IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 17, 2021, accepted December 2, 2021, date of publication December 6, 2021,
date of current version December 29, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3132915

Security Evaluation of n-Choose-k-Sum PUFs Against Modeling Attacks

LIHUI PANG[1,2], HYOUNGSHICK KIM[2], BIN YANG[1], (Member, IEEE), XINLIN WANG[1], AND YANSONG GAO[3], (Member, IEEE)

[1]School of Electrical Engineering, University of South China, Hengyang 421001, China
[2]Department of Software, Sungkyunkwan University, Suwon 440746, South Korea
[3]School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

Corresponding author: Yansong Gao (yansong.gao@njust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61901209, Grant 62002167, and Grant 61871210; in part by National Natural Science Foundation of Jiangsu under Grant BK20200461; in part by the Institute for Information & Communications Technology Promotion funded by the Korea Government under Grant 2018-0-00532; in part by the Hunan Province Key Laboratory for Ultra–Fast Micro/Nano Technology and Advanced Laser Manufacture under Grant 2018TP1041.

ABSTRACT A physical unclonable function (PUF) generates hardware intrinsic volatile secrets by exploiting uncontrollable manufacturing randomness. Several PUF candidates use challenge-response pairs (CRPs) to enhance their security. A practically plausible idea for PUF is to use the technique called n-Choose-k-Sum (nCk) structure/topology, exemplified by the memristive device based on strong PUF (mrSPUF). The previous analysis claimed that nCk-PUF is resilient to modeling attacks. This paper provides numerical security evaluations on simulated CRPs – we first construct a mathematical model of nCk-PUF, and then analyze the possibility of modeling attacks using machine learning (ML). Our analysis demonstrates that our modeling attacks could break the unpredictability of nCk-PUF even though the non-linearity of nCk-PUF could be increased due to XOR-operations on its circuit. Our analysis demonstrates that our modeling attacks could break the unpredictability of nCk-PUF even though the non-linearity of nCk-PUF could be increased due to XOR-operations on its circuit. Consequently, our analysis suggest that n-Choose-k-Sum PUFs (including mrSPUF) sharing similar topology could be insecure. It should be careful when utilizing nCk-PUF for strong PUF applications, and it is suggested to add some security blocks to protect the challenge-response interface, such as the Lockdown-PUF with additional random number generator (RNG), TREVERSE with additional Hash.

INDEX TERMS PUF, mrSPUF, n-Choose-k-Sum PUFs, machine learning, modeling attack.

I. INTRODUCTION

A physical unclonable function (PUF) is a hardware security primitive that takes an input, namely challenge, and produces a corresponding instance-specific output, name response [1]. Generally, the PUF is akin to a hardware fingerprint, which exploits the uncontrollable variations resulted from the manufacturing process [2]. Therefore, no two identical PUF instances can be forged even under the same design and fabrication process. The PUF has enabled various applications such as identification, lightweight authentication, and secure key generation. According to the number of CPRs yielded, the PUF can be divided into two general categories: weak PUF and strong PUF. The former one has a limited number of CRPs and it is mainly used for provisioning cryptographic keys. The typical weak PUFs include memory based PUFs such as SRAM PUF [3], DRAM PUF [4] and Flash PUF [5] and time-delay based PUFs such as ring-oscillator PUF (ROPUF) [6]. In contrast, the strong PUFs yield a much larger CRP space, include not only typical optical PUFs and APUFs, and some other types such as bistable ring (BR) PUF [7]. Overall, in addition to secure key generation, strong PUFs have much wider applications than weak PUFs.

However, the strong PUF construction is non-trivial due to the threat from modeling attacks. That is, an attacker can build a soft-model of the PUF instance to accurately predict its responses given unseen challenges, where the soft-model is trained by a sufficient number of known challenge-response pairs (CRPs), e.g., eavesdropped or simply measured when a short period of physical access to the PUF instance occurs.

The associate editor coordinating the review of this manuscript and approving it for publication was Ismail Butun.

The modeling attacks become a security issue in constructing strong PUFs such as APUF and its variants, e.g., Feed-Forward (FF) Arbiter PUF (FF-APUF) [8], XOR Arbiter PUF [9] and lightweight secure APUF (LS-APUF) [10]. Among these variants, XOR-APUF has received the most attention as being lightweight and compact [1], [11]. Moreover, it has been shown that other strong PUFs are also susceptible to modeling attacks. Xu *et al.* show that a BR-PUF could be vulnerable to modeling attacks, and correspondingly using XOR-BR-PUF to enhance the modeling resilience [12]. Jeroen *et al.* [13] developed efficient impersonation attacks on the following five Arbiter PUF–based authentication protocols, including Poly PUF protocol [14], Obfuscated PUF (OB-PUF) protocol [15], Randomized PUF (RPUF) protocol [16], Light weight Highly Secure PUF (LHS-PUF) protocol [17], Finite-state Machine (FSM-PUF) protocol [18]. Yu *et al.* [19] optimized combined power and modeling attacks to crack lightweight strong PUFs: voltage regulator (VR) PUFs. Santikellur *et al.* [20] employded a novel machine learning-based modeling technique called efficient CANDECOMP/PARAFAC-tensor regression network (CP-TRN), a variant of CP-decomposition-based tensor regression network, to reduce the computational resource requirement of model building attacks on XOR APUF.

There are continuing efforts to design strong PUF alternatives. Wang *et al.* [21] proposed a framework to protect PUF against modeling attacks, called Noisy PUF (NoPUF), which exploited structural unpredictability to improve overall security. Williams *et al.* [22] introduced a serial and a parallel novel 64-bits memory-based controlled PUF (Mc-PUF) architecture for device authentication that has high uniqueness and resilient against modeling attacks. zhang *et al.* [23] proposed a configurable tristate (CT) PUF which can flexibly perform as an arbiter PUF, a ring oscillator (RO) PUF, or a bistable ring (BR) PUF with a bitwise XOR-based mechanism to obfuscate the relationship between the challenge and the response, hence resisting model attacks. Sahoo*et al.* [24] introduced a multiplexer-based composition of APUFs, denoted as MPUF, to against modeling attack. In addition to the basic MPUF design, they proposed two MPUF variants namely cMPUF and rMPUF to improve the robustness against cryptanalysis and reliability-based modeling attack, respectively. However, both the basic MPUF and its two variants have been broken, to a large degree, by logical approximation and global approximation modeling attacks proposed by Shi *et al.* [25], which leverage a machine learning technique, artificial neural network (ANN), to characterize the nonlinear structure of MPUF, rMPUF, cMPUF. Additionally, some of them even turns to exploit high density and unique characteristics in nanotechnology to construct secure strong PUFs [26], [27]. However, nanotechnology is immature. These PUF constructions are typically simulated and loosely claimed to be secure. Solid security examinations on them are usually lacked, which has been noticed by a very recent study [28].

A. PROBLEM STATEMENT

The security race continues between improving or designing a new PUF structure to enhance the PUF security by a defender and attacking it by the attacker. In [28], it rigorously examined the modeling attack resilience of several nanotechnology-enabled strong PUFs and demonstrated that their claimed modeling attacks resilience could not be readily held. A representative nCk-PUF, the mrSPUF [29], is also building upon the nanotechnology but has not been covered by [28]. Therefore, it is imperative to fully assess the security of new strong PUF constructions before adopting them in practice. In addition, the security caveats revealed from the thorough examinations can guide further secure designs, at least by avoiding similar flaws.

B. MOTIVATION AND CONTRIBUTION

Consider the security of PUF, this work examines the security of a strong PUF candidate, building upon the n-Choose-k-Sum (nCk) topology. The nCk-PUF is inclusive of the conventional ring oscillator (RO) PUF [9], where $k = 1$—note this special case only provides limited CRP and is regarded as a weak PUF. From a high level, two non-overlapped groups of elements in n elements are selected, each group has k elements. The values of k elements in each group are summed, and then the summations between two groups are compared to produce a 1-bit response. Such a strong PUF can be realized in various manners. A representative realization is taking advantage of the high density of the nano elements [29]. While the nCk-PUF can also be implemented on silicons, e.g., ASIC chips or FPGAs. For example, in the FPGA platform, one can use n ring oscillators (ROs) to stand for the n elements, while the value is the frequency of each RO. The nCk-PUF security is verified against modeling attacks, but solid experiments have not been conducted for affirmation.

As the nCk-PUF has been proposed to be a compact and promising topology to construct strong PUFs, potentially adopted in practice. Therefore, it is essential to rigorously examine its security, which is the focus of this work. Overall, the results and contributions of this paper are summarized as below:

1) We refutes the claim that the nCk-PUF is secure against modeling attacks through firstly building a mathematical model and solving it with simple machine learning, namely mathematical modeling (MM) attack.
2) We further propose to inject additional non-linearity through XORing multiple nCk-PUF instances, forming l-XOR-nCk-PUF, to increase the modeling resilience against MM attack.
3) We show the l-XOR-nCk-PUF is still breakable with purely powerful machine learning (ML) technique even with l up to 9, which though can well resistant to the MM attack. In this ML attack, we provide an empirical evaluation on the minimal number of CRPs required to model the l-XOR-nCk-PUF to a preset accuracy.

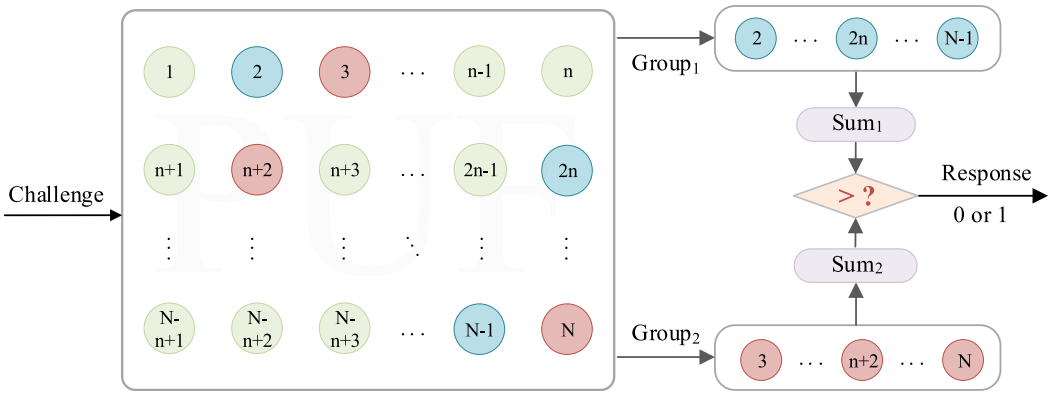


FIGURE 1. n-Choose-k-Sum PUF topological structure.

