Optimal Machine-Learning Attacks on Hybrid PUFs

Fei Hongming[1 [0009−0000−9113−9398]], Prosanta Gope[2 [0000−0001−5636−8726]], Owen Millwood[2 [0000−0002−7250−8271]], and Biplab Sikdar[1 [0000−0002−0084−4647]]

[1] National University of Singapore, Singapore
{fei.hongming,bsikdar}@u.nus.edu
[2] The University of Sheffield, Western Bank, Sheffield
{p.gope,ojwmillwood1}@sheffield.ac.uk

Abstract. Physical Unclonable Functions (PUFs) are a promising, low-cost entropy source and security primitive for Internet-of-Things (IoT) applications, widely used in authentication, key generation and management. As PUFs have been investigated further, they have often been found to be vulnerable to machine-learning attacks (MLA). Despite numerous attempts to fortify PUFs against such vulnerabilities by innovating with different structures and compositions — among which hybrid PUFs were considered a promising approach — the security of these designs against MLA largely remained untested. Specifically, this paper targets the recently introduced hybrid PUFs, namely the heterogeneous Feed-Forward PUFs [1] and OAX PUFs [28], which were claimed to be secure against MLAs. Contrary to these claims, to the best of our knowledge, we are the first to report that even these advanced PUF structures are not immune to MLA. Furthermore, the paper delivers a comprehensive evaluation of the MLA resistance of hybrid PUF structures and proposes the ***Transition Theorem***, which provides a novel insight for performing Hybrid PUF modelling. We successfully apply this theory to three classic attack models, Ruhrmair2010 [18], Mursi2020 [16] and Wisiol2022 [27], and enable them to successfully attack the earlier PUFs modelling failures. This theory contributes to the effectiveness of current strategies and lays the groundwork for future advancements in PUF security.

Keywords: Machine-Learning Attack· Hybrid PUFs · Transition Theorem.

1 Introduction

Internet-of-things (IoT) has spawned numerous lightweight applications and schemes, such as smart homes, smart grids, lightweight authentication, and real-time monitoring systems, facilitating enhanced connectivity and automation across various sectors. However, as the IoT landscape expands, concerns have arisen regarding whether these lightweight mechanisms can adequately protect

privacy, provide confidentiality, and meet other security requirements. Physical Unclonable Functions (PUFs) are derived from the natural, uncontrollable random variations that occur during the manufacturing process. Unclonable, unique, and tamper-resistant, PUFs can be likened to a fingerprint, and when given an input stimulus, PUFs output an unpredictable response. Beyond these attributes, PUFs are lightweight and can be effectively implemented on IoT devices, positioning them as a promising strategy for enhancing the security of IoT devices.

1.1 Problem Statement & Related Work

However, PUFs are proven to be vulnerable to machine-learning attack (MLA) methods, including **Logistic Regression**[18], **Evolution Strategy** [2], and **Neural Network** [27] models. One of the most common countermeasures is to combine several PUFs and add an XOR gate on all of the output responses to obfuscate the interrelationship between the initial input and final output between each individual PUF. The XOR Arbiter-PUF (XOR PUF) was first proposed to combine several basic Arbiter PUFs with a final output XOR, which showed strong resistance against a linear attack model [23]. The Feed-Forward PUF [13] was proposed to introduce a more non-linear relationship into the basic PUF component, increasing the training data requirements for an attacker. Additionally, with the development of machine learning technologies, reliability-based attacks [2] and other attacks which utilise additional side-channel information [3] have been proposed and proved to be significant threats to the PUFs, even when configured with unpractical large parameters. Thus, more complicated designs such as the XOR-Feed-Forward PUF, Interpose PUF [17], OAX-PUF [28] and MPUF [19] were proposed, and yet, each type found successful attacks against them [26,27,20]. With a large configuring parameter, some complex PUFs have not been attacked successfully without using side-channel information. However, these PUFs are not practical in realistic scenarios due to issues with reliability [2]. The more the numbers of PUF compositions involved in the PUF design, the more unreliable the PUF is. As stated in [2], the smaller delay difference causes unreliability in the response. Once the adversary sets this metric as the indicator for the modelling, even 20-XOR PUFs can be broken. In [1], the homogeneous and heterogeneous Feed-Forward were proposed and were shown to be theoretically secure against state-of-the-art MLA. This claim is proved in [27], where a one loop 3-XOR-Feed-Forward Arbiter PUF with heterogeneous structure is demonstrated to be resistant against a neural network model, no matter how much training data is provided, and the prediction accuracy is around 60%. In [28], OR-AND-XOR PUFs are proposed to defend against reliability attacks. We find that the hybrid design which combines several differently configured or different types of PUFs together in one design can help to confuse the adversary who uses both traditional MLA technologies and reliability-based attacks.

MLA against PUFs is typically tackled as a supervised learning problem, where the challenges are used as the input features and the responses are used

as the labels. Then, the modelling task is trained as a binary classification problem (e.g., positive output predicts binary zero, negative output predicts binary one). MLA on PUFs have been a significant pain point when developing so-called Strong PUFs (categorised by a unique challenge/response pair (CRP) space which grows exponentially with PUF size) almost since their conception. MLA is carried out by adversaries collecting a subset of CRPs from an individual PUF's total CRP space to use as an input to a sophisticated ML algorithm, such that a mathematical model can be generated which learns correlative properties between different CRPs. Over the years, almost all Strong PUFs that have been proposed are vulnerable to MLA using many different types of ML algorithms, ranging from traditional ML, which is specifically tailored to a given PUF design/logical structure, to deep learning methods. The first significant work exposing the vulnerability of PUFs to MLA was demonstrated by Rührmair et al. in [18], where the Logistic Regression and Covariance-Matrix Evolutionary Strategy (CMA-ES) algorithms were exploited to model Arbiter PUFs, Ring Oscillator PUFs, XOR Arbiter PUFs, Lightweight Secure PUFs and Feed-Forward Arbiter PUFs. These attacks required varying numbers of CRPs for training the models, with the simple Arbiter PUFs, at a minimum, requiring just 640 CRPs to break the PUF. The more obfuscated PUFs (XOR-Arbiter PUF and Feed-Forward Arbiter PUF), however, generally required many more CRPs before model convergence occurred, at up to 500,000 in most cases. While less efficient than traditional MLA (on PUFs), deep learning-based modelling attacks can learn latent representation without requiring knowledge of the underlying PUF structure, broadening their use cases [20,10]. More extensively, obfuscated APUF designs have shown improved defences against traditional ML attacks; however, Feed-Forward Neural Networks (FNNs) have been shown to model up to 5-XOR APUFs successfully, and (4,4)-iPUFs [20]. A large number of composed PUFs can become a problem for MLAs such that it takes an unacceptable resource and training time to conduct the attack [27]; however, large-scale PUF integration leads to poor reliability and impracticality, reducing the practicality of deploying these PUFs in real life. However, many error correction or noise-tolerant technologies have been proposed [9], though it remains a shortcoming for MLAs targeting complicated PUFs, e.g., 20-XOR Arbiter PUFs. In this case, it requires an unacceptable order of magnitude, additional data, and time costs. Besides CRPs, other side-channel information can be used to help model the PUFs. In [2], a reliability-based machine learning attack using the divide-and-conquer scheme is proposed. Evaluations on XOR PUFs show that with the increasing numbers of XORs, the needed number of challenges increases linearly, which contradicts the prevalent notion that the increase should be exponential. This paper focuses on the traditional MLA based on CRPs, while we also take reliability, bias, and uniformity as important metrics. In [16], the neural network devised for attacking a k-XOR Arbiter PUF is proposed, comprises of three fully connected hidden layers sized $\{2^{k-1}, 2^k, 2^k\}$. It performs well on k-XOR-PUF, with k ranging from 2-9. In [27], a model sized $\{2^k, 2^{k-1}, 2^{k-1}, 2^k\}$ is proposed targeting homogeneous XOR Feed-Forward PUFs. This model performs well with the number of

loops ranging from 1-5 and k ranging from 2-8. However, the model fails on heterogeneous XOR Feed-Forward PUFs. Apparently, models in [16] and [27] are designed for specific PUF instances. Additionally, other modelling methods such as PAC learning[5] and evolution strategy[15] have been applied to model PUFs. However, these methods are not gradient-learnable machine learning approaches and thus fall outside the scope of this work.

