A Multiplexer based Arbiter PUF Composition with Enhanced Reliability and Security

Durga Prasad Sahoo, Debdeep Mukhopadhyay, Rajat Subhra Chakraborty, and Phuong Ha Nguyen

Abstract—Arbiter Physically Unclonable Function (APUF), while being relatively lightweight, is extremely vulnerable to modeling attacks. Hence, various compositions of APUFs such as XOR APUF and Lightweight Secure PUF have been proposed to be secure alternatives. Previous research has demonstrated that PUF compositions have two major challenges to overcome: vulnerability against modeling and statistical attacks, and lack of reliability. In this paper, we introduce a multiplexer based composition of APUFs, denoted as MPUF, to simultaneously overcome these challenges. In addition to the basic MPUF design, we propose two MPUF variants namely cMPUF and rMPUF to improve robustness against cryptanalysis and reliability based modeling attack, respectively. The rMPUF demonstrates enhanced robustness against reliability based modeling attack, while even the well-known XOR APUF, otherwise robust to machine learning based modeling attacks, has been modeled using the same technique with linear data and time complexities. The rMPUF can provide a good trade-off between security and hardware overhead while maintaining a significantly higher reliability level than any practical XOR APUF instance. Moreover, MPUF variants are the first APUF compositions, to the best of our knowledge, that can achieve Strict Avalanche Criterion without any additional hardware. Finally, we validate our theoretical findings using Matlab-based simulations of MPUFs.

Index Terms—Arbiter PUF (APUF), APUF compositions, modeling attack, linear cryptanalysis, reliability based modeling, strict avalanche criteria (SAC), XOR APUF.

1 Introduction

Since the introduction of Arbiter Physically Unclonable Function (APUF) in [1], several applications [2]–[4] of APUF in lightweight and secure protocol design have been proposed. From the very early days of this design, it is known that APUF is vulnerable to functional modeling using machine learning techniques like Logistic Regression (LR) and Support Vector Machine (SVM) [1], [5]. To improve modeling robustness, the following compositions of APUFs were proposed: XOR PUF [5], [6], Lightweight Secure PUF (LSPUF) [7], [8].

One of the most well-studied and exploited composition of APUFs is the x-XOR APUF, in which outputs of x APUF instances are XOR-ed to generate the final output. The XOR PUF is inherently more robust to modeling attacks than standalone APUF. Nevertheless, several attempts of modeling XOR APUF with varying levels of success and practical feasibility were reported in the literature. These attacks on XOR APUF can be classified into two following classes: (i) CRP-only modeling (CM) and (ii) side-channel assisted CRP-based modeling (SCM), where CRP implies challenge-response pair of XOR APUF. The CM attacks of x-XOR APUF using logistic regression (LR) are reported in [5], [9], [10], and SCM attacks are published in [11]–[15]. In general SCM attacks are efficient compared to the CM attacks, but SCM attacks need additional information like power consumption of device, reliability of CRPs, etc. The most powerful SCM attack on XOR APUF, to the best of our knowledge, is the reliability based modeling attack proposed by Becker in [15]. For this attack, an attacker should be able to evaluate PUF instance with the same set of challenges repeatedly. Since LSPUF outputs are generated using XOR PUF, most of the attack strategies developed for XOR PUF is also applicable to LSPUF. In addition, cryptanalysis attacks on LSPUF have also been described in [16], [17].

Beside these security issues, x-XOR APUF also suffers from poor reliability. There are two conflicting properties of x-XOR APUF: reliability of composition reduces exponentially with increasing number of primitive APUF instancess; however, to achieve higher level of security, composition should employ large number of APUF instances.

In this work, we propose a new APUF composition that satisfactorily overcomes the reliability issue of XOR APUF, while providing levels of security comparable to XOR APUF. The proposed composition employs a single multiplexer (MUX) with multiple APUFs; hence, we term it as *Multiplexer PUF* (MPUF). Besides the basic MPUF design, we propose two other variants of MPUF, namely cMPUF and rMPUF, which are robust against cryptanalysis and reliability based modeling attack, respectively. The rMPUF design is the first APUF composition that achieves robustness against reliability based modeling attack, and robustness of a feasible rMPUF is superior than a practical XOR APUF instance (e.g. 10-XOR APUF) while successfully maintains reasonable higher reliability level. Moreover, all these MPUF variants can achieve Strict Avalanche Criterion (SAC), important for robustness against cryptanalysis

The authors are with the Secured Embedded Architecture Laboratory (SEAL), Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur, West Bengal, INDIA–721302. E-mail: {dpsahoo.cs,phuongha.ntu}@gmail.com and {debdeep,rschakraborty}@cse.iitkgp.ernet.in.

attacks [17], without using any additional hardware (unlike the LSPUF which requires an input network).

An application of MUX in the design of robust cryptographic Boolean function can be found in [18]. In the context of PUF, utilization of MUX as a recombination function was first suggested in [19]. The design proposed in [19] used MUX to recombine responses of a single APUF instance, unlike the MPUF where overall output depends on outputs of multiple independent APUFs. However, in [19], authors only suggested this MUX like recombination operator as an example of possible recombination operators without any accompanying security or statistical performance analysis. On the other hand, our work discusses many performance and security advantages of MUX based composition, which, to the best of our knowledge, have not been discussed in the PUF literature earlier.

The rest of the paper is organized as follows. In Section 2, we introduce a notation system to be used in the rest of the paper, and a few desirable properties of an effective PUF composition. In Section 3, we illustrate the basic MPUF design. The general advantages of MPUF are explained in Section 4. The performance metrics for MPUF are analyzed in Section 5. The reliability based model building of the basic MPUF and design of a robust MPUF variant (rMPUF) are discussed in Section 6. Section 7 explains a linear approximation attack on MPUF, and introduces a robust MPUF variant (cMPUF). SAC property of MPUF is discussed in Section 8. Simulation results are reported in Section 9. A discussion on the usages of MPUF variants is provided in Section 10. The concluding remarks and directions of future works are mentioned in Section 11.

2 Preliminaries

2.1 Notations

The following notation system will be used throughout the paper. A vector is represented by lowercase letter in bold font, e.g $\mathbf{a}$. A vector with m-components is represented as: $\mathbf{a} = (\mathbf{a}[0], \ldots, \mathbf{a}[i], \ldots, \mathbf{a}[m-1])$, where $\mathbf{a}[i]$ is the i-th component of $\mathbf{a}$. We use $\mathbf{a}[i:j]$ as a sub-vector $(\mathbf{a}[i], \ldots, \mathbf{a}[j])$. The normal lowercase letter denotes a scalar, e.g. n. The $\oplus$ symbol stands for the Exclusive-OR (XOR) operation or modulo-2 addition. A set is represented in the calligraphic font, e.g. $\mathcal{D}$, and its cardinality is denoted as $|\mathcal{D}|$. A set of natural numbers that are greater than equal to i and less than equal to j is dented as $[i:j]$, whereas (i,j) and $[i,j]$ are used to denote open and closed intervals of real numbers between i and j. Let f and g be Boolean functions. The Hamming distance between f and g is denoted as $\mathrm{HD}(f,g)$ and normalized HD with respect to total number of inputs is denoted by $h_d(f,g)$. Similarly, $\mathrm{HW}(f)$ and $h_w(f)$ denote HW and normalized HW of f. The f and g can be functions or binary strings. The $\langle \mathbf{a}, \mathbf{b} \rangle$ denotes dot product of vector $\mathbf{a}$ and $\mathbf{b}$.

2.2 Desirable Properties of an Effective PUF Composition

The main objective of PUF composition is to employ multiple PUF instances to make the resulting PUF secure against modeling attacks with reasonable reliability level. Let $P_0, \ldots, P_{n-1}$ be PUF instances to be used in the composition, and $\mathcal{B}$ be a Boolean function used to combine all PUF units together, i.e. $o = \mathcal{B}(r_0, \ldots, r_{n-1})$, where r_i is a 1-bit response of PUF instance $P_i, i \in [0 : n-1]$. The exploration for an effective PUF composition $\mathcal{B}$ should be driven by the following factors:

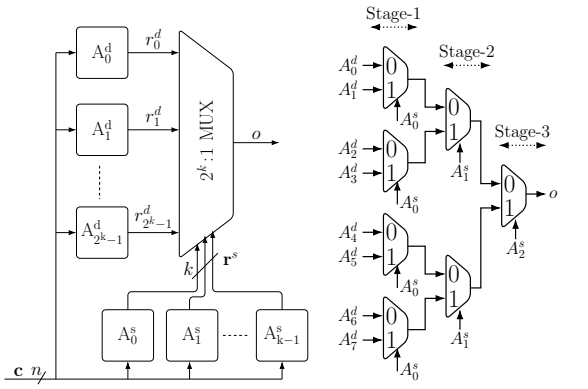
1) **Lightweight vs. Security.** One should not employ a $\mathcal{B}$ with large number of inputs as large number of primitive PUFs is required for generating those inputs. However, this type of complex Boolean function can provide the statistical criteria required for designing a robust PUF composition against cryptanalysis and modeling attacks. So, we need a $\mathcal{B}$ with relatively less number of inputs, while still being robust against modeling and statistical attacks.
2) **Reliability vs. Security.** For a given challenge, if response of $\mathcal{B}$ depends on large number of primitive PUFs, then reliability of composition would suffer. In the case, where a composition employs large number of primitive PUFs to achieve higher degree of security, reliability can be managed if response of $\mathcal{B}$ for a given challenge relies on a relatively small subset of primitive PUFs.
3) **Nonlinearity vs. Linear Approximation.** Nonlinearity of $\mathcal{B}$ should be good enough such that it cannot be approximated by a linear function. For example, an adversary may try to model $\mathcal{B}$ as linear function of its inputs, to enable *linear cryptanalysis* [20].
4) **Balancedness**. To achieve good uniformity property of a PUF composition, the Boolean function employed in composition should be a balanced function.

Next, we discuss multiplexer based APUF compositions, and analyze those designs with respect to above mentioned properties.

3 Overview of Multiplexer based APUF Composition

Figure 1a depicts an architectural overview of multiplexer (MUX) based APUF composition, called MPUF. Here, a MUX is used to combine the responses of primitive APUFs. Two important design parameters of an MPUF are:

1) k: this parameter denotes the number of selection inputs of a MUX. A MUX with k selection inputs has 2^k data inputs and one output. This parameter is an important security parameter, and later we will discuss how to choose the value for this parameter to achieve robustness against modeling attacks.



(a) (n,k)-MPUF (b) (n,3)-MPUF using 2:1 MUXes

Fig. 1: (a) Architectural overview of an (n,k)-MPUF. It generates a 1-bit response o to an n-bit challenge $\mathbf{c}$. The APUF instances $A_i^d, 0 \leq i \leq 2^k-1$ are connected to the data inputs of MUX, and k-bit selection inputs of MUX are generated by k APUF instances $A_j^s, 0 \leq j \leq k-1$. (b) An example of $(n,3)$-MPUF using 2:1 MUXes.