The contributions mentioned above are implemented in Python using simulated CRP, which facilitates to flexibly and thoroughly estimate the modeling attack resilience such as varying the design parameters, e.g., size, of the PUF. In contrast, though it is practical to collect CRPs using silicon data, such as ASIC or FPGA, this appears to be neither flexible nor time and labor efficient, especially when a large amount of CRPs are required to be collected and the strong PUF size varies. In addition, note that one nCk-PUF construction [29] leverages an emerging nano element (i.e., memristor), which is not readily accessible to common labs. Therefore, it is preferable to simulate the nCk-PUF structure to ease its security examination, especially considering the fact that the usage of simulated CRP is a commonly used strong PUF modeling resilience evaluation strategy [11], [13], [14], [16], [20], [30]. For the convenience of the readers and the research community, we have released the source code at https://github.com/sunshine-plh/nCk-PUF-attacking.

C. PAPER ORGANIZATION

In Section II, we describe the topology of the nCk-PUF and propose the nCk-PUF variant of XOR-nCk-PUF. In Section III, we present a mathematical model (MM) to facilitate the response prediction given an yet seen challenge, and resolve the model parameters by a simple machine learning. In Section IV, we examine nCk-PUF and XOR-nCk-PUF modeling resilience against a presented Multiple Layer Perceptron architecture (MLP) based modeling attacks. Section V compares attacking accuracy of MM with MLP attack methods, and compares attacking accuracy of MLP with varying architectural parameters. We further discuss the attacking scenario, e.g., how does an adversary obtain a large number of CRPs, and future work. We conclude this work in Section VI.

II. nCk-PUF AND ITS VARIANTS

In this part, we succinctly summarize the topology of the nCk-PUF, instead of diving into implementation details. Because it is recognized this topology is implementation independent. Secondly, we present a simple nCk-PUF variant, XOR-nCk-PUF, where the XOR logic operation is a commonly used technique to increase the complexity of modeling attacks [1].

A. n-Choose-k-Sum PUF (nCk-PUF)

Fig.1 gives the overall topology of n-Choose-k-Sum strong PUF (nCk-PUF). To be functional as a PUF, the nCk-PUF is comprised by challenge, element array, summation, and comparison blocks.

The challenge provides the address that selects two group of elements in the element array e.g., ring oscillator (RO) array implemented on the FPGA. The array is with N elements. Each group consists of k elements, where $(2 \times k) < N$. Notably, the elements are only used once for a given challenge. In other words, each element will not be concurrently selected by both groups. The values of all elements in one group is summed through the summation block. Then the summation of two groups are compared to produce a 1-bit response corresponding to the challenge. The total number of CRPs, NT_{CRP}, yielded by the nCk-PUF be expressed by Eq.(1).

$$NT_{CRP} = \frac{\binom{N}{k} \times \binom{N-k}{k}}{2} \tag{1}$$

By using a relative large N e.g., 1000 and k e.g., 10, the CRP space of the nCk-PUF can be significantly large, preventing exhaustively characterization of all CRPs in a long period, e.g., hundreds of days. As one typical n-Choose-k-Sum PUF (nCk-PUF), resilience to modeling attacks of the mrSPUF architecture proposed in [29] was analytically evaluated, where it was regard to be secure under modeling attacks [29]. However, there is a lack of actual modeling attacks examinations applied on the nCk-PUF.

This work is to examine nCk-PUF's security with two types of modeling attacks: attacks building upon a mathematical model and attacks with pure machine learning without mathematical model assistance.

B. XOR-nCk-PUF

In order to harden the modeling attack on a single nCk-PUF, we can straight forwardly inject non-linearity

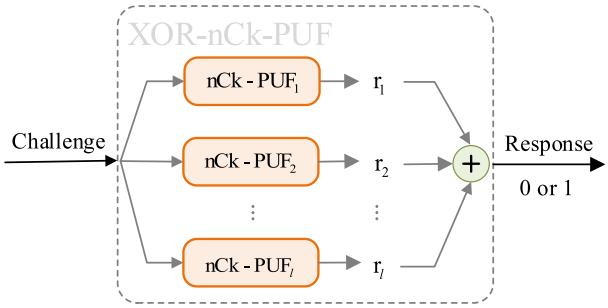


FIGURE 2. XOR-nCk-PUF structure.

via XOR-operation, here we present the nCk-PUF variant, XOR-nCk-PUF, which comprises of multiple independent nCk-PUFs, as shown in Fig. 2. Generally, multiple nCk-PUFs share the same challenge, and the final response bit (0/1) is generated by XORing the responses of each individual nCk-PUF forming the l-XOR-nCk-PUF. The XOR operation is a common technique used to increase PUF modeling attack resilience. for example, the well-known XOR-APUF [1].

III. MATHEMATICAL MODEL ATTACK

We firstly attempt to build a mathematical model to facilitate the response prediction given an yet seen challenge. Then simple machine learning is utilized to resolve the model parameters. This process is similar with other PUF attack studies [31], [32].

A. MATHEMATICAL MODEL

In this section, the estimation of the relative value of each element of nCk-PUF and the construction/learning of a multi-bit XOR classifier are elaborated. Before doing that, we will formalize the element array representation and the challenge vector build the element array representation, followed by response formulation.

1) THE REPRESENTATION OF ELEMENT ARRAY

The element array shown in Fig. 1 can be expressed by Eq. (2).

$$\mathbf{s}=\begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 & \dots & \alpha_{n-1} & \alpha_n \\ \alpha_{n+1} & \alpha_{n+2} & \alpha_{n+3} & \dots & \alpha_{2n-1} & \alpha_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{N-n+1} & \alpha_{N-n+2} & \alpha_{N-n+3} & \dots & \alpha_{N-1} & \alpha_N \end{bmatrix} \tag{2}$$

where, $\mathbf{s}$ is the array, and α_n is the n_{th} $(n = 1, 2, \dots, N)$ element of $\mathbf{s}$. As it is introduced above, the elements responsible for a given CRP of nCk-PUF is fixed, so if the element value α_n can be accurately estimated, it is feasible to accurately predict the responses for any challenge of the nCk-PUF.

Though an attacker has no knowledge of the exact value of α_n, the attacker can reasonably assume that it obeys a distribution, in particular, the common Gaussian distribution. Then, in practice, the problem of attacking nCk-PUF can be transformed into the problem of predicting the approximation value of α_n—not true value—of the nCk-PUF under such a distribution. In this sense, we can set the element array $\mathbf{s}$ as a parameter set that will be estimated through a mathematical model.

2) REPRESENTATION OF THE CHALLENGE AND RESPONSE

The nCk-PUF model challenge is the input vector, which can be represented by three components: the first group selected element (the elements in blue in Fig. 1), the second group selected element (the elements in red in Fig. 1), and the unselected elements (the elements in green in Fig. 1). The number of elements in the first group is k same as that of the second group. Then, the model input array can be represented by an array $\mathbf{I}$ (corresponding to the nCk-PUF shown in Fig. 1), as shown in Eq. (3).

$$\mathbf{I}=\begin{bmatrix} 0 & 1 & -1 & \dots & 0 & 0 \\ 0 & -1 & 0 & \dots & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & -1 \end{bmatrix} \tag{3}$$

where, $'0'$ represents the unselected element, $'1'$ and $'-1'$ represent the first group and second group of selected element, respectively. Generally, the challenge of PUF is a vector, so we can change $\mathbf{s}$ and $\mathbf{I}$ into a vector by Eq. (4) and Eq. (5), as:

$$\begin{aligned} \mathbf{s}' &= \begin{bmatrix} \mathbf{s}_{col.1}^T & \mathbf{s}_{col.2}^T & \dots & \mathbf{s}_{col.n-1}^T & \mathbf{s}_{col.n}^T \end{bmatrix} \\ &= \begin{bmatrix} \alpha_1 & \dots & \alpha_{n-1} & \alpha_n & \dots & \alpha_N \end{bmatrix}_{1\times N} \end{aligned} \tag{4}$$

$$\begin{aligned} \mathbf{c} &= \begin{bmatrix} \mathbf{I}_{col.1}^T & \mathbf{I}_{col.2}^T & \dots & \mathbf{I}_{col.n-1}^T & \mathbf{I}_{col.n}^T \end{bmatrix} \\ &= \begin{bmatrix} 0 & \dots & 1 & 0 & \dots & -1 \end{bmatrix}_{1\times N} \end{aligned} \tag{5}$$

where, $\mathbf{s}'$ is the element original value vector, $\mathbf{s}_{col.n}$ is the n_{th} column of $\mathbf{s}$. $\mathbf{c}$ is the challenge vector or the model input, $\mathbf{I}_{col.n}$ is the n_{th} column of $\mathbf{I}$. T means transpose. The number of $'1'$ and $'-1'$ in vector $\mathbf{c}$ remains to be k, where k is a positive integer and $N > 2 \times k$.

Then, the response is generated by comparing the summations of two groups by Eq. (6).

$$\begin{aligned} \mathbf{r} &= \delta\left(\mathbf{c}(\mathbf{s}')^T\right) \\ &= \delta\left(\alpha_{group_1} - \alpha_{group_2}\right) \\ &= \begin{cases} 1, & \text{if } \alpha_{group_1} > \alpha_{group_2} \\ 0, & \text{otherwise .} \end{cases} \end{aligned} \tag{6}$$

where $\mathbf{c}$ and $\mathbf{r}$ forms a challenge response pair (CRP). $\alpha_{group_1} = \sum_{i=1}^{k} \alpha_i, \alpha_i \in \mathbf{s_{group_1}} = C_N^k \in \mathbf{s}$, $\alpha_{group_2} = \sum_{j=1}^{k} \alpha_j, \alpha_j \in \mathbf{s_{group_2}} = C_{N-k}^k \in \mathbf{s} \setminus \mathbf{s_{group_1}}$. Notably, $\setminus$ ensures that the elements in group_1 are non-overlapped with that in group_2. The $\delta(\cdot)$ is a function, which is used to clip the value greater than 0 to be 1, and the value less than 0 to be 0. In practice, it usually uses the *sigmoid* function instead.

B. MACHINE LEARNING FOR PARAMETER ESTIMATION

As mentioned early, it is not a must to accurately estimate the exact value of each element. Once the approximation value is assessed by priorly assuming that these values follow the normal distribution, it is sufficient to model it to predict response given unseen challenges.

