attacks, particularly False data injection (FDI) attacks, which can compromise the integrity of the CPS as they can target the maximum possible locations in CPS at a time [1]. PUF is a hardware security primitive used for device and data secuirty in CPS applications. FDI attacks can target PUF circuits at any stage manipulating the PUF response that will result in security system failure due to errors in the PUF response.

PUFs are a form of security technology created utilizing the variations in fabrication process of a device, they can be used for authentication and identification. This is because they can generate unique and secret keys through their response [2]. Although PUFs cannot be cloned, the distribution of challenges and responses can be identified by machine learning models given certain keys. Deep learning models can begin to predict responses given certain keys. Additionally, another way that PUFs can be attacked is through glitches that cause bitflips. One bitflip can throw off a response and harm the authentication capability of a PUF device [3].

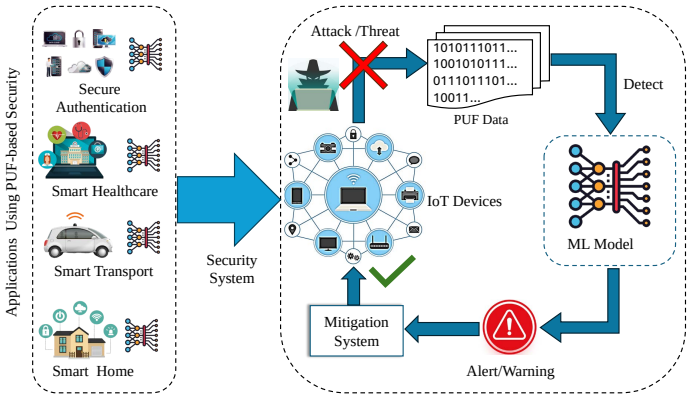

Fig. 1. Security using Machine Learning for PUF based applications

As shown in Fig. 1, the machine-learning model can be employed to detect an attack or threat and mitigate the damage

to keep the IoT devices secure. The PUF data is fed into the machine learning model which looks for patterns or potential attacks in case it needs to trigger an alert.

We propose using GANs to address these issues. GANs, consisting of a generator and a discriminator trained alternately, can distinguish real from fake PUF responses. This enhances security for edge devices and protects valuable information.

The rest of the paper is organized as follows: Section II discusses related research of PUFs and GANs. Section III discusses the problems addresses and proposed solutions along with specific contributions of the current research. Section IV discusses the proposed framework while Section V includes the experimental setup along with discussions on results, and Section VI concludes the research.

II. Related Prior Research

The security of PUF-based systems has been the subject of extensive research, revealing various vulnerabilities and potential countermeasures. The susceptibility of Arbiter PUFs (APUFs) and their variants, such as XOR PUFs, to machine learning attacks has been well-documented [4], [5]. These attacks exploit the inherent predictability of PUF responses, potentially compromising the security of authentication and key generation mechanisms. The research proposes modifications to PUF designs, such as the Double Arbiter PUF (DAPUF) [6], and explores the use of helper data and error correction techniques to enhance their resilience against modeling attacks [7].

Fault injection attacks pose another significant threat to PUF-based systems. These attacks exploit vulnerabilities in the physical implementation of PUFs, potentially leading to unauthorized access and data manipulation. Various countermeasures, including the use of ring oscillator (RO) PUFs as fault injection detectors are proposed in [13], [14]. The sensitivity of RO PUFs to voltage and clock manipulation allows them to serve as effective sensors for detecting and mitigating such attacks.

The relationship between machine learning and PUF security is dual: ML models can both attack and enhance PUFs. Research [15] demonstrates adversarial neural networks improving error-correcting codes, potentially strengthening PUF resilience. Additionally, [16] evaluates PUF security assumptions, clarifying myths and limitations, emphasizing the need for rigorous testing in security-critical applications.

Additionally, other methods of strengthening PUFs involve Error Correction with Multiple CRPs or ECMC, as shown in the work [17]. This method involves splitting up the bits of a response into the number of CRPs.

PUFs are being integrated into edge computing for secure authentication in IoT. Research focuses on robust PUF-based protocols and countermeasures against attacks to protect resource-constrained devices. Table I lists state-of-the-art studies and applications.

