APUF Faults: Impact, Testing, and Diagnosis

Yeqi Wei, Tim Fox, Vincent Dumoulin, Wenjing Rao, Natasha Devroye
Department of Electrical and Computer Engineering
University of Illinois Chicago, Chicago, IL 60607, USA
Email: {ywei30, tfox8, vdumou2, wenjing, devroye}@uic.edu

***Abstract*—Arbiter Physically Unclonable Functions (APUFs) are hardware security primitives that exploit manufacturing randomness to generate unique digital fingerprints for ICs. This paper theoretically and numerically examines the impact of faults native to APUFs – mask parameter faults from the design phase, or process variation (PV) during the manufacturing phase. We model them statistically, and explain quantitatively how these faults affect the resulting APUF bias and uniqueness. On a single APUF instance, these faults manifest as some outlier delta elements in magnitude, thus we focus on such abnormal delta elements when addressing APUF faults. To detect such bad APUF instances and diagnose the abnormal delta elements, we propose a testing methodology which partitions a random set of challenges so that a specific delta element can be targeted, forming a perceivable bias in the responses over these sets. This low-cost approach is highly effective in detecting and diagnosing bad APUFs with abnormal delta element(s).**

***Index Terms*—arbiter PUF, arbiter PUF faults, testing, diagnosis**

I. Introduction

APUFs are promising low-cost hardware security primitives. In an APUF, a series of track pairs with equal delay are designed which, due to the randomness of PV, differ slightly in values. Two racing delay paths that depend on a binary input vector, or "challenge" $\mathbf{c} \in \{0,1\}^n$ are fed into an arbiter. The output is a binary "response" $R(\mathbf{c}) \in \{\pm 1\}$ that depends on which racing path arrives first. This makes it possible to form a unique truth table for each manufactured PUF instance, all from an identical design mask. This truth table consists of 2^n *challenge-response pairs* (CRPs), $(\mathbf{c}, R(\mathbf{c}))$. An APUF is an example of a "strong" PUF that offers CRPs exponential in the number of delay elements, and it is a basic building block for more complex PUFs used in device authentication [1]–[3].

A. Motivation

In contrast to conventional IC production, PUF production *relies on and exploits* PV in the manufacturing process, thus judging whether an individual or a batch of manufactured PUFs is good or bad presents its unique features and challenges, and effective test methodologies rely on statistical fault models that are native to PUFs. Prior work on APUFs and their many variations assumed the delay elements all follow the same Gaussian distribution [4]–[6]. We ask what happens when this assumption is not true, or when a supposedly zero-mean delta element has a particularly large or "abnormal" magnitude. The effect of such faults native to APUFs has not been studied before, thus this paper establishes statistical models for these faults, analyzes their impact on APUF qualities, and proposes a method to test and pinpoint such faults in each individual PUF instance.

B. Prior work

Several papers [7]–[9] have designed tests for APUFs to identify, for example: (i) predictability, (ii) sensitivity to component accuracy, (iii) susceptibility to reverse engineering, (iv) stability, and (v) randomness. In [9], two testing methods using correlation spectra and Welsh's t-test are presented to characterize both good (random-looking) and bad (infected by faults) APUFs. This is different than our focus on testing bad APUFs with abnormal delay elements and diagnosing their locations. The work most similar in spirit is [10], which focuses on identifying stuck-at and delay (not to be confused with delay element) faults in APUFs. None of these prior works focus on testing and diagnosing bad APUFs with outlier delta elements, a novel fault model native and relevant to APUFs.

C. Contribution

(1) By statistically modeling the parameters of APUFs from the design and manufacturing phases, we provide a framework for understanding how different types of faults affect APUF qualities, both analytically and numerically. The effect of the statistically modeled faults on important APUF quality metrics such as bias (percentage of responses that are positive) and uniqueness (fraction of responses to the same challenges that differ between two APUFs) lead to various insights for APUF design constraints and rules.

(2) For each individual APUF instance, we focus on the manifestation of the statistically modeled faults and propose a simple and effective test, based on the response bias over some "target sets" (special sets of challenges that highlight the impact of a specific delay element), that is able to distinguish the stage, type, and sign of (multiple) abnormal delay element(s).

II. APUF Model and Target Sets

The architecture of the well-known APUF is illustrated in Fig. 1, where the input is a challenge vector $\mathbf{c} \in \{0,1\}^n$ and the output is a binary response $R(\mathbf{c}) \in \{\pm 1\}$ produced by a race resolution arbiter which compares which of the two racing signals – denoted by red and blue in the figure – arrives first, after traversing through n stages in series. In each stage i, among the four tracks / delay elements t_i, u_i, r_i and s_i, the two signals traverse via the "parallel" tracks (t_i, u_i) if the challenge bit $c_i = 0$, or the "crossed" tracks (r_i, s_i) if $c_i = 1$. The response is $+1$ (-1) if the upper (lower) entrance to the arbiter arrives first.

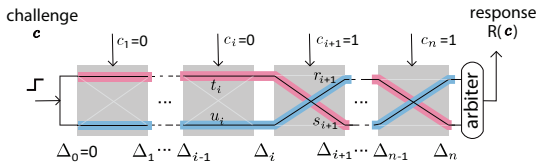


Fig. 1: APUF with challenge bits selecting the parallel (t_i, u_i) or cross (r_i, s_i) tracks to form two racing paths. The response is the sign of the accumulated delay difference $\Delta_n(\mathbf{c})$.

