Challenge Selection for Salvaging Faulty APUFs

Yeqi Wei, Wenjing Rao, Natasha Devroye
Department of Electrical and Computer Engineering
University of Illinois Chicago, Chicago, IL 60607, USA
Email: {ywei30, wenjing, devroye}@uic.edu

***Abstract*—Arbiter-based Physically Unclonable Functions (APUFs) utilize the variability in manufacturing to create distinct digital identifiers for integrated circuits (ICs). Essentially, the input-output functions / truth-tables / full set of "responses" to "challenges", serve as potential hardware security primitives. To fulfill this role, every APUF batch from the same design should exhibit specific features; two of the most important are the response bias and uniqueness. A faulty APUF batch with a μ-fault from the design phase fails to achieve desired uniqueness levels and sometimes exhibits undesired response bias as well, hence is unqualified for security purposes. Instead of discarding such faulty APUFs and re-designing, we present a novel method to salvage a faulty APUF batch with the presence of multiple μ-faults, so that the desired uniqueness and bias are restored. This is done by carefully selecting challenges that can mitigate the impact of the faults. Such a salvaging strategy via challenge selection is intrinsically difficult, due to the enormous size of the challenge set, the black-box nature of APUFs, and the need to perform such tasks efficiently. To overcome these problems, we propose a simple yet effective way to estimate the intensity of the multiple faults and use them to guide the challenge selection process. The proposed method can efficiently find large challenge sets that achieve the desired response bias and uniqueness, thus salvaging a faulty APUF batch in the post-production phase.**

***Index Terms*—arbiter PUF, arbiter PUF faults, fault repair**

I. Introduction

PUFs are promising low-cost hardware security primitives that exploit manufacturing randomness to generate unique digital fingerprints for device authentication [1]–[3]. In an n-stage Arbiter-PUF (APUF), n track pairs are designed to be of equal delay, but due to manufacturing randomness, each pair differs slightly in their delay values. A binary input, or "challenge" $\mathbf{c} \in \{0,1\}^n$, decides which consecutive disjoint delay track pairs are selected to form two racing paths that are fed into an arbiter to produce the output, a binary "response" $R(\mathbf{c}) \in \{\pm 1\}$, which depends on which racing path arrives first. Each manufactured APUF instance has a truth table that consists of 2^n *challenge-response pairs* (CRPs), $\big(\mathbf{c}, R(\mathbf{c})\big)$ that is hopefully unbiased and unique. The APUF is a "strong" PUF with CRPs exponential in the number of delay elements and is a building block for more complex strong PUFs.

Prior work and motivation. To serve as a security primitive, it is critical for APUF instances to meet desired specifications, two of which are especially important: response bias and uniqueness. APUFs suffer from stuck-at faults as well as delay faults, which affect the response bias or uniqueness. Some papers [4]–[9] have designed testing and diagnosing methods for faulty APUFs. In general, those faulty APUFs do not qualify as security primitives and will be discarded or require re-design. In this paper, our focus is on an APUF-native production-line fault caused by unbalanced designs using Electronic Design Automation (EDA) tools, called the μ-fault, and such APUFs are with poor uniqueness (or response bias) [8], [9]. While testing and diagnosing methods are available for μ-fault, repair strategies were not discussed.

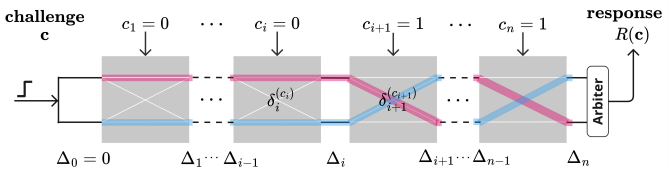

Fig. 1: The n-stage APUF with challenge bits c_i selects the parallel or crossed delay tracks to form two racing paths at each stage i and the delay difference is $\delta_i^{(c_i)}$. The response is the sign of the final accumulated delay difference $\Delta_n(\mathbf{c})$.

Contribution. Instead of discarding these unqualified APUFs, our work, for the first time, proposes to use challenge selection to mitigate the undesired fault impacts. One straightforward way is to use challenges that do not select the faulty delta element(s). However, such a challenge set shrinks quickly as the number of faults increases, and might not even exist. This paper presents a systematic challenge selection-based salvaging strategy where faults "cancel" each other, such that the response bias and uniqueness are as close to ideal as possible. This is achieved based on a method to estimate the intensity of each faulty element for an APUF batch with many μ-faults, and the estimated result is valid for salvaging APUFs generated from this μ-fault production line. The presented strategies are low-cost (need only a small number of APUF samples, no extra hardware), effective, and applicable to multiple faults.

II. Preliminaries

The n-stage APUF architecture is illustrated in Fig. 1, where each stage i consists of "parallel" and "crossed" tracks of delay delta elements. The input (or challenges) is a n-bit binary vector $\mathbf{c} \in \{0,1\}^n$. The signal traverses via the challenge-picked paths: the "parallel" tracks if $c_i = 0$, or the "crossed"

* This work was partially supported by NSF under awards 1909547 and 2244479. The contents of this article are solely the responsibility of the authors and do not necessarily represent the official views of the NSF.

979-8-3315-2144-8/25/$31.00 ©2025 IEEE

tracks if $c_i = 1$. The output (or response) is a binary bit $R(\mathbf{c}) \in \{\pm 1\}$ produced by a race resolution arbiter, which compares which of the two signals (red, blue) arrives first after traversing through n stages in series. The response is $+1$ (-1) if the lower (upper) entrance to the arbiter arrives first. The response relies only on which signal arrives first, i.e. it depends on the *delay difference* between a pair of selected tracks at each stage i, denoted as *delta elements*, $\delta_i^{(0)}$ (selected if $c_i = 0$) and $\delta_i^{(1)}$ (selected if $c_i = 1$), or $\delta_i^{(c_i)}$ for short.