1.2 Contributions

In this study, the focus is directed towards the hybrid architecture in the design of PUFs, with a particular emphasis on conducting a thorough analysis of recently proposed hybrid PUFs, such as heterogeneous Feed-Forward XOR PUFs and OR-AND-XOR PUFs. This investigation includes successful attacks on these systems and evaluates the capability of current models to represent such hybrid PUF structures accurately. Through a detailed examination of these findings, we introduce and define a *novel concept* called ***'Transition Theorem'***, which is then applied to enhance the effectiveness of generic neural-network-based attack methodologies for modelling hybrid PUFs. The contributions of this research are multifaceted and can be summarized as follows:

1. This work shows the first successful demonstration of attacks against heterogeneous Feed-Forward XOR PUFs and OAX-PUFs, illustrating that neural-network-based models, devoid of side-channel information, can achieve a high degree of accuracy in attacking a variety of hybrid PUF configurations and predicting the unseen CRPs.
2. This work identifies significant challenges in modelling hybrid PUFs, such as issues arising from uncertain model structures and ambiguous PUF information, which can lead to overfitting and convergence to local minima. The study utilizes a Mixture-of-Experts structure to learn the hybrid information in heterogeneous Feed-Forward PUFs, OAX PUFs, and other hybrid PUFs using a manageable volume of training data. It provides a detailed analysis of the model's capability to model various PUFs, highlighting its potential for broad application.
3. Based on the analysis of the modelling attacks and the mathematical framework for hybrid PUFs, a ***Transition Theorem*** for neural-network-based attack methods is proposed. This strategy offers guidance from an adversarial perspective on determining appropriate model structures and settings, demonstrating its applicability to other attack methodologies that previously could not conduct successful attacks.
4. This work undertakes a comprehensive evaluation of different instances of hybrid PUFs, providing an exhaustive analysis that establishes a benchmark for the design and assessment of hybrid PUF systems. All the codes and data used in this paper are provided for the use of the research community.[3]

[3] **All the codes and datasets used in this paper are provided in** https://drive.google.com/drive/folders/1t6w-RR2FZKko_Ur3uWkRZHj009dtj0Ro?usp=drive_link

1.3 Paper Organisation

The remainder of this paper is organized as follows. Section 2 introduces the concept and details of Arbiter PUF and its compositions. In Section 3, we present the attack model and operating scheme. Section 4 presents the experiment details and analysis of hybrid designs. In Section 5, we conclude our observations and analysis on enabling the model to process hybrid information.

2 Mathematical Representations of Hybrid PUFs

In this section, we present a formal analysis of mathematical models for different hybrid PUFs. In this paper, we focus on the hybrid structures ending with XOR gate, which combine the results from different basic PUF components using the XOR operations. We first start with the basic XOR Arbiter PUFs, then introduce the Feed-Forward PUFs and OAX-PUFs.

2.1 XOR Arbiter PUF

PUFs utilize sequences of binary numbers as input and output, referred to as challenge-response pairs [22]. Arbiter PUF[14] is one of the most common strong PUFs. An Arbiter PUF is composed of a pair of parallel delay paths and an arbiter at the end. For its operation, a signal is simulated from the beginning of the paths. In general, the challenge determines the paths' routine, which will affect the propagation time of the signal, and the signal arriving time difference, noted as Δ, will decide the response. If $\Delta > 0$, the response is 1, otherwise it is 0. The mathematical formulation of Arbiter PUFs can be written as: $r = \text{sign}(\sum_i^n \Delta d_i \prod_j^i c_j)$. where Δd_i is the delay difference between the upper and lower delay paths at stage i, and c_i is the i-th challenge bit. Here, we note parity x_i as $\prod_j^i C_i$, thus: $r = \text{sign}(\sum_i^n \Delta d_i x_i)$. Then, we can write in the matrix format: $r = \text{sign}(\boldsymbol{D} \times \boldsymbol{X}^T)$. We can see from the expression that the basic relationship between the input and output inside the Arbiter PUFs is linear, which makes it vulnerable against MLA.

To add non-linearity to the PUF, [23] introduced XOR Arbiter PUFs, which are composed of several Arbiter PUFs and an XOR gate. The same challenges are fed to all the PUFs, then the signal goes through all the PUFs parallel, and each PUF outputs a one-bit response. In the end, all the response bits are XORed to generate a one-bit output. Assume a n-XOR Arbiter PUF, whose formulation can be written as:

$$r = \prod_k^n \text{sign}(\sum_i^n \Delta d_i^k x_i) = \text{sign}\left(\prod_k^n \sum_i^n \Delta d_i^k x_i\right). \tag{1}$$

We can also write it in the matrix format:

$$r = \text{sign}(\prod_k^n \boldsymbol{D}^k \times \boldsymbol{X}^T). \tag{2}$$

If we look into the Equation 2, we can find that the only non-linearity to learn is still solely the sign(·) function, while the linear part is more complicated than the part of the Arbiter PUF.

2.2 OR-AND-XOR-PUF

According to [2], the structure of combining multiple identied PUFs and XORing the output suffers from reliability-based attack or hybrid MLA. In [28], the OR-AND-XOR PUF (OAX-PUF) is proposed. OAX-PUF utilizes MAX and MIN (OR and AND) bitwise operators to improve the reliability and confuse the adversary by covering some critical unreliability information. Take a (x, y, z)-OAX-PUF as an example. It is composed of $l = x + y + z$ Arbiter PUFs, among which x Arbiter PUFs are connected to an OR gate, y Arbiter PUFs are connected to an AND gate, and z Arbiter PUFs are connected to an XOR gate. Each gate outputs a one-bit result, and then these bits will be fed to another XOR gate which then outputs one final response bit. From the structure, we can find a minimal structure is $(2, 2, 1)$-OAX PUF which introduces all the new features. With the formulation of XOR PUF shown in Equation 2, we can present a (x, y, z)-OAX-PUF as follow:

$$\begin{aligned} r = \Big(& \text{sign}\left(\text{OR}\left(\text{sign}(\boldsymbol{D}^{O_1} \times X^T), \ldots, \text{sign}(\boldsymbol{D}^{O_x} \times X^T)\right)\right) \\ & \cdot \left(\text{AND}\left(\text{sign}(\boldsymbol{D}^{A_1} \times X^T), \ldots, \text{sign}(\boldsymbol{D}^{A_y} \times X^T)\right)\right) \\ & \cdot \prod_{k}^{z} \boldsymbol{D}^k \times X^T \Big) \end{aligned} \tag{3}$$

We can find from the equation that, different from the PUFs with homogeneous structures, in OAX-PUFs, not every PUF component contributes directly to every CRP, e.g., the PUFs connected to the OR and AND gate. In [28], OAX-PUFs are claimed to be more resistant than XOR PUFs against four powerful attacks: logistic regression (LR), reliability assisted CMA-ES, multilayer perceptron (MLP), and hybrid LR-reliability. We conduct fair comparisons in Section 4.