Though the CRP generation process contains non-linear operations, in particular the comparison operations, we can approximate it by establishing a single-layer fully connected network, as the value estimation layer shown in Fig. 3. We hope to use $\mathbf{W}$ and $\mathbf{C}$ to represent $\mathbf{R}$, as shown in Eq. (7).

$$\mathbf{R} \approx \delta\left(k_0 \mathbf{C}\mathbf{W}^T\right) \tag{7}$$

where, $\mathbf{C} \in \mathbb{R}^{M\times N}$ and $\mathbf{R} \in \mathbb{R}^{M\times 1}$ represent challenge array and its corresponding response array, respectively, and M is the number of CRPs taking part in training. $\mathbf{W} \in \mathbb{R}^{H_1\times N}$ (H_1 is the number of neurons in the second layer as shown in Fig. 3) is the weight matrix of the value estimation layer, k_0 is a large positive number. $\delta(\cdot)$ is implemented by the *sigmoid* function, as Eq. (8).

$$\delta(x) = \frac{e^x}{1+e^x} \tag{8}$$

Then, we can use multiple CPRs to train the single-layer fully connected network, so as to obtain an estimate of the weight matrix $\mathbf{W}$. More precisely, in case of without XOR operation, the more the estimation value of the element in $\mathbf{W}$ matches to the element value of the actual device, the smaller loss of the training model. Therefore, the relative value of the element is transformed into training of the model weight. In other words, each w represents an element in the nCk-PUF.

In actual experiment, there are two points needed to be explained. First of all, in order to make the training converge quickly and improve the matching degree between the weight matrix and the element matrix of the nCk-PUF, the initial value of the weight matrix $\mathbf{W}$ is set to obey normal distribution. For second, although the trained model can successfully approximate the relationship between challenge and response, while there has no XOR operation, however, vanishing gradient problem may be encountered during training, that is because gradient of the *sigmoid* function in its saturated part is approximately 0. Therefore, the Batch Normalization layer is used to replace the constant k_0 in the actual network.

***l*-XOR classifier**: XOR problem has been solved with the construction of multi-layer neural network and the proposal of back propagation [33], and a three-layer fully connected neural network can successfully represent the non-linear relationship of XOR.

For the l-XOR-nCkPUF resolution, we take $l = 2$ ($\mathbf{r}_1$ and $\mathbf{r}_2$)/$l = 3$ ($\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$) inputs with single output $\mathbf{R}_{12}$/ $\mathbf{R}_{123}$, the XOR operation can be summarized by Table. 1. We hope to find optimal network and weight values of the network, so that its output $\mathbf{R}'_{12}$/ $\mathbf{R}'_{123}$ follows the desired output $\mathbf{R}_{12}$/ $\mathbf{R}_{123}$.

TABLE 1. Truth table of XOR function with two/three inputs.

$\mathbf{r}_1$	$\mathbf{r}_2$	$\mathbf{r}_3$	$\mathbf{R}_{12} = \mathbf{r}_1 \odot \mathbf{r}_2$	$\mathbf{R}_{123} = \mathbf{r}_1 \odot \mathbf{r}_2 \odot \mathbf{r}_3$
0	0	0	**0**	**0**
0	1	0	**1**	**1**
1	0	0	**1**	**1**
1	1	0	**0**	**0**
0	0	1	**0**	**1**
0	1	1	**1**	**0**
1	0	1	**1**	**0**
1	1	1	**0**	**1**

We can leverage a fully connected neural network with a hidden layer to represent this nonlinear relationship between inputs and output. Here, we set the number of neurons in the hidden layer of the neural network equal to the l, number of independent nCk-PUF being XORed. As shown in Fig. 3, the XOR classification part has one layer with two/three neurons.

Although one hidden layer can represent the nonlinear relationship between inputs and output, with the l in the XOR increasing, the non-linearity degree of the l-XOR-nCk-PUF increases, it still be can be modeled by other attacking method with high prediction accuracy with shorter training time, a multiple layer perceptron based machine learing attack method will be introduced for l-XOR-nCk-PUF modeling in section IV.

C. SUMMARY

In essence, we have divided the l-XOR-nCk-PUF modeling into two sub-problems: the estimation of the relative value of each element of l-XOR-nCk-PUF and the construction/learning of a l-XOR classifier. In other words, firstly, we estimate the relative value of each element of a single nCk-PUF, so as to obtain its the response to a specific challenge. After that, we establish an XOR classifier to classify the XOR operation of each independent nCk-PUF's response to a specific challenge. Then, the l-XOR-nCk-PUF final response will be obtained. We will adopt a fully connected neural network to implement this mathematical model, which can be divided into three blocks, the estimation of the relative value of the each element of nCk-PUF, the construction/learning of a multi-bit XOR classifier, and the output unit, and those three blocks correspond to the input layer, invisible layer and output layer of the neural network, respectively.

D. ATTACK RESULTS

In this section, the attack results of the model mentioned above will be presented for stand nCk-PUF on simulated, noise-free data, which is a common practice when investigating PUF's modeling resilience [11], [30].

1) CRPs GENERATION

In our experiment, we collected CRPs from Python simulated nCk-PUF models. We assume that all element values of nCk-PUF are positive, independent and identically distributed, and follow Normal distribution with mean $\mu = 100$ and $\sigma = 1$ [29]. We set $\mu = 100$ instead of $\mu = 0$ to

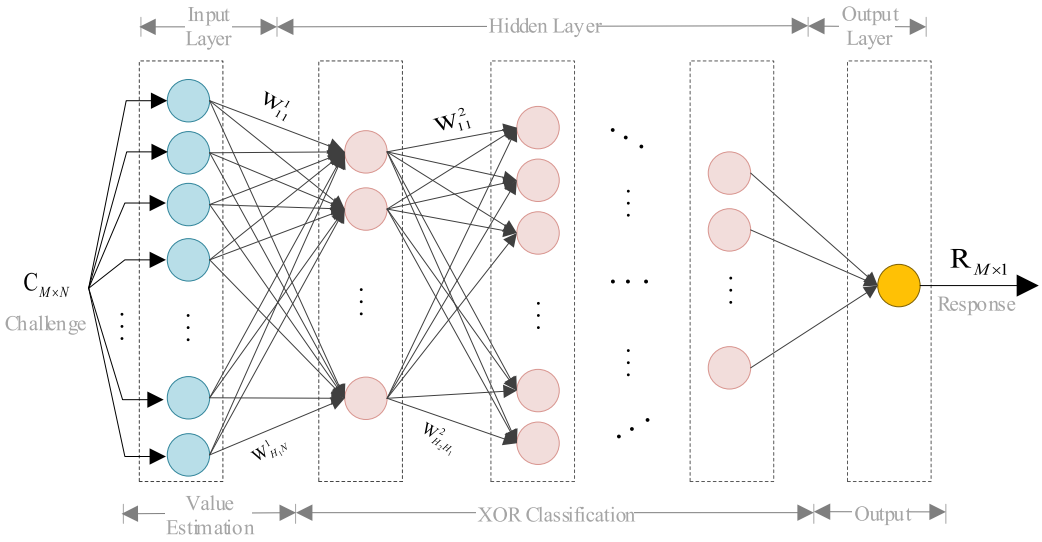


FIGURE 3. Architecture of neural network.

insure all element values of nCk-PUF are positive, which has no effect on the results.

First, we generate all CRPs for each nCk-PUF More specifically, k elements are randomly selected fromthe N-element array, forming the first group. Then k elements are randomly selected from the remaining $(N-k)$ elements to form the second group, then calculate the response by comparing the sum of each group. In the other way around, the key to generating all CRPs is how to generate all positions (namely Address in a physical implementation of the nCk-PUF) of selected element. The address determination method is detailed in Algorithm 1. Although the selected order is different from that of [29], they are equivalent.

2) MODEL SETUP

Taking the total number of CRPs of nCk-PUF in to consideration, its uniformity is nearly 50%, which has also been validated in [29] using one typical nCk-PUF that is mrSPUF. Therefore, it is feasible for us to obtain uniform training data. In CRPs generation processing, we have stored the samples with response '1' and '0' separately, so we only need to take 50% samples from each group, respectively, to ensure the uniformity of training data is 50%. In each attack, we use 80% CRPs for training, 10% for validation and 10% for testing.

Our model was executed on Linux system with CUDNN v7, CUDA v9.1.85, NVIDIA Tesla V100-PCIE-32GB GPU and Intel(R) Xeon(R) CPU E5-2687W v3 @ 3.10GHz. Table. 2 provides the characteristics of the neural networks used in MM attacks.

3) ATTACKING ACCURACY

In order to quantify the effectiveness of the MM attack, we measure the testing CRPs' prediction accuracy. For each set of nCk-PUF parameters, including 36(6), 64(8), 100(10)

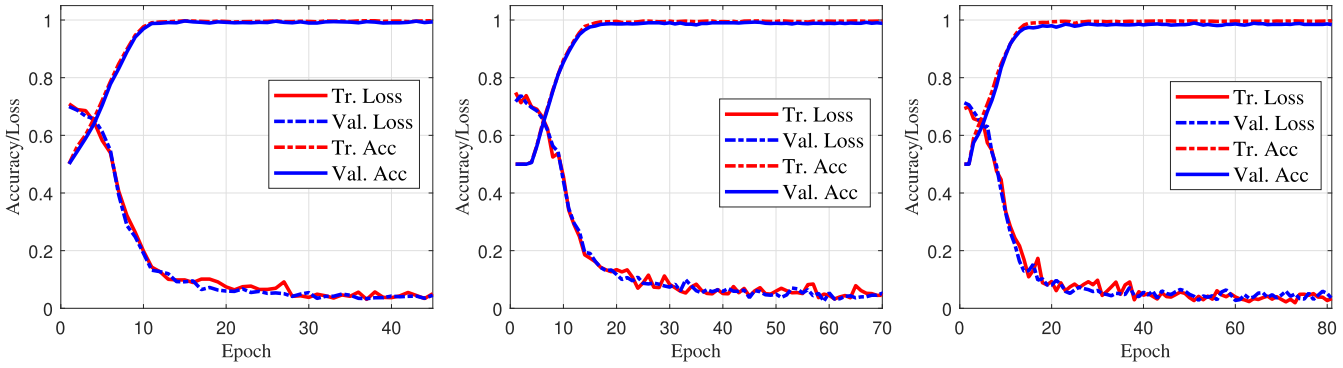
TABLE 2. The characteristics setting for mathematical model attack.

Networks Characteristics	Used in simulation
Weight initializer	Normal distribution ($\mu = 100$ and $\sigma = 1$)
Bias initializer	Zeros
Learning rate	0.01
Optimizer	Adam
Loss function	BCELoss
Output layer activation function	Sigmoid

(n(k) means n-Choose-k) with different XOR operation and different simulation CRPs, five runs are performed and the average is reported.