Since the response relies only on which signal arrives first, it depends on the relative delay difference between the two racing paths. Hence, it depends on the *delay difference* at each stage i, denoted as *delta elements*, $\delta_i^{(0)} := t_i - u_i$ (selected if $c_i = 0$) and $\delta_i^{(1)} := r_i - s_i$ (selected if $c_i = 1$), or $\delta_i^{(c_i)}$ for short.

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

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

where $\Delta_n(\mathbf{c})$ is computed recursively for $i \in [1, n]$, $\Delta_0 = 0$:

$$\Delta_i(\mathbf{c}) = \begin{cases} +\Delta_{i-1}(\mathbf{c}) + \delta_i^{(0)}, & \text{when } c_i = 0 \\ -\Delta_{i-1}(\mathbf{c}) + \delta_i^{(1)}, & \text{when } c_i = 1 \end{cases} \tag{2}$$

and $\Delta_i(\mathbf{c})$ is the accumulated delay difference until stage i.

It is usually assumed for APUFs that all the delta elements follow the same zero-mean Gaussian distribution: $\delta_i^{(x)} \sim \mathcal{N}(0, \sigma^2), \forall i \in [1, n], x \in \{0, 1\}$. We propose a statistical model able to isolate the effects of the mask versus the PV, with parameters of an individual APUF modeled as

$$(t_i, u_i, r_i, s_i) = (t_i^*, u_i^*, r_i^*, s_i^*) + (\epsilon_{t_i}, \epsilon_{u_i}, \epsilon_{r_i}, \epsilon_{s_i}),\ i \in [1, n].$$

The first term represents the design-phase mask parameters (common to all APUFs), and the second the manufacturing-phase PV parameters (unique to each APUF). Hence,

$$\delta_i^{(0)} = t_i - u_i = (t_i^* - u_i^*) + (\epsilon_{t_i} - \epsilon_{u_i}) \tag{3}$$

$$=: \mu_i^{(0)} + \epsilon_i^{(0)} \sim \mathcal{N}(\mu_i^{(0)}, \sigma_{i,0}^2) \tag{4}$$

$$\delta_i^{(1)} = r_i - s_i = (r_i^* - s_i^*) + (\epsilon_{r_i} - \epsilon_{s_i}) \tag{5}$$

$$=: \mu_i^{(1)} + \epsilon_i^{(1)} \sim \mathcal{N}(\mu_i^{(1)}, \sigma_{i,1}^2) \tag{6}$$

where $\mu_i^{(0)}$ and $\mu_i^{(1)}$ are determined by the mask parameters $(t_i^*, u_i^*, r_i^*, s_i^*)$, fixed for all APUFs. The PV is captured by $\epsilon_i^{(0)}$ and $\epsilon_i^{(1)}$, modeled as independent Gaussian random variables.

Our main technical tool and innovation lies in the use of "target sets", or sets of challenges able to extract the influence of a particular delta element on the response bias.

***Definition** 1 (target set):* A target set $\mathcal{C}_{i,+}^{(x)}$ (or $\mathcal{C}_{i,-}^{(x)}$) with $x \in \{0, 1\}, i \in [1, n]$ contains all n-bit challenges preserving (or reversing) the sign of $\delta_i^{(x)}$, which may be derived from (2):

$$\begin{aligned} \mathcal{C}_{i,+}^{(x)} &:= \{\text{challenges with } +\delta_i^{(x)} \text{selected in } \Delta_n\} \\ &= \{\mathbf{c} \in \{0,1\}^n : c_i = x, c_{i+1} + \cdots + c_n \text{ is even}\} \\ \mathcal{C}_{i,-}^{(x)} &:= \{\text{challenges with } -\delta_i^{(x)} \text{selected in } \Delta_n\} \\ &= \{\mathbf{c} \in \{0,1\}^n : c_i = x, c_{i+1} + \cdots + c_n \text{ is odd}\}. \end{aligned}$$

III. Statistical Fault Models and Their Impacts

Intuitively, a good APUF production line is characterized by symmetry, or equal delay for each track pair from the mask, and uniform, unskewed PV for all of the stages of all APUFs:

***Definition** 2 (good APUF production):* Requires $t_i^* = u_i^*$, $r_i^* = s_i^*$ and $\epsilon_{t_i}, \epsilon_{u_i}, \epsilon_{r_i}, \epsilon_{s_i} \sim \mathcal{N}(0, \sigma^2)$, resulting in $\delta_i^{(x)} \sim \mathcal{N}(0, \sigma^2), \forall i \in [1, n], x \in \{0, 1\}$.

Faults in mask or PV may cause deviations from these assumptions in several ways, such as:

a) Mask faults during the design phase can break symmetry, with $t_j^* \neq u_j^*$ or $r_j^* \neq s_j^*$. This could be caused by the lack of strict constraints to the CAD tools, and it will result in the $\delta_j^{(x)}$ of all the APUFs having a non-zero mean.
b) PV faults from the manufacturing phase can break the assumption of $(\epsilon_{t_i}, \epsilon_{u_i}, \epsilon_{r_i}, \epsilon_{s_i}) \sim \mathcal{N}(0, \sigma^2)$, with two possibilities:
   1) Some ϵ could have non-zero mean, caused by some asymmetric characteristics of the fabrication process, unilaterally enlarging (or shrinking) some values of delay elements.
   2) Alternatively, some ϵ could have an uncommon σ, perhaps caused by spatial asymmetry during fabrication across the delay elements.

