Using Standardized Quantization for Multi-Party PPUF Matching: Foundations and Applications

Saro Meguerdichian and Miodrag Potkonjak
Computer Science Department
University of California, Los Angeles
{saro, miodrag}@cs.ucla.edu

***Abstract*—Hardware-based physically unclonable functions (PUFs) are elegant security primitives that leverage process variation inherent in modern integrated circuits. Recently proposed matched public PUFs (mPPUFs) use a combination of coordinated device aging and gate disabling to create two PUFs that securely realize identical input-output mappings. However, mPPUFs of any reasonable size allow for protocols between only a very limited number of parties. We propose quantization of possible delay values to enable matching of an unbounded number of arbitrary PPUF instances, improving stability in the presence of fluctuations in temperature or supply voltage while maintaining resiliency against a wide number of attacks.**

I. Introduction

Essential wireless security requirements include resiliency against physical and side channel attacks, low energy for communication, storage, and computation, and the ability to realize a variety of public key protocols. Hardware-based physically unclonable functions (PUFs) and more specifically public PUFs (PPUFs) have emerged as the hardware security primitive of choice for low-power embedded systems. A PUF is a multi-input multi-output system that is extremely difficult to reproduce due to physical and technological constraints, with functional dependencies between outputs and inputs that are very difficult to predict. The first silicon PUFs demonstrated by Gassend et al. in 2002 used uniqueness in gate delays caused by process variation (PV) as the secret key, but were limited in the range of applications and the need for storage of challenge-response pairs [1]. In 2009, Beckmann et al. proposed that the gate-level characteristics of a PUF, such as delay or leakage energy, can be published and serve instead as a public key, creating the first PPUF [2].

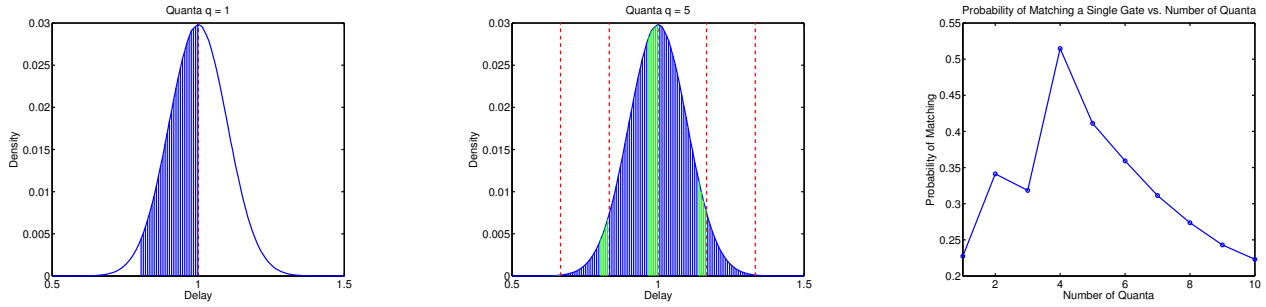
Traditionally, PUFs and PPUFs have been used as authentication devices. Targeted to wireless devices in particular, recently proposed matched public PUFs (mPPUFs) have introduced the ability to conduct secure public-key communication protocols [3]. The mPPUF input-output mapping is dependent on delays of gates and interconnect on the device. Specifically, signal transitions *race* from input flip-flops through an exponential number of paths comprised of multiple levels of logic gates–some of which multiply signal transition frequency and some of which repress signal transitions unpredictably–to *arbiters*, each of which outputs 0 or 1 depending on which of its own input signals (one from the device and one from the clock) transitions first.

The matching technique uses a combination of coordinated device aging (to exactly match delays of corresponding gates) and gate disabling (to disable those gates that cannot be matched) to create two PPUFs that securely realize identical input-output mappings. In the most realistic scenario, a pair or small number of mPPUFs would be matched just after manufacturing, then issued to parties who can then use the matched mPPUF pair for secure communication. A small number of matched mPPUFs can also be issued to a single sensor owner, who can then deploy the matched mPPUFs on sensor nodes. Note that in the preferred realization, the matching is done by the mPPUF owners themselves, removing any requirement to trust the foundry or issuer.

However, secure wireless communication using mPPUFs is restricted to at most 2 or very few parties. In other words, one would need to own 1 separate mPPUF per party or small group of parties in order to communicate securely. This is because matching requires the delays of corresponding gates on each device to be within some delay difference defined by the maximum possible amount of aging for a gate; as the number of devices to be matched increases, the probability that a single gate on all devices has delay within this threshold decreases exponentially. In realistic wireless systems, n-to-n public key communication, where an unbounded number of arbitrary parties can communicate securely with each other, is a crucial security protocol.