Table. 3 details the MM attacking results. The column "No. of XOR (l)" and "Tr. CRPs size" in the Table. 3 are the number of nCk-PUF been XORed and the number CRPs used in training, respectively. The column "Test Acc" and "Tr. Time" are the prediction accuracy and training time for the nCk-PUF under the simulation conditions being given in columns 1 to 3. The total number of CRPs yielded by 36(6), 64(8), 100(10) are 1.9477×10^7, 1.5492×10^{11} and 2.1811×10^{15}, respectively, calculated by Eq. (1).

For nCk-PUF without injecting XOR non-linearity, the MM attack method can model nCk-PUF with high accuracy, the prediction accuracies are all over 98% for 36(6)-PUF, 64(8)-PUF and 100(10)-PUF while only 0.5×10^4 CPRs are used in training, and the average training time is less than 15 seconds. When 2×10^4 CRPs take part in training, the attacking accuracy will achieve 99.5% regardless it is 36(6)-PUF, 64(8)-PUF or 100(10)-PUF, and the training time is no more than 50 seconds. For the $l = 2$-XOR-nCk-PUF, the attacking accuracy varies for 36(6)-PUF, 64(8)-PUF and 100(10)-PUF, more precisely, as the complexity or size of the nCk-PUF increases, the prediction accuracy will decrease while the training CRPs size is the same. In order to achieve



(a) 36-Choose-6 PUF. (b) 64-Choose-8 PUF. (c) 100-Choose-10 PUF.

FIGURE 4. Training and validation losses and accuracies over epochs of mathematical model attacking for different nCk-PUF.

Algorithm 1 CRPs Address Generation

$NT_{CRP} = \frac{\binom{N}{2k} \times \binom{2k}{k}}{2}$ ▷ *the total CRPs number of nCk-PUF*
$N_1 = \binom{N}{2k}$ ▷ *the set total number of randomly selected* $2k$ *elements from* N
$N_2 = \binom{2k}{k}$ ▷ *the set total number of randomly selected* k *elements from* $2k$ *selected elements*
$Addr = zeros[N, 2k]$, $C_1 = zeros[N_1, 2k]$, $C_2 = zeros[N_2, k]$, $Count = 0$, $AddrIndex$ ▷ *Initialization, 'AddrIndex' is a vector, whose elements are natural numbers 0 to* $(NT_{CRP} - 1)$ *randomly arranged in random order*
procedure **AddrGene** (NT_{CRP}, N_1, N_2,$Addr$,C_1, C_2,$Count$, $AddrIndex$)
  for i *in range* N_1 **do**
    $C_2 = list(itertools.combinations(C_1[i, :], k))$
    for j *in range* $int(N_2/2)$ **do**
      $Index = AddrIndex[Count]$
      $Addr[Index, :] = list(C_2[j, :]) + list(C_2[-(j+1), :])$
      $Count = Count + 1$
    end for
  end for
return $Addr$
end procedure

98.5% prediction accuracy, it requires 5×10^5 CPRs for 36(6)-PUF, and 1×10^5 CPRs for 64(8)-PUF and 100(10)-PUF, and their average training time are 102.90 seconds, 206.48 seconds and 212.22 seconds, respectively. This is about 10 to 20 times of nCk-PUF without being XOR. For the l = 3-XOR-nCk-PUF, to obtain the desired high prediction accuracy, the number of CRPs required will greatly increase and the training time also will increase dramatically. 5×10^5 training CPRs provided 91.12% prediction accuracy after 912.54 seconds training and 2×10^6 training CRPs given 91.88% and 90.43% prediction accuracy for 64(8)-PUF and 100(10)-PUF, respectively, and it takes more than one hour for training.

TABLE 3. Prediction accuracy and training time of MM attacks.

nCk-PUF type	No. of XOR (l)	Tr. CRPs Size (10^4)	Test Acc. (%)	Tr. time (sec)
36(6)	1	0.5	99.45	9.78
		1	99.48	17.04
		2	99.53	33.43
	2	1	82.38	21.97
		2	97.46	44.79
		5	98.80	102.90
	3	10	79.38	169.79
		20	84.47	310.24
		50	91.12	912.54
64(8)	1	0.5	98.80	14.80
		1	99.36	23.80
		2	99.72	45.70
	2	2	83.10	44.70
		5	89.96	113.53
		10	98.68	206.48
	3	20	79.03	407.05
		100	84.35	2105.60
		200	91.88	4530.77
100(10)	1	0.5	98.64	13.71
		1	99.29	24.06
		2	99.50	46.66
	2	2	71.81	46.77
		5	87.58	115.21
		10	98.57	212.22
	3	100	83.45	2164.12
		150	89.75	3907.19
		200	90.43	4922.97

In order to illustrates the convergence of the mathematical model attack, we present the loss function values and accuracies of training and test in the training process in Fig. 4. The losses and accuracies are plotted over the epochs for 36(6)-PUF, 64(8)-PUF and 100(10)-PUF without being XOR operated, respectively, using 0.5×10^4 training CRPs. The MM attack on nCk-PUF will converge while the epoch reaches 30, and the loss function value reaches a stable level at around 0.08 and thus the attack accuracy achieves 99%.

4) ATTACK SUMMARY

From the attack results shown in Table. 3 and its corresponding analysis, we conclude that the MM attack can model the

nCk-PUF with high accuracy, especially for nCk-PUF without XOR. The prediction accuracy given a single nCk-PUF can achieve 99% even while a small number training CRPs is available, and it is insensitive to nCk-PUF's complexity. This is validated by prediction accuracy of 36(6)-PUF, 64(8)-PUF and 100(10)-PUF, which are similar with each other simulated under the same conditions.

As for l-XOR-nCk-PUF with $l = 2, 3$, the MM attack also demonstrates high prediction accuracy above 90%, however, it requires much more CRPs, and nCk-PUF's complexity become a factor affecting prediction accuracy. For 2-XOR-nCk-PUF, 64(8)-PUF and 100(10)-PUF need more than 5×10^4 training CRPs to obtain 98% attack accuracy compared with 36(6)-PUF. For 3-XOR-nCk-PUF, this phenomenon will become even more significant, with the model parameters given in Table. 2, 90% prediction accuracy requires 2×10^6 CPRs for 100(10)-PUF. At the same time, the training time also increase greatly with l-XOR-nCk-PUF as l increases. The reason is that XOR is non-linear operation, making the MM attack being more difficult with a larger l.

From the above analysis, we conclude that the MM attack is the first choice to attack nCk-PUF without or with only a small l in the XOR operation. Although its structure is simple, it can achieve high prediction accuracy with a small number of training CRPs. For l-XOR-nCk-PUF with a larger l, the MM attack becomes less effective, we will investigate its modeling resilience with more powerful pure machine learning method.

IV. MODELING ATTACK WITH MACHINE LEARNING

The XOR PUFs (XPUFs) are intended to against CRP based machine learning (ML) attacks, it is found if XOR gate is large enough, the number of training CRPs and the computational time required for modeling XPUF will increase rapidly with respect to the number of PUFs being XORed. However, many studied have reported XPUFs can be successfully attacked. For example, a logisitic regression (LR) ML attack method was published [34], its prediction accuracy can reach 99% for 9 Arbiter PUFs XORed; a neural network-based attack was reported [35], which can attack XPUFs with fewer CRPs and shorter learning time when compared with ML attack methods. In this section, Multiple Layer Perceptron (MLP) based ML method will be leveraged for l-XOR-nCk-PUF attacking, because MLP is more effective than using the LR in terms of the training time [35]–[38].

A. MLP ATTACK SETUP

Python 3.8 is used to implement our MLP based ML attack method, and Keras with Tensorflow v2.4.1 back-end (tf.keras) is selected as the ML library for the training model. The learning rate equals to 0.01, and the initializer adopts uniform distribution, the hidden layer and output layer activation function use ReLU and Sigmoid function, respectively. The number of layers and the neuron number of each layer will be listed in each experiment. Other model parameters and simulation computer parameters keep the same as that of illustrated in section III-D.

B. RESULTS

To measure the effectiveness of MLP attacking on l-XOR-nCk-PUF, we select three different targeted attacking accuracy of 97%, 98% and 99%, respectively, for 36(6), 64(8) and 100(10) PUF while l ranges from 1 to 6(7-XOR, 8-XOR and 9-XOR nCk-PUFs attack results will be elaborated in section V). In this experiment, the MLP consists of three hidden layers and the neurons number of each hidden layer were (64, 32, 16), respectively, the simulation results as shown in Table. 4., and all figures were obtained by averaging over 5 different train sets.

From Table. 4, we can see that about 2.5×10^4 CRPs are needed to get 99% prediction accuracy for 1-XOR-36(6)-PUF, and the training time it takes is about 17 seconds, however, the required CRPs number will be increased gradually for 1-XOR 64(8) and 100(10) PUF to achieve 99% prediction accuracy—7.1×10^4 and 15.1×10^4, and their corresponding training time are 40 seconds and 68 seconds, respectively. In the other word, for 36(6), 64(8) and 100(10) PUF, the number of CRPs required as well as the training time needed will increase accordingly to achieve similar prediction accuracy while l takes the same value, that is because the elements contained in these PUFs are increasing accordingly. This tendency is similar when the l ranges from 2 to 6.

For l-XOR-36(6)-PUF, we can see about 13.2×10^4, 22.4×10^4, 32.7×10^4, 54.5×10^4 and 131.4×10^4 CRPs are used to achieve 99% prediction accuracy while l changes from 2 to 6, and their corresponding training time are about 146, 262, 734, 805 and 1509 seconds, respectively. That is to say, when elements number of nCk-PUF N and the selected elements $2 * k$ remain the same, the number of CRPs required as well as the training timed will increase with l—the number of independent nCk-PUF being XORed, so as to achieve similar prediction accuracy, that is because the non-linearity degree of l-XOR-nCk-PUF dependents on l increases.This conclusion also can be verified by the attacking results of l-XOR-64(8)-PUF and l-XOR-100(10)-PUF with l changing from 1 to 6 shown in Table. 4.