2) n. this parameter denotes the challenge size of primitive APUFs as well as MPUF, and all primitive APUFs receive n-bit challenge.

We use the notation (n,k)-MPUF (cf. Fig. 1a) to represent an MPUF with n-bit challenge and a MUX with k selection inputs. Let $\mathcal{X}$ be a set of 2^k+k primitive APUFs used in (n,k)-MPUF, and we partition this set into two disjoint sets $\mathcal{D}$ and $\mathcal{S}$, depending on their usage purpose. The sets $\mathcal{D} = \{A_0^d, \ldots, A_{2^k-1}^d\}$ and $\mathcal{S} = \{A_0^s, \ldots, A_{k-1}^s\}$ represent the primitive APUFs connected to data inputs and selection inputs of MUX, respectively. The 1-bit responses of $A_i^d \in \mathcal{D}$ and $A_j^s \in \mathcal{S}$ are denoted by r_i^d and r_j^s, respectively.

Since, for a given challenge $\mathbf{c}$, response of MPUF is the same as response of an APUF in $\mathcal{D}$, we can use a control circuitry to activate the corresponding data input APUF only, instead of activating all APUFs in $\mathcal{D}$. Selection of an APUF from $\mathcal{D}$ depends on the concatenated responses of APUFs in $\mathcal{S}$. We denote the concatenated cumulative response bits of $A_j^s \in \mathcal{S}$ by $\mathbf{r}^s = r_0^s|\cdots|r_{k-1}^s$. By the statement "$\mathbf{r}^s = i$", let us denote that i is an unsigned integer corresponding to binary vector $\mathbf{r}^s$.

An (k,n)-MPUF instance generates 1-bit response to an n-bit challenge $\mathbf{c}$, and size of challenge-response space is 2^n. There are 2^k APUFs connected to MUX data inputs, and their challenge-response pairs (CRPs) form CRP space $\mathcal{C}$ of (n,k)-MPUF. Note that all CRPs of an individual $A_i^d, i \in [0, 2^k-1]$ do not belong to $\mathcal{C}$. Ideally, data input APUFs $A_i^d, i \in [0, 2^k-1]$ contribute equally to $\mathcal{C}$, and under such conditions, the number of CRPs of each $A_i^d, i \in [0, 2^k-1]$ belonging to $\mathcal{C}$ is $2^n/2^k$. In practice, we might observe bias in selection of APUFs $A_i^d, i \in [0, 2^k-1]$ due to non-uniform distribution of k-bit selection inputs generated by $A_j^s \in \mathcal{S}$. Hence, data input APUF instances might not contribute equally to $\mathcal{C}$.

The modeling robustness of an MPUF instance heavily relies on the fact that an adversary does not have access to responses of primitive APUFs used in composition, i.e., all inputs (data and control inputs) of MUX in MPUF are secret. This security assumption about the "opaqueness" of design, that denies an adversary the opportunity to directly observe the internal nodes of a PUF circuit, is also made for XOR APUF and LSPUF designs.

4 Advantages of MUX Based Composition

In light of Section 2.2 on the desirable properties of an effective composition function, we are in a position to explain the reasons behind the choice of MUX as composition operator. Let k be the number of selection inputs of a MUX; then total number of inputs of a MUX is $m = 2^k + k$.

4.1 Balancedness

As depicted in Fig. 1a that MPUF instance produces 1-bit response o to a given n-bit challenge $\mathbf{c}$. This composition is said to be a *balanced composition* if distributions of 0's and 1's in response string are uniform, i.e., $\Pr(o=0) = \Pr(o=1) = \frac{1}{2}$.

We now estimate the probability $\Pr(o=0)$ for (n,k)-MPUF. Let us consider the response of each APUF instance $A_i^d, i \in [1 : 2^k-1]$ to be a binary random variable A_i^d. The $\Pr(A_i^d=0)$ and $\Pr(A_i^d=1)$ denote the probabilities that PUF instance A_i^d generates 0 and 1, respectively, with the obvious constraint $\Pr(A_i^d=0)+\Pr(A_i^d=1)=1$. The random variable A^s corresponds to PUF instance $A^s = A_0^s|\cdots|A_{k-1}^s$, and it takes values in $[0:2^k-1]$. Since A_i^d and A^s are statistically independent, we compute $\Pr(o=0)$ for (n,k)-MPUF as:

$$\begin{aligned}\Pr(o=0) &= \sum_{i=0}^{2^k-1} \Pr(A^s=i)\Pr(A_i^d=0|A^s=i) \\ &= \sum_{i=0}^{2^k-1} \Pr(A^s=i)\Pr(A_i^d=0), \end{aligned} \tag{1}$$

where $\Pr(A_i^d=0) = \Pr(A_i^d=0|A^s=i)$ and $\Pr(A_i^d=1) = \Pr(A_i^d=1|A^s=i)$ always hold. Let us assume that the random variable corresponding to each APUF instance used in composition follows uniform distribution, and thus $\Pr(A_i^d=0) = \Pr(A_i^d=1) = \frac{1}{2}$, $\Pr(A^s=j) = \frac{1}{2^k}$ for $j \in [0:2^k-1]$. Based on these assumptions, we can rewrite Eq. (1) as follows:

$$\begin{aligned}\Pr(o=0) &= \sum_{i=0}^{2^k-1} \Pr(A^s=i)\Pr(A_i^d=0) \\ &= \frac{1}{2^k}\sum_{i=0}^{2^k-1}\Pr(A_i^d=0) = \frac{1}{2^k}\sum_{i=0}^{2^k-1}\frac{1}{2} = \frac{2^{k-1}}{2^k} = \frac{1}{2}. \end{aligned} \tag{2}$$

Thus, output distribution of an MPUF would be uniform if none of its input distributions is biased. There can be another special case where the output distribution of MUX is uniform, even when all the input distributions are not uniform. We consider such a situation next; this special construction would be used later in Section 7.2.

Let us consider a special case of MPUF with the following data input connection pattern: *half of the data inputs are the bitwise complements of the other half*. Hence, we need 2^{k-1} APUF instances to generate 2^k data inputs of MUX. Let random variable $\bar{A}_i^d$ be complement of the binary random variable A_i^d defined earlier. In this case, we only assume that the random variable A^s is uniform, i.e., $\Pr(A^s = j) = \frac{1}{2^k}$ for $j \in [0 : 2^k - 1]$. Based on these assumptions and special input connection pattern, we can rewrite Eq. (1) as:

$$\begin{aligned}\Pr(o = 0) &= \sum_{i=0}^{2^k-1} \Pr(A^s = i)\Pr(A_i^d = 0) \qquad (3)\\ &= \frac{1}{2^k}\sum_{i=0}^{2^{k-1}-1}\left[\Pr(A_i^d = 0) + \Pr(\bar{A}_i^d = 0)\right]\\ &= \frac{1}{2^k}\sum_{i=0}^{2^{k-1}-1}\left[\Pr(A_i^d = 0) + \Pr(A_i^d = 1)\right] = \frac{2^{k-1}}{2^k} = \frac{1}{2}.\end{aligned}$$

According to Eq. (3), it is sufficient for selection input APUFs of MPUF, with special data input connection, to be balanced to make MPUF output balanced —it is not essential for the data input APUFs to be balanced. This is an important advantage. However, at the same time, the APUFs connected to data inputs of MUX should not be completely biased, as this reduces response unpredictability of MPUF. For example, let us consider a case where $\Pr(A_i^d = 0) = 1$ and $\Pr(\bar{A}_i^d = 1) = 1$. Intuitively, the unpredictability of response o must be less in the above case compared to the case where $\Pr(A_i^d = 0) = \frac{1}{2}$ and $\Pr(\bar{A}_i^d = 1) = \frac{1}{2}$. Both the cases maintain uniform distributions of 0's and 1's in o.

4.2 Nonlinearity

For an m-variable Boolean function f, its nonlinearity N_f is defined as:

$$N_f = \min_{i\in[0:2^{m+1}-1]} \mathrm{HD}(f, \psi_i),$$

where ψ_i is an m-variable affine function. Nonlinearity bounds of a balanced Boolean function is provided in the following lemma [21]:

Lemma 1 (Nonlinearity of balanced Boolean function). *Let $f : \{0,1\}^m \to \{0,1\}$ be a balanced Boolean function where $n \geq 3$. Then, nonlinearity N_f of f satisfies the following expression:*

$$N_f \leq \begin{cases} 2^{m-1} - 2^{m/2-1} - 2, & m \quad \textit{is even}\\ 2^{m-1} - 2^{\lfloor m/2\rfloor - 1}, & m \quad \textit{is odd}\end{cases} \qquad (4)$$

where $\lfloor x\rfloor$ denotes the maximum integer less than x.

For the sake of analysis of an m-variable Boolean function, let us use:

$$E_f = N_f/2^m \qquad (5)$$

which represents the normalized N_f, and it lies in range [0,0.5). $E_f \approx 0.5$ is desirable for robustness against linear cryptanalysis [20].

In case of MUX (with k selection inputs) in (n,k)-MPUF, an affine Boolean function that results in the best linear approximation has the following form:

$$\psi_{best}(\mathbf{c}; i, \mathbf{u}) = A_i^d(\mathbf{c}) \oplus \left(\bigoplus_{j=0}^{k-1} \mathbf{u}[j]A_j^s(\mathbf{c})\right), \qquad (6)$$

where $A_i^d, i \in [0 : 2^k - 1]$ is the i-th data input APUF of MUX and A_j^s is the j-th selection input. The $\mathbf{u}[j] \in \{0,1\}$ decides whether A_j^s will appear in ψ_{best}.