Mask and PV faults can be respectively modeled as:

***Definition** 3 (μ-fault of APUF production):* Some abnormal $\delta_j^{(y)} \sim \mathcal{N}(\mu, \sigma^2)$ where $\mu = K\sigma \neq 0$ for some scalar K, while all other $\delta_i^{(x)} \sim \mathcal{N}(0, \sigma^2), \forall i \neq j \in [1, n]$ or $x \neq y \in \{0, 1\}$.

***Definition** 4 (σ-fault of APUF production):* Some abnormal $\delta_j^{(y)} \sim \mathcal{N}(0, \xi^2)$ where $\xi = L\sigma$ for some scalar $L \gg 1$, while all other $\delta_i^{(x)} \sim \mathcal{N}(0, \sigma^2), \forall i \neq j \in [1, n]$ or $x \neq y \in \{0, 1\}$.

A μ-fault or σ-fault with large K or L will, with high probability, result in one (or more) $\delta_j^{(y)}$ having a very large value. We observe the effect of such large $\delta_j^{(y)}$ on APUF quality metrics, and learn how to detect such abnormalities.

A. Bias and uniqueness metrics: definitions and analysis

We look at the impact of μ-faults and σ-faults on the bias and uniqueness, two (of many) PUF-quality metrics [11], [12].

1) Impact of μ-fault on bias and uniqueness:

We first consider a generic μ-fault model, in which

$$\delta_j^{(0)} \sim \mathcal{N}(K_0\sigma, \sigma^2),\ \delta_j^{(1)} \sim \mathcal{N}(K_1\sigma, \sigma^2). \tag{7}$$

Bias represents whether a PUF generates responses ± 1 with equal percentage and is ideally 0.5.

***Definition** 5:* The response bias of *multiple* PUFs over a set of challenges $\mathcal{C}$, denoted as $B_m(\mathcal{C})$, is defined as the average of a *single* PUF's bias over $\mathcal{C}$, $B_s(\mathcal{C})$ (with $B_s(\emptyset) = 0.5$) as

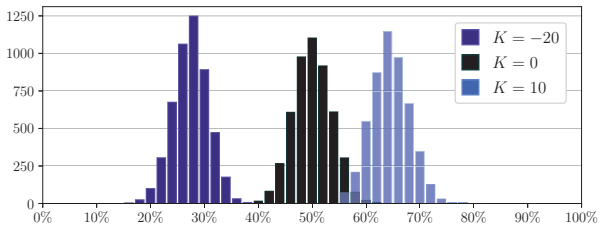


Fig. 2: Histograms of $B_s(\mathcal{C})$ for good PUFs ($K = 0$) vs. bad PUFs with last-stage μ-fault ($K = -20, 10$).

in (8), which can be empirically approximated as in (9) and corresponds to the fraction of positive responses over $\mathcal{C}$:

$$B_m(\mathcal{C}) := E_{\text{PUF}}\left[B_s(\mathcal{C})\right] = E_{\text{PUF}}\left[P_{\mathbf{c}\in\mathcal{C}}\left[\Delta_n(\mathbf{c}) > 0\right]\right] \quad (8)$$

$$\approx \frac{1}{\#\text{ PUFs}} \sum_{\text{PUFs}} \left[\frac{1}{|\mathcal{C}|}\sum_{\mathbf{c}\in\mathcal{C}} \mathbb{1}(R(\mathbf{c}) = +1)\right]. \quad (9)$$

For the generic μ-fault model, one can easily see that if the fault is at stage $j \neq n$ (the last stage), then a quarter of the challenges pick each of the $\delta_j^{(0)}$ with a positive sign, a quarter negative, a quarter pick $\delta_j^{(1)}$ with a negative and a quarter with a positive sign, leading to the bias $B_m(\mathcal{C})$ over $\mathcal{C} = \{0,1\}^n$ as

$$B_m(\mathcal{C}) = E_{\text{PUF}}[P(\Delta_n(\mathbf{c}) > 0)] = 0.5,\ j \neq n. \quad (10)$$

When $j = n$ (last stage has an abnormally large mean), then

$$B_m(\mathcal{C}) = \frac{1}{2}\left(Q\left(\frac{-K_0}{\sqrt{n}}\right) + Q\left(\frac{-K_1}{\sqrt{n}}\right)\right), \quad (11)$$

where $Q(x)$ denotes the tail distribution function of the standard normal distribution, i.e. $Q(x) := \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-u^2/2}du = P(X > x)$, if $X \sim \mathcal{N}(0,1)$. Note that while the set of all challenges remains unbiased under different fault models (with the exception of when the final delta is abnormal), if the target set is specially selected, we will observe a bias of

$$B_m(\mathcal{C}_{j,+}^{(x)}) = P(\Delta_n(\mathbf{c}) > 0 | \mathbf{c} \in \mathcal{C}_{j,+}^{(x)}) = Q\left(\frac{-K_x}{\sqrt{n}}\right), \quad (12)$$

which will allow us to identify faults in $\delta_j^{(x)}$.