Overall, Table. 4 shows MLP attack can achieve 99% prediction accuracy attacking on l-XOR-36(6)-PUF, l-XOR-64(8)-PUF and l-XOR-100(10)-PUF with l changing from 1 to 6. What's more, even for 6-XOR-100(10)-PUF, it only needs 369.3×10^4 CRPs to obtain 99% prediction accuracy, which is very small compared with the total CRP space that is 2.18×10^{15} for 6-XOR-100(10)-PUF according to Eq. 1. The ratios between the number of training CRPs required to achieve 99% attacking accuracy and its corresponding nCk-PUF's total CRPs number are summarized in Table. 5. In all cases, a small ratio of CRPs are sufficient for attacking the given l-XOR-nCk-PUF successfully. Notably, this ratio decreases magnitude for larger size l-XOR-nCk-PUFs, which means intuitively increase the PUF

TABLE 4. Prediction accuracy and training timing of MLP based machine learning attack.

	1-XOR-nCk-PUF			2-XOR-nCk-PUF			3-XOR-nCk-PUF		
	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)
	0.0782	97.02	2.15	1.8085	97.99	16.23	3.0551	97.05	29.07
	0.3339	98.07	3.87	1.9810	98.03	17.04	6.1785	97.98	63.20
36(6)	2.5267	99.03	17.03	13.2550	99.05	145.68	22.4225	98.99	261.93
	4-XOR-nCk-PUF			5-XOR-nCk-PUF			6-XOR-nCk-PUF		
	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)
	3.7482	96.97	99.10	5.4975	96.99	171.04	8.6687	96.98	307.61
	7.4580	98.07	145.69	13.1718	98.03	303.14	24.4661	98.05	564.48
	32.7250	99.02	733.84	54.4680	99.01	805.33	131.3930	99.04	1509.28
	1-XOR-nCk-PUF			2-XOR-nCk-PUF			3-XOR-nCk-PUF		
	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)
	0.2293	97.00	3.48	2.1673	97.02	15.99	5.7911	97.01	55.36
	0.8553	98.03	8.80	5.3172	98.01	45.30	11.7954	97.97	120.06
64(8)	7.1417	98.99	39.24	23.2850	98.98	273.22	46.6847	99.04	536.48
	4-XOR-nCk-PUF			5-XOR-nCk-PUF			6-XOR-nCk-PUF		
	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)
	7.6120	96.99	139.01	9.8572	97.01	241.48	15.9603	97.03	418.27
	17.2837	98.00	307.93	22.2924	97.98	406.84	41.1800	97.97	714.05
	59.9960	98.96	885.90	120.0500	99.08	1180.39	272.9699	99.05	2076.01
	1-XOR-nCk-PUF			2-XOR-nCk-PUF			3-XOR-nCk-PUF		
	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)
	0.4114	97.08	4.13	4.0644	97.07	24.96	10.6709	97.05	105.41
	1.5449	98.01	8.32	10.2916	98.03	79.33	20.9271	97.98	217.07
100(10)	15.1183	99.02	68.08	43.7349	98.99	411.51	71.7932	98.99	814.36
	4-XOR-nCk-PUF			5-XOR-nCk-PUF			6-XOR-nCk-PUF		
	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)	Tr. CRPS($\times 10^4$)	Test Acc.(%)	Tr. time(sec)
	13.1094	97.02	221.58	16.7537	96.97	340.40	32.2380	96.99	708.32
	27.2143	97.98	413.29	40.6955	98.03	600.57	84.9895	98.05	1149.25
	94.7120	98.96	1198.05	158.0641	98.98	1430.70	369.2940	98.99	2393.00

size is unnecessarily linearly increasing the modeling attack resilience.

In order to illustrates its convergence, we present the loss function values and accuracies of training and validation in the training process in Fig. 5 for 1-XOR-36(6)-PUF, 1-XOR-64(8)-PUF and 1-XOR-100(10)-PUF, the corresponding training CRP number are 2500, 4000 and 6000, respectively. We can see the convergence speed of the training process is very fast, it will reach a stable phase after 20 epochs, to be exactly, both of the training and validation accuracy will reach 99%, and both the training and validation loss function value will be lower than 0.04.

1) SCALABILITY

For n-stage APUF attacking, Rührmair *et al*. [30] theoretical considerations (dimension of the feature space, Vapnik-Chervonenkis dimension) suggested the required number of CRPs N_{CRPs} that is necessary to model n-stage APUF with a misclassification rate of ϵ should obey the relationship shown in Eq. (9).

$$N_{CRPs} = O\left(n \setminus \epsilon\right) \tag{9}$$

Rührmair *et al.* [30] validate this it by extensive experimental results. In practical PUF applications, it is imperative to know the concrete number of CRPs that may become known before the PUF-security breaks down. For example, the lockdown-PUF discards the PUF instance for further authenticaiton usage once this number is reached [39]. Similarly, we also run scalability estimations on l-XOR-nCk-PUF with l ranging from 1 to 6, as shown in Fig.6, it affirms that the minimal number of CRPs that is necessary to model l-XOR-nCk-PUF with a misclassification rate would obey the relationship shown in Eq. (10).

$$N_{XOR-CRPs} = O\left((n+1)^l \setminus \epsilon\right) \tag{10}$$

where n is the original elements number of nCk-PUF and ϵ is the prediction error. Assuming an approximate linear functional dependency $y = ax + c$ in the double logarithmic plot of Fig.6 with a slope of a slight dynamic changing with l. Meanwhile, we can estimate the relation between the corresponding training time of the experiments shown in Fig.6 and its training CRPs, the experimental results are shown in Fig.7, it shows that the training time is proportional to $\log(N_{XOR-CRPs} \setminus ((n+1) \times l))$.

V. DISCUSSION

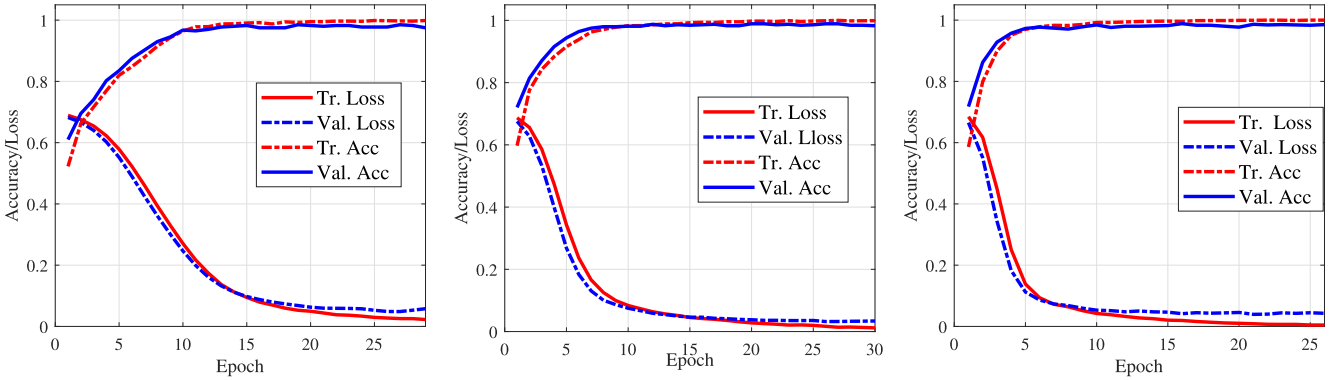
We firstly compare the MM attack with the MLP attack. Then, we discuss the hidden layer parameter of MLP attacks by comparative experiments. Based on the comparison we discuss the attack scenario and followed with a discussion on future work.

A. MM ATTACK AND MLP ATTACK COMPARISON

First of all, the attacking accuracy of MM is better than that of MLP for nCk-PUF without XOR opeartion (1-XOR-nCk-PUF). From Table. 3, we see that MM can achieve 99.45% prediction accuracy using 0.5×10^4 training CPRs for 1-XOR-36(6)-PUF, and the training time is short—about 10 seconds. Compared with that of MLP attack shown in Table. 4, prediction accuracy and training time are 99.03% and 17.03 seconds, respectively, using 2.5×10^4 training CPRs. For 1-XOR-64(8)-PUF/1-XOR-100(10)-PUF, MM attack using 1×10^4/1×10^4 training CPRs gets

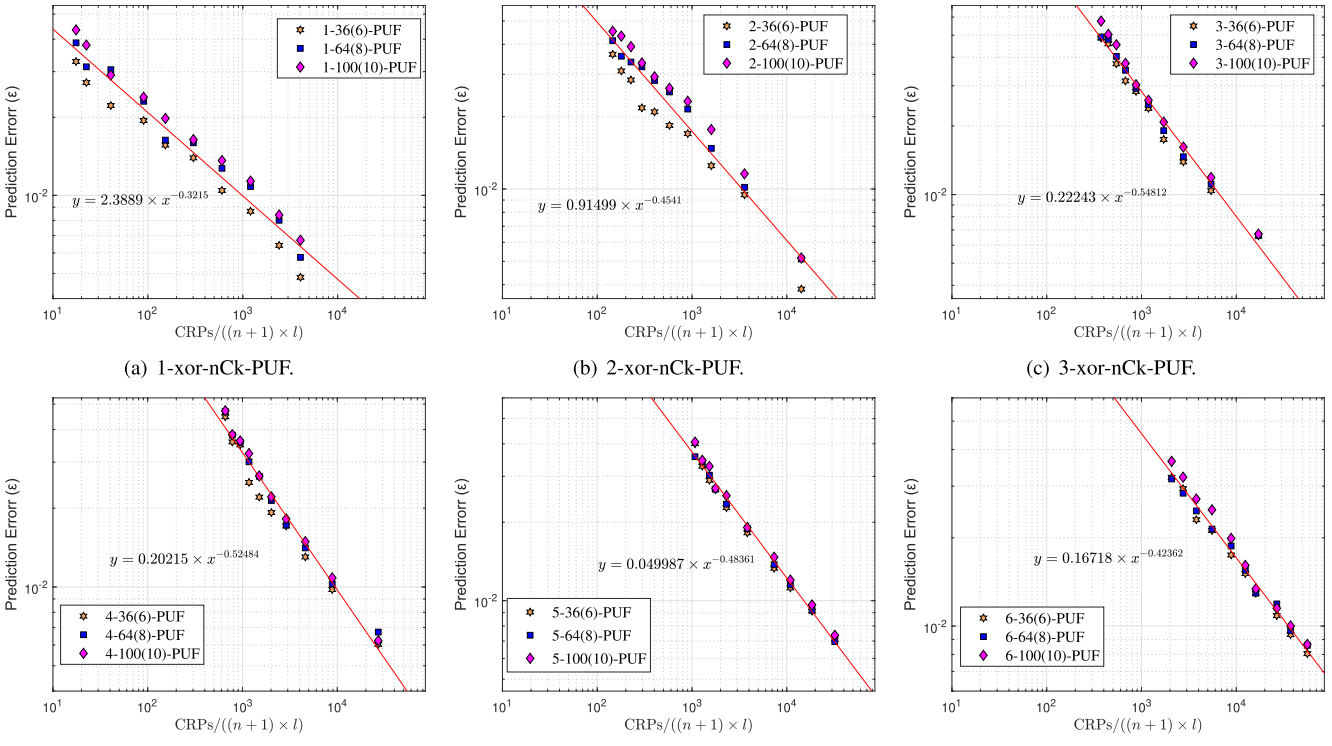
TABLE 5. The ratio between the number of training CRPs required to achieve 99% prediction accuracy and nCk-PUF's total CRPs number.