Let us recall the notation A^s used to represent concatenation of $A_j^s \in \mathcal{S}$ and assume it as a random variable with values in $[0 : 2^k - 1]$. Now we compute the probability that output o of (n,k)-MPUF can be approximated by using single data input APUF $A_i^d, i \in [0, 2^k - 1]$ as:

$$\begin{aligned}\Pr(o = A_i^d) &= \Pr(A^s = i)\Pr(o = A_i^d | A^s = i)\\ &\quad + \Pr(A^s \neq i)\Pr(o = A_i^d | A^s \neq i)\\ &= \frac{1}{2^k} + \sum_{\substack{j\in[0:2^k-1]\\ j\neq i}} \Pr(A^s = j)\Pr(o = A_i^d | A^s = j)\\ &= \frac{1}{2^k} + \frac{1}{2^k}\sum_{\substack{j\in[0:2^k-1]\\ j\neq i}} \Pr(o = A_i^d | A^s = j)\\ &= \frac{1}{2^k} + \frac{1}{2^k}[(2^k - 1) \times \frac{1}{2}] = \frac{1}{2} + \frac{1}{2^{k+1}}. \qquad (7)\end{aligned}$$

Let us consider $\psi_{best} = A_i^d$, and then normalized HD $h_d(\mathrm{mux}, \psi_{best}))$ between MUX and ψ_{best} is as:

$$h_d(\mathrm{mux}, \psi_{best}) = 1 - \Pr(o = A_i^d) = \frac{1}{2} - \frac{1}{2^{k+1}}. \qquad (8)$$

If $\psi_{best} = A_i^d \oplus 1$, then $h_d(\mathrm{mux}, \psi_{best}) = \frac{1}{2} + \frac{1}{2^{k+1}}$. Since $\bigoplus_{j=0}^{k-1} u_j A_j^s$ is always 0 or 1 whenever A_i^d is selected as output of MUX, for any instance of ψ_{best} as in Eq. (8), $h_d(\mathrm{mux}, \psi_{best}) = \frac{1}{2} \pm \frac{1}{2^{k+1}}$. Thus, the normalized nonlinearity value (cf. Eq. (5)) of (n,k)-MPUF is:

$$E_{(n,k)-\mathrm{MPUF}} = \frac{1}{2} - \frac{1}{2^{k+1}}. \qquad (9)$$

This fact suggests that we need to choose parameter k of MUX such that $E_{(n,k)-\mathrm{MPUF}} \to \frac{1}{2}$ as $\frac{1}{2^{k+1}} \to 0$. However, in case of (n,k)-MPUF, we cannot use arbitrarily large k value as it results in large number of APUFs for composition. For $k = 4$, $E_{\psi_{best}} = \frac{1}{2} - 0.03 = 0.47$, and its implies that (n,k)-MPUF with $k = 4$ has reasonably good nonlinearity.

4.3 Reliability

An interesting and useful fact about (n,k)-MPUF is that response to a given challenge depends only on a data input APUF and k selection input APUFs, which follows from the working principle of MUX. This helps to achieve relatively good reliability of MPUF, although the composition exploits many APUF instances. *This is one of the major advantages of MUX based composition over XOR based compositions, namely XOR APUF and LSPUF.*

5 Performance Metrics of MPUF

Typically, performance quality is measured using following metrics: uniformity, uniqueness, and reliability [22]. Since uniformity is equivalent to the balancedness property discussed in Section 4.1, we exclude its discussion in this section.

5.1 Uniqueness

Uniqueness property is used as estimate of differences in challenge-response behavior of PUF instance pairs [22]. The uniqueness metric, denoted by U, has a value $0 \leq U \leq 1$, and the ideal value U is 0.5, i.e., behavior of two arbitrary PUF instances of the same type are uncorrelated. Next, we derive an analytical expression for uniqueness property of (n,k)-MPUF.

Let us recall the fact that response o to a challenge $\mathbf{c}$ in (n,k)-MPUF depends on $(k+1)$ APUF instances. Without loss of generality and for simplicity of analysis, let us assume that uniqueness values of all APUFs are similar, and it is Q. Let P_1 and P_2 be two instances of (n,k)-MPUF, and their responses are denoted by o_{P1} and o_{P2}, respectively. The uniqueness metric Q_{M} of (n,k)-MPUF can be expressed as:

$$Q_{\mathrm{M}} = \Pr(o_{\mathrm{P1}} \neq o_{\mathrm{P2}}) = 1 - \Pr(o_{\mathrm{P1}} = o_{\mathrm{P2}}). \tag{10}$$

To estimate Q_{M}, we consider the following events based on primitive PUF responses, for a given challenge, in P_1 and P_2:

- **E1**: The $(k+1)$-bit responses due to $(k+1)$ APUFs contributing to response generation in MPUF instances P_1 and P_2 are equal; then, $\Pr(E1) = (1-Q)^{k+1}$.
- **E2**: There is at least 1-bit difference between k-bit responses generated by k selection input APUFs of MUXes in P_1 and P_2. The probability $\Pr(E2) = 1-(1-Q)^k$.
- **E3**: This is an extension of event **E2** with an additional constraint that outputs o_{P1} and o_{P2} are equal. This implies that two different data inputs of MUXes in P_1 and P_2 are selected for challenge $\mathbf{c}$, but responses of the selected data input APUFs are equal. Ideally, the probability that outputs of two different APUF instances being equal is $(1-Q)$. Then, $\Pr(E3) = (1-Q) \times (1-(1-Q)^k)$.

So, two MPUF instances P_1 and P_2 have similar outputs with probability: $\Pr(o_{\mathrm{P1}} = o_{\mathrm{P2}}) = \Pr(E1) + \Pr(E3) = (1-Q)$. Hence, according to Eq. (10), uniqueness of (n,k)-MPUF is: $Q_{\mathrm{M}} = 1-(1-Q) = Q$. In other words, for a given challenge $\mathbf{c}$, uniqueness of response depends on the probability that responses of data input APUFs selected in MPUF instances P_1 and P_2 are different. Thus, MPUF can achieve at least the uniqueness of primitive APUF.

5.2 Reliability

Reliability [22] is a major issue in case of PUF compositions. For instance, reliability values of XOR APUF and LSPUF reduce with increasing number of component APUFs. But, for improved security of APUF compositions, more primitive APUF instances should be employed. In case of MPUF (also holds for other MPUF variants to be discussed later), we show that its reliability is superior to reliability of XOR APUF. In this context, we have the following lemma:

Lemma 2. *The number of APUF present in (n,k)-MPUF is $x_{\mathrm{M}} = k + 2^k$. Though an MPUF employs x_{M} number of APUFs, reliability of response to a challenge $\mathbf{c}$ depends only on reliability values of $(k+1)$ APUFs. An (n,k)-MPUF instance is **at least as reliable** as $(k+1)$-XOR APUF.*

Proof. Without loss of generality and to simplify the following analysis, we assume that all APUFs used in the composition have similar reliability, and it is R, $0 \leq R \leq 1$. We now quantify reliability of (n,k)-MPUF and $(k+1)$-XOR APUF. First, we consider $(k+1)$-XOR APUF and denote its reliability by R_{X}. In case of $(k+1)$-XOR APUF, the response to a given challenge $\mathbf{c}$ is reliable when either responses of all $(k+1)$ APUFs are reliable or there are even number of APUFs with unreliable responses. So, we can compute reliability of $(k+1)$-XOR APUF as follows:

$$\begin{aligned} R_{\mathrm{X}} &= \sum_{\substack{i \in [0,k+1] \\ i \text{ is even}}} \binom{k+1}{i} (1-R)^i R^{k+1-i} \\ &= \frac{(R+(1-R))^{k+1} + (R-(1-R))^{k+1}}{2} \\ &= \frac{1}{2} + \frac{(2R-1)^{k+1}}{2}, \end{aligned} \tag{11}$$

as $[(a+b)^{k+1} + (a-b)^{k+1}] = 2 \times [$ sum of terms with even i in Binomial expansion of $(a+b)^{k+1}]$, where $a = R$ and $b = (1-R)$.

To estimate reliability of (n,k)-MPUF, we consider the following events based on the behavior of primitive APUFs for a given challenge $\mathbf{c}$:

- **E1**: The responses of k selection input APUFs and response of the selected data input APUF of MUX are reliable. The probability of this event is $\Pr(E1) = R^{k+1}$.
- **E2**: There is at least one APUF which is not reliable among k selection input APUFs of MUX. The probability of this event is $\Pr(E2) = 1 - R^k$.

- **E3**: This is an extension of **E2** with an additional constraint that output o of MPUF is reliable. Due to the event **E2**, a different data input APUF A_j^d (when expected data input APUF is $A_i^d, i \neq j$) will be selected. The probability that $A_j^d(\mathbf{c}) = A_i^d(\mathbf{c}), i \neq j$ is $\frac{1}{2}$. Thus, the probability $\Pr(E3) = (1-R^k)/2$.

So, the reliability of (n,k)-MPUF is:

$$R_{\mathrm{M}} = \Pr(E1) + \Pr(E3) = R^{k+1} + \frac{(1-R^k)}{2} = \frac{1}{2} + \frac{R^k(2R-1)}{2}. \quad (12)$$

Form Eqs. (11) and (12), we can write:

$$R_{\mathrm{M}} - R_{\mathrm{X}} = \frac{R^k(2R-1)}{2} - \frac{(2R-1)^{k+1}}{2} \geq 0 \Rightarrow R_{\mathrm{M}} \geq R_{\mathrm{X}},$$

as $(2R-1) \leq R$ and $(2R-1)^k \leq R^k$. This proves Lemma 2. □

6 Reliability Based Modeling of MPUF

In [15], Becker proposed a linear time modeling attack on XOR APUF, which seems to be the strongest implementation attack. In this attack, adversary exploits reliability information of challenges in model building, instead of corresponding responses. To the best of our knowledge, this attack is applicable (with certain modifications) to majority of existing strong delay PUF compositions, when PUF primitives themselves are not robust. The basic MPUF design is also vulnerable to this attack. In this section, we first demonstrate reliability based modeling attack on the basic MPUF as its security analysis, and then we discuss design philosophy of a robust MPUF design variant, called rMPUF.