The key problem is that although the mPPUF and all other previously proposed silicon PUF and PPUF architectures are composed entirely of digital logic, they are essentially analog systems in the sense that their functionality is derived from delay values, which are continuous. We propose that quantization of gate delays into a relatively few number of acceptable values allows two PPUFs to match the same configuration without requiring the aging to be coordinated. In principle, because aging (quantization) of any PPUF instance is done independently of any other PPUF instance,

IEEE/ACM International Conference on Computer-Aided Design (ICCAD) 2012, November 5-8, 2012, San Jose, California, USA



(a) (b) (c)

Fig. 1: Motivating example, showing possible quanta (red dashed) and gate delays that can be aged to match some quantum (shaded), for (a) 1 and (b) 5 quanta, with (c) probability of matching a single gate vs. degree of quantization. In (b), those gates with delay in the green shaded regions can age to two quanta (one is chosen at random).

this approach enables matching of an unbounded number of arbitrary PPUF instances, creating new n-to-n communication protocols where each participant is only required to possess a single PPUF instance. We also demonstrate that this approach fundamentally improves stability to fluctuations in temperature or supply voltage while maintaining resiliency to a wide number of attacks.

A. Motivating Example

Perhaps the best way to understand how PPUF matching works, the limitations of the previously proposed mPPUF matching procedure, and how delay quantization can be used to address these limitations is to consider the following small example, where two PPUF owners Alice and Bob want to match their PPUFs. Assume a very simple process variation (PV) model for delay where individual gate delays follow a Gaussian random distribution with $\mu = 1$ and $\sigma = 0.1$, with min and max delay bounded at 0.5 and 1.5, respectively. Also assume an aging model where maximal aging of a gate increases its delay by 0.2. Meguerdichian et al. proposed a scheme for coordinated PPUF matching where Alice ages all her gates that are faster by less than 0.2 than Bob's corresponding gates, Bob does the same, and Alice and Bob both disable all other gates [3].

Although the authors show that this matching strategy results in a small and low power PPUF that Alice and Bob can use to conduct public key communication, it has two key limitations that we seek to address in this paper. The first is that the resulting PPUF does not afford any additional stability against variations in temperature and supply voltage over the original PPUF. This is due to the fact that for any pair of gates consisting of one gate from Alice's PPUF and its corresponding gate on Bob's PPUF, at most one is aged to match the exact delay of the other. Consequently, there is a high resolution of gate delays on the resulting PPUF, derived from already existing delays on one PPUF or the other. The second and perhaps more crucial limitation is that the approach does not scale to multi-party communication with a large number of parties. In other words, one party would require a separate PPUF to communicate with each other party (or very small groups of parties).

In this paper, we solve both of these problems by quantizing the possible gate delays for each gate. Now, for each gate, there are one or more preset delay values, or quanta, which are considered valid. In this scheme, for each gate on her PPUF, there are 3 distinct cases for the actions that a PPUF owner, Alice, can take (independently, upon receiving the PPUF): (i) if the gate can be aged to match a single quantum, Alice ages the gate to that quantum; (ii) if the gate can be aged to match more than one quantum, Alice selects a quantum at random and ages the gate to that quantum; or (iii) If the gate cannot be aged to match any quanta, Alice disables the gate.

With quantization, to match their PPUFs, Alice and Bob first take one of the above steps for each gate independently of one other. Figures 1a-1b show the described PV and aging models and specified quanta for 1 and 5 quanta, respectively; shaded regions correspond to gate delays that can be aged to match some quantum. Note that for ease of illustration we present here a naive quantization strategy that simply divides the region of possible delay values uniformly.

There are three key observations regarding the quantization process. The first is that the acceptable delay values (quanta) are specified pre-silicon and are the same for all devices, but differ between gates on a single device. The second is that each PPUF owner quantizes his or her own PPUF instance to match the specified quanta, removing the requirement of trust of any third party. Finally, because each PPUF is quantized independently, without any coordination, it can be matched dynamically to any other arbitrary PPUF instance.

The probability of matching a single gate between two parties using this strategy is shown in Figure 1c. Note that probability of matching decreases after the number of quanta increases larger than $q = 4$, since quanta begin to overlap at that point and are chosen at random for matching. Nevertheless, a subset of Alice's PPUF matches exactly, in terms of delay, a subset of Bob's PPUF. If Alice and Bob disable all remaining gates (i.e. prevent them from switching), they will have PPUF instances that realize identical input-output mappings. According to Figure 1c, for $q = 2$, Alice and Bob match roughly 35% of their PPUF; in other words, after matching they each have an identical PPUF roughly 35% of the size of the original PPUF.

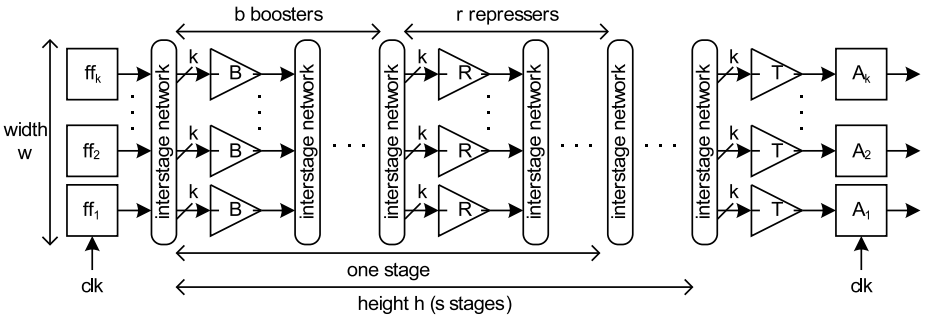


Fig. 2: mPPUF architecture of width w and height h, consisting of s stages of b booster cells (B) and r represser cells (R), with a final level of terminator cells (T) [3]. Signals from input flip flops (ff) race against the clock to arbiters (A).