	1-XOR-nCk-PUF	2-XOR-nCk-PUF	3-XOR-nCk-PUF	4-XOR-nCk-PUF	5-XOR-nCk-PUF	6-XOR-nCk-PUF
36(6)	1.30×10^{-3}	6.81×10^{-3}	1.15×10^{-2}	1.68×10^{-2}	2.80×10^{-2}	6.75×10^{-2}
64(8)	4.61×10^{-7}	1.50×10^{-6}	3.01×10^{-6}	3.87×10^{-6}	7.75×10^{-6}	1.76×10^{-5}
100(10)	6.93×10^{-11}	2.01×10^{-10}	3.29×10^{-10}	4.34×10^{-10}	7.25×10^{-10}	1.69×10^{-9}



(a) 36-Choose-6 PUF. (b) 64-Choose-8 PUF. (c) 100-Choose-10 PUF.

FIGURE 5. Training and validation losses and accuracies over epochs of MLP based ML attacking for different nCk-PUF.

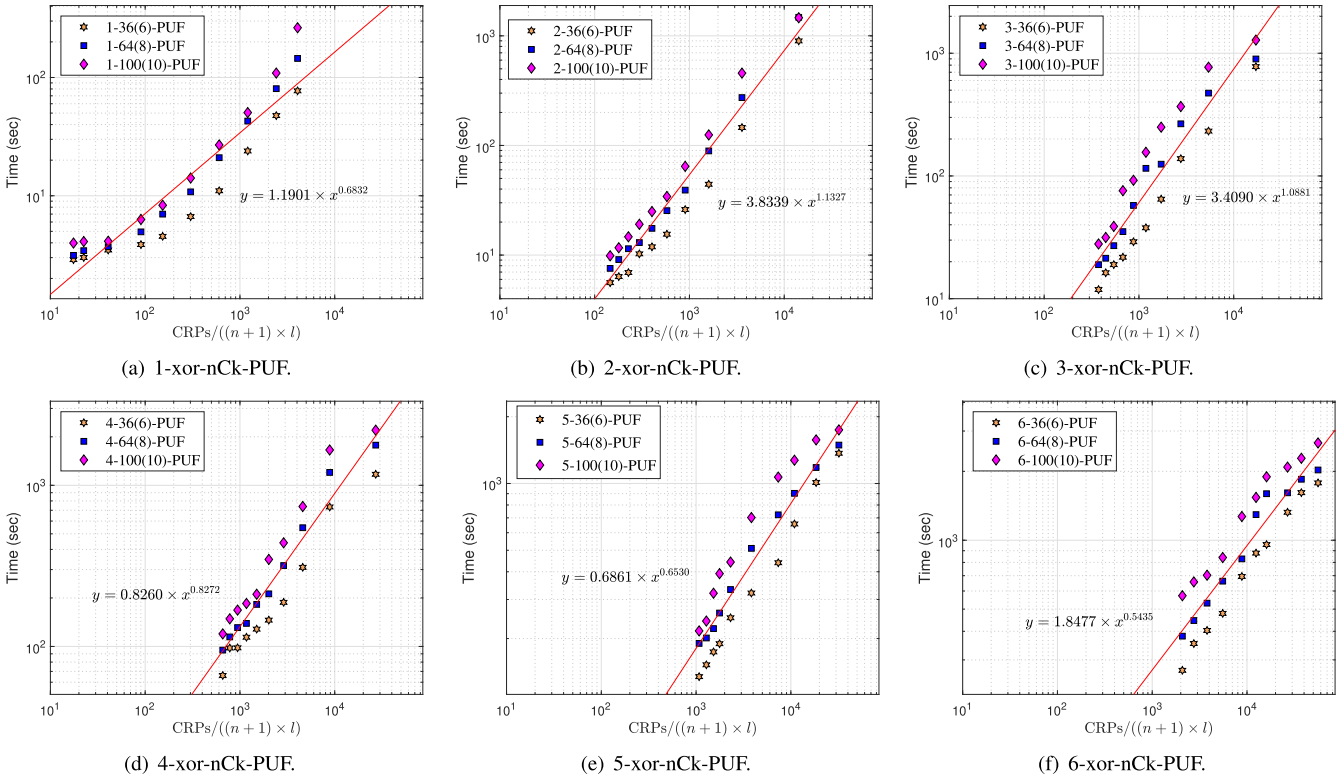


(a) 1-xor-nCk-PUF. (b) 2-xor-nCk-PUF. (c) 3-xor-nCk-PUF.

(d) 4-xor-nCk-PUF. (e) 5-xor-nCk-PUF. (f) 6-xor-nCk-PUF.

FIGURE 6. Double logarithmic plot of misclassification rate on the ratio of training CRPs for 1 to 6-XOR-nCk-PUF.

99.36%/99.29% prediction accuracy and takes 23.80/24.06 seconds, compared with that of MLP attack, 98.99%/99.02% prediction accuracy and 39.24/68.08 seconds using 7.1×10^4/15.1×10^4 training CPRs. By comparing the attacking accuracy of MM and MLP attacks for 2-XOR-nCk-PUF, as shown in Table 3 and Table 4, the former also shows its advantage over the latter. Though significant as that of 1-XOR-nCk-PUF. Additionally, except for fewer CRPs, higher prediction accuracy and shorter training time, for 1/2-XOR-nCk-PUF attacking, since the structure of MM is simple and requires less computing resources compared with MLP.



(a) 1-xor-nCk-PUF. (b) 2-xor-nCk-PUF. (c) 3-xor-nCk-PUF.

(d) 4-xor-nCk-PUF. (e) 5-xor-nCk-PUF. (f) 6-xor-nCk-PUF.

FIGURE 7. Double logarithmic plot of training time on the ratio of training CRPs for 1 to 6-XOR-nCk-PUF.

Second, in case of more XOR operation, the MLP attack has a strong superiority compared with MM attack. The MM attack faces hardness when high non-linearity is injected into the PUF structure. From the analysis of section IV-B, MLP attack can still achieve high prediction accuracy even for nCk-PUF with multiple XOR operation, as shown in Table 4 and Fig. 6, its prediction accuracy can achieve as high as 99%. As a matter of fact, MLP attack is able to accurately model l-XOR-nCk-PUF even when $l > 6$ as discussed later. In addition to strong attack ability against multiple XOR nCk-PUF, when the training process reaches the convergence stage, the loss function's stability of MLP attack is better than that of MM attack, as shown in Fig. 4 and Fig. 5, respectively.

B. THE HIDDEN LAYER PARAMETER OF MLP

As it mentioned in section IV-B, the MLP in previous attacks has three hidden layers with 64, 32 and 16 neurons in each hidden layer, respectively. Khalid *et al.* [35] present a neural network-based method that can successfully attack XOR-PUFs with significantly fewer CRPs and shorter learning time when compared with other ML attack methods [30], [34], [38], [40], the MLP comprised of three hidden layers, with 2^{l-1} neurons for the first and last layers and 2^l for the middle layer, l is the number of XOR operations. Therefore, we will compare the prediction accuracy and timing of those two method for 4 to 9-XOR-nCkPUF, the simulation results as shown in Table. 6.

Firstly, from Table. 6 we can note our MLP attack setting can break 7, 8 and 9-XOR-nCk-PUF. To be exactly, it can achieve 98.71%, 98.56% and 96.58% prediction accuracy for 7-XOR, 8-XOR and 9-XOR 36(6)-PUF with 150×10^4, 300×10^4 and 400×10^4 CRPs, respectively. For 7-XOR, 8-XOR and 9-XOR 68(8)-PUF, the corresponding prediction accuracy are 98.84%, 98.49% and 97.23% using 250×10^4, 400×10^4 and 500×10^4 training CRPs, respectively. For 7-XOR, 8-XOR and 9-XOR 100(10)-PUF, it can still obtain about 98% prediction accuracy with 400×10^4, 600×10^4 and 800×10^4 CRPs. Additionally, the training time for all the experiment shown in Table. 6 is less than 40 minutes.

Secondly, for 4-XOR-nCk-PUF, the neuron number of hidden layer used in [35] is $(2^3, 2^4, 2^3)$, which is smaller than that of our method $(64, 32, 16)$. From Table. 6 we can see the prediction accuracy of our method is higher and training time is shorter, taking 4-XOR-100(10)-PUF for example, the prediction accuracy of our method is 99.01% compared with Khalid *et al.* of 98.53% with 95×10^4 CPRs talking part in training. Meanwhile, the training time of our method is about 1200 seconds compared with their training time 1600 seconds. For 5-XOR-nCk-PUF, the comparison of the two methods' attack results is consistent with that of 4-XOR-nCk-PUF. For 6-XOR-nCk-PUF, the number neurons of hidden layer of their setting is $(2^4, 2^5, 2^4)$, the neuron sum is slightly larger than that of our method, and the attack results of those two method are similar as shown in

TABLE 6. Prediction accuracy and timing comparison between our attacking method and other method.