Uniqueness is measured by how much a pair of PUFs differ in their responses using the "inter-PUF distance" metric (called uniqueness here for short) from [12, Equation (3)], and is ideally 0.5. It is formally defined as in (13) and may be empirically approximated as in (14) if given M PUFs:

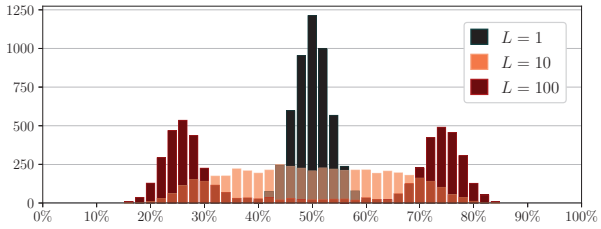


Fig. 3: Histograms of $B_s(\mathcal{C})$ for good PUFs ($L = 1$) vs. bad PUFs of last-stage σ-fault ($L = 10, 100$).

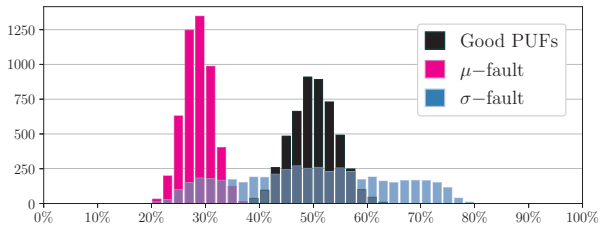


Fig. 4: Histograms of $\text{HD}_s(\mathcal{C})$ for good PUFs vs. bad PUFs with a single μ-fault ($K = 20$) or a single σ-fault ($L = 20$).

***Definition** 6:* The *uniqueness*, $\text{HD}_m(\mathcal{C})$ of *multiple* PUFs over challenge set $\mathcal{C}$ is the expectation of the *single pair distance* between two PUF instances, $\text{HD}_s(\mathcal{C})$ over $\mathcal{C}$ as:

$$\text{HD}_m(\mathcal{C}) := E_{\text{PUF}}\left[\text{HD}_s(\mathcal{C})\right] \quad (13)$$

$$\approx \left[\frac{2}{(M)(M-1)}\sum_{i=1}^{M-1}\sum_{j=i+1}^{M}\frac{\text{HD}(R_i(\mathcal{C}), R_j(\mathcal{C}))}{|\mathcal{C}|}\right] \quad (14)$$

where $R_i(\mathbf{c})$ is the response of PUF i, and $\text{HD}(R_i(\mathcal{C}), R_j(\mathcal{C}))$ is the Hamming distance between the response vectors of PUFs i and j (M total PUFs), taken over the $|\mathcal{C}|$ challenges in set $\mathcal{C}$. For the generic μ-fault model, a little more work yields:

$$\text{HD}_m(\mathcal{C}) = P_{\text{PUF},\mathbf{c}\in\mathcal{C}}(R_1(\mathbf{c}) \neq R_2(\mathbf{c})) \quad (15)$$

$$= \left[Q\left(\frac{-K_0}{\sqrt{n}}\right)Q\left(\frac{K_0}{\sqrt{n}}\right) + Q\left(\frac{-K_1}{\sqrt{n}}\right)Q\left(\frac{K_1}{\sqrt{n}}\right)\right]. \quad (16)$$

2) Impact of σ-fault on bias and uniqueness:

Consider a generic σ-fault model, in which

$$\delta_j^{(0)} \sim \mathcal{N}(0, L_0\sigma^2),\ \delta_j^{(1)} \sim \mathcal{N}(0, L_1\sigma^2). \quad (17)$$

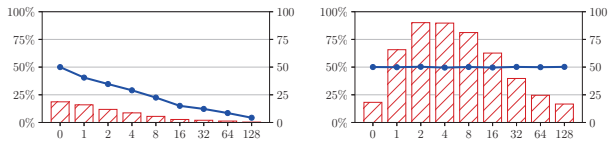
It is easy to see that the bias and uniqueness will be 0.5 since $P[\Delta_n(\mathbf{c}) > 0] = 0.5$ (probability over PUFs) for all challenges.

Note: The response bias and uniqueness are defined as expected values over the distributions of the manufactured delay differences and over challenges in a set $\mathcal{C}$. For a fixed PUF instance, the delay differences are fixed and so the per-PUF bias and uniqueness are taken over challenges. Hence, we can view a "distribution" of bias and uniqueness over different PUF instances. Ideally these distributions are deterministic peaks at 0.5. However, in practice there are ranges of per-PUF bias and uniqueness values: the larger the spread, the worse.

B. Simulation results: bias and uniqueness of faults

The data is presented using [13], obtained by applying 1000 random challenges as the set $\mathcal{C}$ over a large number (1000 to 5000) of 64-bit APUFs, all injected with the same configuration of μ-faults (with scalar K) or σ-faults (with scalar L).