The response $R(\mathbf{c})$ can be represented as the sign of the *accumulated delay difference* at the final stage, $\Delta_n(\mathbf{c})$, as

$$R(\mathbf{c}) = \text{sign}\big(\Delta_n(\mathbf{c})\big) \in \{\pm 1\},$$

where $\Delta_n(\mathbf{c})$ is computed recursively: for $i \in [1, n]$, $\Delta_0 = 0$, and $\Delta_i(\mathbf{c}) = (-1)^{c_i}\Delta_{i-1}(\mathbf{c}) + \delta_i^{(c_i)}$, where $\Delta_i(\mathbf{c})$ is the accumulated delay difference after stage i. By expanding the recursive equation, $\Delta_n(\mathbf{c})$ can also be represented as the summation of delta elements (δs) with different signs:

$$\Delta_n(\mathbf{c}) = \sum_{i=1}^{n} s_i^{(0)}\delta_i^{(0)} + \sum_{i=1}^{n} s_i^{(1)}\delta_i^{(1)},$$

where $s_i^{(c_i)}$ is the sign of $\delta_i^{(c_i)}$ is determined by $\mathbf{c}$ as in Table I.

c_i	Parity$(c_{i+1} : c_n)$	effect in Δ_n	$s_i^{(0)}$	$s_i^{(1)}$
0	0	$+\delta_i^{(0)}$ picked	$+1$	0
0	1	$-\delta_i^{(0)}$ picked	-1	0
1	0	$+\delta_i^{(1)}$ picked	0	$+1$
1	1	$-\delta_i^{(1)}$ picked	0	-1

TABLE I: Sign determination for delta elements.

As the signs are determined by a given challenge, not all 3^n sign cases are possible. For example, for stage i, one and only one delta element (either $\delta_i^{(0)}$ or $\delta_i^{(1)}$) must be selected, thus either $s_i^{(0)}$ or $s_i^{(1)}$ must be 0. Similarly, the following sign constraints exist:

- (same-stage only one element) $[s_i^{(0)}, s_i^{(1)}]$ must be one of $\{[0,-1],[-1,0],[0,1],[1,0]\}$.
- (final-stage non-negative) $s_n^{(c_n)} \neq -1$.
- (adjacent-stage correlation) if $s_{j+1}^{(0)} \neq 0$ and $s_j^{(x)} \neq 0$, then $s_j^{(x)} = s_{j+1}^{(0)}$; if $s_{j+1}^{(1)} \neq 0$ and $s_j^{(x)} \neq 0$, then $s_j^{(x)} \neq s_{j+1}^{(1)}$, where $x \in \{0, 1\}$.

A. Metrics: response bias and uniqueness

We focus on two PUF metrics of great importance: the response bias and the uniqueness [10], [11], defined as:

***Definition 1 (*response bias):** $Bias(\mathcal{A}, \mathcal{C})$, of a set of PUFs, $\mathcal{A}$, taken over a set of challenges, $\mathcal{C}$, is defined as

$$Bias(\mathcal{A}, \mathcal{C}) := Prob\left[R(\mathbf{c}) = +1\right] = \frac{\#\text{ of +1 in } \mathbf{R}(\mathcal{A}, \mathcal{C})}{|\mathcal{A}| \times |\mathcal{C}|}$$

where $\mathbf{R}(\mathcal{A}, \mathcal{C})$ are the responses of the PUFs in set $\mathcal{A}$ to the challenges in set $\mathcal{C}$. The response bias lies in the range $[0, 1]$, and is ideally 0.5.

Uniqueness is for hardware primitives with respect to authentication, i.e. whether each PUF is distinguishable from the others. We adopt a "modified" uniqueness from [9], which corrects an error in a widely-used uniqueness:

***Definition 2 (*uniqueness):** $U(\mathcal{A}, \mathcal{C})$ of PUFs $\mathcal{A}$, over a set of challenges, $\mathcal{C}$, is $U(\mathcal{A}, \mathcal{C}) := 1 - 2Dev$ where the deviation from the ideal average pairwise PUF Hamming distance is

$$Dev := \frac{2}{|\mathcal{A}|(|\mathcal{A}| - 1)} \sum_{i=1}^{|\mathcal{A}|-1} \sum_{j=i+1}^{|\mathcal{A}|} \left| \frac{\text{HD}\left(R_i(\mathcal{C}), R_j(\mathcal{C})\right)}{|\mathcal{C}|} - 0.5 \right|,$$

and $\text{HD}\left(R_i(\mathcal{C}), R_j(\mathcal{C})\right)$ is the Hamming distance between the responses of PUF i and PUF j to a challenge set $\mathcal{C}$. The uniqueness lies in the range [0, 1]: 0 is worst and 1 is ideal.

B. Fault models

In [8], [9] it was shown a manufacturer should ideally aim for every delta element in an APUF batch to follow the same zero-mean Gaussian distribution: $\forall i \in \{1, \cdots, n\}$, $x \in \{0, 1\}$. $\delta_i^{(x)} \sim \mathcal{N}(\mu_{\delta_i^{(x)}}, \sigma^2_{\delta_i^{(x)}}) = \mathcal{N}(0, \sigma^2)$. However, production lines will inevitably deviate from this due to for example EDA tool imbalance, leading to non-ideal bias and uniqueness:

Definition 3: **(μ-fault)** An APUF batch suffers from a μ-fault if there exists a δ element index, say, $\delta_j^{(x)}$, with non-zero mean Gaussian distribution, i.e. for some $K_j^{(x)} \neq 0$,

$$\exists j \in \{1, \cdots, n\}, x \in \{0, 1\} : \ \delta_j^{(x)} \sim \mathcal{N}(K_j^{(x)}, \sigma^2),$$

and $K_j^{(x)}$ is the fault intensity of the faulty delta element $\delta_j^{(x)}$.