TABLE I
Comparative Table for State-of-the-Art Literature

Research	Year	ML Algorithm	Application
Chen et. al [8]	2018	Fault Injection Module	Increase PUF Attack resistance
Wen et. al [9]	2017	Fuzzy Extractor	PUF reliability
Long et. al [10]	2019	Double PUF-based Model	PUF Authentication System
Yoon et. al [11]	2020	Generative Adversarial Network	PPUF testing and restructuring
Aarella et. al [12]	2024	K-Mer Sequence	PUF Bit Error Correction
Current Work (GAN-Fortified PUF, iPUF)	2024	Generative Adversarial Network	PUF Bitflip detection

III. Research Contributions

A. Problems Addressed

There are many ways in which the PUF system can be attacked or manipulated, either by external attackers or by environmental variations, both accounting for instability in PUF response and eventual failure of the PUF-based systems like authentication or authorization.

B. Proposed Solutions

This research proposes the use of GANs for detecting the errors in the PUF response, the research involves studying the properties GANs concerning n-bit binary PUF response. The GANs generators are tuned to generate PUF responses that are similar to the original reponse and the discriminator can be fine tuned to identify the real from the error response.

C. Novel Contributions of the Current Research

This research proposes a novel GAN-based approach to detect threats to a PUF operation, that is primarily based on binary sequence of n-bit PUF response data.

- PUF response data preparation, analysis and Preprocessing, data organization and dataset generation suitable for GAN training
- The Application of GANs for detecting errors in PUF responses is novel compared to traditional statistical methods
- Machine Learning for improving error detection enhancing the robustness and security of PUF-based security applications
- Fine-tuning GAN to handle binary input, testing the capabilities beyond the usual application domains like image synthesis
- Attack pattern recognition by training the GAN on real and synthetic glitch data

IV. The Proposed Novel Framework - The iPUF

The research uses PUF 64-bit CRP response from XORArbiter PUF [18], the data is preprocessed and labelled. The unique responses are labled as real, fake response is created by introducing bitflips randomly along the 64-bit sequence,

and they are labelled as fake. The GAN model is trained to classify the real and fake responses based on their labels.

The framework using GAN for predicting errors in the PUF response is shown in Figure 2. The model is trained on multiple instances of responses to study the variations in the threshold values that differentiate the original response from the corrupted response. The threshold value is used as a reference for identifying the real and fake data. Once the model is trained, it can predict the threshold value of the new data based on which it can make accurate decisions. If the threshold value of new data is the same as the 't', it will be tagged as a real response, and a new value that is greater than the value 't' is flagged as an error, and the system is directed for further mitigation.

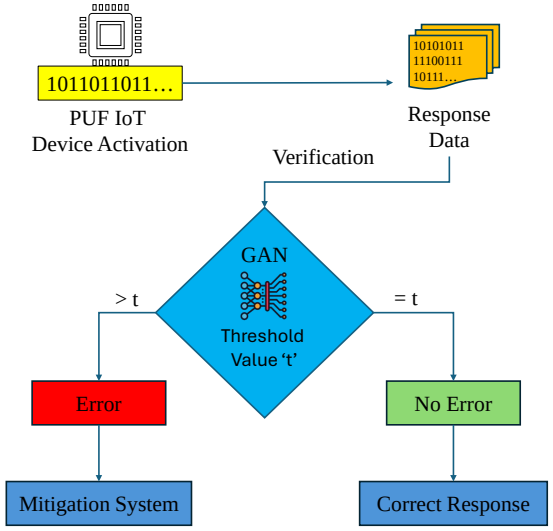

Fig. 2. Proposed GAN framework for error detection.

V. Experimental Results

A. Experimental Setup

The dataset consists of 10K data of repeated responses from 1000 unique challenges. The responses are grouped into 100 unique responses for each challenge, with some groups having glitched data. The GAN model setup for error detection in PUF response data is shown in Figure 3. The necessary libraries are imported for defining GAN, and Keras to build models and layers. The parameters are adjusted to suitable values, *batch_size* is set to 64, *latent_dim* of the noise generator is set to 100. The dataset is preprocessed, and columns and labels are added. The real data is labeled as '1' and fake data is labeled as '0'. The *real_data_batch()* and *generator(noise)* functions are used to select random batches from real samples and generate random noise.

The generator model takes noise as input and outputs a 64-dimensional binary-like vector like the shape of the PUF response. It includes dense layers with LeakyReLu activations and BatchNormalization layers to stabilize the training and prevent mode collapse. The output layer uses sigmoid function generating values between 0 and 1.