1) Impact on response bias $B_m(\mathcal{C})$: Interestingly, μ-faults and σ-faults do not affect $B_m(\mathcal{C})$, unless they occur at the last stage n of an APUF as shown in Figs. 2 and 3. Since usually there are an equal number of challenges selecting a delta element (exception: last stage) in either positive or negative forms, even if a delta element is large, its impact on the percent



(a) μ-fault count (b) σ-fault count

Fig. 5: Impact of multiple faults on uniqueness: how the mean (blue line, left y-axis) and variance (red bar, right y-axis) of $\text{HD}_m(\mathcal{C})$ are affected by μ- and σ-faults with $K = L = 10$.

of positive and negative responses will cancel out. However, when the μ-fault or σ-fault is in the last stage, bias is affected, as seen in Figs. 2 and 3 and equation (11). *This emphasizes the importance of ensuring that last stage delta elements do not deviate from the ideal symmetric assumptions.*

2) Impact on uniqueness $HD_m(\mathcal{C})$: Fig. 4 shows a histogram of the uniqueness $\text{HD}_s(\mathcal{C})$ of a single μ-fault or a single σ-fault, taken over the manufacturing randomness of the APUFs. A single μ fault shifts the distribution of $\text{HD}_s(\mathcal{C})$ leftwards, and a single σ fault increases its variance (flattens it out) relative to good APUFs. Both cases are bad for APUF uniqueness: the distribution of $\text{HD}_s(\mathcal{C})$ should ideally be a single mass point at 50%. Deviation from this, either by a decreased mean or increased variance, results in some (large number of) APUF pairs yielding similar responses.

Fig. 5 shows how increasing the number of faults affects $\text{HD}_m(\mathcal{C})$'s mean (the lower the worse) and variance (the higher the worse) over APUFs. Here, as the number of faults increases from 0 to $2n$ on the x-axis (in logarithmic scale), we observe the impact of increasing number of μ vs. σ faults on uniqueness. The mean (blue lines) of $\text{HD}_m(\mathcal{C})$ is not affected by σ-faults, but it decreases significantly as the μ-fault count increases. When all δs have μ-faults, ($x = 2n$) the mean is reduced to $\approx 5\%$ as a function of (K, n), as seen in Fig. 6 (M4). We can explain this analytically, but omit details due to space.

The red bars indicate the variance of APUF uniqueness ($\text{HD}_m(\mathcal{C})$), which remains small as the μ-fault count increases but is affected drastically even with a single σ-fault occurrence.

3) Impact of μ-fault count and locations: To show the fault count and location impacts, we consider the following models, all of which are μ-faults with identical PV, and $K, K_i, K_j \neq 0$:

- **M1:** single μ-fault: $\delta_i^{(x)} \sim \mathcal{N}(K\sigma, \sigma^2)$.
- **M2:** dual μ-faults at the same stage: $\delta_i^{(0)} \sim \mathcal{N}(K_0\sigma, \sigma^2)$ and $\delta_i^{(1)} \sim \mathcal{N}(K_1\sigma, \sigma^2)$.
- **M3:** dual μ-faults at different stages: $\delta_i^{(x)} \sim \mathcal{N}(K_i\sigma, \sigma^2)$, and $\delta_j^{(y)} \sim \mathcal{N}(K_j\sigma, \sigma^2)$.
- **M4:** maximum μ-faults: $\forall i \in [1, n]$, $\delta_i^{(x)} \sim \mathcal{N}(K\sigma, \sigma^2)$.

Fig. 6 shows simulation results for how μ-faults affect the uniqueness $\text{HD}_m(\mathcal{C})$ of the 4 models for some random set $\mathcal{C}$ of 1000 challenges. We are able to derive these plots analytically (omitted due to space constraints), and these theoretical results match our Monte Carlo simulations over 1000 APUFs and 1000 challenges, where the $|K|$ in all models is the same. We notice that M2 and M4 are the "worst" as they relatively quickly converge to a low value for the uniqueness (i.e. for high $|K|$, most APUFs produce the same outputs to the same challenges). M1 and M3 level out at about 25%, meaning that about a quarter of the challenges produced will lead to different responses, which may also be verified analytically.

Fault impact: The results here indicate the severeness of μ-faults in decreasing the APUF uniqueness $\text{HD}_m(\mathcal{C})$. *One must aim for $t_i^* = u_i^*, r_i^* = s_i^*$ in the mask for all elements.* These figures also show the impact of σ-faults in increasing the variance of $\text{HD}_m(\mathcal{C})$: many produced APUFs will have a $\text{HD}_m(\mathcal{C})$ that deviates from 50%, jeopardizing uniqueness. *It is thus important to maintain uniform PV across elements.*

IV. Test & Diagnosis of APUF Instances

In this section, we present how to *detect* whether *an individual APUF instance* has unusually large delta elements (thus a "bad" APUF), and how to *diagnose* these "abnormal" delta elements (location, type, and sign). Our method is based on using the bias of target sets as the test statistic.

When presented with a single PUF instance, one cannot differentiate whether a particular large $|\delta_i^{(x)}|$ came from a μ- or σ-fault; all that matters is that it is large and hence will affect the bias and uniqueness. This motivates us to define:

***Definition** 7 (δ-fault of a bad APUF instance):* A $\delta_j^{(y)}$ is *abnormal* if $|\delta_j^{(y)}| \geq T$, where $T = K\sigma$ is a constant. An APUF is *bad* if it has at least one abnormal delta element.