2.3 Homogeneous and Heterogeneous Feed-Forward XOR Arbiter PUF

The XOR PUF is composed of n basic Arbiter PUFs, the combined outputs of which form the final output following an XOR operation. In [2], it was determined that as n increases, the reliability decreases significantly. When n increases to 8, the reliability is around 86.2%, making the PUF impractical without heavy error correction. Although efforts [8,9,6,25] have been devoted to creating reliable PUFs, the trend of reliability decreasing makes a large number of XOR PUFs unrealistic. Besides, a large number of basic PUFs consume high hardware resources. To introduce more non-linearity and bypass the above-mentioned drawbacks, the Feed-Forward PUF [13] was proposed, in which loops are introduced

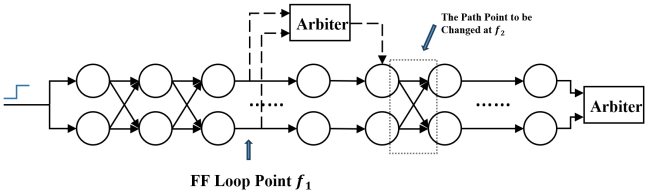

Fig. 1. Feed-Forward PUF

into the delay path of Arbiter PUFs. As shown in Figure 1, there is a loop that begins from the end of stage f_1, goes through an arbiter and then connects to the input of stage f_2. The output of the front delay paths can decide one challenge bit of one posterior stage.

Here we analyze the mathematical model of Feed-Forward PUFs. Take a n-stage (f_1, f_2)-Feed-Forward PUF as an example. Then we can write the formulation as:

$$\begin{aligned} r =& \text{sign}\left(\text{sign}(\sum_{i=1}^{f_1} w_i \cdot x_i) \cdot (\sum_{i=f_2}^{n} w_i \cdot x_i) + \sum_{i=1}^{f_2} w_i \cdot x_i\right) \\ =& \text{sign}\left(\sum_{i=1}^{f_2} w_i \cdot x_i + \sum_{i=f_2}^{n} \text{sign}(\sum_{i=1}^{f_1} w_i \cdot x_i) \cdot w_i \cdot x_i\right). \end{aligned} \tag{4}$$

where the whole n-stage delay paths are divided into two parts, from 1 to f_2 and from f_2 to n. The first part's first f_1 stages contribute to the parity value of the second part, which brings more non-linearity. If we turn the format into a matrix:

$$\begin{aligned} r = \text{sign}\big(&\boldsymbol{D}_{[1,f_2]} \times \boldsymbol{X}_{[1,f_2]} \\ &+ \text{sign}(\boldsymbol{D}_{[1,f_1]} \times \boldsymbol{X}_{[1,f_1]}) \cdot (\boldsymbol{D}_{[f_2,n]} \times \boldsymbol{X}_{[f_2,n]})\big). \end{aligned} \tag{5}$$

From Equation 5, we can find that Feed-Forward PUF contains a cascade sign logic, which appears more complicated than Equation 2. In [23], an XOR-Feed-Forward Arbiter PUF is proposed, which is constructed using an XOR gate and several Feed-Forward PUFs, and all the PUF responses are fed to the XOR gate and generate a one-bit response in the end. The formulation is given as follows:

$$\begin{aligned} r = \prod_{k}^{n} \text{sign}\big(&\boldsymbol{D}_{[1,f_2]} \times \boldsymbol{X}_{[1,f_2]} \\ &+ \text{sign}(\boldsymbol{D}_{[1,f_1]} \times \boldsymbol{X}_{[1,f_1]}) \cdot (\boldsymbol{D}_{[f_2,n]} \times \boldsymbol{X}_{[f_2,n]})\big). \end{aligned} \tag{6}$$

2.4 Other Hybrid PUFs

To resist MLA, more complicated designs have been proposed. In [17], the Interpose PUF is proposed, and it is claimed to be secure against developed MLAs,

including reliability-based attacks. In [26], the divide-and-conquer technique is proposed to analyze the two building blocks of the Interpose PUF separately. For a $(k_{\text{up}}, k_{\text{down}})$-Interpose PUF, the security level is downgraded to a $\max\{k_{\text{up}}\}$-XOR Arbiter PUF. In [1], the concept of homogeneous and heterogeneous XOR-Feed-Forward PUFs is proposed. The homogeneous XOR-Feed-Forward PUF is the traditional one, where all the settings, e.g., the position of loops, are the same for all the component PUFs. The heterogeneous XOR-Feed-Forward PUF, however, has different settings for all the component PUFs, which introduce more complexity without extra resource consumption. In [1,27], the MLA resistance of the heterogeneous XOR-Feed-Forward PUF has been evaluated and it is shown that even a 3-stage 1-loop heterogeneous XOR-Feed-Forward PUF can hardly be broken. Overall, the delay-based PUF and the compositions are still vulnerable against MLAs, with the concern of limited entropy contained in the physical structures. If we go inside the equation of the PUFs, we can find that the parameters to model are solely the delay matrix $\boldsymbol{D}$, which contains at most $n \times k$ parameters. From MLAs, we learn that $\boldsymbol{D}$ can be learned easily if the structure is simple or known and modelled easily by the adversary. Thus, the designers need to design a complex enough structure to cover the relationship between the challenges and responses. In the meantime, we need to be careful with the balance between reliability and structure complexity.

2.5 State-of-Art Modelling Structures

In [18,16,27], general attack models against k-XOR Arbiter PUFs and n-bit XOR Feed-Forward Arbiter PUFs are proposed. In [16], a $\{2^{k-1}, 2^k, 2^{k-1}\}$ multilayer perception model is proposed with k ranging from 5 to 9. In [27] a $\{n, \frac{n}{2}, \frac{n}{2}, n\}$ multilayer perception model is proposed with $n = 64$. In this paper, we choose these three models as the benchmark, as they achieve good performance when modelling XOR PUFs and Feed-Forward XOR PUFs using CRPs, and the model hyper-parameters are determined according to the PUFs structure. We denote them as Ruhrmair2010, Mursi2020 and Wisiol2022. **However, it should be noted that none of these models can attack OAX-PUFs and heterogeneous Feed-Forward PUFs.** In Table 1, the hyperparameters used in [18,16,27] are listed. They both use tanh as the activation function, since for the arbiter PUFs, using tanh instead of relu can guarantee the data normalization between the layers, i.e., with zero mean[12]. It should be noted that the structures of Mursi2020 and Wisiol2022 are slightly different regarding different PUFs, even from the same category. For Mursi2020, the number of neurons is determined by the stages of XOR PUFs.

3 Methodology

In this section, we first formally analyse the problem encountered when modelling PUFs using a machine learning model. Then, we present the Mixture of the PUF-Expert (MoPE) structure and the technical design from the perspective of PUF modelling.

Table 1. Hyperparameter Value Used in Ruhrmair[18], Mursi2020[16] and Wisiol2022[27].