We assume the number of stages, the fault count, and the fault locations are known and obtained by the methods in [8], [9], yielding fault list $\mathcal{F}$ and the following notation for the sign of elements in this list: $\mathcal{F} = \{f_1, f_2, ..., f_m : f_j \text{ is a faulty element } \delta_j^{(x)}\}$. However, the fault *intensities* are not – finding the fault intensities is not addressed in [8], [9].

C. The big picture

The response matrix (with rows as APUF instances and columns as the challenges) can be used to visualize the response bias and uniqueness [9]. In Fig. 2 (top), three response matrices illustrate an ideal production line and a μ-fault one with a single faulty delta element $\delta_j^{(x)}$. The ideal case shows the (ideal) random pattern. In contrast, the μ-fault scenario exhibits a "striped" pattern, where some challenges yield identical responses across APUFs generated from the same μ-fault production line, reducing uniqueness. In authentication protocols, an APUF claims its identity by responding to random challenges. The server verifies its claim based on the accuracy of the responses against a stored database. As shown in Fig. 2 (middle), for APUFs from an ideal line, a threshold (e.g., 0.8) allows clear distinction; however, APUFs from the same faulty production line cannot be distinguished using the same threshold[1]. This paper proposes a fault intensity-based method to generate a salvaging challenge set, restoring

[1]Note: if APUFs are generated from different production lines (either ideal or with μ-faults with varying fault intensities, the server can distinguish APUFs, and no new threshold is needed.

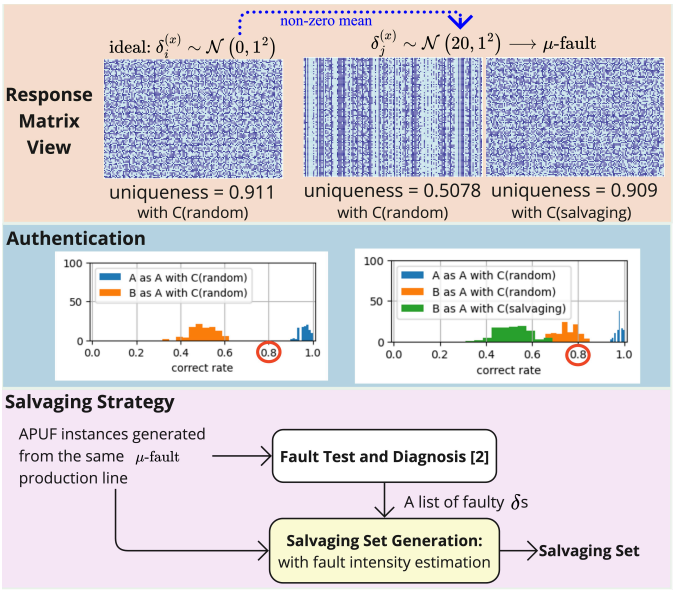

Fig. 2: Authenticating APUFs generated from a μ-fault production line requires a new threshold for random challenges; but not for challenges generated from salvaging strategy.

ideal uniqueness for faulty APUFs without requiring new authentication thresholds, as shown in Fig. 2 (bottom).

III. Candidate Sets

In this section, we discuss how each individual delta element affects the response bias, which helps us define the candidate set that highlights the impact of specific delta elements on response bias. Finally, we show that some candidate sets can "salvage" faulty APUFs, i.e., under these challenges, these faulty APUFs achieve ideal response bias and uniqueness.

The response bias is determined by the distribution of the final accumulated delay difference $\Delta_n(\mathbf{c})$, which can be expressed as $\Delta_n(\mathbf{c}) \sim \mathcal{N}(\mu_D, \sigma_D^2)$, where

$$\mu_D = \sum_{i=1}^{n} \left(s_i^{(0)} \mu_{\delta_i^{(0)}} + s_i^{(1)} \mu_{\delta_i^{(1)}} \right), \tag{1}$$

$$\sigma_D^2 = \sum_{i=1}^{n} \left(|s_i^{(0)}| \sigma^2_{\delta_i^{(0)}} + |s_i^{(1)}| \sigma^2_{\delta_i^{(1)}} \right),$$

and hence the response bias can be further represented as

$$Bias(\mathcal{A}, C) = Prob\left[\Delta_n(\mathbf{c}) > 0\right] = Q\left(\frac{-\mu_D}{\sigma_D}\right), \tag{2}$$

where $Q(\frac{t-\mu}{\sigma}) = Prob[Y > t]$ for a random variable $Y \sim \mathcal{N}(\mu, \sigma^2)$, where the $Q(\cdot)$ is the tail of the standard Gaussian function ($Q(x) := \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-t^2/2} dt$) that satisfies $Q(0) = 0.5$ and $Q(\frac{t-\mu}{\sigma}) + Q(-\frac{t-\mu}{\sigma}) = 1$.

Achieving an ideal response bias requires $\mu_D = 0$, which depends on the delta elements' mean and their signs. One way to guarantee $\mu_D = 0$ is to force all $\mu_{\delta_i^{(x)}} = 0$, which would result in a bias of 0.5 as $Q(0) = 0.5$. For example, APUFs generated from an ideal production line satisfy this.