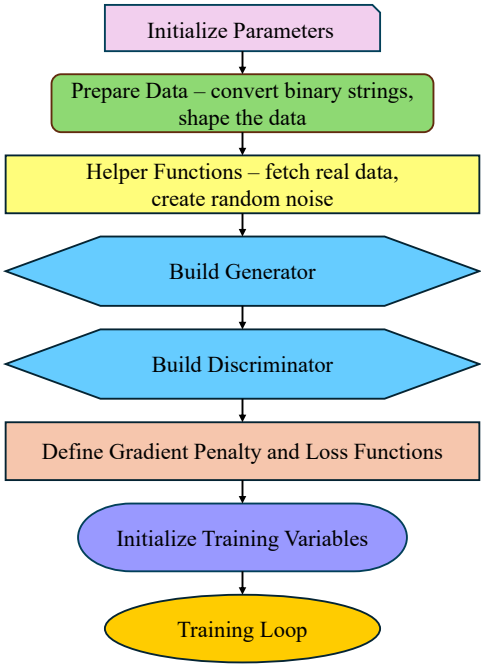

Fig. 3. Training process of the GAN model

The discriminator model takes 64-dimensional binary vectors as input and outputs a single probability, sigmoid activation is used in the output layer. *feature_matching_loss()* is used to calculate the mean squared error between real and fake features, making the generator produce similar data that is close to real data.

The GAN generator and discriminator training steps are shown in Algorithm 1. The gradient penalty tuning parameter λ_{gp} is adjusted to be close to 5 to avoid over-regularizing the discriminator; any lower value will make the model unstable. Noise strength is adjusted to a value close to 0.1 after testing the over various values. These values are adjusted accordingly during testing to prevent the discriminator from overfitting for various datasets.

B. Results and Analysis

The discriminator output shown in Figure 4 ranges between 0 and 1, where outputs close to 1 likely represent real data, and those close to 0 indicate fake data for a given set of responses. Analyzing the real data distribution shows that it sharply peaked at 1, meaning the discriminator is confident that real data samples are indeed real. The distribution for fake data is spread mostly between 0.0 and 0.7, with a peak between 0.2 and 0.4, suggesting the discriminator correctly classifies most fake data as fake, with outputs concentrated in the lower range.

The trained model is confidently detecting the real and fake labels with a small outliers as shown by the confusion matrix in Figure 5.

The Receiver Operating Characteristic (ROC) curve shown in Figure 6 shows the trade-off between the True Positive Rate (TPR) and False Positive Rate (FPR) for various threshold

Algorithm 1: GAN Training Loop

Input: Number of epochs $E = 1000$, Batch size B, Gradient penalty coefficient λ_{gp}, Label smoothing value s
for *epoch* $= 1$ to E **do**
  Step 1: Train Discriminator
    Fetch real samples:
  real_batch $\leftarrow$ *real_data_batch()*
    Generate noise: *noise* $\sim \mathcal{N}(0, 1)$
    Create fake samples:
  fake_batch $\leftarrow$ *generator(noise)*
    Add noise regularization to real and fake samples
    Apply label smoothing:
  real_labels $\leftarrow s \cdot$ *real_labels*,
  fake_labels $\leftarrow (1 - s) \cdot$ *fake_labels*
    Train discriminator on real and fake samples
    Calculate gradient penalty:
  gp $\leftarrow \lambda_{gp} \cdot$ gradient_penalty(*real_batch*, *fake_batch*)
    Update discriminator loss: *d_loss* $\leftarrow$ *d_loss* $+$ *gp*
  Step 2: Train Generator
    Generate fake samples:
  fake_batch $\leftarrow$ *generator(noise)*
    Calculate feature matching loss between real and fake samples
    Use *tf.GradientTape* to compute gradients for generator
    Update generator weights
end for

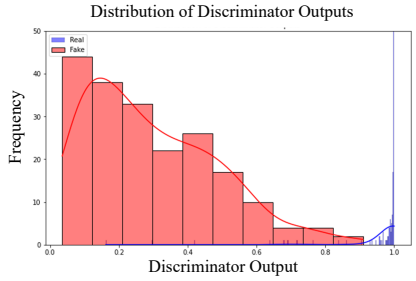

Fig. 4. The plot for the discriminator output

settings of the classifier. The ROC curve is very close to the top-left corner, showing that the model achieves a high true positive rate with a very low false positive rate. This steep curve indicates that the model can separate real and fake samples effectively with little error. The Area Under the Curve (AUC) score of 1 represents a perfect classifier, while an AUC of 0.5 represents random guessing. In this case, the AUC is 0.99, which is very close to 1.