As in the previous approach proposed by Meguerdichian et al., after matching Alice and Bob have identical resulting PPUFs that they can use to conduct secure communication. In our proposed approach, because Alice and Bob aged their gates independently of each other, each one can match their PPUF with any arbitrary number of other parties by simply enabling and disabling different subsets of gates. Because enabling and disabling gates can be done in real time by applying the proper inputs to gates, there is no additional overhead in communication or computation latency.

Furthermore, because there are a discrete and relatively small number of possible delay values for each gate, quantization improves stability of the PPUF to variations in temperature or supply voltage, since there are larger gaps between acceptable delay values at each level of gates. However, this comes at the cost of security properties like entropy and unpredictability, which are direct consequences of the size of the PPUF, since with quantization the resulting PPUF is smaller than in the previously proposed approach. This is rectifiable by increasing the size of the PPUF, which is already orders of magnitude smaller and more energy efficient than other hardware security primitives.

II. Related Work

The first PUF was introduced in 2002 by Pappu et al. and used a mesoscopic optical system [4]. Shortly thereafter, Gassend et al. extended the PUF paradigm into the silicon world, providing a silicon PUF realization and laying a foundation for a decade of rapid advancement in hardware security [1]. In 2009, Beckmann et al. proposed the first PPUF structure and used it to construct secure protocols for the exchange of secret keys and for public key cryptographic communication [2]. A conceptually similar approach was proposed at the University of Munich [5]. Since then, multiple PPUF architectures have been presented [6][7].

More recently, device aging-based PUFs have emerged, which leverage both PV and device aging [8][9]. The mPPUF introduced the ability of PPUFs to perform secure public key communication between a single pair or a very limited number of parties [3]. In this paper, we use quantization to enhance the security protocols afforded by PPUF matching, both enabling n-to-n public key communication and enhancing resiliency to variations in temperature and supply voltage.

III. Preliminaries

We use the delay model proposed by Markovic et al [10], shown in Equation 1, with supply voltage V_{dd}, subthreshold slope n, mobility μ, oxide capacitance C_{ox}, gate width W, gate length L, thermal voltage $\phi_t = (kT/q)$, drain induced barrier lowering (DIBL) factor σ, threshold voltage V_{th}, load capacitance C_L, and delay and model fitting parameters k_{tp} and k_{fit}.

$$Delay = \frac{k_{tp} \cdot C_L \cdot V_{dd}}{2 \cdot n \cdot \mu \cdot C_{ox} \cdot \frac{W}{L} \cdot (\frac{kT}{q})^2} \cdot \frac{k_{fit}}{(ln(e^{\frac{(1+\sigma)V_{dd}-V_{th}}{2 \cdot n \cdot (kT/q)}} + 1))^2} \quad (1)$$

For the effect of PV on L, we follow the quad-tree model proposed by Cline et al. [11]. For V_{th}, we adopt the Gaussian distribution proposed by Asenov et al. [12]. Note that gates are not correlated in terms of V_{th}, and no other parameters in Equation 1 are significantly impacted by PV.

NBTI occurs when a negative voltage is applied between the gate and source of a PMOS transistor, placing the transistor under stress and causing its V_{th} to increase over time [13]. We use the aging model proposed by Chakravarthi et al. and shown in Equation (2) for the effect of device aging due to NBTI on V_{th} shift, where A and β are constants, V_G is the applied gate voltage, E_α is the measured activation energy of the NBTI process, T is the temperature, and t is time [14].

$$\Delta V_{th} = A \cdot e^{\beta V_G} \cdot e^{-E_\alpha / kT} \cdot t^{0.25} \quad (2)$$

Note that aging can be done to a great extent relatively quickly if done on purpose (i.e. if the stress is applied continuously). Furthermore, some input vectors will age a particular gate more than others. For example, in a standard CMOS NAND gate with 2 PMOS transistors, the input vector 11 will not place either PMOS transistor under stress, since both the gate and source will have voltage V_{dd}. Input vector 00, on the other hand, will place both transistors under stress, as the source will have voltage V_{dd} and the gate will have voltage 0. Finally, input vectors 01 and 10 will each place one of the PMOS transistors under stress. Note that for a CMOS NAND

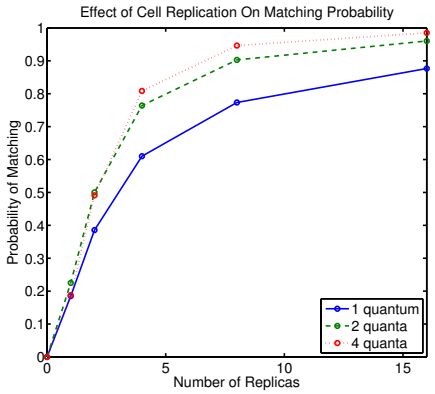


Fig. 3: Effect of cell replication on probability of matching, for 1 (blue solid), 2 (green dashed), and 4 (red dotted) quanta.