If APUFs are generated from a μ-fault production line, it results in non-ideal bias when this bias is taken over some challenges. *The idea now is to select challenges which are unbiased (where $\mu_D = 0$).* Since μ_D is a summation of $s_i^{(c_i)} \mu_{\delta_i^{(c_i)}}$ terms, and $\mu_{\delta_i^{(c_i)}}$ are fixed, we start with grouping challenges based on the sign $(s_i^{(0)}, s_i^{(1)})$ of some delta elements, defined as *candidate set*, such that the μ_D are known for each group, as follows:

Definition 4: **(candidate set)** Given a delta set $\mathcal{D}$ with m delta elements $\{d_1, d_2, ..., d_m\}$, and a corresponding sign vector $\mathbf{s} = [s_1, s_2, ..., s_m] : s_j \in \{0, \pm 1\}, j \in [1, m]$. A candidate set $\mathcal{C}(\mathcal{D}, \mathbf{s})$ contains all the challenges selecting each delta element $d_j \in \mathcal{D}$ with sign s_j:

$$\mathbf{c} \in \mathcal{C}(\mathcal{D}, \mathbf{s}) \iff \forall d_j \in \mathcal{D}, sign(d_j, \mathbf{c}) = s_j.$$

Note: The construction of a candidate set can be done via the cases shown in Table I. If a sign vector $\mathbf{s}$ violates the sign constraints imposed by the APUF architecture, then the corresponding candidate set is necessarily empty (or does not exist). Otherwise, we call the sign vector $\mathbf{s}$ "available".

There are two particular candidate sets that deserve special attention: 1) the "avoidance set" $\mathcal{C}(\mathcal{D}, \mathbf{0})$ when $\mathbf{s} = [0, \ldots, 0]$ and none of the deltas in $\mathcal{D}$ are selected; and 2) the "target set" $\mathcal{C}(\{\delta_i^{(x)}\}, [\pm 1])$ which selects a single delta element with a specific sign. Note that this was denoted as $\mathcal{C}_{j,\pm}^{(x)}$ in [8], [9].

A. Faulty APUF in authentication

Many lightweight authentication protocols with strong PUFs are explored [12]–[16]. In this paper, we consider the basic authentication protocol in [17] and show how faulty APUFs with poor uniqueness affect the authentication protocol design.

The authentication protocol employs two phases: a one-time enrollment in a secure environment and the in-the-field authentication where the information is vulnerable to attacks. In the enrollment phase, the server collects arbitrary CRPs and saves them in a database for future authentication.

In an authentication process, an APUF A claims its identity as A to the server, and the server randomly selects a set of challenges and sends them to A. APUF instance A calculates the responses and returns them to the server. The server compares the responses in the database with the received CRPs and calculates the percentage that matches. The server accepts A's claim based on this percentage: if it exceeds some threshold $\gamma \in [0, 1]$, the server is convinced of A's claim and accepts it; otherwise, the server rejects it. For example, in Fig. 3 (left), two APUFs, A and B, are generated from an ideal 64-stage APUF production line, and both claim their identity as A. The match percentages are shown for different types of challenge sets. When A claims as A, the correct rate is high; when B claims as A, the correct rate is significantly smaller. It is easy to find a threshold γ, e.g. 0.8, so the server can distinguish APUF instance A from APUF instance B.

However, faulty APUFs generated from the same production line have poor uniqueness, and their responses to some challenges are the same. In Fig. 3 (right), both APUFs are from the

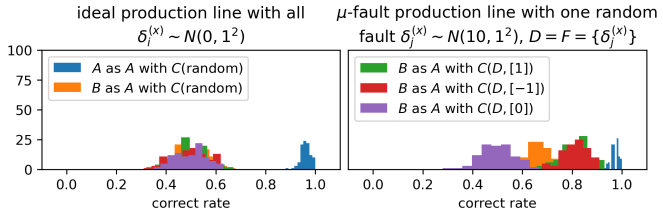

Fig. 3: Histogram of correct rate over 100 runs for two 64-stage APUFs (A and B) generated from the same ideal or μ-fault production line, with 100 authentication challenges.

TABLE II: Average accept rate of two 64-stage APUFs (A and B) under different thresholds. Both APUFs are generated from the same μ-fault production line with $\mathcal{F} = \{\delta_3^{(0)}\}, K_3^{(0)} = 10$.

threshold γ	0.75	0.8	0.85	0.9
A as A with $\mathcal{C}$(random)	1	1	1	0.99
B as A with $\mathcal{C}$(random)	0.03	0.01	0	0
B as A with $\mathcal{C}(\mathcal{F}, [\mathbf{1}])$	0.78	0.48	0.15	0.02
B as A with $\mathcal{C}(\mathcal{F}, [-\mathbf{1}])$	0.80	0.49	0.2	0.03
B as A with $\mathcal{C}(\mathcal{F}, [\mathbf{0}])$	0	0	0	0

same μ-fault production line with fault intensity $K_j^{(x)} = 10$. The correct rate of challenges from the random set increases when B claims as A. Specifically, the increase is related to the challenges from the candidate sets of the faulty delta element. Table II shows the accept rate under different thresholds and, when $\gamma = 0.8$, the accept rate of B claims as A with challenges from $\mathcal{C}(\mathcal{F}, [\pm\mathbf{1}])$ is around 0.5. This means **the threshold must be adjusted so the server can distinguish A from B, and there could be a trade-off between the accept rate and the reject rate.**

IV. Salvaging APUFs via Challenge Selection

We showed that faulty APUFs have poor response bias and uniqueness, and would fail authentication. We now present how to salvage these faulty APUFs by challenge selection such that the response bias and uniqueness are ideal within the selected challenges and, hence can be used for authentication.