Hyper Parameters	**Ruhrmair2010**	**Mursi2020**	**Wisiol2022**
Architecture	Logistic Regression	$(2^{k-1}, 2^k, 2^{k-1})$	$(n, \frac{n}{2}, \frac{n}{2}, n)$
Kernel Initializer	Normal dist.	Normal dist.	Normal dist.
Optimizer	Adam[11]	Adam	Adam
Hid. Lay. active.	-	tanh	tanh
Learning rate	Adaptive	Adaptive	Adaptive
Loss function	BCELoss	BCELoss	BCELoss

3.1 Local Minima Problem

The local minima problem is a common problem in machine learning, which will stop the optimisation with a low accuracy performance. The issue arises when an optimization algorithm, tasked with minimizing a loss function that measures the discrepancy between the predicted and actual PUF responses, becomes trapped in a local minimum point of the overall feature space, where the function value is lower than at neighbouring points but not necessarily the lowest possible value globally. This scenario is especially prevalent in high-dimensional spaces common to PUF modelling, where the landscape of the loss function is riddled with numerous local minima. Local minima hinder the modelling process by preventing convergence to the global minimum, where the most accurate model resides. This results in suboptimal PUF models that fail to capture the intricate mappings between challenges and responses accurately. The ramifications are twofold: firstly, the reliability of PUF-based security systems may be compromised due to inaccuracies in authentication or key generation processes. Secondly, the resilience of PUFs against modelling attacks, wherein an adversary attempts to construct a predictive model of the PUF, may be overstated if the models used for evaluation are themselves trapped in local minima and thus are not representative of the best possible modelling efforts.

Several strategies have been proposed to mitigate the local minima problem in modelling. These include the use of advanced optimization techniques such as simulated annealing, genetic algorithms, or gradient-based methods with momentum terms that can potentially escape shallow local minima [7]. Additionally, employing regularization methods to simplify the model or initializing the optimization process from multiple random starting points can also increase the likelihood of converging to a global minimum [21]. Furthermore, hybrid approaches that combine machine learning models with domain-specific knowledge about the PUF architecture and behaviour have shown promise in enhancing model accuracy and robustness [1]. Thus, the problem of local minima represents a significant challenge in the modelling of PUFs. Overcoming this hurdle is essential for the development of reliable and secure PUF-based systems. Continued research into sophisticated optimization methods and model architectures

is critical for advancing the state-of-the-art in PUF modelling and ensuring the security and integrity of hardware-based cryptographic systems.

There are two main factors contributing to local minimal in PUF modelling attacks:

1. First, the **inappropriate structure** of the model. For example, in [16], to model k-XOR-PUF, the model structure is $\{2^k, 2^{k-1}, 2^k\}$, which can ensure a high training accuracy and test accuracy. However, if the middle hidden layer is removed and the model structure is $\{2^k, 2^k\}$, it is hard to conduct a successful attack even on 2-XOR-PUF such that the training accuracy might vary from $60\% - 99\%$, and the test accuracy gets trapped around 60%, regardless of how much data is used for training. This is a classic local minima problem resulting in overfitting or failures in training.
2. Second, the insufficient amount of training data. From many MLA work [18,27,16,10], a necessary amount of training data is required to conduct a successful attack. If less training data is supplied, the model does not work. We also observe that the performance of one type of PUF fluctuates with different set-up random seeds and different choices of CRPs have a significant influence on the success of modelling. Thus we argue that we should not evaluate the performance of one type of PUF. The problem arises when the adversary tries to attack a new PUF, without knowing the ideal amount of training data or model structure. The training accuracy will still increase to a certain value, e.g., 70% and 75%, with training; however, the test accuracy is stuck at 50%. The model is stuck in the local minimum point and fails to complete the task. In this case the wrong judgement on the MLA resistance would be given.

Finding the global minimal loss for the model instead of the local minimal loss is an important question when modelling PUFs, especially for increasingly complicated designs, e.g., Feed-Forward PUF, Interpose PUF, and OAX-PUF. This kind of hybrid structure makes it harder to deal with. Based on the aforementioned two factors, we need to first, find the most suitable model structure for the specific PUFs. Second, we need to find the most appropriate training data for it.

3.2 Modelling PUFs using Miture-of-Experts

We incorporate the generic model, which utilizes an MoE structure [24] to attack multiple types of PUFs without modifying any hyperparameters. This idea is presented in the left part of Figure 2, where different experts are trained to learn different involved PUFs in the hybrid PUF. In the right part of Figure 2, the model accepts a challenge as input and produces the predicted response as output. Challenges are processed by an input layer connected to three experts. These experts are tailored to handle the distinct features of CRPs. Each expert comprises of two hidden layers, each with 32 neurons. The first layer is directly connected to the input layer, while the second links to the gate function. The gate function assigns weights to the experts, amalgamates their outputs, and

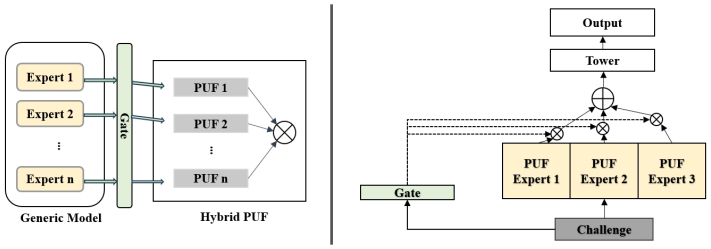

Fig. 2. Generic Model for a Hybrid PUF.

channels them to the tower. Initially, the challenge bits C^N are converted (where N represents the PUF stages) into the feature vector X^N, aligning with the structure of the arbiter-based PUF, $x_i = \prod_{j=i}^{n} c_j$. This transformation aids the model in perceiving the decision boundary as a hyperplane. The response r serves as the label and is adjusted to the range $0, 1$, if not already within it, to align with the activation function. Post-feature engineering, the input layer is structured to accommodate these features. In the MoPE layer, we establish K experts, with the count being adaptable based on the complexity of targeted PUFs. Normally, we set $k = 5$. The expert structure remains consistent across all PUF types, as two hidden layers equipped with non-linear activation functions are believed to model any function, given sufficient parameters. The $k-$th expert, denoted as $f_k(\cdot)$, is designed to extract specific insights or features from the input. Each expert delivers their unique interpretation of the input: $h_k(X) = f_k(X^n)$.

To harness the expertise of various experts without overburdening the model with excessive parameters, the gate function $g(x)$ is introduced. This function evaluates the features and determines the weight. The $softmax(\cdot)$ activation function posts the $N \times K$ kernel W_{gk} to distribute weights among experts and ensure the model prioritises the most apt one. Consequently, weights are computed as: $g(X) = softmax(W_{NK}(X))$. The weight assigned to the k-th expert is represented as $g^k(X)$, ensuring that $\sum_{k=1}^{K} g^k(X) = 1$. Subsequently, the MoE layer's output is derived by amalgamating the outputs of the experts: $MoPE(X) = \sum_{i=1}^{K} g^i(X)h_i(X)$. Then, the tower layer, $T(\cdot)$, is established, tasked with processing the composite information supplied by the experts. This layer then connects to the output layer, which employs the $sigmoid(\cdot)$ activation function to restrict the prediction output to the range $\{0, 1\}$. The MoPE structure's inherent flexibility allows the gate function to integrate multiple experts, facilitating network scalability to accommodate the diverse complexities inherent to PUFs.