Let us briefly describe Becker's attack on XOR APUF and then we discuss how it can be adopted for MPUF. Main objective of this attack is the model building of individual primitive APUF based on reliability information of random challenges applied to XOR APUF. Hence, this attack has linear time and data complexities. Reliability information h_i of challenge $\mathbf{c}_i$ is estimated based on the responses obtained from l measurements at a fixed operating condition. The h_i is defined in [15] as follows:

$$h_i = \left| \frac{l}{2} - \sum_{t=1}^{l} r_{i,t} \right|, \quad (13)$$

where $r_{i,t}$ is the response to $\mathbf{c}_i$ in t-th measurement. Note that h_i lies in $[0, l/2]$; $h_i = 0$ holds for challenge with completely unreliable response, and $h_i = l/2$ holds for challenge with completely reliable response. The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [15], [23] has been used to learn the individual APUF based on $(\mathbf{c}_i, h_i)$ pairs of XOR APUF. In CMA-ES, fitness of a APUF model $\mathbf{w}$ for parity vector $\mathbf{\Phi}_i$ (derived from challenge $\mathbf{c}_i$) is defined as:

$$\widetilde{h}_i = \begin{cases} 1, & \text{if } \langle \mathbf{w}, \mathbf{\Phi}_i \rangle > \epsilon \\ 0, & \text{if } \langle \mathbf{w}, \mathbf{\Phi}_i \rangle < \epsilon, \end{cases} \quad (14)$$

Algorithm 1 Modeling of MPUF using reliability information

Input:
i. Parameters n and k of an (n,k)-MPUF instance
ii. A set $\mathcal{D}$ of 3-tuple $(\mathbf{c}, o, h)$ where o and h are the response and reliability values of MPUF, respectively, for challenge c

Output:
i. Set $\mathcal{M}$ of APUF models corresponding to data and selection inputs

1: Using $\mathcal{D}$, build models for selection input APUFs $A_j^s, j \in [0:k-1]$ following the Becker's scheme discussed in [15] for XOR APUF. Let $M_j^s, j \in [0:k-1]$ denote a model of A_j^s, and models are built using reliability information h.
2: Partition set $\mathcal{D}$ into 2^k disjoint sets $\{\mathcal{D}_i | i \in [0:2^k-1]\}$ according to the predicted selection inputs values of MUX using $M_j^s, j \in [0:k-1]$. Partition $\mathcal{D}_i$ is corresponding to data input APUF A_i^d.
3: Build a model M_j^d for $A_j^d, j \in [0:2^k-1]$ using CRP set $\mathcal{D}_j$ and LR [5] based modeling of APUF. In this step, response o to challenge $\mathbf{c}$ is used instead of reliability information h.
4: **return** $\mathcal{M} = \{M_0^d, \ldots, M_{2^k-1}^d, M_0^s, \ldots, M_{k-1}^s\}$

where parameter ϵ should also be learned for each APUF using CMA-ES along with weight vector $\mathbf{w}$. For each APUF with n-bit challenge, number of parameters to be learned using CMA-ES is $n+2$. In case of a good APUF model $\mathbf{w}$, correlation between $h_0, \ldots, h_N$ and $\widetilde{h}_0, \ldots, \widetilde{h}_N$ would be high, where N denotes number of challenges required for modeling. According to Becker's modeling technique, the same challenge set can be reused for modeling of APUF instances in XOR APUF with multiple runs of CMA-ES; this is feasible as CMA-ES is a randomized algorithm. Thus, Becker's modeling technique does not produce the same model for XOR APUF in different runs of the algorithm, and time required for model building might be different for each model.

6.1 Reliability based Modeling of the Basic MPUF

The outline of reliability based modeling of MPUF is given in Algorithm 1. The input dataset $\mathcal{D}$ has 3-tuple $(\mathbf{c}_i, o_i, h_i)$ where h_i (cf. Eq. (13)) is the reliability information of response o_i (majority voted) to challenge $\mathbf{c}_i$ computed over l measurements of MPUF at a fixed operating condition. Like XOR APUF, selection input APUFs of MPUF have contributed to entire CRP space of MPUF. However, individual data input APUF has contributed to only $(2^{-k} \times 100)\%$ of MPUF CRP space. Thus, there is significantly less correlation between reliability of data input APUF and reliability of MPUF. This implies that the adversary can efficiently learn only k selection input APUFs of (n,k)-MPUF using reliability information. But this does not imply that MPUF is secure because models

of selection input APUFs can be used to predict selection inputs of MUX for a given challenge. Based on these predicted selection inputs, the adversary can partition set $\mathcal{D}$ into 2^k partitions $\{\mathcal{D}_0, \ldots, \mathcal{D}_{2^k-1}\}$ corresponding to 2^k data input APUFs $A_j^d, j \in [0 : 2^k - 1]$ of MPUF. Next, the adversary employs LR [5] approach to build model M_j^d for $A_j^d, j \in [0 : 2^k - 1]$ using $(\mathbf{c}_i, o_i) \in \mathcal{D}_j$.

In case of XOR APUF, all models are built independently of each other, which is not true for MPUF modeling. First selection input APUFs are modeled, and then data input APUFs are modeled based on these models. Thus, in case of MPUF, prediction accuracy of data input APUFs should be less than that of selection input APUFs. This results in a small reduction in overall prediction accuracy of MPUF. Let p_s be the probability that model of selection input APUF predicts its output correctly. For the sake of simplified worst-case analysis (from designer perspective), we assume that these probabilities are the same for all selection input APUFs. Let us assume that $p_{sa} = (p_s)^k$ be the probability that all selection inputs are predicted correctly, and this probability is an estimation of correct CRPs in $\mathcal{D}_i$ which are used for modeling of A_i^d.

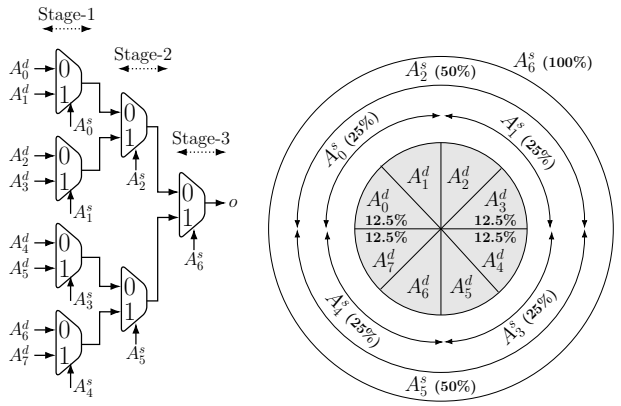
Let N_k^d be the number of CRPs (with $1-p_{sa}$ probability of noise in CRPs) required for building a model M_i^d of $A_i^d, i \in [0 : 2^k - 1]$ in (n, k)-MPUF. Then, $N_k^D = N_k^d \times 2^k$ denotes total number of CRPs required for modeling all data input APUFs. Let N_k^s be the number of CRPs required for modeling a selection input APUF, and the same set of CRPs can be reused for modeling other selection input APUFs. Since the CRPs used for modeling selection input APUFs can be reused for modeling of data input APUFs, data complexity of complete (n, k)-MPUF modeling is $N_k^M = \max\{N_k^D, N_k^s\}$. Typically, $N_k^s \gg N_k^d$, and if 2^k is not a large value, then $N_k^M = N_k^s$.

6.2 rMPUF: a Robust MPUF Variant

Following two facts can be observed from the reliability based modeling attack on MPUF:

1) Reliability based attack is not feasible on data input APUFs A_i^d as the individual contribution of each A_i^d on MPUF output is significantly less compared to the size of total CRP space.
2) Each selection input APUF A_j^s has contributed to entire CRP space of MPUF, and thus there is a significant correlation between reliability of MPUF and A_j^s outputs.

To make MPUF design robust against this attack, we need to ensure that each A_j^s contributes to a small subset of the entire CRP space like A_i^d. Less contribution on CRP space implies that the adversary needs more MPUF CRPs to build a high accuracy model of APUF, as reliability based modeling exploits unreliable CRPs in the training set. Based on this design philosophy, we propose an MPUF variant, as shown in Fig. 2a. We have termed this MPUF variant "rMPUF" as it is robust against *reliability* based modeling attack. An



(a) (n,3)-rMPUF (b) Impact of A_i^s on o in (n,3)-rMPUF

Fig. 2: (n, k)-rMPUF: (a) architectural overview of $(n, 3)$-rMPUF, and (b) impacts of selection (A_j^s) and data (A_i^d) input APUFs on CRP space of $(n, 3)$-rMPUF. The gray circle in (b) denotes the entire CRP space of $(n, 3)$-rMPUF, and this CRP space is divided into 2^3 equal size partitions corresponding to each data input APUF.

(n, k)-rMPUF consists of k stages of 2:1 MUXes, where selection inputs of 2:1 MUXes are generated by independent APUFs. On the other hand, in the basic (n, k)-MPUF, selection inputs of all 2:1 MUXes belonging to a specific stage are generated by a single APUF instance (cf. Fig. 1b). Thus an (n, k)-rMPUF employs more APUF instances compared to an (n, k)-MPUF.

In case of (n, k)-rMPUF, for a given challenge, response of an $A_i^d, i \in [0 : 2^k - 1]$ propagates to output o of rMPUF. Selection of A_i^d depends on concatenated responses of A_j^s's lying on the path from A_i^d to o. In an (n, k)-rMPUF, there are $k+1$ APUFs, including the data input APUF, on the path from a particular data input to MPUF output o (cf. Fig. 2a). Thus, like MPUF, reliability of (n, k)-rMPUF is similar to $(k+1)$-XOR APUF. Nonlinearity and balancedness properties of (n, k)-rMPUF are the same as (n, k)-MPUF.

Next, we analyze robustness of rMPUF against reliability based modeling attack. By "robustness against modeling attack", we mean that an adversary needs more CRPs to execute the attack. Unlike MPUF where model building of all selection input APUFs are performed simultaneously using the same CRP training set, selection input APUFs in rMPUF lying on a path from data input to o are modeled sequentially. We explain the modeling strategy using an example of $(n, 3)$-rMPUF as shown in Figs. 2a and 2b. The adversary first builds a model for A_6^s using reliability of MPUF responses, and this is feasible as contribution of A_6^s on rMPUF CRP space is 100% (cf. Fig. 2b). Let $\mathcal{S}_3$ be a set of CRPs required to build a model M_6^s for A_6^s and $N = |\mathcal{S}_3|$. Once M_6^s is built, the adversary partitions $\mathcal{S}_3$ into two sets $\mathcal{S}_3^0$ and $\mathcal{S}_3^1$ corresponding to the two predicted response classes 0 and 1, respectively. Next, $\mathcal{S}_3^0$ and $\mathcal{S}_3^1$ are used in model building of A_2^s and A_5^s, respectively,

but these sets do not have sufficient number of CRPs for reliability based model building. At least another set $\mathcal{S}_2$ of N random CRPs is required. Like $\mathcal{S}_3$, set $\mathcal{S}_2$ is to be partitioned into $\mathcal{S}_2^0$ and $\mathcal{S}_2^1$ based on the predicted response values of M_6^s. The sets $\mathcal{S}_3^0 \cup \mathcal{S}_2^0$ and $\mathcal{S}_3^1 \cup \mathcal{S}_2^1$ are used for modeling A_2^s and A_5^s, respectively. Once models M_2^s, M_5^s, M_6^s are built, the adversary follows a similar approach for modeling of $A_0^s, A_1^s, A_3^s, A_4^s$ based on reliability information. After successful modeling of all selection input APUFs, the adversary builds models for data input APUFs by following the same approach used for MPUF modeling in Algorithm 1.