A. Intuitive solution: using the avoidance set $\mathcal{C}(\mathcal{F}, \mathbf{0})$

The presence of abnormal delta elements is the root cause of non-ideal response bias and uniqueness. One immediate and intuitive approach to mitigating their impact is to select challenges that "avoid" the faulty elements at all (i.e. $\mathbf{s} = \mathbf{0}$).

1) Avoidance set with a single fault: In [8], [9], the authors consider a single μ-fault on $\delta_j^{(x)}$ and define the target and avoidance sets. They show that avoidance set challenges (instead of random sets) yield ideal response bias and uniqueness, and hence this set can be used to salvage the APUFs.

2) Avoidance set with multiple faults: We now extend the investigation from a single fault to multiple faults scenario. We start with the fault impact of two μ-faults, as shown in Fig. 4. The uniqueness for μ-faults decreases as the number of faults increases. The test and diagnosis methodologies from [9] are directly applied to the multiple-faults scenario.

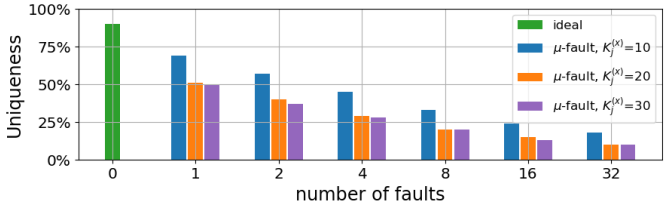

Fig. 4: Fault impact on uniqueness for 100 64-stage APUFs and 100 random challenges under different number of faults.

Next, we theoretically analyze the fault impact on response bias. We start from an 8-stage APUF with a fault list $\mathcal{F} = \{\delta_3^{(0)}, \delta_6^{(0)}\}$ (meaning $\delta_3^{(0)}$ and $\delta_6^{(0)}$ are faulty) and the fault intensities are $K_3^{(0)} = K_6^{(0)} = 5$. Then, there are a total of 3^2 possible vectors of $\mathbf{s}$ (recalling that each $s_i \in \{-1, 0, +1\}$), from $[-1, -1]$ all the way to $[+1, +1]$, and each defines a candidate set presented in Table III. For example, $\mathcal{C}(\mathcal{F}, [-1, 0])$ contains all the challenges with $s_3^{(0)} = -1$, $s_6^{(0)} = 0$, and this translates into selecting challenges $\mathbf{c}$ for which: a) the first faulty element $\delta_3^{(0)}$ is picked with negative sign in Δ_n, and b) the second faulty element $\delta_6^{(0)}$ is NOT selected. The underlined challenge bits correspond to the faulty delta elements at locations 3 and 6. The mean of $\Delta_n(\mathbf{c})$ can be derived as $\mu_D = s_3^{(0)} K_3^{(0)} + s_6^{(0)} K_6^{(0)}$, and there exist three candidate sets with ideal bias and uniqueness namely with $[s_1, s_2]$ equal to $[-1, +1], [0, 0], [+1, -1]$, indicated by a colored background.

$s_3^{(0)}$	$s_6^{(0)}$	bias	uniq.	μ_D	example challenge	$\lvert\widehat{\mu_D}\rvert$
-1	-1	0.00	0.00	-10	01000010	10.91
-1	0	0.04	0.13	-5	01000100	5.24
-1	+1	0.51	0.62	0	01010011	0.43
0	-1	0.02	0.07	-5	01110001	5.67
0	0	0.49	0.62	0	01110111	0.00
0	+1	0.94	0.21	+5	01110011	5.67
+1	-1	0.47	0.65	0	01010001	0.43
+1	0	0.97	0.12	+5	01010111	5.24
+1	+1	1.00	0.00	+10	01011000	10.91

TABLE III: Examples of candidate sets for 8-stage APUFs.

The response bias can be theoretically calculated based on the mean of $\Delta_n(\mathbf{c})$. For an n-stage APUF production line which suffers two μ-faults, $\delta_{j_1}^{(x_1)} \sim \mathcal{N}(K_{j_1}^{(x_1)} \neq 0, \sigma^2)$ and $\delta_{j_2}^{(x_2)} \sim \mathcal{N}(K_{j_2}^{(x_2)} \neq 0, \sigma^2))$ (all other normal delta elements $\delta_i^{(c_i)} \sim \mathcal{N}(0, \sigma^2)$), for challenges from a candidate set $\mathcal{C}\left([s_{j_1}^{(x_1)}, s_{j_2}^{(x_2)}]\right)$, the μ_D, by linearity of expectation, is $\mu_D = s_{j_1}^{(x_1)} K_{j_1}^{(x_1)} + s_{j_2}^{(x_2)} K_{j_2}^{(x_2)}$. The response bias over this candidate set is $Bias\left(\mathcal{A}, \mathcal{C}\left([s_{j_1}^{(x_1)}, s_{j_2}^{(x_2)}]\right)\right) = Q\left(\frac{-\mu_D}{\sigma_D}\right)$, where $\mu_D = s_{j_1}^{(x_1)} K_{j_1}^{(x_1)} + s_{j_2}^{(x_2)} K_{j_2}^{(x_2)}$ and $\sigma_D = \sqrt{n\sigma^2}$. This can be generalized to any number of faults.