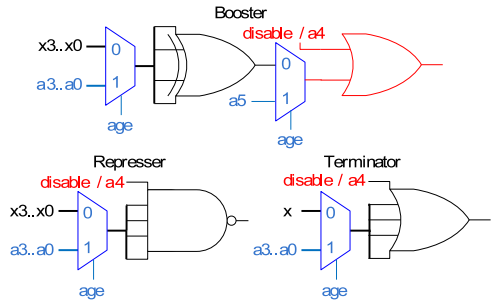


Fig. 4: Aging (blue) and disabling (red) logic for booster, represser, and terminator cells (black). Setting the "disable" input to 1 (XOR, OR) or 0 (NAND) freezes the cell's output to 1. Setting the "age" input to 1 places the cell in aging mode, where aging inputs a_i are applied for maximal aging.

gate, each input-output path (single PMOS transistor) can be aged independently. In normal operation, the gates are placed under and removed from stress very often, and not always with the maximally aging input vector, resulting in immediate and nearly full recovery after each aging cycle.

IV. Architecture

Our quantized PPUF uses an augmentation of the mPPUF architecture proposed by Meguerdichian et al., which we reproduce in Figure 2 for convenience [3]. We briefly summarize the key architectural features of the mPPUF, then present our augmentations in Sections IV-A and IV-B, concluding with a discussion about strategies for quantization in Section IV-C.

Inputs from w flip-flops race against the clock through the various input-output paths in the circuit to w arbiters. An arbiter has 2 inputs, x and y; it outputs 0 if input x transitions from $0 \to 1$ before input y and 1 otherwise. In other words, each arbiter outputs 1 if and only if the first $0 \to 1$ transition of its input (from the final stage) occurs before the $0 \to 1$ clock transition.

The mPPUF architecture has width w and consists of s stages. Each stage is comprised of b k-input booster cells and r k-input represser cells. Booster cells serve to increase the switching frequency of a propagating signal while represser cells repress frontier signal transitions and increase simulation complexity and unpredictability. The design has height $h = s \cdot (b + r)$, where the w cells in each sequential level take as inputs the outputs of the k cells in the previous level. A final level of w 1-input terminator cells are added to enhance stability. While the previous authors describe the implementations of booster, represser, and terminator cells in detail, for the purposes of our discussion we can consider them to be essentially k-input XOR, NAND, and OR gates.

A. Cell Replication

One of the ramifications of delay quantization compared to coordinated gate delay matching is that a significantly smaller subset of cells can be matched between two arbitrary parties. A naive way to circumvent this is to simply increase the size of the PPUF to maintain the size of the resulting matched subset. However, the probability of matching is not the only determinant of whether or not a cell remains active in the resulting PPUF; at least one of the cell's inputs must also match and be active.

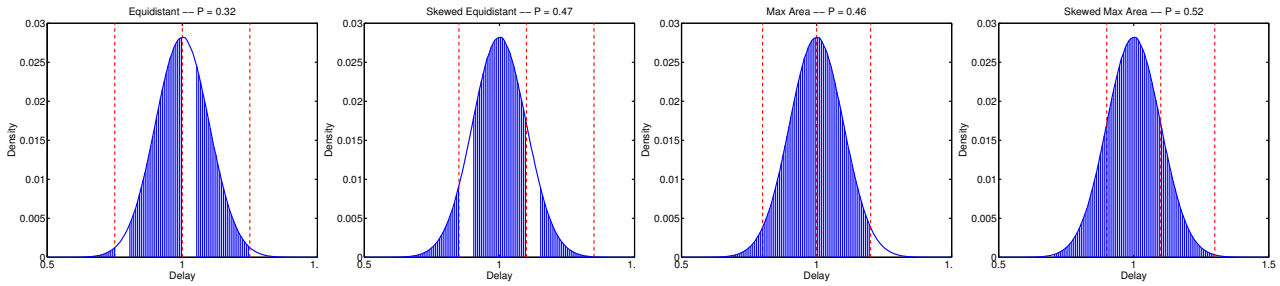
Consider an example where the probability of matching a cell is 0.2, and each cell on the PPUF has 4 inputs. After matching, then, a first-level cell is active with probability 0.2. A second level cell is active if and only if it matches and one of its inputs match. This probability is in fact $0.2 \cdot (1 - 0.8^4) = 0.12$, and continues to decrease every level, dipping below 1% at just the fourth level of cells. Therefore, increasing the size of the PPUF is not a solution by itself.

To augment this strategy, we replicate and multiplex PPUF cells. Now, quantization is done with respect to the combined delay of the computation cell and the multiplexer. Consequently, for r replicas, a cell on one PPUF has r^2 chances to match the corresponding cell on the other PPUF, instead of just one, resulting in a matching probability of $P(r) = 1 - (1 - P(1))^{r^2}$, where r is the number of replicas and $P(1)$ is the original probability of matching with only one replica. For our architecture, the probability of matching vs. number of replicas is shown in Figure 3, where $P(1) \approx 0.2$. Note that the plotted probabilities do not follow the formula exactly, since gate delays are correlated, especially the r input-output delays of each cell's multiplexer; the formula represents an upper bound.