For an (n,k)-rMPUF, total number of distinct CRPs required for modeling all selection input APUFs is:

$$N_k^S = N_k^s + \sum_{j=1}^{k-1} \left(\frac{N_k^s}{2} \times 2^j \right) = 2^{k-1} N_k^s, \tag{15}$$

where N_k^s denotes number of CRPs required for reliability based modeling of a selection input APUF instance in (n,k)-rMPUF. The value of N_k^s in (n,k)-rMPUF is similar to data complexities of (n,k)-MPUF and k-XOR APUF for reliability based modeling, and it increases linearly in k. For example, $N_3^S = N_3^s + N_3^s + 2N_3^s = 4N_3^s$ for $(n,3)$-rMPUF, $N_4^S = N_4^s + N_4^s + 2N_4^s + 4N_4^s = 8N_4^s$ for $(n,4)$-rMPUF, and $N_3^s < N_4^s$. Since modeling of data input APUFs are performed based on the responses instead of reliability information, less number of CRPs is required for this purpose. Note that adversary can reuse N_k^S CRPs in the modeling of data input APUFs. Hence, data complexity of complete (n,k)-rMPUF modeling is N_k^S, and this value increases exponentially in k. On the other hand, data complexity of k-XOR APUF is linear in k. This implies that rMPUF is more robust against modeling attack than XOR APUF. As we mentioned earlier, advantage of rMPUF is that it can achieve a good balance between reliability and security at the cost of hardware overhead as number of APUFs used in composition is large. However, in practice, this balance is not feasible for XOR APUF.

Although the APUFs in XOR APUF can be learned parallely to reduce attack time, this parallel approach is only applicable to the selection input APUFs in a specific stage of an (n,k)-rMPUF. The selection input APUFs belonging to different stages of rMPUF must be learned sequentially, and thus, time required for modeling an rMPUF is longer than XOR APUF.

Compared to the $(n,3)$-MPUF, $(n,3)$-rMPUF requires four extra APUFs which seems acceptable in practice, and hence we consider $(n,3)$-rMPUF as the preferred rMPUF variant in practice. In (n,k)-rMPUF, any response depends on the k selection input APUFs, and it is intuitive that number of CRPs required for reliability based modeling of a selection input APUF is approximately similar as that of k-XOR APUF modeling. Later in Section 9.3, we validate this observation using simulation results for $(64,3)$-rMPUF. In reference to the results from Becker's paper [15, Table 2], the reliability based modeling of 128-bit 3-XOR APUF requires $\approx 100 \times 10^3$ CRPs and for $(128,3)$-rMPUF, it should be $\approx 400 \times 10^3$ (as theoretically estimated in Eq. (15)) which is similar to the data complexity of a 10-XOR APUF. **The interesting fact is that the reliability of $(n,3)$-rMPUF is similar to reliability of 4-XOR APUF which is significantly lesser than reliability of 10-XOR APUF. Thus, $(n,3)$-rMPUF achieves the same security level of 10-XOR APUF with significantly higher reliability at the cost of additional five APUF instances.**

7 Linear Cryptanalysis of MPUF

In this section, we first discuss linear cryptanalysis of the basic MPUF, and then propose a MPUF variant (cMPUF) which is robust against this cryptanalysis attack.

7.1 Linear Cryptanalysis (LC)

Adversary's objective here is to find a linear approximation of MPUF. From Section 4.2, it can be observed that the probability of MPUF output being identical to the output of an arbitrary data input APUF is:

$$p_{match} = \Pr(o = A_i^d) = \frac{1}{2} + \frac{1}{2^{k+1}}. \tag{16}$$

In other words, E_o drifts from its ideal value 0.5 (cf. Eq. (5)), and makes linear cryptanalysis feasible for small k value (cf. Table 1). Since adversary does not have access to output of data input APUF A_i^d, she needs to build a model of APUF based on CRPs of MPUF. In this scenario, she needs to deal with noisy (erroneous) CRPs during model building, and the probability that a CRP is noisy:

$$\epsilon = 1 - p_{match} = \frac{1}{2} - \frac{1}{2^{k+1}}. \tag{17}$$

Table 1 shows values of p_{match} and ϵ with varying k values. Note that $o = A_i^d$ is not the only approximation that satisfy p_{match} and ϵ expression as stated in Eqs. (16) and (17). Any linear expression in the form of ψ_{best} in Eq. (6) has the same p_{match} and ϵ values. This implies that amount of noisy CRPs increases with increasing value of k.

It should be an obvious choice to the adversary to use a ψ_{best} expression that can be efficiently modeled using noisy CRPs with higher degree of prediction accuracy (i.e., expected accuracy should be greater than 99%). If adversary cannot find any such ψ_{best}, then linear approximation based attack will not be feasible. It can be observed from Eq. (6) that ψ_{best} expression composes of either a single data input APUF only or XOR of a data

TABLE 1: p_{match} and ϵ for (n,k)-MPUF with different k

k	1	2	3	4
p_{match} (%)	75	62.5	56.25	53.13
ϵ (%)	25	37.5	43.75	46.87
m†	3	6	11	20

† No. of APUFs used in MPUF.

input APUF and a few selection input APUFs. If ψ_{best} is a XOR APUF, then computational complexity of model building would be significantly higher while number of noisy training CRPs is large.

This linear approximation technique can be used to develop a modeling attack. The basic idea is to build a model for each primitive APUF using linear approximation as follows:

- $o = A_i^d$ for modeling $A_i^d, i \in [0 : 2^k - 1]$, and
- $o = A_0^d \oplus A_j^s$ for modeling of $A_j^s, j \in [0 : k - 1]$.

Next we propose an MPUF variant (called cMPUF) to prevent $o = A_i^d$ linear approximation attack, as modeling of an APUF is comparatively efficient with noisy CRPs compared to 2-XOR APUF.

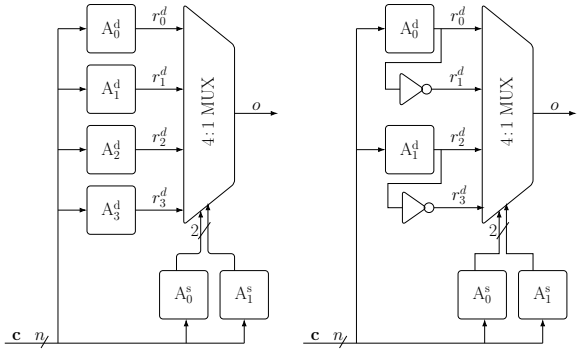
7.2 cMPUF: a Robust MPUF Variant Against LC

We aim to prevent linear approximation like $o = A_i^d$ by modifying the data input connection pattern of basic MPUF design as: **half of the data inputs of MUX are complement of the other data inputs** (cf. Figure 3b for $k = 2$). We have already introduced this connection pattern in Section 4.1. We have termed this modified MPUF design as "cMPUF" due to its *complemented* data input. Here, our objective with cMPUF is to introduce 50% noise (i.e., $\Pr(o = A_i^d) = 0.5$) in training CRPs when the adversary attempts to build a model of APUF using cMPUF's CRPs, for any $k > 1$. The reader can find similar philosophy being used in noise bifurcation based PUF protocol in [24]. It withstands the adversary to take advantage of poor nonlinearity of MUX in linear approximation attack using single APUF. We compute the probability $\Pr(o = A_i^d)$ for cMPUF as:

$$\begin{aligned}
\Pr(o = A_i^d) &= \Pr(A^s = i)\Pr(o = A_i^d | A^s = i) \\
&\quad + \Pr(A^s \neq i)\Pr(o = A_i^d | A^s \neq i) \\
&= \frac{1}{2^k} + \sum_{\substack{j \in [0, 2^k - 1] \\ j \neq i}} \Pr(A^s = j)\Pr(o = A_i^d | A^s = j) \\
&= \frac{1}{2^k} + \frac{1}{2^k} \sum_{\substack{j \in [0, 2^k - 1] \\ j \neq i}} \Pr(o = A_i^d | A^s = j) \\
&= \frac{1}{2^k} + \frac{1}{2^k}[(2^k - 2) \times \frac{1}{2}] = \frac{1}{2}. \qquad (18)
\end{aligned}$$

In the last line of Eq. (18), we have used $(2^k - 2) \times \frac{1}{2}$, as we need to exclude the case $j = i$ and another value of j where complement of A_i^d is connected. If $A_j^d = \bar{A}_i^d$ for some j, then $\Pr(o = A_i^d | A^s = j) = 0$. Hence, we use this data input connection pattern in cMPUF to make it robust against easier form of linear approximation attack. The expression of ψ_{best} that provides minimum HD with cMPUF has following form:

$$\psi_{best}(\mathbf{c}; i, \mathbf{u}) = A_i^d(\mathbf{c}) \oplus \left(\bigoplus_{j=0}^{k-1} \mathbf{u}[j] A_j^s(\mathbf{c}) \right), \qquad (19)$$



(a) (n,2)-MPUF (b) (n,2)-cMPUF

Fig. 3: Example of MPUF variants: (a) the basic $(n, 2)$-MPUF and (b) $(n, 2)$-cMPUF.

where $\mathbf{u}[j] \in \{0, 1\}$, but all $\mathbf{u}[j]$ cannot be 0 together. This implies that for linear approximation attack, the adversary needs to build a model for at least 2-XOR APUF using cMPUF's CRPs, which is infeasible for (n, k)-cMPUF with $k >= 3$ due to large amount noise. So, cMPUF is a preferred MPUF variant than the basic MPUF to prevent linear approximation attack.

7.3 Nonlinearity Comparison: MPUF vs. cMPUF

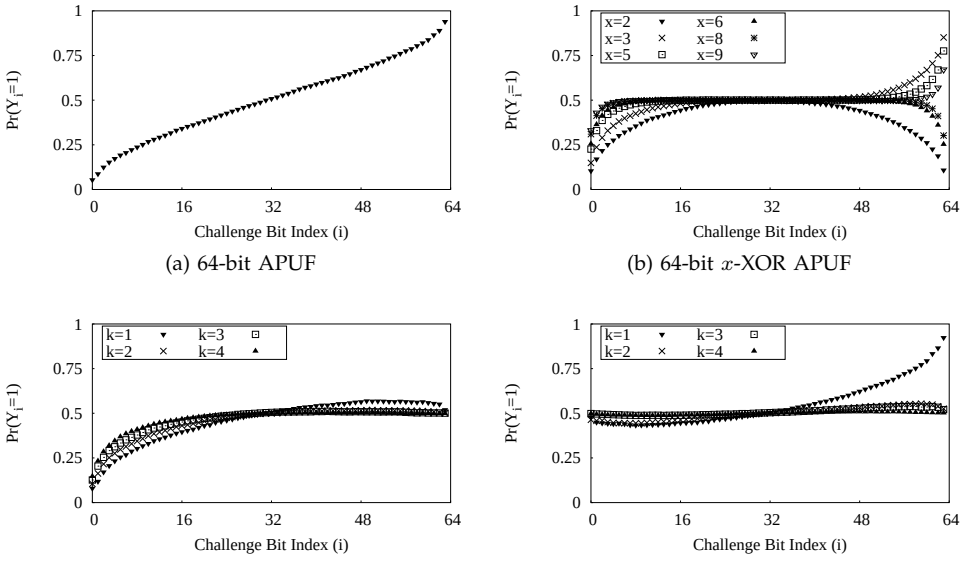
The analysis presented above establishes that cMPUF can only prevent linear approximation of the form $o = A_i^d$. One obvious question in this context is: *do the (n, k)-MPUF and (n, k)-cMPUF have the same nonlinearity for a given k?* Note that $(n, 1)$-cMPUF is nothing but a 2-XOR APUF, and for $k > 1$, we have the following lemma:

Lemma 3. *The nonlinearity of (n, k)-MPUF and $(n, k+1)$-cMPUF are the same, i.e., $E_{(n,k)-\text{MPUF}} = E_{(n,k+1)-\text{cMPUF}}$ where E represents the normalized* HD *between a given MPUF variant and any affine/linear function.*

Proof. According to Eq. (19), the best linear approximation for cMPUF can be achieved using linear expression $A_i^d \oplus A_j^s$ for some $i \in [0 : 2^k - 1]$ and $j \in [0 : k - 1]$. For each assignment of $A_j^s, j \in [0 : k - 1]$, one data input APUF A_i^d will be selected when $\mathbf{r}^s = i$. Due to complemented data input connection of cMPUF, output of cMPUF can be either A_i^d or $\bar{A}_i^d$. Let us consider linear expression $A_0^d \oplus A_j^s$. Since A_0^d and $\bar{A}_0^d$ are connected to two different MUX inputs in cMPUF, an adversary can select an A_j^s such that:

$$A_j^s = \begin{cases} 0, & \text{for } A_0^d \\ 1, & \text{for } \bar{A}_0^d. \end{cases} \qquad (20)$$

Thus, if cMPUF outputs A_0^d, then $A_0^d \oplus A_j^s = A_0^d \oplus 0 = A_0^d$ also holds. Similarly, if cMPUF outputs $\bar{A}_0^d$, then $A_0^d \oplus A_j^s = A_0^d \oplus 1 = \bar{A}_0^d$ holds. Whatever be the data input connection pattern of cMPUF, for $(A_0^d, \bar{A}_0^d)$, there exists an A_j^s that satisfies condition mentioned in Eq. (20)). If we



(a) 64-bit APUF (b) 64-bit x-XOR APUF

(c) $(64, k)$-MPUF (d) $(64, k)$-MPUF with improved SAC

Fig. 4: SAC properties of APUF, XOR APUF and MPUF. The $\Pr(Y_i = 1)$ is the output transition probability of PUF while i-th challenge bit is complemented. These results are based on the Matlab simulation of APUF, XOR APUF and the basic MPUF.

assume that A^s_x satisfies this condition for A^d_0 in (n, k)-cMPUF, then:

$$\Pr(o = A^d_0 \oplus A^s_x) = \frac{2}{2^k} + \frac{2^k - 2}{2^k} \times \frac{1}{2} = \frac{1}{2} + \frac{1}{2^k}, \text{ and}$$
$$E_{(n,k)-\text{cMPUF}} = 1 - \Pr(o = A^d_0 \oplus A^d_x) = \frac{1}{2} - \frac{1}{2^k}.$$

Hence it is proved that $E_{(n,k)-\text{MPUF}} = E_{(n,k+1)-\text{cMPUF}}$ (cf. Eq. (9)). □

From this discussion, it is evident that all possible connection patterns for complemented data input APUFs are equivalent with respect to their nonlinearity properties. Hence, in the rest of the paper, we use the following connection pattern as default pattern for (n, k)-cMPUF:

$$A^d_0 \to 0, \bar{A}^d_0 \to 1,$$
$$\ldots,$$
$$A^d_i \to i, \bar{A}^d_i \to (i + 1),$$
$$\ldots,$$
$$A^d_{2^{k-1}-1} \to (2^k - 2), \bar{A}^d_{2^{k-1}-1} \to (2^k - 1),$$

where $A^d_i \to x$ implies that data input APUF A^d_i in connected to x-th data input of MUX in (n, k)-cMPUF.

8 Strict Avalanche Criterion of MPUF

In the context of statistical attack on PUFs, an adversary exploits correlation among CRPs explicitly. The *Strict Avalanche Criterion* (SAC) [7], [25] of a PUF is important to make it robust against statistical attacks. A PUF instance P is said to satisfy SAC property if its output changes with probability 0.5, whenever a single challenge bit is complemented. Let us consider a random variable Y_i defined as:

$$Y_i = \begin{cases} 1 & \text{if } P(\mathbf{c}) \oplus P(\mathbf{c} \oplus \mathbf{e}_i) = 1 \\ 0 & \text{if } P(\mathbf{c}) \oplus P(\mathbf{c} \oplus \mathbf{e}_i) = 0 \end{cases} \quad (21)$$

where $\mathbf{c}$ is a n-bit challenge to PUF P, $\mathbf{e}_i \in \{0, 1\}^n$ and only the i-th bit of $\mathbf{e}_i$ is one. Ideally, $\Pr(Y_i = 1) = 0.5$ should hold for all $i \in [0 : n - 1]$ for n-bit challenge to satisfy SAC property. If a PUF fails to achieve that, then adversary can generate related CRPs from a set of known CRPs.

Since APUF is used as primitive in MPUF variants, we first discuss SAC property of APUF. Figure 4a depicts SAC property of 64-bit APUF. Note that $i = 0$ implies LSB (least significant bit) of challenge, and $i = n - 1$ is MSB (most significant bit) of challenge. APUF achieves good SAC property when $i \to \frac{n}{2}$, and it becomes poor when either $i \to 0$ or $i \to (n - 1)$.

The SAC property for the basic MPUF is reported in Fig. 4c based on Matlab simulation results (refer to Section 9.1 for simulation setup). From SAC properties of APUF (cf. Fig. 4a) and XOR APUF (cf. Fig. 4b), it can be observed that SAC values, i.e., $\Pr(Y_i = 1)$, are relatively poor for challenge bits towards both left and right ends of challenge $\mathbf{c}$. On the other hand, in case of MPUF, $\Pr(Y_i = 1) \to 0.1$ when $i \to 0$, and thus, only least significant challenge bits show poor SAC property. This nature of SAC property for MUPF can be explained based on SAC property of APUF. Let us consider the following two extreme cases for APUF: $\Pr(Y_0 = 1) \approx 0$ and $\Pr(Y_{n-1} = 1) \approx 1$. For $i = 0$, value of A^s remains the same with very high probability for challenge pair

$(\mathbf{c}, \mathbf{c} \oplus \mathbf{e}_0)$, and same data input APUF will be selected for both challenges. Hence, MPUF also produces the same output for $\mathbf{c}$ and $\mathbf{c} \oplus \mathbf{e}_0$ with high probability. However, for $i = n-1$, A^s produces different values for challenge pair $(\mathbf{c}, \mathbf{c} \oplus \mathbf{e}_{n-1})$ with probability approximately 1, and it results in different response values for MPUF with probability approximately 0.5, as two different data input APUFs are selected for challenges $\mathbf{c}$ and $\mathbf{c} \oplus \mathbf{e}_{n-1}$.

The SAC property of LSPUF [25] was improved using a special XOR input network. Now, we show that the same can be achieved for MPUF without any input network. Let $\mathbf{c}_{\text{rev}}$ be reverse of challenge $\mathbf{c}$. If we apply challenge $\mathbf{c}$ and its reverse challenge $\mathbf{c}_{\text{rev}}$ alternatively to selection input APUFs in MPUF, for example $A^s(\mathbf{c}) = A^s_0(\mathbf{c}_{\text{rev}})|A^s_1(\mathbf{c})|A^s_2(\mathbf{c}_{\text{rev}})|A^s_3(\mathbf{c})$ for $k = 4$, then outputs of $A^s(\mathbf{c})$ and $A^s(\mathbf{c} \oplus \mathbf{e}_i)$ would be different with high probability, for all $i \in [0 : n-1]$. Hence, two different data input APUFs would be selected for $\mathbf{c}$ and $\mathbf{c}_{\text{rev}}$. It results in improved SAC property for (n, k)-MPUF with $k > 1$ as shown in Fig. 4d. A similar SAC property can also be observed for cMPUF and rMPUF.

9 Simulation Results

We currently do not have access to ASIC fabrication. Since APUFs exhibit poor uniqueness property on FPGAs as observed by us and reported in multiple publications previously [26], [27], we do not report FPGA-based results here. As an alternative, we perform all PUF characterizations and analyses based on simulation of the proposed PUF circuits in Matlab, similar to the approach adopted in [5], [15].

9.1 Simulation Setup

In Matlab simulation, we have considered that all delay components of APUF are independent and identically distributed and follow a normal distribution with $\mu = 10$ and $\sigma = 0.05$. To observe the effect of experimental noise on simulated APUF behavior (that happens in real PUF due to temperature and supply voltage variations), we have used an additive noise which follows a normal distribution $\mathcal{N}(0, \sigma^2_{\text{noise}})$ [15]. In the presence of noise, each delay component of APUF follows $\mathcal{N}(10, \sigma^2 + \sigma^2_{\text{noise}})$. To control reliability of APUF as well as MPUF, we have employed the following relationship between σ and σ_{noise}: $\sigma_{\text{noise}} = \alpha\sigma$, where $0 \leq \alpha \leq 1$. Both APUF and MPUF were treated to be 100% reliable when $\alpha = 0$. We have simulated (n, k)-MPUF for $k = 3, 4$, (n, k)-cMPUF for $k = 4, 5$, and (n, k)-rMPUF for $k = 3$ with different challenge size $n = 64, 128$. For each MPUF variant, we simulated 50 instances. Each instance was evaluated 25 times with 1×10^5 challenges for three different noise levels $\sigma_{\text{noise}} = \alpha\sigma$ with $\alpha = 1/2, 1/20, 1/80$, to estimate their performance metrics—uniformity, uniqueness and reliability.

TABLE 2: Performance metrics (%) of simulated APUFs

n	$\alpha^\dagger$	Uniformity (Avg.,Std.)	Uniqueness (Avg.,Std.)	Reliability (Avg.,Std.)
64	1/2	(50.34,3.58)	(50.04,3.96)	(96.03,0.46)
	1/20	(50.33,3.62)	(50.05,3.99)	(99.60,0.05)
	1/80	(50.33,3.61)	(50.05,3.99)	(99.90,0.01)
128	1/2	(50.43,2.58)	(50.00,2.81)	(96.05,0.32)
	1/20	(50.42,2.60)	(50.01,2.83)	(99.60,0.03)
	1/80	(50.42,2.60)	(50.01,2.83)	(99.90,0.01)

$^\dagger \sigma_{\text{noise}} = \sigma\alpha$, where σ is the standard deviation of delay distribution of delay component.

9.2 Results for Performance Metrics

The performance metrics of primitive APUFs (64-bit and 128-bit) used in MUX based compositions are provided Table 2. Table 3 depicts of performance metrics for MPUF, cMPUF and rMPUF. For each performance metric, we should consider both average (Avg.) and standard deviation (Std.), as average value might not provide an good estimation when values of a given metric for different PUF instances are significantly different from each other.