	4-XOR-nCk-PUF					5-XOR-nCk-PUF				
	Tr. CRPs	$(64, 32, 16)$		$(2^3, 2^4, 2^3)$		Tr. CRPs	$(64, 32, 16)$		$(2^4, 2^5, 2^4)$	
	$(\times 10^4)$	Test Acc.$(\%)$	Tr. time(Sec)	Test Acc.$(\%)$	Tr. time(Sec)	$(\times 10^4)$	Test Acc.$(\%)$	Tr. time(Sec)	Test Acc.$(\%)$	Tr. time(Sec)
36(6)	33	0.9903	539.4373	98.53	782.25	55	99.02	810.67	98.67	1223.07
64(8)	60	0.9896	885.9444	98.21	1140.83	120	99.1	1193.29	98.91	1313.97
100(10)	95	0.9901	1201.21	98.53	1605.00	158	98.98	1430.39	97.78	2069.93
	6-XOR-nCk-PUF					**7-XOR-nCk-PUF**				
	Tr. CRPs	$(64, 32, 16)$		$(2^5, 2^6, 2^5)$		**Tr. CRPs**	$(64, 32, 16)$		$(2^6, 2^7, 2^6)$	
	$(\times 10^4)$	Test Acc.$(\%)$	Tr. time(Sec)	Test Acc.$(\%)$	Tr. time(Sec)	$(\times 10^4)$	Test Acc.$(\%)$	Tr. time(Sec)	Test Acc.$(\%)$	Tr. time(Sec)
36(6)	132	99.04	1513.39	98.90	1752.04	150	98.71	1689.86	99.08	2176.25
64(8)	273	99.05	2066.39	99.03	2123.94	250	98.84	1938.66	99.07	2226.39
100(10)	370	99.00	2395.43	99.01	2189.55	400	98.64	2045.97	99.13	2460.64
	8-XOR-nCk-PUF					**9-XOR-nCk-PUF**				
	Tr. CRPs	$(64, 32, 16)$		$(2^7, 2^8, 2^7)$		**Tr. CRPs**	$(64, 32, 16)$		$(2^8, 2^9, 2^8)$	
	$(\times 10^4)$	Test Acc.$(\%)$	Tr. time(Sec)	Test Acc.$(\%)$	Tr. time(Sec)	$(\times 10^4)$	Test Acc.$(\%)$	Tr. time(Sec)	Test Acc.$(\%)$	Tr. time(Sec)
36(6)	300	98.56	1501.98	98.96	1346.44	400	96.58	1441.2	98.7	1038.41
64(8)	400	98.49	1729.8	98.93	1527.97	500	97.23	1637.87	98.73	1545.38
100(10)	600	98.02	2014.08	98.96	1934.13	800	97.68	1789.95	98.92	2134.45

* $(64, 32, 16)$ is hidden layer parameter of our attack method, it means the neural network has three hidden layers, and the number of neurons in each hidden layer is 64, 32 and 16, respectively. $(2^{l-1}, 2^l, 2^{l-1})$ is hidden layer parameter of the attack method proposed in [35], the neurons number of its three hidden layers is 2^{l-1}, 2^l and 2^{l-1}), respectively, where refers to the times of XOR operation, e.g. when $l = 6$, it is $(2^5, 2^6, 2^5)$.

Table. 6. For 7-XOR-nCk-PUF, the hidden layers' neurons number of their setting is $(2^6, 2^7, 2^6)$, which is larger that of our setting (64, 32, 16). The prediction accuracy of our method is slightly lower than that of the method displayed in [35], however, our setting has a slight advantage in training time, as shown in Table. 6, the prediction accuracy and train time of our method for 7-XOR-100(10)-PUF are 98.64% and 2045.97 seconds compared with that of the other method's 99.13% and 2460.64 seconds. For 8-XOR-nCk-PUF and 9-XOR-nCk-PUF, the comparison of the two methods' attack results are consistent with that of 7-XOR-nCk-PUF.

From the above analysis, we can draw a conclusion that our MLP attack, which contains one input layer, three hidden layers (64, 32, 16) and one output layer, can successfully break l-XOR-nCk-PUF. Even when l is up to 9, it still can achieve 98% prediction accuracy. Additionally, compared with the MLP structure given in [35], which also has three hidden layers but its parameter is $(2^{l-1}, 2^l, 2^{l-1})$, when the sum of neuron is smaller than that of our method (64, 32, 16), the attack results of our method are better. On the contrary, the method proposed in [35] works well. However, when $l <= 9$, the prediction accuracy difference between the two methods is very small. Therefore, we set the hidden layer parameters to (64, 32, 16), and it is especially beneficial for the situation that the number of XOR operation is unknown, we can use the least resources to get the best attack results.

C. ATTACKING SCENARIO

From Table. 4 and Table. 6, we can see the number of CRP needed to attack l-XOR-nCk-PUF increases with l changing from 1 to 9. To be precise, about 800 CRPs are required for 1-XOR-36(6)-PUF to achieve 97% prediction accuracy, and about 3.4 thousand and 25 thousand CRPs are needed to get 98% and 99% prediction accuracy. As the required number of CRPs is not large, it is feasible for an attacker to collect the required CRPs through eavesdropping on the authentication channel. However, for 6-XOR-36(6)-PUF, the number of CRPs required to achieve 97%, 98% and 99% accuracy will increase to about 86 thousand, 245 thousand and 1.314 million, it becomes infeasible to collect through eavesdropping.

Nonetheless, there are chances of the attacker to gain physical access to the PUF embedded device in practice through the device supply chain to acquire a large number of CRPs for machine learning training. For example, the attacker is a deliver party who transfers the PUF device to the PUF end-user. In this case, the attacker who directly accesses the PUF can arbitrarily query the PUF to gain a large number of CRPs, e.g.,2,000,000, in a short time, for building an accurate soft model of the physical PUF. Though exhaustively characterizing all CRPs (i.e., total CRPs of 2.1811×10^{15} yielded by the 9-XOR-100(10)-PUF) is still infeasible in a limited time period, e.g., months.

D. FUTURE WORK

Future work can be carried out from the perspective of both attacks and defenses.

Efficient Attack. First of all, from the perspective of attack, one can focus on improving the modeling attack accuracy while minimizing computation overhead and required number of CRPs.

For simple 1-XOR-nCk-PUF, the prediction process of MM attack can be considered as classify the output/response r (0 or 1) into two groups, which is equivalent to two points on a plane. This can be easily separated by a line. For 2-XOR-nCk-PUF, the output of PUF r is determined by r_1 and r_2, to be exactly, $(0, 0) \rightarrow 0$, $(0, 1) \rightarrow 1$, $(1, 0) \rightarrow 1$ and $(1, 1) \rightarrow 0$. Therefore, the prediction process of MM attack can be considered to classify those four points ($(0, 0)$, $(0, 1)$, $(1, 0)$ and $(1, 1)$) into to two groups based on the finally output of nCk-PUF r (0 or 1), and it is easy to find two lines to model the classification task. Then, from the perspective of classification, we can continue to explore high-dimensional separable space to model l-XOR-nCk-PUF when $l > 2$.

One can explore other machine learning techniques such as RNN, CNN other than MLP to model l-XOR-nCk-PUF and compare its corresponding attack results with that of MLP, so as to obtain the best network to model l-XOR-nCk-PUF with lower CRPs, higher prediction accuracy and shorter training time.

From a defensive point of view, one can combine the nCk-PUF with other secure designs, though straightforwardly using the nCk-PUF is rather insecure.

In other words, because of the attack shown by us, simply using the nCk-PUF is insecure in practice, in particular, for its main authentication application that takes advantage of its large CRP space. However, it is possible to incorporate it into a secure design.

Notably, the security of a PUF system can be greatly ensured if additional security blocks are used to protect the challenge-response interface [15], [18], [39], [41], [42] such as the Lockdown-PUF with additional RNG, TREVERSE with additional Hash [15], [18], [39], [41], [42]. The nCk-PUF is complementary with these designs for a secure usage. For example, XOR-nCk-PUF can replace XOR-APUF that is used in Lockdown-PUF system [39] to still ensure a number of secure authentication rounds. Specifically, whenever the XOR-nCk-PUF used in Lockdown-PUF system has exposed a pre-estimated number of usable CRPs, these CRPs can be exploited to build an accurate model of the XOR-nCk-PUF. In such a case, the lockdown-PUF instance can no longer be used for the upcoming CRP-based authentication and thus being discarded. Note the lock-down PUF prevents an attacker from gaining new CRPs unless a new authentication is explicitly issued and released by a server.

VI. CONCLUSION

This paper firstly thoroughly examines the modeling attack resilience of nCk-PUF with mathematical model (MM) based attack, which easily achieves 99% attacking accuracy regardless of the PUF size. This has trivially refuted the previous security claim of the nCk-PUF. As XOR is a common practice to increase the robustness of PUF against modeling attack, we further investigate the l-XOR-nCk-PUF security. Though the MM attack becomes inefficient when l is slightly larger, e.g., more than 3. The MLP attack can still successfully break it even when the l is up to 9 with relatively small number of CRPs and limited training time for achieving a close to 100% attacking accuracy. In this context, an empirical conclusion is that the PUFs sharing similar topology with nCk-PUF are not as secure as they were claimed even for its l-XOR-nCk-PUF variants. Care should be taken when using them in practice, and it is suggested to add some security blocks to protect the challenge-response interface as discussed in Section V.

REFERENCES

[1] Y. Gao, S. F. Al-Sarawi, and D. Abbott, "Physical unclonable functions," *Nature Electron.*, vol. 3, no. 2, pp. 81–91, 2020.

[2] C. Herder, M.-D. Yu, F. Koushanfar, and S. Devadas, "Physical unclonable functions and applications: A tutorial," *Proc. IEEE*, vol. 102, no. 8, pp. 1126–1141, Aug. 2014.

[3] Y. Gao, Y. Su, L. Xu, and D. C. Ranasinghe, "Lightweight (reverse) fuzzy extractor with multiple reference PUF responses," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 7, pp. 1887–1901, Jul. 2019.

[4] F. Tehranipoor, N. Karimian, W. Yan, and J. A. Chandy, "DRAM-based intrinsic physically unclonable functions for system-level security and authentication," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 3, pp. 1085–1097, Mar. 2017.

[5] Y. Wang, W.-K. Yu, S. Wu, G. Malysa, G. E. Suh, and E. C. Kan, "Flash memory for ubiquitous hardware security functions: True random number generation and device fingerprints," in *Proc. IEEE Symp. Secur. Privacy*, May 2012, pp. 33–47.

[6] A. Maiti, J. Casarona, L. McHale, and P. Schaumont, "A large scale characterization of RO-PUF," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust (HOST)*, Jun. 2010, pp. 94–99.

[7] Q. Chen, G. Csaba, P. Lugli, U. Schlichtmann, and U. Rührmair, "The bistable ring PUF: A new architecture for strong physical unclonable functions," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust (HOST)*, Jun. 2011, pp. 134–141.

[8] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *Symp. VLSI Circuits Dig. Tech. Papers*, 2004, pp. 176–179.

[9] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. 44th ACM/IEEE Design Automat. Conf.*, Jun. 2007, pp. 9–14.

[10] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure PUFs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2008, pp. 670–673.

[11] P. H. Nguyen, D. P. Sahoo, C. Jin, K. Mahmood, U. Rührmair, and M. Van Dijk, "The interpose PUF: Secure PUF design against state-of-the-art machine learning attacks," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2019, no. 4, pp. 243–290, Aug. 2019.