3.3 Routine Algorithm

In the mixture-of-experts (MoPE) architecture, the routing algorithm plays a crucial role in determining which experts are activated for a given input. This

selection process is typically governed by a trainable gating network, which evaluates the input and allocates weights to each expert based on their relevance to the current task. The gating network, often implemented as a softmax layer, produces a distribution over the experts, where the weights reflect the confidence in each expert's ability to contribute to the task at hand. This mechanism enables the MoPE model to dynamically allocate computation across different experts, allowing it to leverage specialized knowledge and improve overall performance. The choice of experts is thus data-driven, guided by the learning process where the gating network adjusts its parameters through backpropagation, optimizing the allocation of inputs to experts based on the loss minimization criterion. This approach ensures that the MoPE architecture can adaptively focus on the most relevant experts for processing diverse and complex inputs, enhancing the model's flexibility and efficiency. The benefits of selecting the most suitable experts can be comprehended through the *top-1* strategy [4], where only one of the most suitable experts will be used for the modelling task. However, for the experts' structure being fixed as 32×32, it is hard for such a structure to model complex PUFs, e.g., 7-XOR-PUF. In [24], 2.4 million training data are used for a successful attack. However, the modelling fails when the number of used experts is fixed as 1. This indicates that only one expert is insufficient to model complex PUFs, for they contain complicated information that requires multiple experts to work together. Thus, in this paper, we argue that multiple experts can learn different parts of the information contained in the PUF. Thus, for PUF modelling, the routine gate does not only perform as an optimization-selector of experts but also as a structure-information learner who can learn the structure information contained in the hybrid design.

3.4 Proposed Transition Theorem

In this section, we consider the mathematical models of hybrid PUFs, and evaluate the relationship between the modelling resistance and structures.

Theorem 1 (Transition Theorem: OR, AND). *The OR and AND gate compress multiple PUFs into one stable PUF. Thus, a $\{x, y, z\}$-OAX PUF, is equivalent to $(z+2)$-XOR PUFs.*

Proof. As shown in Equation (3), the number of multiplication factors equals the number of XOR PUFs plus two from the AND gate and OR gate. We can understand the OR operation as the MAX operation. Thus, we can get the following:

$$\begin{aligned} &\mathrm{OR}\left(\mathrm{sign}(\boldsymbol{D}^{O_1} \times X^T), \ldots, \mathrm{sign}(\boldsymbol{D}^{O_x} \times X^T)\right) \\ &= \mathrm{MAX}\left(\mathrm{sign}(\boldsymbol{D}^{O_1} \times X^T), \ldots, \mathrm{sign}(\boldsymbol{D}^{O_x} \times X^T)\right) \\ &= \mathrm{MAX}\left(\mathrm{sign}(\boldsymbol{D}^{O_1}, \ldots, \mathrm{sign}(\boldsymbol{D}^{O_x})\right) \times X^T \\ &= \mathrm{sign}(\boldsymbol{D}^{max} \times X^T). \end{aligned} \tag{7}$$

Consider the following relation:

$$\begin{aligned} &\mathrm{MAX}(\boldsymbol{A} \times \boldsymbol{X}, \boldsymbol{B} \times \boldsymbol{X}) \\ =&\mathrm{MAX}(\sum_{i}^{n} a_i \cdot x_i, \sum_{i}^{n} b_i \cdot x_i) \end{aligned} \tag{8}$$

Here we create a matrix $\boldsymbol{C} = [c_0, c_1, \ldots, c_n]$, where

$$c_i = \begin{cases} \max(a_i, b_i), & \text{if } x_i = 1 \\ \min(a_i, b_i), & \text{if } x_i = -1 \end{cases} \tag{9}$$

We can find that the representation collapses into a single arbiter PUF.

Theorem 2 (Transition Theorem: Feed-Forward Loop). *For every additional loop added to the PUF, the complexity increases approximately two times.*

Proof. From Equation (5), we can find that the original delay path is divided into two parts, $D_{[1,f_2]}$ and $D_{[1,f_2]}$; the first represents the delay path from the start to the loop point f_2. The second represents the dot result of the front and back two parts, where the format is similar to the mathematical representation of XOR PUFs $(\mathrm{sign}(\boldsymbol{D}_{[1,f_1]} \times \boldsymbol{X}_{[1,f_1]}) \cdot (\boldsymbol{D}_{[f_2,n]} \times \boldsymbol{X}_{[f_2,n]}))$. To summarize, the total delay added up is from one delay path of normal Arbiter PUF with the length of f_2 and an XOR PUF with different challenge input to different PUFs. Thus, we argue that the complexity introduced by the loop is between the arbiter PUF and XOR PUF since the length of the delay path is shorter than the original one.

Theorem 3 (Transition Theorem: Heterogeneous Feed-Forward XOR PUFs). *A $\{n, k\}$-Heterogeneous Feed-Forward XOR PUF, is equivalent to $\{n * 2^k\}$-XOR PUF.*

Proof. For a heterogeneous Feed-Forward XOR PUF, the loop positions are different for every involved PUF. Thus, no information can be shared between modelling the basic PUFs, and they can be considered to be independent of each other. For every additional loop, the delay path is divided into two parts one time further. Thus in total, we can view a $\{n, k\}$-Heterogeneous Feed-Forward XOR PUF as an XOR PUF with $\{n * 2^k\}$ PUFs involved.

4 Experiments and Evaluation

In this section, we evaluate the results of modelling different hybrid PUFs using MoPE. Based on the results of the successful attack, we then analyse how the modelling capability can be achieved and turn it into a modelling attack targeting new PUFs. We analyse the mathematical model of the instance of the hybrid PUF and evaluate the proposed ***transition theorem***. We apply the theorem to other modelling attack strategies that can not break certain PUFs. We show that with the help of modifications, they can break previously unbreakable PUFs.

Table 2. Modelling results for hybrid PUFs using the generic model.

	Or, And, XOR	crp	time	acc
OAX PUF	(4, 4, 0)	24k	<1min	>93%
	(0, 4, 4)	240k	<1min	>96%
	(4, 0, 4)	240k	<1min	>96%
	(4, 4, 4)	300k	<1min	>96%
Type	**k**	**Loops**	**crp**	**time**	**acc**
Homogeneous FF-PUF	1	1	20k	<2min	>94%
		2	120k	<1min	>97%
		3	250k	<1min	>98%
		4	500k	<1min	>98%
		5	1M	<4min	>94%
	3	1	120k	<1min	>90%
		2	400k	<5min	>93%
Heterogeneous FF-PUF	2	1	160k	<1min	>98%
	3	1	640k	<5min	>98%
	2	2	400k	<1min	>95%