From Table 3, it can be observed that both MPUF and cMPUF have similar performance metrics. Since nonlinearity of (n, k)-MPUF and $(n, k+1)$-cMPUF are same, for the sake of security against linear approximation attack, $(n, k+1)$-cMPUF is preferable to an application engineer. But reliability of (n, k)-MPUF is slightly better than that of $(n, k+1)$-cMPUF, as $(n, k+1)$-cMPUF has employed one extra APUF for selection input of MUX. Performance metrics of rMPUF is also similar to other MPUF variants with sightly higher standard deviation for each metric. One reason might the use of independent selection input APUFs. In Table 3, reliability results of MPUF variants are compared with reliability values of $(k+1)$-XOR APUF and x-XOR APUF, where x is the number of APUFs employed in compositions. It was observed that reliability of MPUF variants are better than that of $(k+1)$-XOR APUF and significantly better than x-XOR APUF. From Tables 2 and 3, it can be observed that uniformity and uniqueness of MPUF variants are at least as that of primitive APUFs. Due to composition of APUFs, only reliability is reduced.

9.3 Results for Reliability based Modeling of MPUF Variants

As discussed in Section 6.1, the basic MPUF as well as cMPUF variant are not secure against reliability based modeling attack. Table 4 depicts modeling accuracy values and required number of CRPs (data complexity) for each selection input APUF belonging to $(64, 4)$-MPUF with different reliability levels. A same set of CRPs was used to learn all selection input APUFs. The number of CRPs required for modeling a data input APUF is significantly less compared to a selection input APUF, as data input APUFs are modeled based on CRPs, without any reliability information. For modeling of data input APUFs, we have reused the CRPs collected for

TABLE 3: Performance metrics (%) of simulated (n, k)-MPUF/cMPUF/rMPUF

PUF	n	k	$\alpha^\star$	Uniformity (Avg.,Std.)	Uniqueness (Avg.,Std.)	Reliability (Avg.,Std.)			$x^\dagger$
						(n,k)-MPUF Variant	(k+1)-XOR APUF	x-XOR APUF	
MPUF	64	3	1/2	(50.82,3.32)	(50.04,1.43)	(86.24,0.63)	(77.47,1.22)	(53.60,0.50)	11
			1/20	(50.33,2.72)	(50.02,0.61)	(98.91,0.05)	(98.28,0.10)	(94.62,0.23)	
			1/80	(50.33,2.71)	(50.02,0.61)	(99.74,0.01)	(99.59,0.03)	(98.82,0.05)	
		4	1/2	(49.57,2.03)	(50.01,0.61)	(83.57,0.59)	(71.99,1.07)	(50.16,0.09)	20
			1/20	(49.80,1.67)	(50.01,0.32)	(98.67,0.06)	(97.82,0.11)	(89.40,0.35)	
			1/80	(49.79,1.66)	(50.01,0.32)	(99.69,0.01)	(99.49,0.02)	(97.79,0.06)	
	128	3	1/2	(49.68,2.08)	(50.02,0.64)	(86.40,0.48)	(77.83,0.91)	(53.74,0.40)	11
			1/20	(49.79,1.82)	(50.01,0.39)	(98.92,0.04)	(98.31,0.08)	(94.69,0.15)	
			1/80	(49.79,1.81)	(50.01,0.39)	(99.74,0.01)	(99.60,0.02)	(98.84,0.04)	
		4	1/2	(49.87,2.24)	(50.00,0.39)	(83.61,0.36)	(72.13,0.74)	(50.15,0.08)	20
			1/20	(49.95,1.73)	(50.00,0.21)	(98.68,0.04)	(97.85,0.08)	(89.47,0.25)	
			1/80	(49.95,1.72)	(50.00,0.20)	(99.70,0.01)	(99.50,0.02)	(97.80,0.05)	
cMPUF	64	4	1/2	(49.80,2.27)	(49.99,0.23)	(83.76,0.63)	(71.80,0.89)	(52.64,0.43)	12
			1/20	(49.86,1.77)	(50.00,0.16)	(98.69,0.06)	(97.82,0.08)	(94.09,0.25)	
			1/80	(49.86,1.76)	(50.00,0.16)	(99.69,0.01)	(99.48,0.02)	(98.71,0.05)	
		5	1/2	(50.25,2.09)	(50.00,0.18)	(78.82,0.70)	(64.85,1.15)	(50.13,0.08)	21
			1/20	(50.23,1.64)	(50.00,0.13)	(98.10,0.08)	(96.71,0.17)	(87.09,0.29)	
			1/80	(50.23,1.62)	(50.00,0.13)	(99.55,0.02)	(99.22,0.04)	(97.12,0.08)	
	128	4	1/2	(49.65,1.83)	(50.00,0.14)	(83.78,0.43)	(71.83,0.98)	(52.66,0.23)	12
			1/20	(49.75,1.46)	(50.00,0.10)	(98.69,0.04)	(97.82,0.11)	(94.12,0.17)	
			1/80	(49.74,1.45)	(50.00,0.10)	(99.70,0.01)	(99.49,0.03)	(98.72,0.04)	
		5	1/2	(49.89,1.34)	(50.00,0.11)	(78.84,0.40)	(64.85,0.64)	(50.12,0.08)	21
			1/20	(49.90,1.03)	(50.00,0.10)	(98.11,0.05)	(96.73,0.10)	(87.05,0.20)	
			1/80	(49.91,1.02)	(50.00,0.10)	(99.55,0.01)	(99.23,0.03)	(97.11,0.05)	
rMPUF	64	3	1/2	(50.01,4.40)	(49.88,1.02)	(85.10,0.41)	(78.67,0.78)	(60.25,0.45)	15
			1/20	(50.04,3.80)	(49.95,0.69)	(98.67,0.07)	(98.02,0.11)	(92.93,0.17)	
			1/80	(50.02,3.81)	(49.95,0.69)	(99.68,0.02)	(99.50,0.04)	(98.14,0.08)	
	128	3	1/2	(49.60,2.24)	(49.98,0.62)	(85.02,0.32)	(78.74,0.57)	(60.28,0.35)	
			1/20	(49.58,1.98)	(49.99,0.41)	(98.66,0.05)	(98.01,0.10)	(92.95,0.14)	
			1/80	(49.59,1.97)	(49.99,0.41)	(99.68,0.02)	(99.50,0.03)	(98.13,0.05)	

† x represents the number of APUFs used in the MPUF or rMPUF. * $\sigma_{\text{noise}} = \alpha\sigma$, where $0 \leq \alpha \leq 1$.

TABLE 4: Modeling accuracy of selection input APUFs in $(64, 4)$-MPUF

α	# CRPs	Rel. of MPUF (%)	Acc. of APUF (%)
1/2	0.8×10^5	83.6	95.60 – 96.80
1/20	1.4×10^5	98.7	99.59 – 99.62
1/80	2.8×10^5	99.7	99.88 – 99.90

Note Rel.—Reliability Acc.—Prediction Accuracy

TABLE 5: Modeling accuracy of APUFs in k-XOR APUF

k	α	# CRPs	Rel. of XOR APUF (%)	Acc. of APUF (%)
4	1/2	0.8×10^5	79.55	96.55 – 96.95
	1/20	1.4×10^5	98.32	99.69 – 99.70
	1/80	1.7×10^5	99.60	99.90 – 99.92
10	1/2		53.93	90 – 91
	1/20	5.5×10^5	94.11	99.60 – 99.67
	1/80		98.64	99.92 – 99.90

Note Rel.—Reliability Acc.—Prediction Accuracy

modeling of selection input APUFs. This implies that the number of CRPs required for a successful reliability based modeling of $(64, 4)$-MPUF is similar to 4-XOR APUF (cf. Table 5). Thus, the basic $(64, 4)$-MPUF design is vulnerable to reliability based modeling attack even though it has exploited 2^k extra APUFs compared to k-XOR APUF. This implies that data input APUFs do not play any significant role to improve robustness against reliability based modeling attack.

Table 6 shows a range of prediction accuracy values for a partial set of selection input APUFs used in $(64, 3)$-rMPUF (cf. Fig. 2a). We have evaluated only this rMPUF instance as its resource overhead is in practical range. The only three selection input APUFs belonging to *Stage-3* (A_6) and *Stage-2* (A_2 and A_5) of $(64, 3)$-rMPUF are modeled successfully. There are no successful models for other selection input APUFs belonging to *Stage-1* (A_0, A_1, A_3, and A_4) when total number of training CRPs is 6×10^5 and 30 different runs of CMA-ES are used. We have considered three different levels of noise in APUF simulation. From results of 64-bit 10-XOR APUF (cf. Table 5) and $(64, 3)$-rMPUF, it can be observed that there is no successful model for rMPUF even using more training CRPs than that are used for successful modeling of a 10-XOR APUF. Thus, $(64, 3)$-rMPUF is at least as robust as 10-XOR APUF. Another interesting advantage of $(64, 3)$-rMPUF is that its reliability level is similar to 4-XOR APUF (cf. Table 3) which is significantly higher than reliability level of 10-XOR APUF.

Table 7 shows the two partitions of 6×10^5 CRPs corresponding to outputs 0 and 1 of the model of A_6. It is very difficult to have equal size partitions even in case of simulation, and it implies that adversary needs more CRPs to model an APUF with smaller partition. This results in increase of total number of CRPs required to build a successful model for rMPUF. In model building of an selection input APUF, we have used partially overlapped CRPs in different runs of CMA-ES. For this reason, the reader can observe that number of CRPs used for each APUF modeling (cf. Table 6) is lesser than the size of CRP partition assigned for modeling corresponding APUF (cf. Table 7).