B. Aging and Disabling Logic

To allow for fast and low-energy device aging, we augment our PPUF architecture by adding individual gate input control. The target gates on the PPUF can be aged by their respective desired amounts by supplying the gates with their maximally aging inputs. Furthermore, we add the following disabling logic to facilitate gate disabling: *booster* – a 2-input OR gate is added to the output of the XOR gate and input 1 is applied; *represser* – a primary input is added to the NAND gate and set to 0; or *terminator* – a primary input is added to the OR gate and set to 1. Note that disabling a gate simply prevents it from switching. This is akin to setting its delay to ∞.

We present our proposed aging (blue) and disabling (red)



(a) (b) (c) (d)

Fig. 5: Four quantization strategies and corresponding probabilities of matching: (a) Equidistant (P = 0.32); (b) Skewed Equidistant (P = 0.47); (c) Max Area (P = 0.46); and (d) Skewed Max Area (P = 0.52).

logic for booster, represser, and terminator cells (black) in Figure 4. A single disable input is added to represser and terminator cells, while an OR gate and disable input are added to booster cells. Data inputs to the cells are multiplexed with inputs that can be used for aging if the age select input is activated. The disable input can be used as another input for aging control. Note the aging input vector can be applied to target specific input-output delays or to age the gate maximally.

C. Quantization

There are two directly conflicting yet crucial desiderata in the selection of valid delay values (quanta). The first is *high probability of matching between two legitimate parties.* The probability that a cell aged to a particular quantum on one PPUF matches its corresponding cell aged to a quantum on another PPUF is dependent on the delay distribution from PV, the maximum delay increase possible from device aging, and the distribution of delay quanta. One goal of the quantization strategy for a cell, then, is to ensure that for every possible delay value determined by PV, at least one quantum is reachable by increasing delay from device aging. However, even to achieve this single goal, quanta placement is a non-trivial optimization problem, since the delay distribution and aging possibilities follow complex correlated PV and aging models that vary from gate to gate.

Furthermore, distributing quanta across the distribution to maximize the area covered, for example, is likely suboptimal, since cell delays are not uniformly distributed. Consider the case where delay follows a Gaussian distribution with $\mu = 1$ and $\sigma = 0.1$, and max aging results in a delay increase of 0.2. Here, although placing quanta at 1 and 1.2 results in a greater probability for a cell to be able to reach a quantum ($P(0.8 < d < 1.2 = 0.95)$ than placing quanta at 0.9 and 1.1 ($P(0.7 < d < 1.1) = 0.84$), we see that the probability of matching is greater in the latter case ($P^2(0.7 < d < 0.9) + P^2(0.9 < d < 1.1) = 0.49$ while $P^2(0.8 < d < 1.0) + P^2(1.0 < d < 1.2) = 0.46$).

The second desideratum is *low probability of matching of an attacker's PPUF to the same configuration legitimately matched by a pair of PPUFs.* In order for protocols to maintain security, we must guarantee that the subset of cells that match between two PPUFs is unique between every pair of PPUFs. In other words, if a third PPUF can match the same configuration as two matched PPUFs, the security of the communication link between the two legitimately matched parties is compromised.

We make the conservative assumption that the third PPUF is malicious, and can configure itself with the goal of matching a pair of matched PPUFs. In this case, the attacker's cell matches the configuration if: a) the corresponding matched cell is disabled; or b) the attacker's cell can be set using device aging to the same quantum to which the enabled cell is set.

For example, assume the case that there is only a single possible quantum, and the resulting probability of matching is 0.2. The probability of attack, then, is $0.8 + 0.2^2 = 0.84$, the sum of the probabilities of the two mutually exclusive cases a) and b) described above. As the number of quanta and the probability of matching increase, this probability in general decreases. However, if there are overlapping quanta, and cells on legitimate PPUFs are configured to available quanta at random, the attacker's chance in reality *increases*. This is because the attacker has the freedom to choose the quanta to which he ages his gates *a posteriori*, unlike the two legitimate parties who age their gates *a priori* to enable real-time matching of an arbitrary number of other parties.

The first desideratum is crucial due to the fact that security of the PPUF is a direct consequence of its size. As the probability of matching increases, so does the size of the matched PPUF subset. Therefore, a high probability of matching enhances security properties of the resulting PPUF.

However, it is important to note that the second desideratum is in direct conflict with the first. An extraordinarily high probability of matching (near 1.0) leads in turn to a very high probability of attacking (or matching one PPUF to the configuration matched by another pair of PPUFs). At its extreme, this results in the matched PPUF being the entire PPUF without any disabled cells. However, a defense against this type of attack can in principle be satisfied to any arbitrary degree of certainty (as long as the probability of matching is not 1.0) by increasing the size of the PPUF, as the attacker would need to match the exact configuration across the entire PPUF, for every cell.

We propose and analyze the following four different strategies for delay quanta placement, summarized in Figures 5a-5d for a simple Gaussian delay PV model with $\mu = 1$ and

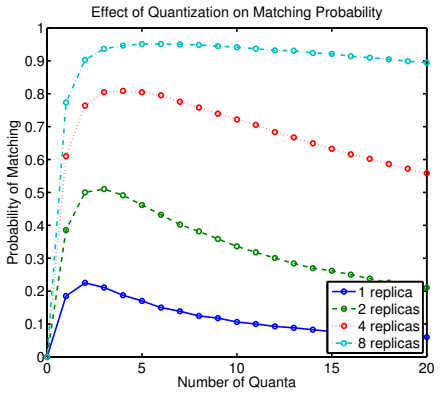


Fig. 6: Effect of quantization strategy (ii) on probability of matching, for 1 (blue solid), 2 (green dashed), 4 (red dotted), and 8 (teal dashdot) replicas.