4.1 Modelling Hybrid PUFs using the Generic Model

In this section, we present the modelling results on hybrid PUFs using the generic model proposed in Section 3. Avvaru et al. introduced the homogeneous and heterogeneous Feed-Forward XOR PUFs in [1]. Subsequent to their work, a multitude of machine learning models were proposed to target FF-APUFs [27]. A large portion of these models capitalize on the inconsistent reliability of PUF designs, focusing particularly on homogeneous XOR FF PUFs with uniform loop positions. In contrast, heterogeneous XOR-FF-APUFs are largely considered resilient against modelling attacks. As evidenced in Table 2, we successfully modelled 2−loop FF PUFs with 2 XOR stages and 1−loop FF PUFs with 3 XOR stages, achieving accuracy exceeding 95% and 98%, respectively. As shown in Table 2, the homogeneous/heterogeneous Feed-Forward XOR PUFs and OAX PUFs are modelled with high accuracy beyond 90%, which is a successful attack. From the results, by comparing the cost of training data, we observe that the cost of heterogeneous Feed-Forward PUFs has parameter-related similarity with XOR PUFs. For the generic model, it consumes $80k$ and $240k$ CPRs for 4-XOR-PUFs and 5-XOR-PUFs separately, and $160k$ CRPs for a $\{2, 1\}$-heterogeneous Feed-Forward PUF. Since the model is generic and fixed, and it does not need any prior knowledge of the PUFs, we can reach an easy observation that the modelling resistance of $\{2, 1\}$-heterogeneous Feed-Forward PUF is between the 4-XOR-PUFs and 5-XOR-PUFs. For the same reason, we can say the modelling resistance of $\{2, 2\}$-heterogeneous Feed-Forward and $\{3, 1\}$-heterogeneous Feed-Forward PUF are between the 5-XOR-PUFs and 6-XOR-PUFs. We can find that the observations match our analysis of the mathematical representation in Sec-

Table 3. Transition details for Ruhrmair2010, Mursi2020 and Wilsiol2022.

Target PUF	**Ruhrmair2010 Before Transition**	**Ruhrmair2010 After Transition**
(x, y, z)-OAX PUF	LR	LR based on Equation (3)
(k, l_{loop})-Homo. FF-PUF	LR	LR based on Equation (6)
Target PUF	**Mursi2020 Before Transition**	**Mursi2020 After Transition**
(x, y, z)-OAX PUF	-	$(2^{k+1}, 2^{k+2}, 2^{k+1})$
(k, l_{loop})-Homo. FF-PUF	$(2^{k-1}, 2^{k}, 2^{k-1})$	$(2^{(k-1)*l_{loop}}, 2^{k*l_{loop}}, 2^{(k-1)*l_{loop}})$
(k, l_{loop})-Hete. FF-PUF	$(2^{k-1}, 2^{k}, 2^{k-1})$	$(2^{(k-1)*2^{l_{loop}}}, 2^{k*2^{l_{loop}}}, 2^{(k-1)*2^{l_{loop}}})$
Target PUF	**Wilsio2022 Before Transition**	**Wilsiol2022 After Transition**
(x, y, z)-OAX PUF	-	$(2^{(x+y+z)}, 2^{(x+y+z-1)}, 2^{(x+y+z-1)}, 2^{(x+y+z)})$
(k, l_{loop})-Homo. FF-PUF	$(n, \frac{n}{2}, \frac{n}{2}, n)$	$(2^{k*l_{loop}}, 2^{(k-1)*l_{loop}}, 2^{(k-1)*l_{loop}}, 2^{k*l_{loop}})$
(k, l_{loop})-Hete. FF-PUF	$(n, \frac{n}{2}, \frac{n}{2}, n)$	$(2^{k*l_{loop}}, 2^{(k-1)*l_{loop}}, 2^{(k-1)*l_{loop}}, 2^{k*l_{loop}})$

*LR: Logistic Regression

tion 2 and the theorems proposed in Section 3. In the next section, we present the performance of Ruhrmair2010, Mursi2020 and Wilsiol2022 after applying the transition theorem.

4.2 Modelling Hybrid PUFs Using the Proposed *Transition Theorem*

As discussed in Section 3, Ruhrmair2010, Mursi2020 and Wisiol2022 were initially proposed for modelling XOR PUFs and homogeneous Feed-Forward XOR PUFs at the first beginning. In [27], Wisiols et al. have shown the capability of neural networks to model PUFs. However, they failed to model heterogeneous Feed-Forward XOR PUFs and claimed a major modification of the model structure is needed. We argue that the major problem is the structure mismatch between the models and the PUFs, especially for hybrid PUFs. The hybrid PUFs introduce complicated structure designs into the PUFs, which confuses the adversary. In [13], it has been evaluated that the basic Feed-Forward PUF is harder to model compared to the basic Arbiter PUF; thus, when combining them together into an XOR construction, we cannot simply consider Feed-Forward XOR PUF to be the same as XOR Arbiter PUFs. From the experiments, we notice the local minimal problem occurs when the accuracy gets stuck at around 60%, or the overfitting problem occurs when the test accuracy does not match the training accuracy. According to the observations in Section 4.1 and the *transition theorem*, we optimize the structure for Mursi2020 and Wisiol2022 to fit them for the hybrid PUFs modelling tasks, as shown in Table 3. Specially, when modelling homogeneous Feed-Forward XOR PUF, we modify the structure according to the proposed theorem and treat a (k, l_{loop})-homogeneous Feed-Forward XOR PUF as a $k*l_{loop}$-XOR PUF, a (k, l_{loop})-heterogeneous Feed-Forward XOR PUF as a $k * 2^{l_{loop}}$-XOR PUF, a (x, y, z)-OAX PUF as a $(x + y + z)$-XOR PUF. As

Table 4. Modelling results using OPTIMIZED Ruhrmair2010 [18], Mursi2020 [16] and Wisiol2022 [27] after Applying the ***Transition Theorem***.

Type	k	Loops	crp	time	acc
Homo. FF-PUF	2	1	30k†, 120k††, 120k‡	<1min†, 1min††, 1min‡	>97%†, 95%††, 95%‡
		3	240k†, 360k††, 360k‡	<1min†, 1min††, 1min‡	>97%†, 97%††, 96%‡
		4	700k†, 720k††, 720k‡	<5min†, 2min††, 2min‡	>98%†, 99%††, 98%‡
		5	1.4M†, 1.4M††, 1.4M‡	<10min†, 5min††, 5min‡	>97%†, 98%††, 97%‡
	3	1	240k†, 240k††, 240k‡	<1min†, 1min††, 1min‡	>95%†, 98%††, 96%‡
		2	480k†, 480k††, 480k‡	<2min†, 2min††, 2min‡	>95%†, 93%††, 93%‡
Hete. FF-PUF	2	1	200k††, 200k‡	< 2min††, min‡	> 98%††, 97%‡
	3	1	640k††, 640k‡	< 5min††, 5min‡	> 98%††, 98%‡
	2	2	400k††, 400k‡	< 1min††, 1min‡	> 95%††, 95%‡
	Or, And, XOR		**crp**	**time**	**acc**
OAX PUF	(4, 4, 0)		20k†, 20k††, 20k‡	<1min†, 1min††, 1min‡	>97%†, 95%††, 95%‡
	(0, 4, 4)		20k†, 20k††, 20k‡	<1min†, 1min††, 1min‡	>97%†, 95%††, 95%‡
	(4, 0, 4)		240k†, 240k††, 300k‡	<2min†, 2min††, 2min‡	>98%†, 97%††, 97%‡
	(4, 4, 4)		1M†, 800k††, 800k‡	<3min†, 3min††, 3min‡	>97%†, 98%††, 98%‡

† Ruhrmair2010[18]; †† Mursi2020 [16]; ‡ Wisiol2022 [27].

shown in 4, we can find that these two models achieve good accuracy beyond 90% for all the hybrid PUFs, which are MLA-resistant before the modifications. From the successful attacks and cost of CRPs, we find that consumption generally follows the transition theorem. For example, a $(4, 0, 0)$-OAX PUF costs 24k CRPs, similar to a 2-XOR PUF. For a $(3, 1)$-heterogeneous FF-PUF, it costs 640k CRPs, which is similar to a 6-XOR PUF. Therefore, we conclude that the transition theorem is feasible on the two neural network-based attacks.