The discriminator and generator losses over 1000 epochs are shown in Figure 7. The discriminator loss starts at a lower value and eventually stabilizes at 3.0, indicating that the discriminator is performing well and likely reached an optimal value at this point the discriminator is successfully

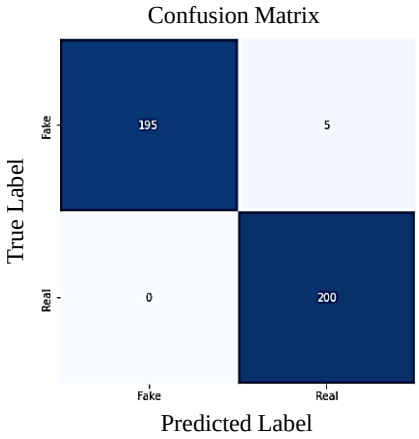

Fig. 5. Confusion matrix

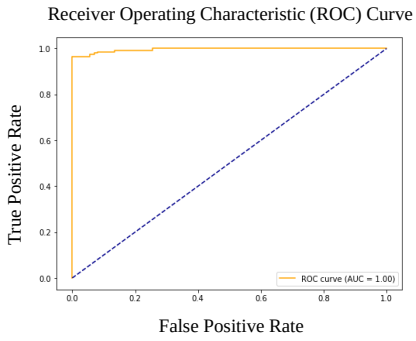

Fig. 6. The plot for ROC

distinguishing between real and fake data with high confidence. The stable value shows that the discriminator is not overly confident.

The generator loss remains constantly low after initially starting at a high value and eventually decreasing, producing samples that are quite realistic and harder for the discriminator to distinguish.

It is to be noted that the dataset used has few bit flips in the range of 1-4 in each binary sequence. However a more degraded dataset will generate different results from the GAN. This model is quite effective in detecting minor flips upto 1 bit.

The prediction results are shown in Figure 8, where the predictions are made based on the threshold value from the discriminator. The model is efficient in generating threshold values that are unique for real and fake data, as displayed in the results.

The comparative results from various models studied in related research are shown in Table II.

VI. Conclusion

This research demonstrates a robust GAN training approach, applying advanced techniques like gradient penalty, noise regularization, and feature matching tailored for binary data. These methods help stabilize the training and improve the

TABLE II
COMPARATIVE TABLE FOR STATE-OF-THE-ART LITERATURE

Research	Year	Algorithm	Accuracy
Chen et. al [8]	2018	Fault Injection Module	65.10
Wen et. al [9]	2017	Fuzzy Extractor	98.00
Long et. al [10]	2019	Double PUF-based Model	N/A
Yoon et. al [11]	2020	Generative Adversarial Network	66.00
Aarella et. al [12]	2024	K-Mer Sequence	99.74
Current Work (iPUF)	2024	Generative Adversarial Network	99.00

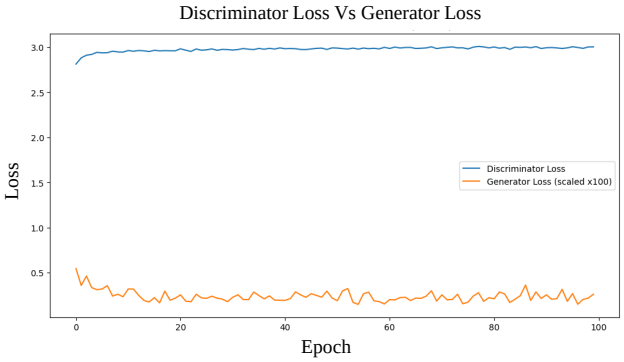

Fig. 7. Evaluation of the discriminator and generator loss

```
Input sample 1 (binary response): 1111111111000010101100111110101011100100010101111100111111111
Original label: REAL
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step
Predicted label by discriminator: REAL
Discriminator score: 0.4201115667819977

Input sample 2 (binary response): 1111111111000010101100111110101011100100010101111100111111111
Original label: REAL
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
Predicted label by discriminator: REAL
Discriminator score: 0.4201115667819977

Input sample 3 (binary response): 1111111111000010101100111110101011100100010101111100111111111
Original label: FAKE
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step
Predicted label by discriminator: FAKE
Discriminator score: 0.43325427174568176
```

Fig. 8. Prediction results with corresponding threshold Values

generator's ability to mimic real data distribution as shown in the results.

The threshold value is the key to making better predictions. The current model is 99% accurate in detecting fake responses, a feedback system-based model can further enhance the performance by updating the threshold values for a variety of data, and a continuous learning GAN can be implemented to update and detect the latest threats using new data. Overall, This GAN model works as a robust framework for generating realistic samples, providing a valuable foundation for applications in anomaly detection and hardware security in PUF-based systems. The model needs testing in real-time on features like voltage variations or current variations for further development.