The reader might think that how an adversary can estimate the accuracy values of APUFs as reported in Table 6. It is true that the adversary cannot compute these values, and she has to select APUF models based on fitness value the model generated by CMA-ES. Typically, a APUF model with best fitness value implies best prediction accuracy in CMA-ES. Since we are the designer and performing security evaluation of the design, we have access to some secret information, such as outputs of each APUF instances used in PUF composition, to

TABLE 6: Modeling accuracy of selection input APUFs in $(64, 3)$-rMPUF

α	Total # CRP	#CRPs and Acc. of Selection Input APUF Models					
		A_2		A_5		A_6	
		Acc. (%)	# CRP	Acc. (%)	# CRP	Acc. (%)	# CRP
1/2	6×10^5	$90 - 91$	1.9×10^5	$94.93 - 95.43$	2.39×10^5	$94.95 - 95.56$	1×10^5
1/20		$99.59 - 99.64$	2.05×10^5	$99.66 - 99.68$	2.68×10^5	$99.58 - 99.61$	1.5×10^5
1/80		$99.82 - 98.87$	2.69×10^5	$99.89 - 99.96$	3.01×10^5	$99.87 - 99.90$	3×10^5

Note Acc.—Prediction Accuracy

TABLE 7: Two partitions of 6×10^6 training CRPs based on the model of A_6

α	$A_6 = 0$ A_2	$A_6 = 1$ A_5
1/2	258479	341521
1/20	264630	335370
1/80	264985	335015

TABLE 8: Modeling accuracy of data input APUFs built based on CRPs of $(64, k)$-MPUF with 100% reliability

k	# CRPs	Acc. of APUF (%)	$\Pr(o = A_i^d) \times 100$
1	2×10^5	$70 - 95$	$70 - 76$
2		$63 - 78$	$64 - 67$
3		$55 - 69$	$60 - 65$
4		$51 - 58$	$56 - 59$

Note Acc.—Prediction Accuracy

evaluate security level of the design. When CMA-ES generates a model of selection input APUF, we have checked its predicted responses with responses of all selection input APUFs. This approach helps us to quickly verify whether modeling of all APUF instances is feasible or not, and what are the corresponding prediction accuracy values. Thus, the adversary needs significantly long time and large number of CRPs to build a successful model for rMPUF than that are required by a security evaluation engineer having access to some secret information. For this reason, we have not reported time required for attacking rMPUF and XOR APUF based on reliability based information.

9.4 Results for Cryptanalysis of MPUF Variants

Now we present some results for linear approximation attack discussed in Section 7.1. Table 8 reports modeling accuracy of data input 64-bit APUFs which are built based on CRPs of MPUF. With increasing value of k, both the modeling accuracy of APUF and the probability of an APUF model approximating MPUF output reduce gradually. We have provided a range of accuracy values as different challenge sets of the same size (2×10^5 CRPs) were used in different runs of CMA-ES. There is no successful model for 64-bit 2-XOR APUF which is formed using one data input APUF and one selection input APUF even using 4×10^5 CRPs for $(64, 2)$-MPUF. So, 2-XOR APUF modeling using CRPs of MPUF is more difficult compared to a single APUF modeling. Hence, cMPUF is more robust against linear approximation attack than the basic MPUF. Robustness of rMPUF against linear cryptanalysis is the same as MPUF while they follow the same data input connection pattern.

In our reported result in Table 8, we have assumed that APUFs are perfectly reliable. It is obvious that in the presence of noise, accuracy will be reduced or the adversary needs more CRPs to achieve the same accuracy value reported in Table 8.

10 Discussion

Till now, we have introduced two variants of the basic MPUF, namely cMPUF and rMPUF. To improve robustness against linear approximation attack, one can also employ complemented data input connection in rMPUF like cMPUF. Since for $k = 3$, linear approximation of rMPUF (without complemented data inputs) is not leaking significant information, we suggest to use $(64, 3)$-rMPUF without complemented data inputs. According to our discussion, rMPUF is the best variant, and it should be used in practice where the adversary might have access to reliability information.

Although main objectives of rMPUF are to improve security and reliability metrics of APUF composition, uniformity and uniqueness metrics of rMPUF (also true for other MPUF variants) can suffer when primitive PUFs do not have good uniformity and uniqueness metrics. It is obvious that XOR APUF can improve uniformity and uniqueness metrics even if primitive APUFs do not have good values for these metrics. One can include this feature in rMPUF to a certain extent without reducing reliability significantly. One such approach is the use of 2-XOR APUF for each data input instead of single APUF. To reduce hardware overhead for data inputs with 2-XOR scheme, we suggest to use partially overlapped 2-XOR APUFs for 2^k data inputs like $(A_0^d \oplus A_1^d), (A_1^d \oplus A_2^d), \ldots, (A_{2^k-1}^d \oplus A_{2^k}^d)$. In this scheme, only one extra APUF instance is required and reliability of (n, k)-rMPUF is now similar to $(k+2)$-XOR APUF. As a cautionary note, one should not use 2-XOR scheme for selection inputs as it results in significant reliability reduction. Note that there will be no reduction in security level due to the overlapped 2-XOR scheme for data inputs of MUX, as outputs of two consecutive overlapped 2-XOR APUFs for a given challenge would never be available to rMPUF output for cryptanalysis. In reliability based modeling of rMPUF with 2-XOR scheme, the adversary needs to model 2-XOR APUF for data input APUFs.

The reader should not be confused with our introductions of many MPUF variants. This was done to explore useful MPUF design space. However, there is a recent proposal for PUF based lockdown authentication scheme

using XOR APUF in [28], and our proposed rMPUF can be used in that protocol to achieve higher number of secure authentications compared to XOR APUF.

11 Conclusion

In this paper, we have introduced three MUX based APUF compositions, namely MPUF, cMPUF and rMPUF. Major advantage of MUX composition is that it can achieve higher reliability than any practical XOR APUF instance. In addition, we have shown that the proposed rMPUF is robust against reliability based modeling while XOR APUF is vulnerable to the same attack. Based on our experimental results, it is shown that $(64, 3)$-rMPUF is at least as robust as 10-XOR APUF with 64-bit challenge, and its reliability is as high as the reliability of 4-XOR APUF. The rMPUF can be considered as a secure and reliable alternative for well-known XOR APUF in practice.

References

[1] D. Lim, "Extracting Secret Keys from Integrated Circuits," Master's thesis, MIT, USA, 2004.

[2] J. Delvaux, D. Gu, D. Schellekens, and I. Verbauwhede, "Secure Lightweight Entity Authentication with Strong PUFs: Mission Impossible?" in *Proc. of CHES*, 2014, pp. 451–475.

[3] R. Maes, V. van der Leest, E. van der Sluis, and F. Willems, "Secure Key Generation from Biased PUFs," in *Proc. of CHES*, 2015, pp. 517–534.

[4] A. Aysu, E. Gulcan, D. Moriyama, P. Schaumont, and M. Yung, "End-to-end design of a puf-based privacy preserving authentication protocol," in *Proc. of CHES*, 2015, pp. 556–576.

[5] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proc. of 17th ACM CCS*. New York, NY, USA: ACM, 2010, pp. 237–249.

[6] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. of DAC*. New York, NY, USA: ACM Press, 2007, pp. 9–14.

[7] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure PUFs," in *Proc. of ICCAD*. Piscataway, NJ, USA: IEEE Press, 2008, pp. 670–673.

[8] ——, "Techniques for Design and Implementation of Secure Reconfigurable PUFs," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 2, no. 1, pp. 1–33, 2009.

[9] 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 TIFS*, vol. 8, no. 11, pp. 1876–1891, 2013.

[10] J. Tobisch and G. T. Becker, "On the Scaling of Machine Learning Attacks on PUFs with Application to Noise Bifurcation," in *Proc. of RFIDsec*, 2015, pp. 17–31.

[11] U. Rührmair, X. Xu, J. Sölter, A. Mahmoud, M. Majzoobi, F. Koushanfar, and W. P. Burleson, "Efficient Power and Timing Side Channels for Physical Unclonable Functions," in *Proc. of CHES*, 2014, pp. 476–492.

[12] S. Tajik, E. Dietz, S. Frohmann, J. Seifert, D. Nedospasov, C. Helfmeier, C. Boit, and H. Dittrich, "Physical Characterization of Arbiter PUFs," in *Proc of CHES*, 2014, pp. 493–509.

[13] S. Tajik, H. Lohrke, F. Ganji, J. P. Seifert, and C. Boit, "Laser Fault Attack on Physically Unclonable Functions," in *12th Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, 2015.

[14] F. Ganji, J. Krämer, J. Seifert, and S. Tajik, "Lattice Basis Reduction Attack against Physically Unclonable Functions," in *Proc. of ACM SIGSAC CCS*, 2015, pp. 1070–1080.

[15] G. T. Becker, "The Gap Between Promise and Reality: On the Insecurity of XOR Arbiter PUFs," in *Proc. of CHES*, 2015.

[16] D. P. Sahoo, P. H. Nguyen, D. Mukhopadhyay, and R. S. Chakraborty, "A Case of Lightweight PUF Constructions: Cryptanalysis and Machine Learning Attacks," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 34, no. 8, pp. 1334–1343, 2015.

[17] P. H. Nguyen and D. P. Sahoo, "An Efficient and Scalable Modeling Attack on Lightweight Secure Physically Unclonable Function," 2016, http://eprint.iacr.org/2016/428.

[18] D. Mukhopadhyay and D. R. Chowdhury, "A Parallel Efficient Architecture for Large Cryptographically Robust $n \times k (k > n/2)$ Mappings," *IEEE Trans. Computers*, vol. 60, no. 3, pp. 375–385, 2011.

[19] M. day Yu and S. Devadas, "Recombination of Physical Unclonable Functions," in *GOMACTech-10 Conference*, 2010.

[20] M. Matsui, "Linear Cryptanalysis Method for DES Cipher," in *Proc. of Advances in Cryptology - EUROCRYPT*, 1993, pp. 386–397.

[21] J. Seberry, X. Zhang, and Y. Zheng, "Nonlinearity and propagation characteristics of balanced boolean functions," *Inf. Comput.*, vol. 119, no. 1, pp. 1–13, 1995.

[22] A. Maiti, V. Gunreddy, and P. Schaumont, "A Systematic Method to Evaluate and Compare the Performance of Physical Unclonable Functions," *IACR Cryptology ePrint Archive*, vol. 2011, p. 657, 2011.

[23] N. Hansen and A. Auger, "CMA-ES: evolution strategies and covariance matrix adaptation," in *13th Annual Genetic and Evolutionary Computation Conference (GECCO)*, 2011, pp. 991–1010.

[24] M. M. Yu, D. M'Raïhi, I. Verbauwhede, and S. Devadas, "A Noise Bifurcation Architecture for Linear Additive Physical Functions," in *Proc. of IEEE HOST*, 2014, pp. 124–129.

[25] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Testing Techniques for Hardware Security," in *Proc. of IEEE International Test Conference(ITC)*, Oct. 2008, pp. 1–10.

[26] Y. Hori, H. Kang, T. Katashita, A. Satoh, S. Kawamura, and K. Kobara, "Evaluation of Physical Unclonable Functions for 28-nm Process Field-Programmable Gate Arrays," *Journal of Information Processing (JIP)*, vol. 22, no. 2, pp. 344–356, 2014.

[27] S. Morozov, A. Maiti, and P. Schaumont, "An Analysis of Delay Based PUF Implementations on FPGA," in *Reconfigurable Computing: Architectures, Tools and Applications*, ser. Lecture Notes in Computer Science, P. Sirisuk, F. Morgan, T. El-Ghazawi, and H. Amano, Eds. Springer Berlin / Heidelberg, 2010, vol. 5992, pp. 382–387.

[28] M.-D. M. 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 Transactions on Multi-Scale Computing Systems*, 2016.