Acknowledgments. This research is supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research Development Programme, under grant FCP-NUS-RG-2022- 019. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the National Research Foundation, Singapore and Infocomm Media Development Authority. The work of Prosanta Gope was supported by The Royal Society Research Grant under grant RGS\R1\221183.

5 Conclusion

In this study, we critically assess the strategy of integrating diverse PUF components into Hybrid PUFs. To the best of our knowledge, our investigation presents the first achievement in modelling two widely recognized PUFs: the heterogeneous Feed-Forward (FF) XOR PUF and the OAX PUFs. Furthermore, we introduce the *transition theorem* based on the mathematical representations. Our analysis suggests that incorporating a FF loops and OR/AND logic both

reduce the overall PUF complexity, rendering the design more susceptible to machine learning attacks. We apply the *transition theorem* to two prominent attack frameworks, Mursi2020[16] and Wilsol2022[27], which were previously ineffective against these PUFs. By modifying their structures with transition theorem, we successfully executed attacks against these PUFs, confirming it's validity. We not only offer insights into the design of novel Hybrid PUFs but also underscore vulnerabilities in PUFs previously deemed secure, suggesting a key contribution to both the development and security analysis of PUF technologies.

A Transition Theorem and Proofs

We present the full analysis and proofs of the *transition theorem*.

A.1 OAX-PUF

Theorem 1 (Transition Theorem: OR, AND). *The OR and AND gate compress multiple PUFs into one stable PUF. Thus for a $\{x, y, z\}$-OAX PUF, it is equivalent to $(z+2)$-XOR PUFs.*

The Theorem 1 argues that, for a hybrid PUF composed of multiple basic Arbiter PUFs and an OR or AND gate, it is equal to one Arbiter PUF. Figure 3 shows an example how to convert a 2-OR PUF into an Arbiter PUF.

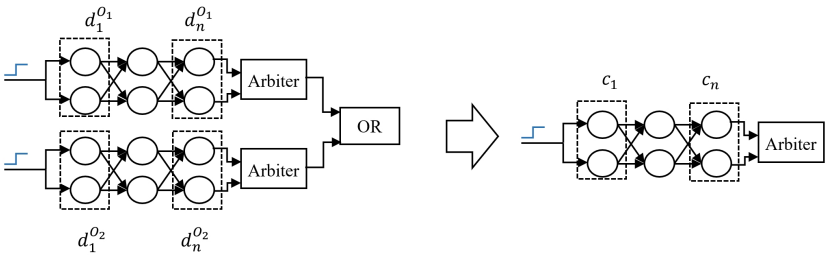

Fig. 3. Convert a 2-OR PUF into an Arbiter PUF

Proof.

$$
\begin{aligned}
r &= \mathrm{OR}\left(\mathrm{sign}(\boldsymbol{D}^{O_1} \times X^T), \ldots, \mathrm{sign}(\boldsymbol{D}^{O_x} \times X^T)\right) && (1)\ \textit{The formation of OR PUF.}\\
&= \mathrm{MAX}\left(\mathrm{sign}(\boldsymbol{D}^{O_1} \times X^T), \ldots, \mathrm{sign}(\boldsymbol{D}^{O_x} \times X^T)\right) && (2)\ \textit{OR and MAX are equivalent in boolean fields.}\\
&= \mathrm{MAX}\left(\mathrm{sign}\left(\textstyle\sum_i^n d_i^{O_1} \cdot x_i\right), \ldots, \left(\textstyle\sum_i^n d_i^{O_x} \cdot x_i\right)\right) && (3)\ \boldsymbol{D}^{O_1} \times X^T = \textstyle\sum_i^n d_i^{O_x} \cdot x_i\\
&= \mathrm{sign}\left(\mathrm{MAX}\left(\textstyle\sum_i^n d_i^{O_1} \cdot x_i\right), \ldots, \left(\textstyle\sum_i^n d_i^{O_x} \cdot x_i\right)\right) && (4)\ \textit{The MAX and sign}(\cdot)\ \textit{satisfy the law of commutation.}\\
&= \mathrm{sign}\left(\textstyle\sum_i^n \mathrm{MAX}\left(d_i^{O_1} \cdot x_i, \ldots, d_i^{O_x} \cdot x_i\right)\right) && (5)\ \textit{Put MAX inside.}
\end{aligned}
$$

Since all the $d_i^{O_j}$ are fixed for any i, j, we can write Equation (5) as:

$$r = \text{sign}\left(\sum_{i}^{n} c_i \cdot x_i\right) \tag{6}$$

where

$$c_i = \begin{cases} \max\left(d_i^{O_1}, \ldots, d_i^{O_x}\right), & \text{if } x_i = 1 \\ \min\left(d_i^{O_1}, \ldots, d_i^{O_x}\right), & \text{if } x_i = -1 \end{cases} \tag{7}$$

We can find Equation 6 is the same as the formulation of the arbiter PUF that $r = \boldsymbol{C} \times \boldsymbol{X}^T$. For the AND gate, we analyse in a similar way with OR gate, that we only need to change the MAX logic in Equation (2-5) to MIN logic.
Thus for a $\{x, y, z\}$-OAX PUF, the x-OR-PUF and y-AND-PUF can be treated as two Arbiter PUFs. Thus the $\{x, y, z\}$-OAX PUF is in fact equivalent to $(z+2)$-XOR PUFs.

B Feed-Forward PUF

Theorem 2 (Transition Theorem: Feed-Forward Loop). *For every additional loop added to the PUF, an XOR composition is added to the delay path and the complexity increases approximately two times.*

The Theorem 2 argues that, for a Feed-Forward PUF, the Feed-Forward loop derives an XOR-PUF from the original delay path. Figure 4 shows an example of converting a 1-Feed-Forward-loop-PUF into a combination of an Arbiter PUF and a 2-XOR PUF.

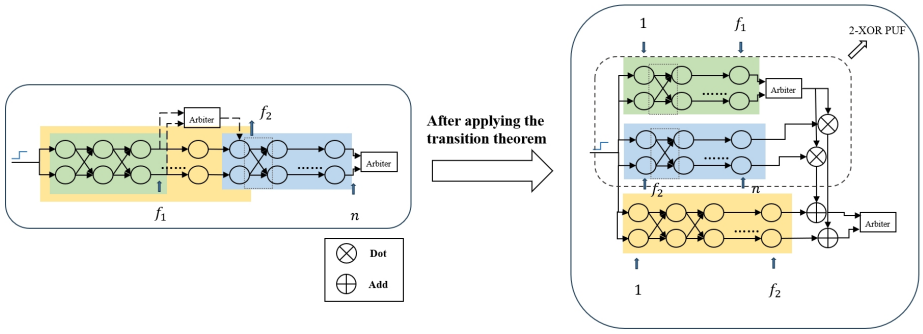

Fig. 4. Convert a Feed-Forward PUF into an Arbiter PUF

Proof. The Feed-Forward PUF can be formulated as:

$$\begin{aligned} r = &\text{sign}\left(\boldsymbol{D}_{[1,f_2]} \times \boldsymbol{X}_{[1,f_2]}\right. \\ &\left. + \text{sign}\left(\boldsymbol{D}_{[1,f_1]} \times \boldsymbol{X}_{[1,f_1]}\right) \cdot \left(\boldsymbol{D}_{[f_2,n]} \times \boldsymbol{X}_{[f_2,n]}\right)\right) \end{aligned} \tag{8}$$