The value of K here can be solved for by setting $\text{HD}_m(\mathcal{C}) = P^*$ for the desired uniqueness. For example, for M1, this becomes a quadratic to solve for $Q(K/\sqrt{n})$, and invert for K. The manufacturer might approximately obtain σ for PV, so that T may be obtained once K is found.

To find abnormal delta elements, we isolate the effect of a single delta element on Δ_n, and hence on $R(\mathbf{c})$, using the *target sets* of Definition 1. As seen by equation (12), the response bias over a target set (in contrast to the entire or random challenge set) will deviate from 0.5 when a delta element is abnormal.

Fig. 7 illustrates this with histograms of Δ_n of 10,000 random challenges for 4 64-bit APUF instances: a good APUF, and 3 bad APUFs with an abnormal $\delta_{25}^{(0)} = 10, 25, 50$, respectively, and normal delta elements $\sim \mathcal{N}(0, 1)$. All histograms are symmetric about 0, with response biases of mean $B_s(\mathcal{C}) = 0.5$

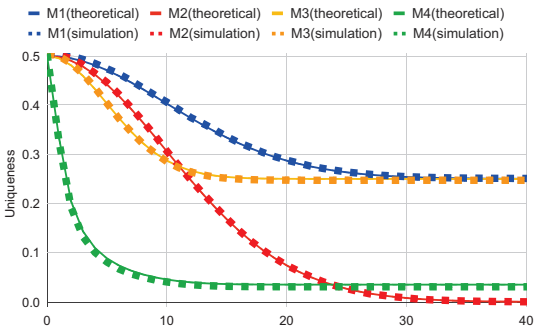


Fig. 6: Uniqueness $\text{HD}_m(\mathcal{C})$ (y-axis) deteriorates as $|K|$ (x-axis) increases (under the 4 models M1-M4, via theoretical results from (14) and by Monte-Carlo simulations).

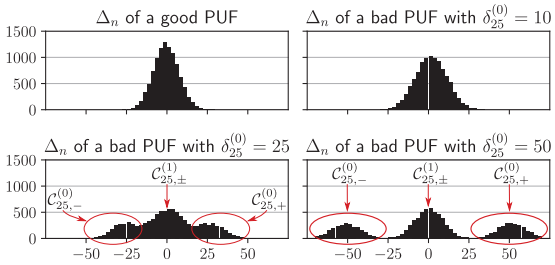


Fig. 7: Δ_n histograms for good vs. bad APUF instances.

for a random set $\mathcal{C}$. The histograms of bad APUFs, with mean of 0.5, have three humps rather than one. The smaller humps are formed from the target sets $\mathcal{C}_{25,\pm}^{(0)}$, selecting the abnormal $\delta_{25}^{(0)}$ with positive and negative sign, respectively. Thus, the response bias over a target set $B_s(\mathcal{C}_{i,\pm}^{(x)})$ will be biased, and can be used to "identify" an abnormal $\delta_i^{(x)}$.

A. Difference scores for delta elements: $D_i^{(x)}$

Based on using the response biases of target sets, $B_s(\mathcal{C}_{i,\pm}^{(x)})$, to identify an abnormal $\delta_i^{(x)}$, we propose the following metric:

***Definition** 8 (difference score):* For an APUF instance with responses obtained from a random challenge set $\mathcal{C}$, the difference score $D_i^{(x)}$ for each delta element $\delta_i^{(x)}$ is defined as

$$D_i^{(x)} := B_s(\mathcal{C}_{i,+}^{(x)}) - B_s(\mathcal{C}_{i,-}^{(x)}) \in [-1, 1] \tag{18}$$

A normal $\delta_i^{(x)}$ tends to have $D_i^{(x)} \approx 0$. When $D_i^{(x)}$ deviates from 0 ($|D_i^{(x)}| > \gamma$), it can be used to indicate the corresponding $\delta_i^{(x)}$ is abnormal, with the same sign, as follows:

Approximately optimal test: If

$$\begin{cases} D_i^{(x)} > \gamma, & \text{decide abnormal } \delta_i^{(x)} > T = K\sigma \\ |D_i^{(x)}| < \gamma, & \text{decide normal } |\delta_i^{(x)}| \leq T = K\sigma \\ D_i^{(x)} < -\gamma, & \text{decide abnormal } \delta_i^{(x)} < -T = -K\sigma \end{cases}.$$

For $i = n$, $\mathcal{C}_{n,-}^{(x)} = \emptyset$, $D_n^{(x)} = B_s(\mathcal{C}_{n,+}^{(x)}) - 0.5$. The proof for using difference score as the test statistic is omitted.

Fig. 8 illustrates how these difference scores "identify" an abnormal delta element: $D_{25}^{(0)} \approx 1.0$ spikes for the abnormal $\delta_{25}^{(0)}$. Naturally, there will be a trade-off in identifying abnormal stages as abnormal and misdiagnosing normal stages as abnormal, which will depend on the threshold $\gamma \in [0, 1]$. To pick γ to obtain a desired false positive rate $0 < p_{\text{FP}} < 1$ we must solve $P(|D_i^{(x)}| > \gamma) \leq p_{\text{FP}}$ for γ. This can be done numerically or by looking at the ROC curve (shown in Fig. 11) for the given K. Note that K is either solved for as described below Definition 7, or it may be estimated by using (12) to obtain $K \approx -\sqrt{n}Q^{-1}(B_s(\mathcal{C}_{i,+}^{(x)}))$.