$\sigma = 0.1$, max delay aging of 0.2, and 3 quanta.

(i) *Equidistant (Matching Probability: 0.32).* Quanta are placed such that they span uniformly across the entire distribution of delays. This is the simplest strategy and is reasonable if expected delay distributions and maximal aging amounts are unknown ahead of time.

(ii) *Skewed Equidistant (Matching Probability: 0.47).* Quanta are placed as in (i) but offset by half the maximum amount of delay aging. The motivation behind the forward offset is that aging can only increase delay but cannot decrease it. Skewing the quanta by this amount balances (and therefore increases) the portion of the delay distribution that can be aged to match a quantum. This strategy is preferred over (i) when maximal aging of a cell can be predicted easily.

(iii) *Max Area (Matching Probability: 0.46).* Quanta are placed such that the maximum area of cell delays lie between each quanta. For delay distributions that are non-uniform, the strategy proposed in (i) is suboptimal, since the goal is to maximize the number of cells that can match quanta. However, this approach is only feasible if the delay distribution is known *a priori*.

(iv) *Skewed Max Area (Matching Probability: 0.52).* Quanta are placed as in (iii) but offset by half the maximum amount of delay aging, following the same motivation as (ii). This is the preferred strategy if both the delay distribution and the maximal delay aging of the cell are known or can be predicted relatively accurately *a priori*, since it maximizes the probability of matching.

There are a number of caveats to selecting an effective quantization strategy. The first is that for some protocols it is important that a cell can be set to one of multiple quanta. Therefore, when the number of quanta is high and quantization intervals overlap, a quantum is chosen at random for a particular cell. Although this may decrease the probability of matching (Figure 6) and increase the probability of attacking (since the attacker can choose *a posteriori* which quantum to match), it can instead result in increased size (and therefore security) of the matched PPUF subset if the delay distribution cannot be predicted accurately.

Algorithm 1 n-Party Public Key Exchange

1: **for all** $1 \leq i \leq n$, $1 \leq j \leq n$, $i \neq j$ **do**
2: **for all** gates g on the PPUF of party P_i **do**
3: P_i sends P_j $\log(q)$ bits denoting to which quantum gate g has been aged.
4: **end for**
5: **for all** gates g on the PPUF of party P_j **do**
6: P_j sends P_i a single bit denoting whether or not that gate matches and should be enabled for communication.
7: **end for**
8: **end for**

Another important observation is that if all cells have identical quanta, then security of the PPUF is compromised as there are only very few possible values for the initial arrival time of PPUF output signals. Therefore, we add a small random component to the chosen quanta at each cell to ensure that each cell has a unique set of achievable quanta.

V. Operation

A. n-Party Public Key Exchange

In this section, we propose a procedure through which two PPUF owners Alice and Bob can match a subset of their PPUFs to the same configuration. For the purposes of the following discussion, assume that each cell has been either aged such that its delay matches a quantum or disabled if it could not match any quanta, following one of the strategies posed in Section IV-C. Recall that in the previous proposed mPPUF scheme, Alice and Bob conducted coordinated device aging for matching. In our approach, Alice and Bob perform the aging process (quantization) independently of each other and only once, upon receiving their devices. Consequently, the matching process requires only the following sequential exchange of information between Alice and Bob.

First, Alice sends Bob her quantization information for each cell, denoting to which quantum that cell's delay is set (if any). This corresponds to $\log(q) \cdot n$ bits of communication, where q is the number of quanta and n is the number of cells on the PPUF. At this point, Bob knows which cells can match between his and Alice's PPUFs. Then, Bob transmits to Alice a single bit for each cell, denoting whether or not Alice should disable that cell in order to communicate with Bob. This adds another n bits of communication.

As a result, Alice and Bob both know which cells they must disable in order to have only an identical subset of their PPUFs enabled. When Alice and Bob wish to communicate, they must complete the matching process by applying the disable inputs to the proper gates, which can be done in the same clock cycle and is by no means irreversible. Only $(\log(q)+1)\cdot n$ bits worth of communication energy is expended, only once per pair of communicating parties.

We present the exchange of information required for matching in Algorithm 1 as a public key exchange protocol between

Algorithm 2 Public Key Communication

1: P_i and P_j match their PPUFs by applying the enable/disable bits from the public key exchange, each enabling a subset of the PPUF that realizes the same encryption function E_{ij}.
2: P_i chooses a random string r_{ij} and computes the hash $R_{ij} = E_{ij}(r_{ij})$.
3: P_i computes the ciphertext $y_{ij} = x_{ij} \oplus R_{ij}$.
4: P_i transmits y_{ij} and r_{ij} to P_j
5: P_j computes the hash $R_{ij} = E_{ij}(r_{ij})$.
6: P_j recovers the original plaintext message $x_{ij} = y_{ij} \oplus R_{ij}$.

n arbitrary parties $P_1 ... P_n$, each with a unique quantized PPUF. Note that no confidential information is leaked, since the configurations of the involved PPUFs are public information regardless. Note also that this protocol must be conducted only one time for each pair of parties for the lifetime of the device.