We can find that the original delay path is divided into two parts, $D_{[1,f_2]}$ and $D_{[1,f_2]}$; the first represents the delay path from the start to the loop point f_2. The second represents the dot result of the front and back two parts, where the format is similar to the mathematical representation of XOR PUFs ($\text{sign}(\boldsymbol{D}_{[1,f_1]} \times \boldsymbol{X}_{[1,f_1]}) \cdot (\boldsymbol{D}_{[f_2,n]} \times \boldsymbol{X}_{[f_2,n]})$). To summarize, the total delay added up is from one delay path of normal Arbiter PUF with the length of f_2 and an XOR PUF with different challenge input to different PUFs. Thus, we argue that the complexity introduced by the loop is between the arbiter PUF and XOR PUF since the length of the delay path is shorter than the original one.

References

1. Avvaru, S.S., Zeng, Z., Parhi, K.K.: Homogeneous and heterogeneous feed-forward xor physical unclonable functions. IEEE Transactions on Information Forensics and Security **15**, 2485–2498 (2020)
2. Becker, G.T.: The gap between promise and reality: On the insecurity of xor arbiter pufs. In: Cryptographic Hardware and Embedded Systems–CHES 2015: 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings 17. pp. 535–555. Springer (2015)
3. Delvaux, J., Verbauwhede, I.: Side channel modeling attacks on 65nm arbiter pufs exploiting cmos device noise. In: 2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST). pp. 137–142. IEEE (2013)
4. Fedus, W., Zoph, B., Shazeer, N.: Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research **23**(120), 1–39 (2022)
5. Ganji, F., Tajik, S., Seifert, J.P.: Pac learning of arbiter pufs. Journal of Cryptographic Engineering **6**, 249–258 (2016)
6. Golanbari, M.S., Kiamehr, S., Bishnoi, R., Tahoori, M.B.: Reliable memory puf design for low-power applications. In: 2018 19th International Symposium on Quality Electronic Design (ISQED). pp. 207–213. IEEE (2018)
7. Guilmeau, T., Chouzenoux, E., Elvira, V.: Simulated annealing: A review and a new scheme. In: 2021 IEEE Statistical Signal Processing Workshop (SSP). pp. 101–105. IEEE (2021)
8. He, Z., Wan, M., Deng, J., Bai, C., Dai, K.: A reliable strong puf based on switched-capacitor circuit. IEEE Transactions on Very Large Scale Integration (VLSI) Systems **26**(6), 1073–1083 (2018)
9. Hiller, M., Kürzinger, L., Sigl, G.: Review of error correction for pufs and evaluation on state-of-the-art fpgas. Journal of cryptographic engineering **10**(3), 229–247 (2020)
10. Khalafalla, M., Gebotys, C.: 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). pp. 204–209 (2019). https://doi.org/10.23919/DATE.2019.8714862
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
12. LeCun, Y., Bottou, L., Orr, G.B., Müller, K.R.: Efficient backprop. In: Neural networks: Tricks of the trade, pp. 9–50. Springer (2002)

13. Lee, J.W., Lim, D., Gassend, B., Suh, G.E., Van Dijk, M., Devadas, S.: A technique to build a secret key in integrated circuits for identification and authentication applications. In: 2004 Symposium on VLSI Circuits. Digest of Technical Papers (IEEE Cat. No. 04CH37525). pp. 176–179. IEEE (2004)
14. Lim, D., Lee, J.W., Gassend, B., Suh, G.E., Van Dijk, M., Devadas, S.: Extracting secret keys from integrated circuits. IEEE Transactions on Very Large Scale Integration (VLSI) Systems **13**(10), 1200–1205 (2005)
15. Mishra, N., Pratihar, K., Mandal, S., Chakraborty, A., Rührmair, U., Mukhopadhyay, D.: Calypso: An enhanced search optimization based framework to model delay-based pufs. IACR Transactions on Cryptographic Hardware and Embedded Systems **2024**(1), 501–526 (2024)
16. Mursi, K.T., Thapaliya, B., Zhuang, Y., Aseeri, A.O., Alkatheiri, M.S.: A fast deep learning method for security vulnerability study of xor pufs. Electronics **9**(10), 1715 (2020)
17. Nguyen, P.H., Sahoo, D.P., Jin, C., Mahmood, K., Rührmair, U., Van Dijk, M.: The interpose puf: Secure puf design against state-of-the-art machine learning attacks. Cryptology ePrint Archive (2018)
18. Rührmair, U., Sehnke, F., Sölter, J., Dror, G., Devadas, S., Schmidhuber, J.: Modeling attacks on physical unclonable functions. In: Proceedings of the 17th ACM conference on Computer and communications security. pp. 237–249 (2010)
19. Sahoo, D.P., Mukhopadhyay, D., Chakraborty, R.S., Nguyen, P.H.: A multiplexer-based arbiter puf composition with enhanced reliability and security. IEEE Transactions on Computers **67**(3), 403–417 (2017)
20. Santikellur, P., Bhattacharyay, A., Chakraborty, R.S.: Deep learning based model building attacks on arbiter puf compositions. Cryptology ePrint Archive (2019)
21. Santikellur, P., Prakash, S.R., Chakraborty, R.S., et al.: A computationally efficient tensor regression network based modeling attack on xor apuf. In: 2019 Asian Hardware Oriented Security and Trust Symposium (AsianHOST). pp. 1–6. IEEE (2019)
22. Shi, J., Lu, Y., Zhang, J.: Approximation attacks on strong pufs. IEEE transactions on computer-aided design of integrated circuits and systems **39**(10), 2138–2151 (2019)
23. Suh, G.E., Devadas, S.: Physical unclonable functions for device authentication and secret key generation. In: Proceedings of the 44th annual design automation conference. pp. 9–14 (2007)
24. Wang, S., Li, Y., Li, H., Zhu, T., Li, Z., Ou, W.: Multi-task learning with calibrated mixture of insightful experts. In: 2022 IEEE 38th International Conference on Data Engineering (ICDE). pp. 3307–3319. IEEE (2022)
25. Wang, W.C., Yona, Y., Diggavi, S.N., Gupta, P.: Design and analysis of stability-guaranteed pufs. IEEE Transactions on Information Forensics and Security **13**(4), 978–992 (2017)
26. Wisiol, N., Mühl, C., Pirnay, N., Nguyen, P.H., Margraf, M., Seifert, J.P., Van Dijk, M., Rührmair, U.: Splitting the interpose puf: A novel modeling attack strategy. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 97–120 (2020)
27. Wisiol, N., Thapaliya, B., Mursi, K.T., Seifert, J.P., Zhuang, Y.: Neural network modeling attacks on arbiter-puf-based designs. IEEE Transactions on Information Forensics and Security **17**, 2719–2731 (2022)
28. Yao, J., Pang, L., Su, Y., Zhang, Z., Yang, W., Fu, A., Gao, Y.: Design and evaluate recomposited or-and-xor-puf. IEEE Transactions on Emerging Topics in Computing **10**(2), 662–677 (2022)