[12] X. Xu, U. Rührmair, D. E. Holcomb, and W. Burleson, "Security evaluation and enhancement of bistable ring PUFs," in *Proc. Int. Workshop Radio Freq. Identificat., Secur. Privacy Issues*. Cham, Switzerland: Springer, 2015, pp. 3–16.

[13] J. Delvaux, "Machine-learning attacks on polyPUFs, OB-PUFs, RPUFs, LHS-PUFs, and PUF-FSMs," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 8, pp. 2043–2058, Aug. 2019.

[14] S. T. C. Konigsmark, D. Chen, and M. D. F. Wong, "PolyPUF: Physically secure self-divergence," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 7, pp. 1053–1066, Jul. 2016.

[15] Y. Gao, G. Li, H. Ma, S. F. Al-Sarawi, O. Kavehei, D. Abbott, and D. C. Ranasinghe, "Obfuscated challenge-response: A secure lightweight authentication mechanism for PUF-based pervasive devices," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2016, pp. 1–6.

[16] J. Ye, Y. Hu, and X. Li, "RPUF: Physical unclonable function with randomized challenge to resist modeling attack," in *Proc. IEEE Asian Hardw.-Oriented Secur. Trust (AsianHOST)*, Dec. 2016, pp. 1–6.

[17] T. Idriss and M. Bayoumi, "Lightweight highly secure PUF protocol for mutual authentication and secret message exchange," in *Proc. IEEE Int. Conf. RFID Technol. Appl. (RFID-TA)*, Sep. 2017, pp. 214–219.

[18] Y. Gao, H. Ma, S. F. Al-Sarawi, D. Abbott, and D. C. Ranasinghe, "PUF-FSM: A controlled strong PUF," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 5, pp. 1104–1108, May 2018.

[19] W. Yu, "Optimization of combined power and modeling attacks on VR PUFs with Lagrange multipliers," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 11, pp. 2512–2516, Nov. 2020.

[20] P. Santikellur and R. S. Chakraborty, "A computationally efficient tensor regression network-based modeling attack on XOR arbiter PUF and its variants," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 6, pp. 1197–1206, Jun. 2021.

[21] A. Wang, W. Tan, Y. Wen, and Y. Lao, "NoPUF: A novel PUF design framework toward modeling attack resistant PUFs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 6, pp. 2508–2521, Jun. 2021.

[22] P. Williams, H. Idriss, and M. Bayoumi, "Mc-PUF: Memory-based and machine learning resilient strong PUF for device authentication in Internet of Things," in *Proc. IEEE Int. Conf. Cyber Secur. Resilience (CSR)*, Jul. 2021, pp. 61–65.

[23] J. Zhang, C. Shen, Z. Guo, Q. Wu, and W. Chang, "CT PUF: Configurable tristate PUF against machine learning attacks for IoT security," *IEEE Internet Things J.*, early access, Jun. 18, 2021, doi: 10.1109/JIOT.2021.3090475.

[24] D. P. Sahoo, D. Mukhopadhyay, R. S. Chakraborty, and P. H. Nguyen, "A multiplexer-based arbiter PUF composition with enhanced reliability and security," *IEEE Trans. Comput.*, vol. 67, no. 3, pp. 403–417, Mar. 2018.
[25] J. Shi, Y. Lu, and J. Zhang, "Approximation attacks on strong PUFs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 10, pp. 2138–2151, Oct. 2019.
[26] J. Mathew, R. S. Chakraborty, D. P. Sahoo, Y. Yang, and D. K. Pradhan, "A novel memristor-based hardware security primitive," *ACM Trans. Embedded Comput. Syst.*, vol. 14, no. 3, pp. 1–20, May 2015.
[27] H. Nili, G. C. Adam, B. Hoskins, M. Prezioso, J. Kim, M. R. Mahmoodi, F. M. Bayat, O. Kavehei, and D. B. Strukov, "Hardware-intrinsic security primitives enabled by analogue state and nonlinear conductance variations in integrated memristors," *Nature Electron.*, vol. 1, no. 3, pp. 197–202, Mar. 2018.
[28] S. Zeitouni, E. Stapf, H. Fereidooni, and A.-R. Sadeghi, "On the security of strong memristor-based physically unclonable functions," in *Proc. 57th ACM/IEEE Design Automat. Conf. (DAC)*, Jul. 2020, pp. 1–6.
[29] Y. Gao, D. C. Ranasinghe, S. F. Al-Sarawi, O. Kavehei, and D. Abbott, "Memristive crypto primitive for building highly secure physical unclonable functions," *Sci. Rep.*, vol. 5, no. 1, Oct. 2015, Art. no. 12785.
[30] U. R Ührmair, F. Sehnke, J. S Ölter, G. Dror, S. Devadas, and J. Ü. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proc. 17th ACM Conf. Comput. Commun. Secur. (CCS)*, 2010, pp. 237–249.
[31] U. Rührmair, J. Sölter, F. Sehnke, X. Xu, A. Mahmoud, V. Stoyanova, G. Dror, J. Schmidhuber, W. Burleson, and S. Devadas, "PUF modeling attacks on simulated and silicon data," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 11, pp. 1876–1891, Nov. 2013.
[32] P. Santikellur and R. S. Chakraborty, "A computationally efficient tensor regression network-based modeling attack on XOR arbiter PUF and its variants," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 6, pp. 1197–1206, Jun. 2021.
[33] L. Samir, G. Said, K. Abderrahmen, and S. Youcef, "A new training method for solving the XOR problem," in *Proc. 5th Int. Conf. Electr. Eng.-Boumerdes (ICEE-B)*, Oct. 2017, pp. 1–4, doi: 10.1109/ICEE-B.2017.8192143.
[34] J. Tobisch and G. T. Becker, "On the scaling of machine learning attacks on PUFs with application to noise bifurcation," in *Proc. Int. Workshop Radio Freq. Identificat., Secur. Privacy Issues*. Cham, Switzerland: Springer, 2015, pp. 17–31.
[35] K. T. Mursi, B. Thapaliya, Y. Zhuang, A. O. Aseeri, and M. S. Alkatheiri, "A fast deep learning method for security vulnerability study of XOR PUFs," *Electronics*, vol. 9, no. 10, p. 1715, Oct. 2020, doi: 10.3390/electronics9101715.
[36] G. Hospodar, R. Maes, and I. Verbauwhede, "Machine learning attacks on 65 nm arbiter PUFs: Accurate modeling poses strict bounds on usability," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2012, pp. 37–42, doi: 10.1109/WIFS.2012.6412622.
[37] M. S. Alkatheiri and Y. Zhuang, "Towards fast and accurate machine learning attacks of feed-forward arbiter PUFs," in *Proc. IEEE Conf. Dependable Secure Comput.*, Aug. 2017, pp. 181–187, doi: 10.1109/DESEC.2017.8073845.
[38] 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 Int. Congr. Internet Things (ICIOT)*, Jul. 2018, pp. 49–56, doi: 10.1109/ICIOT.2018.00014.
[39] M. D. Yu, M. Hiller, J. Delvaux, R. Sowell, S. Devadas, and I. Verbauwhede, "A lockdown technique to prevent machine learning on PUFs for lightweight authentication," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 2, no. 3, pp. 146–159, Apr. 2016.
[40] P. Santikellur, A. Bhattacharyay, and R. Chakraborty, "Deep learning based model building attacks on arbiter PUF compositions," Cryptol. ePrint Arch., Tech. Rep. 2019/566, 2019. [Online]. Available: https://ia.cr/2019/566
[41] C. Herder, L. Ren, M. van Dijk, M.-D. Yu, and S. Devadas, "Trapdoor computational fuzzy extractors and stateless cryptographically-secure physical unclonable functions," *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 1, pp. 65–82, Jan. 2017.
[42] Y. Gao, M. Van Dijk, L. Xu, W. Yang, S. Nepal, and D. Ranasinghe, "TREVERSE: Trial-and-error lightweight secure reverse authentication with simulatable PUFs," *IEEE Trans. Dependable Secure Comput.*, early access, May 11, 2020, doi: 10.1109/TDSC.2020.2993802.

LIHUI PANG received the bachelor's degree from the Henan University of Science and Technology, the M.Sc. degree from Xihua University, in 2011, and the Ph.D. degree from the University of Electronic Science and Technology of China, China, in 2015. She is currently with the University of South China, as a Lecturer, and Sungkyunkwan University, as a Visiting Scholar. Her current research interests include physical unclonable functions designing and attacking, signal separation, and recognition.

HYOUNGSHICK KIM received the B.S. degree from the Department of Information Engineering, Sungkyunkwan University, the M.S. degree from the Department of Computer Science, Korea Advanced Institute of Science and Technology (KAIST), and the Ph.D. degree from the Computer Laboratory, University of Cambridge, in 1999, 2001, and 2012, respectively. From 2004 to 2008, he worked with Samsung Electronics, as a Senior Engineer. He worked with the Department of Electrical and Computer Engineering, The University of British Columbia, as a Postdoctoral Fellow. He is currently with the Department of Software, Sungkyunkwan University, as an Associate Professor. His current research interests include usable security and security engineering.

BIN YANG (Member, IEEE) was born in Henan, China, in 1980. He received the B.S. degree in automation from the Zhengzhou University of Light Industry, in 2005, and the Ph.D. degree in control science and engineering from Hunan University, in 2010. He is currently a Professor with the University of South China. He is the author of more than 30 journal articles. His current research interests include image processing, pattern recognition, and deep learning.

XINLIN WANG received the B.S. degree in opto-electronical technology from the Huazhong University of Science and Technology, in 1992, the M.Sc. degree from the National University of Defense Technology, and the Ph.D. degree from the Huazhong University of Science and Technology, in 2001 and 2007, respectively. He is currently a Professor with the University of South China and works as the Dean of the Graduate School. His current research interests include the basic theory and numerical simulation of the interaction of high-power laser, ultrafast laser and matter, surface plasmon optics, intelligent laser processing systems and advanced laser manufacturing technology.

YANSONG GAO (Member, IEEE) received the bachelor's degree from the Henan University of Science and Technology, the M.Sc. degree from the University of Electronic Science and Technology of China, in 2013, and the Ph.D. degree from the University of Adelaide, Australia, in 2017. He was with Data61, CSIRO, Sydney, Australia, as a Postdoctoral Research Fellow. He is currently with the Nanjing University of Science and Technology, China, as an Associate Professor. His current research interests include hardware security with focus on physical unclonable functions, AI security, and system security.