As a future research, a broader dataset can be used to study the behavior of the GAN model to suit the needs of security and data analysis involving binary sequences. They need to be optimized for low-power edge devices and tested for practical applications, and they can be employed in deep-fake image detection. GANs can also be trained to generate strong PUFs to enhance the reliability of the system.

REFERENCES

[1] S. Padhan and A. K. Turuk, "Design of False Data Injection Attacks and Their Detection and Mitigation in Cyber-Physical Systems," in *Proc. 27th International Conference on Advanced Computing and Communications (ADCOM 2022)*, 2023, pp. 41–45.

[2] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. 9th ACM Conference on Computer and Communications Security*, 2002, pp. 148–160.

[3] J. Ruchti, M. Gruber, and M. Pehl, "When the decoder has to look twice: Glitching a puf error correction," Cryptology ePrint Archive, Paper 2021/958, 2021, https://eprint.iacr.org/2021/958. [Online]. Available: https://eprint.iacr.org/2021/958

[4] A. O. Aseeri, Y. Zhuang, and M. S. Alkatheiri, "A Machine Learning-Based Security Vulnerability Study on XOR PUFs for Resource-Constraint Internet of Things," in *Proc. IEEE International Congress on Internet of Things (ICIOT)*, 2018, pp. 49–56.

[5] H. T. Nguyen, S. Bottone, K. T. Kim, M. Chiang, and H. V. Poor, "Adversarial Neural Networks for Error Correcting Codes," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 01–06.

[6] R. Yashiro, T. Machida, M. Iwamoto, and K. Sakiyama, "Deep-Learning-Based Security Evaluation on Authentication Systems Using Arbiter PUF and Its Variants," vol. 9836, 09 2016, pp. 267–285.

[7] R. Maes and V. van der Leest, "Countering the effects of silicon aging on SRAM PUFs," in *Proc. IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2014, pp. 148–153.

[8] J. Chen, J. Wen, F. Dan, Z. Li, B. Liu, Y. Xu, S. Chen, and B. Li, "A Modeling Attack Resistant Scheme Based on Fault Injection," in *Proc. IEEE 3rd International Conference on Signal and Image Processing (ICSIP)*, 2018, pp. 582–585.

[9] Y. Wen and Y. Lao, "Efficient fuzzy extractor implementations for PUF based authentication," 10 2017, pp. 119–125.

[10] J. Long, W. Liang, K.-C. Li, D. Zhang, M. Tang, and H. Luo, "PUF-Based Anonymous Authentication Scheme for Hardware Devices and IPs in Edge Computing Environment," *IEEE Access*, vol. 7, pp. 124 785–124 796, 2019.

[11] J. Yoon and H. Lee, "PUFGAN: Embracing a Self-Adversarial Agent for Building a Defensible Edge Security Architecture," in *Proc. IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, 2020, pp. 904–913.

[12] S. G. Aarella, V. P. Yanambaka, S. P. Mohanty, and E. Kougianos, "Fortified-Edge 4.0: A ML-Based Error Correction Framework for Secure Authentication in Collaborative Edge Computing," in *Proc. Great Lakes Symposium on VLSI*, 2024, p. 639–644. [Online]. Available: https://doi.org/10.1145/3649476.3660384

[13] T. Köylü, L. Garaffa, C. Reinbrecht, M. Zahedi, S. Hamdioui, and M. Taouil, "Exploiting PUF Variation to Detect Fault Injection Attacks," in *Proc. 25th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, 2022, pp. 74–79.

[14] S. Tajik, "On the Physical Security of Physically Unclonable Functions," 01 2019.

[15] G. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," 07 2007, pp. 9–14.

[16] 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 *Proc. Cryptographic Hardware and Embedded Systems – CHES*, 2012, pp. 283–301.

[17] K. Sun, Y. Shen, Y. Lao, Z. Zhang, X. You, and C. Zhang, "A New Error Correction Scheme for Physical Unclonable Function," in *Proc. IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 2018, pp. 374–377.

[18] S. G. Aarella, S. P. Mohanty, E. Kougianos, and D. Puthal, "PUF-based Authentication Scheme for Edge Data Centers in Collaborative Edge Computing," in *Proc. IEEE International Symposium on Smart Electronic Systems (iSES)*, 2022, pp. 433–438.