B. Detection and diagnosis simulation results

We now present the results of the proposed testing methodology, outlined in Algorithm 1. The detection and diagnosis results will refer to the typical metrics used in a confusion matrix, including True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). We mainly focus on the *True Positive Rate (TPR)*, defined as TP / (TP + FN), and the *False Positive Rate (FPR),* defined as FP / (FP + TN). Ideally, TPR = 1 and FPR = 0.

Algorithm 1: Testing a single APUF instance

Input: A random set of challenges $\mathcal{C}$ and responses $\{R_{\mathbf{c}} : \mathbf{c} \in \mathcal{C}\}$ from an APUF instance, $\gamma \in [0, 1]$.
Output: abnormality decision for each $\delta_i^{(x)}$.

for $i \leftarrow 1$ **to** n **do**
  // Form 4 sets of responses for $\mathcal{C}_{i,\pm}^{(x)}$
  initialize $pos0 = neg0 = pos1 = neg1 = \emptyset$
  for $\mathbf{c} = (c_1, c_2, \cdots, c_n) \in \mathcal{C}$ **do**
    case $c_i = 0, c_{i+1} + \cdots + c_n$ *is even* **do**
      $pos0 = pos0 \bigcup \{R_{\mathbf{c}}\}$
    case $c_i = 0, c_{i+1} + \cdots + c_n$ *is odd* **do**
      $neg0 = neg0 \bigcup \{R_{\mathbf{c}}\}$
    case $c_i = 1, c_{i+1} + \cdots + c_n$ *is even* **do**
      $pos1 = pos1 \bigcup \{R_{\mathbf{c}}\}$
    case $c_i = 1, c_{i+1} + \cdots + c_n$ *is odd* **do**
      $neg1 = neg1 \bigcup \{R_{\mathbf{c}}\}$
  end
  // Compute $B_s(\mathcal{C}_{i,\pm}^{(x)})$ and $D_i^{(x)}$ at i.
  $Bpos0 \leftarrow$ ratio of $+1$ in $pos0$
  $Bneg0 \leftarrow$ ratio of $+1$ in $neg0$
  $Bpos1 \leftarrow$ ratio of $+1$ in $pos1$
  $Bneg1 \leftarrow$ ratio of $+1$ in $neg1$
  $D0 \leftarrow Bpos0 - Bneg0$
  $D1 \leftarrow Bpos1 - Bneg1$
  /* Diagnose on $\delta_i^{(0)}$ and $\delta_i^{(1)}$ based on the given threshold of γ. */
  if $|D0| > \gamma$ **then**
    report $\delta_i^{(0)}$ as abnormal with $sign(D0)$
  if $|D1| > \gamma$ **then**
    report $\delta_i^{(1)}$ as abnormal with $sign(D1)$
end

1) Effectiveness for detection and diagnosis: The Monte-Carlo simulation considers 1000 bad APUFs with one to eight

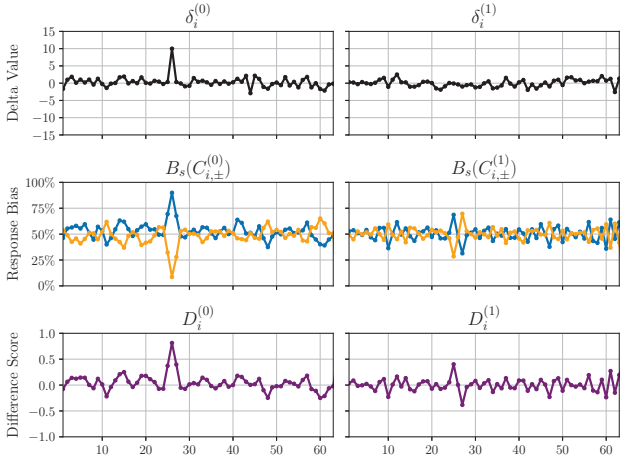


Fig. 8: δ-element values (top), corresponding response biases (middle, $B_s(\mathcal{C}_{i,+}^{(x)})$ in blue and $B_s(\mathcal{C}_{i,-}^{(x)})$ in orange), and the difference scores (bottom) of 1000 randomly chosen challenges on a 64-bit APUF with an abnormal $\delta_{25}^{(0)} = 10$.

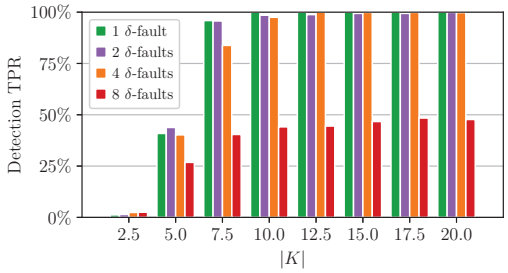


Fig. 9: Detection rate of bad 64-bit APUFs increases with larger $|K|$, and decreases with more δ-faults.