In theory then, if we let $\mathbf{s}$ be all zeros, i.e. $s_i^{(x)} = 0$ for all faults in $\mathcal{F}$, then this is the avoidance set $\mathcal{C}(\mathcal{F}, \mathbf{0})$ which yields ideal bias and uniqueness, as all faults are avoided. However, as we show next, because of the sign constraints, **the avoidance set is not always available (i.e., it is an empty set)**

$[s_3^{(0)}, s_4^{(0)}]$	bias	uniq.	μ_D	example challenge
$[-1,-1]$	0.00	0.00	-10	01000001
$[-1,\ 0]$	0.04	0.16	-5	01010101
$[-1,+1]$	/	/	/	/
$[\ 0,-1]$	0.04	0.17	-5	01100001
$[\ 0,\ 0]$	0.48	0.72	0	01110101
$[\ 0,+1]$	0.96	0.17	+5	01100101
$[+1,-1]$	/	/	/	/
$[+1,\ 0]$	0.96	0.15	+5	01011101
$[+1,+1]$	1.00	0.00	+10	01000101

(a) Two faults $\delta_3^{(0)}, \delta_4^{(0)}$ at adjacent stages.

$[s_3^{(0)}, s_8^{(0)}]$	bias	uniq.	μ_D	example challenge
$[-1,-1]$	/	/	/	/
$[-1,\ 0]$	0.04	0.17	-5	01000001
$[-1,+1]$	0.50	0.73	0	01000100
$[\ 0,-1]$	/	/	/	/
$[\ 0,\ 0]$	0.49	0.71	0	01100101
$[\ 0,+1]$	0.96	0.14	+5	01100100
$[+1,-1]$	/	/	/	/
$[+1,\ 0]$	0.96	0.14	+5	01000101
$[+1,+1]$	1.00	0.00	+10	01001100

(b) Two faults $\delta_3^{(0)}, \delta_8^{(0)}$ with one ($\delta_8^{(0)}$) at the final stage.

TABLE IV: Examples of two μ-faults at varying locations where some candidate sets are empty (denoted by "/").

or is not large enough as the number of faults increases.

B. The problem of the avoidance set $\mathcal{C}(\mathcal{F}, \mathbf{0})$

Consider two-fault examples. If the designed signs at these two faults violate the sign constraints in Table I, the corresponding candidate set is empty or does not exist, e.g. when:

1) Two faults at the same stage: e.g. when $\delta_i^{(0)}$ and $\delta_i^{(1)}$ (i not the final stage) are faulty, since one delta element *must* be selected at each stage, either $\delta_i^{(0)}$ or $\delta_i^{(1)}$ is selected and hence sign cases $[0,0], [\pm 1, \mp 1]$ are not available.

2) Two faults at adjacent stages: e.g. when $\delta_3^{(0)}$ and $\delta_4^{(0)}$ are abnormal, as shown in Table IVa, if $\delta_4^{(0)}$ is selected with a non-zero sign $s_4^{(0)}$, then the sign $s_3^{(0)}$ must equal to $s_4^{(0)}$. Hence, sign cases $[-1,+1]$ and $[+1,-1]$ are not available.

3) Two faults with one at the final stage: e.g. when $\delta_3^{(0)}$ and $\delta_8^{(0)}$ are abnormal as shown in Table IVb, since the sign of a final-stage delta element must be positive, all sign cases with the final stage of sign -1 are not available.

As an alternative, we present a general way to systematically find challenges based on the fault intensity estimation method, to construct a challenge set with close to ideal response bias and uniqueness. This set can serve as a set for salvaging faulty APUFs when the avoidance set is not available.

V. Fault intensity-based challenge selection

Consider APUFs generated from a μ-fault production line with an abnormal delta element list $\mathcal{F}$. The goal is to find a challenge set ideal for both response bias and uniqueness. One brute-force approach is to directly evaluate the bias and uniqueness of all possible candidate sets. This always works. However, the number of candidate sets increases exponentially in the number of faults. We notice, from Table III and IV, that the candidate sets with ideal performance have $\mu_D = 0$ (including the avoidance set). This is no coincidence: the response bias may be expressed as in (2) and hence finding candidate sets with $\mu_D = 0$ as close to zero as possible will be optimal. To do so in an efficient manner, notice that **if the fault intensities are known, then μ_D, expressed as a linear combination of the signs and the means of delta elements as in (1) is easily calculated for any candidate set.** Hence, candidate sets with $\mu_D = 0$ can be found without a brute-force approach IF the fault intensities of all abnormal delta elements are known; we present a method for estimating these next.

A. Fault intensity estimation

To efficiently find sets whose $|\mu_D| \approx 0$ in (1) (i.e. sets that can salvage the PUFs), the fault intensities $K_i^{(x)}$ of the faulty elements $\delta_i^{(x)} \in \mathcal{F}$ need to be estimated. We propose a way to estimate these directly. This may be done in a clever manner again using (1) and (2): since the response bias is a Q-function of the unknown μ_D and the Q-function is invertible through direct look-up (very accurate tables are available), for different challenge sets, we can estimate μ_D for a challenge set $\mathcal{C}$ as:

$$\widehat{\mu_D}(\mathcal{C}) = \sum_{(i,x)\in\mathcal{F}} s_i^{(x)} K_i^{(x)} \approx -Q^{-1}\left(Bias(\mathcal{A}, \mathcal{C}) \times \sigma_D\right).$$