B. n-to-n Public Key Communication

Enabling secure, trusted n-to-n public key communication is one of our key motivations and contributions. We define n-to-n public key communication as that between an unbounded number of arbitrary parties. Public key communication has not been realized using PUFs until very recently (2011) with the introduction of the mPPUF, but even then it was limited between essentially only two parties.

Algorithm 2 describes the public key communication protocol as a natural extension of the public key exchange protocol, where a party P_i wants to send securely a secret plaintext message x_{ij} to an arbitrary party P_j. The communication is secure because the only information transmitted in the exchange are the random string r_{ij}, which is independent of the plaintext, and the ciphertext y_{ij}. Note that x_{ij} cannot be recovered from y_{ij} unless $E_{ij}(r_{ij})$ is computed, which due to the security properties we show in Section VI cannot be done (or predicted using knowledge of r_{ij}) by anyone other than the owners of the matched PPUF subset, in this case only P_i and P_j.

It is crucial to note that because PPUF reconfiguration by disabling different subsets can be done in the same clock cycle as the PPUF computation, any party can communicate securely with any other party using the corresponding public key and very little overhead. The only ramification is that one party cannot broadcast a message to all parties in the group, but must instead send a message using the correct key individually to each recipient.

VI. Resiliency against Attacks

In this section, we identify potential security attacks against (i) individual PPUFs and (ii) matched PPUFs. For statistical attacks, we conducted comprehensive simulations using PPUFs of width 128, consisting of 7 stages of 2 boosters and 1 represser, with 8 replicas and 2 quanta for each cell (unless otherwise specified). We present the results for each simulation using 10,000 input vectors.

A. Individual PPUF

Potential security attacks against individual PPUFs include guessing, simulation and technological attacks. In guessing attacks, the attacker observes a polynomial number of challenge-response pairs and tries to statistically analyze them in order to predict the answer to an unseen challenge. It is important to observe that in this case, there is no real benefit to selecting a training set for statistical modeling. Technological attacks are mainly based on cloning, emulation, side channel, and physical attacks.

1) Prediction attacks: In these attacks, the attacker tries to predict each output O_i using knowledge of previously observed input-output pairs. Specifically, his goal is to predict $P(O_i = c)$, where $c = 0$ or 1. Figure 7a shows the results of simulation for one representative PPUF subset after matching. Ideally, an output is 0 or 1 with equal likelihood (probability 0.5, shown as a red dashed line in the figure). However, protocols can choose to use only those outputs which are most unpredictable.

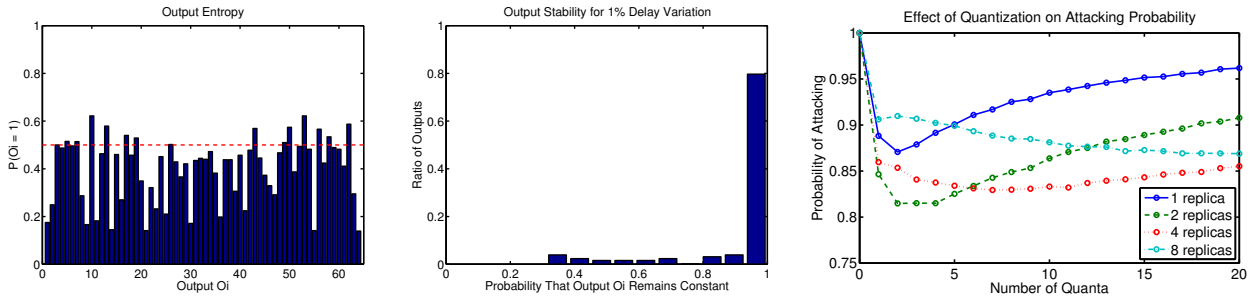
2) Side channel attacks: Side-channel attacks are not a threat, since the power profile of a gate would not reveal any new information. Gate delays are available as a public key.

3) Emulation attacks: An emulation attack is possible if the attacker can create an IC with gates with a factor larger timing delay, but with the same relative timing characteristics, using either older technology that is not as susceptible to PV or a field programmable gate array (FPGA). However, because gate delays are quantized, we ensure resiliency to this type of attack by using multi-level input staggering. With this scheme, we can apply primary input signals to any cell using the aging inputs shown in Figure 4, thereby rendering it impossible for an attacker to use only relative gate delays. Any cloning or emulation attack must match all delays exactly.

B. Matched PPUFs

Finally, attacks against matched PPUFs are essentially protocol attacks that try to exploit vulnerabilities in policies for the exchange of data between two matched PPUF holders. In order to perpetrate this kind of attack, an attacker must configure using aging a third PPUF to match the two honestly matched PPUFs. In order to *a posteriori* match the same configuration, the attacker must be able to match every honestly matched cell in terms of delay.

We present in Figure 7c simulation results for the probability of such an attack for a single cell, vs. number of quanta. The key observation here is that an attacker must match every cell across the entire PPUF as the legitimately matched parties in order to launch a successful attack; we can see that for even modest PPUF sizes this probability is effectively zero. For instance, for a relatively high probability of 0.9 of attacking a single gate (high replication and quantization), the probably of matching the entire PPUF is on the order of 10^{-92} for even just 2000 gates, while the legitimate parties match roughly 1800 gates.