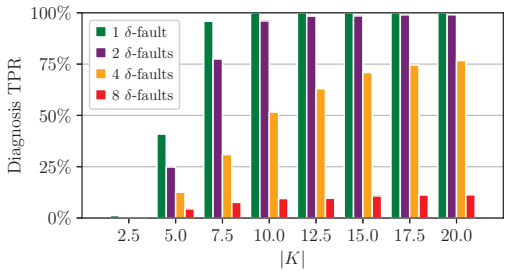


Fig. 10: Diagnosis rate of abnormal δs in bad 64-bit APUFs increases with larger $|K|$, and decreases with more δ-faults.

randomly selected δ-faults with $\delta_i^{(x)} = K\sigma$ ($|K| \in [2.5, 20]$), and $\gamma = 0.5$[1]. For TPR, detection results are shown in Fig. 9, and diagnosis results in Fig. 10. FPR results are in Table I.

TPR for bad APUF detection: As shown in Fig. 9, when $|K| \geq 10$ our method delivers TPR $> 97.5\%$ for bad APUFs with up to 4 abnormal delta elements. When $|K| \leq 5$, bad APUFs are naturally hard to detect, as they are not that different from good APUFs (and their uniqueness from (16) is close to ideal, at $(1-Q(5/\sqrt{64}))Q(5/\sqrt{64}) \approx 0.459$). Detection rate is reduced to $< 50\%$ as the number of abnormal δs increases to 8. The existence of more abnormal δs reduces the "significance" of each delta in the difference score, and other large deltas "cancel" the boosting effect of the targeted delta.

TPR for abnormal $\delta_i^{(x)}$ diagnosis: Fig. 10 shows similar trends in the diagnosis ability of the proposed method: abnormal delta elements can be precisely identified when there are roughly 4 or fewer of them having large magnitudes ($|K| > 5$).

FPR for detection and diagnosis: As shown in Table I, FPR can be kept quite low with $\gamma \geq 0.5$, for both detection and diagnosis. This means very few good APUFs (or normal delta elements) are mistaken for bad ones (or abnormal ones).

2) Trade-off between TPR and FPR: Both the value of K (threshold abnormal delta elements) and the choice of γ (test threshold for $D_i^{(x)}$) have a large impact on TPR and FPR. Fig. 11 shows the Receiver-Operating Characteristics (ROC) curves indicating the trade-offs between FPR (x-axis) and TPR (y-axis) for various K and γ selections. Plots like this can be used to choose γ for a desired TPR or FPR. An ideal choice should reside on the upper-left corner (FPR $= 0$, TPR $= 1$).

[1] $\gamma = 0.5$ works well for large K, but results could be improved for small K values by choosing a different γ, as will be described in the next subsection.

γ	0.1	0.2	0.3	0.4	0.5	0.6	0.7
detection FPR%	100	100	100	58.40	7.00	0.10	0
diagnosis FPR%	50.37	17.75	4.18	0.67	0.06	0	0

TABLE I: False positive results from 1000 good 64-bit APUFs.

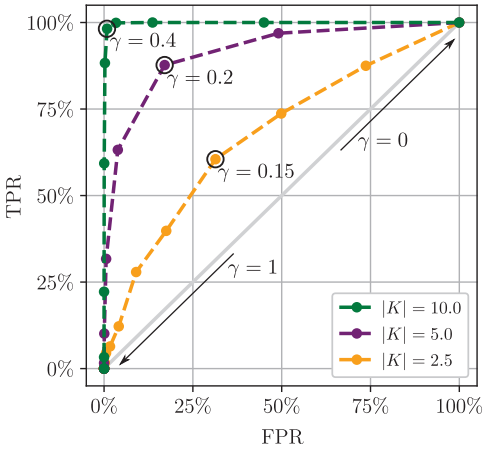


Fig. 11: Trade-off between TPR and FPR via ROC plot of diagnosis results, on 64-bit bad APUFs with a single δ-fault.

V. Conclusion

We have presented new statistical models for native APUF faults and demonstrated the impact of these faults on two metrics of importance to APUFs: bias and uniqueness. We then presented a simple, effective testing methodology that detects and diagnoses bad delta elements in individual APUF instances.

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] A. Chandrakasan, W. J. Bowhill, and F. Fox, *Models of Process Variations in Device and Interconnect*, 2001, pp. 98–115.

[5] M. Pelgrom, A. Duinmaijer, and A. Welbers, "Matching properties of mos transistors," *IEEE JSSC*, vol. 24, no. 5, pp. 1433–1439, 1989.

[6] M. M. Yu, D. M'Raïhi, R. Sowell, and S. Devadas, "Lightweight and secure puf key storage using limits of machine learning," in *International Workshop on CHES*. Springer, 2011, pp. 358–373.

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

[8] 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.

[9] D. Chatterjee, A. Hazra, and D. Mukhopadhyay, "Testability Analysis of PUFs Leveraging Correlation-Spectra in Boolean Functions," *arXiv:1810.08821*, 2018.

[10] 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.

[11] D. Lim, "Extracting secret keys from integrated circuits," Master's thesis, MIT, Cambridge, MA, May 2004.

[12] Y. Lao and K. K. Parhi, "Statistical analysis of mux-based physical unclonable functions," *IEEE TCAD*, vol. 33, no. 5, pp. 649–662, 2014.

[13] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.