If the abnormal delta list $\mathcal{F}$ is of size m then we have m unknown fault intensities to estimate, which may be done by inverting an $m \times m$ system of linear equations. This invertible system of linear equations needs to be constructed based on candidate sets and the sign cases they correspond to: each candidate set corresponds to a specific choice of signs for the faulty elements, represented as a length m vector $\mathbf{s} \in \{-1, 0, 1\}^m$. We thus need to find m candidate sets chosen with corresponding sign cases $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m\}$ to ensure that the matrix $\mathbf{S} := [\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m]$ is invertible and solve:

$$\underbrace{\mathbf{S}}_{\text{known by selection}} \cdot \underbrace{\begin{bmatrix} K_1 \\ K_2 \\ \vdots \\ K_m \end{bmatrix}}_{\text{unknown}} \approx \underbrace{\begin{bmatrix} -Q^{-1}\left(Bias(\mathcal{A}, \mathcal{C}(\mathbf{s}_1))\sigma_D\right) \\ -Q^{-1}\left(Bias(\mathcal{A}, \mathcal{C}(\mathbf{s}_2))\sigma_D\right) \\ \vdots \\ -Q^{-1}\left(Bias(\mathcal{A}, \mathcal{C}(\mathbf{s}_m))\sigma_D\right) \end{bmatrix}}_{\text{obtained from querying the APUF}}$$
$$\implies \mathbf{S} \cdot \mathbf{K} \approx \mathbf{Q}$$

where the $\mathbf{S}$ is the sign case matrix (dimension $m \times m$ which is a matrix with entries of $\{-1, 0, +1\}$, $\mathbf{K}$ is the fault intensity vector of dimension $m \times 1$, and $\mathbf{Q}$ is the estimated μ_D vector of dimension $m \times 1$. If there exists an invertible sign matrix $\mathbf{S}_p$, the fault intensities can be estimated as:

$$\widehat{\mathbf{K}} \approx \mathbf{S}_p^{-1}\mathbf{Q}.$$

1) Example: Consider the example in Table III, where 100 APUFs and 16 challenges are used for fault intensity estimation. The first step is to find an invertible sign matrix; the identity matrix is always a great starting point. We calculate the response bias for challenges selected from candidate sets with sign cases $[+1, 0]$ and $[0, +1]$. By using $\widehat{\mathbf{K}} \approx \mathbf{S}_p^{-1}\mathbf{Q}$, the estimated fault intensities are $[5.24, 5.67]$.

2) Discussion: (a) The invertible sign matrix can be found using various approaches. We suggest first checking whether all sign cases in the identity matrix are available; if so the estimation result is ideal. (b) The μ_D is estimated by the inverse of the Q-function. Note that for $|\mu_D|$ greater than 30 (the fault mean is 30 times its variance, quite an outlier!), the gradient of the Q-function tends to be zero which leads to inaccurate $\widehat{\mu}_D$ estimation. (c) When only one APUF instance is available, the fault intensity estimation requires more challenges to achieve a small error rate.

3) Finding an invertible sign case matrix: Finding one $\mathbf{S}_p$ may be accomplished using a simple search or by randomly trying different possibilities. The only fault pattern for which such an invertible sign matrix does not exist is when there are two same-stage faults occurring at adjacent stages. From Monte Carlo simulations in a 64-stage APUF and average over 100000 runs (in each run, 10000 attempts are made to find a proper sign matrix), the probability of finding a proper sign matrix for different randomly placed fault locations is essentially 1. If the number of faults is less than 10, the success rate is 1.0 and we will be able to estimate the fault intensities.

B. Salvaging faulty APUFs: finding the optimal salvaging set

The avoidance set is always preferred if this is a valid set of signs given the fault locations; this corresponds to $\mathbf{s} = \{0\}^m$ (recalling that $|\mathcal{F}| = m$). This is the best set to use if it is available and large enough; otherwise, we suggest an ordered list of "salvaging sets" which are candidate sets ordered from smallest $|\mu_D|$ to largest. Once the fault intensities are estimated, it is trivial to calculate all possible μ_D.

To create the salvaging set of a desired size, select the avoidance set and then proceed to add more and more challenges to the salvaging set from the candidate sets, starting with the set with the smallest $|\mu_D|$ and so forth. Consider the previous example in which the estimated fault intensities are $[5.24, 5.67]$. The $|\widehat{\mu}_D|$ are known for the candidate sets of all sign cases, as shown in Table III. The challenges are selected by first picking the set with sign case $[0, 0]$ (the avoidance set). If more challenges are needed, then proceed to the candidate set with the next smallest $|\widehat{\mu}_D|$, etc. **Note: once the salvaging set is found, it is applicable to salvage all APUFs generated from this faulty μ-fault production line.**

VI. Simulation Results

Next, we show how to apply the challenge selection-based salvaging method on μ-fault APUF production line faults. We assume multiple faults at locations in $\mathcal{F}$, and each fault leads to $\delta_i^{(x)} \sim \mathcal{N}(K_i^{(x)}, \sigma^2)$ rather than the desired $\mathcal{N}(0, \sigma^2)$. Such μ-faults have a great impact on the response bias and uniqueness but are also relatively easy to salvage in the sense that the abnormal delta elements of APUF instances have similar values and hence can be salvaged by the same set of challenges. **That is – all μ-faults from one faulty APUF batch can be salvaged by the same set of challenges, which is highly beneficial from a practical engineering perspective – that whole batch needs to use different challenges from the salvaging set in the protocols.**

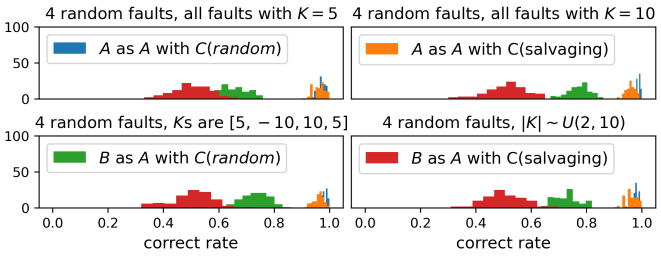

Fig. 5: Histograms of correct rate over 100 runs for two 64-stage APUFs (A and B) generated from the same μ-fault production lines, with 100 authentication challenges.