(a) (b) (c)

Fig. 7: Probability that an output bit of a matched PPUF (a) equals 1 and (b) remains the same in the presence of delay variations of up to 1%. (c) Effects of delay quantization for 1, 2, 4, and 8 replicas on the probability of attacking for a single cell, using quantization strategy (ii) and random arbitration of quanta.

VII. Stability against VT Variations

For authentication protocols, there are two key mechanisms for attaining stability against environmental and random effects that cause variations in supply voltage and temperature (VT) and therefore device delay. The first is that because of the difficulty of predicting or simulating PPUF outputs, it is not necessary to verify that the party in question can produce *all* output bits without error. The second is a direct application of the above observation: if an output bit is mismatched, the authenticating party requests the exact measured arrival time of the output bit and accepts it if it is close enough to the expected value.

In many other protocols it is crucial that every bit is transmitted correctly. For example, in public key communication, the transmitted message can only be recovered if the random string's hash was computed identically by both parties. In this case, both parties can capture their PPUF outputs at multiple clock rates, also transmitting error correction code encoding the differences between the various cases.

However, delay quantization provides intrinsic stability of PPUF outputs. As a simple but powerful illustration, consider the case where all delay quanta across all cells are set with granularity 0.5. Then, all signal transitions should occur at multiples of 0.5. This inherently creates stability in the sense that an output signal that is measured to arrive at time 4.397 can be assumed to actually arrive at time 4.5, since this is the closest possible acceptable time as specified by the granularity of the quantization strategy. We model delay variation due to any effects as uniform random noise in our simulations, and present the results for 1% delay variation in Figure 7b. We see that a majority of the outputs remain constant in the presence of delay variations.

VIII. Conclusion

We have introduced the concept of delay quantization for PPUF matching, where acceptable delays for PPUF cells are restricted to a small number of pre-specified quanta at coarse granularities. As a result, a PPUF can be dynamically matched to any other arbitrary PPUF instance by disabling different gates and leaving only that subset active whose cells are aged to the same quanta as the target PPUF. This benefit comes at the requirement of a slightly larger PPUF, which we have shown to maintain all the security properties that make PPUFs so powerful, while also retaining a low probability of coincidence with any malicious PPUF. Thus, we have presented the first n-to-n public key communication protocols using PPUFs, enabling secure communication between an unbounded number of arbitrary PPUF owners. Furthermore, we have demonstrated that delay quantization inherently provides stability against delay fluctuations.

Acknowledgment

This work was supported in part by the NSF under awards CNS-0958369, CNS-1059435, and CCF-0926127, and is based upon research performed in collaborative facilities renovated with funds from the National Science Foundation under Grant No. 0963183, an award funded under the American Recovery and Reinvestment Act of 2009 (ARRA).

References

[1] B. Gassend et al., "Silicon physical random functions," *ACM CCS*, pp. 148–160, 2002.
[2] N. Beckmann and M. Potkonjak, "Hardware-based public-key cryptography with public physically unclonable functions," *IH*, pp. 206–220, 2009.
[3] S. Meguerdichian and M. Potkonjak, "Matched public PUF: ultra low energy security platform," *IEEE/ACM ISLPED*, pp. 45–50, 2011.
[4] R. Pappu et al., "Physical one-way functions," *Science*, vol. 297, no. 5589, pp. 2026–2030, 2002.
[5] U. Rührmair, "SIMPL systems, or: can we design cryptographic hardware without secret key information?" *SOFSEM*, vol. 6543, pp. 26–45, 2011.
[6] M. Potkonjak, S. Meguerdichian, and J. L. Wong, "Trusted sensors and remove sensing," *IEEE Sensors*, pp. 1104–1107, 2010.
[7] M. Potkonjak et al., "Differential public physically unclonable functions: architecture and applications," *IEEE/ACM DAC*, pp. 242–247, 2011.
[8] S. Meguerdichian and M. Potkonjak, "Device aging-based physically unclonable functions," *IEEE/ACM DAC*, pp. 288–289, 2011.
[9] S. Meguerdichian and M. Potkonjak, "Security primitives and protocols for ultra low power sensor systems," *IEEE Sensors*, pp. 1225–1228, 2011.
[10] D. Markovic et al., "Ultralow-power design in near-threshold region," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 237–252, 2010.
[11] B. Cline et al., "Analysis and modeling of CD variation for statistical static timing," *IEEE ICCAD*, pp. 60–66, 2006.
[12] A. Asenov, "Random dopant induced threshold voltage lowering and fluctuations in sub-0.1 um MOSFETs: a 3-D atomistic simulation study," *IEEE T-ED*. vol. 45, no. 12, pp. 2505–2513, 1998.
[13] M. A. Alam et al., "A comprehensive model of PMOS NBTI degradation," *Microelectronics Reliability*, vol. 45, pp. 71–81, 2005.
[14] S. Chakravarthi et al., "A comprehensive framework for predictive modeling of negative bias temperature instability," *IRPS*, pp. 273–282, 2004.