Ks	5	10	[5,-10, 10, 5]	U(2,10)
bias	$0.5 \rightarrow \mathbf{0.5}$	$0.5 \rightarrow \mathbf{0.5}$	$0.5 \rightarrow \mathbf{0.5}$	$0.5 \rightarrow \mathbf{0.48}$
uniq.	$0.71 \rightarrow \mathbf{0.91}$	$0.45 \rightarrow \mathbf{0.91}$	$0.53 \rightarrow \mathbf{0.91}$	$0.68 \rightarrow \mathbf{0.91}$

TABLE V: The response bias and uniqueness under random challenges and proposed salvaging set (bold) for 64-stage APUFs with four faults and different fault intensities Ks.

The Python simulation results of 100 64-stage faulty APUFs generated from μ-fault production lines, with 4 randomly picked faults are shown in Table V. The fault intensities of abnormal delta elements are fixed to 5 or 10, or randomly generated from a uniform distribution between 2 and 10. The number of challenges for the fault intensity estimation procedure and for salvaging are 100 and 10000, respectively. To evaluate the performance of challenges from the non-zero sign candidate sets, we skip the avoidance set[2]. The bias and uniqueness over 10000 challenges from a random challenge set and a salvaging set with $|\widehat{\mu_D}| < 1$ are given. The result shows that regardless of the fault locations, faulty APUFs perform ideally within the salvaging set, i.e. the response bias and uniqueness are close to the ideal, 0.5 and 1.0 respectively, and significantly outperform the random challenge sets usually used in APUF authentication protocols, as shown in Fig. 5.

VII. Conclusion

APUFs generated from APUF production lines with μ-faults have non-ideal response bias and uniqueness and are unqualified for authentication purposes. The root cause of faults is the abnormal delta elements. Choosing challenges that avoid these faults (i.e. from the avoidance set) yields ideal bias and uniqueness. However, the avoidance set is not always available for multiple faults, and its size shrinks as the number of faults increases. As an alternative, we have designed a low-cost, efficient fault intensity-based challenge selection method, for μ-fault affected APUFs without requiring extra hardware. Our method results in creating optimal sets of challenges that will yield close to ideal response bias and uniqueness.

[2]Often, challenges are selected from candidate sets with two non-zero signs rather than one non-zero sign as two can balance out, whereas one must be below the threshold to be selected.

REFERENCES

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

[2] B. Gassend, D. Lim, D. Clarke, M. van Dijk, and S. Devadas, "Identification and authentication of integrated circuits," *Concurrency - Practice and Experience*, vol. 16, pp. 1077–1098, 09 2004.

[3] W. Che, F. Saqib, and J. Plusquellic, "Puf-based authentication," in *IEEE ICCAD*, 2015, pp. 337–344.

[4] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Testing techniques for hardware security," in *IEEE ITC*, 2008, pp. 1–10.

[5] S. U. Hussain, S. Yellapantula, M. Majzoobi, and F. Koushanfar, "Bist-puf: Online, hardware-based evaluation of physically unclonable circuit identifiers," in *IEEE ICCAD*, 2014, pp. 162–169.

[6] D. Chatterjee, A. Hazra, and D. Mukhopadhyay, "Testability analysis of pufs leveraging correlation-spectra in boolean functions," 2018.

[7] J. Ye, Q. Guo, Y. Hu, and X. Li, "Deterministic and probabilistic diagnostic challenge generation for arbiter physical unclonable function," *IEEE TCAD*, vol. 37, no. 12, pp. 3186–3197, 2018.

[8] Y. Wei, T. Fox, V. Dumoulin, W. Rao, and N. Devroye, "Apuf faults: impact, testing, and diagnosis," in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2022, pp. 442–447.

[9] Y. Wei, W. Rao, and N. Devroye, "Apuf production line faults: Uniqueness and testing," in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2023, pp. 1–6.

[10] L. Feiten, M. Sauer, and B. Becker, "On metrics to quantify the inter-device uniqueness of PUFs," Cryptology ePrint Archive, Paper 2016/320, 2016. [Online]. Available: https://eprint.iacr.org/2016/320

[11] Y. Hori, T. Yoshida, T. Katashita, and A. Satoh, "Quantitative and statistical performance evaluation of arbiter physical unclonable functions on fpgas," *2010 International Conference on Reconfigurable Computing and FPGAs*, pp. 298–303, 2010.

[12] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Controlled physical random functions," in *18th Annual Computer Security Applications Conference, 2002. Proceedings.* IEEE, 2002, pp. 149–160.

[13] E. Öztürk, G. Hammouri, and B. Sunar, "Towards robust low cost authentication for pervasive devices," in *2008 Sixth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2008, pp. 170–178.

[14] S. Katzenbeisser, Ü. Kocabaş, V. Van Der Leest, A.-R. Sadeghi, G.-J. Schrijen, and C. Wachsmann, "Recyclable pufs: Logically reconfigurable pufs," *Journal of Cryptographic Engineering*, vol. 1, pp. 177–186, 2011.

[15] M. Majzoobi, M. Rostami, F. Koushanfar, D. S. Wallach, and S. Devadas, "Slender puf protocol: A lightweight, robust, and secure authentication by substring matching," in *2012 IEEE Symposium on Security and Privacy Workshops*. IEEE, 2012, pp. 33–44.

[16] Ü. Kocabaş, A. Peter, S. Katzenbeisser, and A.-R. Sadeghi, "Converse puf-based authentication," in *Trust and Trustworthy Computing: 5th International Conference, TRUST 2012, Vienna, Austria, June 13-15, 2012. Proceedings 5*. Springer, 2012, pp. 142–158.

[17] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Science*, vol. 297, no. 5589, pp. 2026–2030, 2002.