Reliable Physical Unclonable Functions Using Data Retention Voltage of SRAM Cells

Xiaolin Xu, *Student Member, IEEE*, Amir Rahmati, Daniel E. Holcomb, *Member, IEEE*, Kevin Fu, *Senior Member, IEEE*, and Wayne Burleson, *Fellow, IEEE*

***Abstract*—Physical unclonable functions (PUFs) are circuits that produce outputs determined by random physical variations from fabrication. The PUF studied in this paper utilizes the variation sensitivity of static random access memory (SRAM) data retention voltage (DRV), the minimum voltage at which each cell can retain state. Prior work shows that DRV can uniquely identify circuit instances with 28% greater success than SRAM power-up states that are used in PUFs [1]. However, DRV is highly sensitive to temperature, and until now this makes it unreliable and unsuitable for use in a PUF. In this paper, we enable DRV PUFs by proposing a DRV-based hash function that is insensitive to temperature. The new hash function, denoted DRV-based hashing (DH), is reliable across temperatures because it utilizes the temperature-insensitive ordering of DRVs across cells, instead of using the DRVs in absolute terms. To evaluate the security and performance of the DRV PUF, we use DRV measurements from commercially available SRAM chips, and use data from a novel DRV prediction algorithm. The prediction algorithm uses machine learning for fast and accurate simulation-free estimation of any cell's DRV, and the prediction error in comparison to circuit simulation has a standard deviation of 0.35 mV. We demonstrate the DRV PUF using two applications—secret key generation and identification. In secret key generation, we introduce a new circuit-level reliability knob as an alternative to error correcting codes. In the identification application, our approach is compared to prior work and shown to result in a smaller false-positive identification rate for any desired true-positive identification rate.**

***Index Terms*—Chip identification, data retention voltage (DRV), key generation, machine learning (ML), physical unclonable function (PUF).**

I. Introduction

INTEGRATED circuit instances can be identified or authenticated using nonvolatile static identifiers or through the use of distinguishing physical characteristics. Physical characteristics have several security advantages over static identifiers, including immutability and resistance to cloning and tampering. The physical characteristics can be viewed as an identifying fingerprint of a given device instance. More formally, physical fingerprints are a component of a particular type of physical unclonable function (PUF) [2], [3] that is originally described as a physically obfuscated key [4], and more recently as a weak PUF [5].

If used for identification or constructing secret keys, fingerprint observations must be consistent over time and across different environmental conditions. A fundamental concern in PUFs is to minimize the impact of noise and environmental fluctuations while still being sensitive to the microscopic variations that make each device unique. A common way of minimizing the impact of noise and environment is to use differential circuits. Yet small variations in the fingerprint of a device are inevitable, and much effort is spent on error correction of somewhat-unreliable fingerprints or PUF outputs, or adding significant extra circuitry for calibration [6]. However, error correcting codes and calibration circuitry are expensive in terms of the number of raw bits and silicon resource required to create a reliable key, and more so if a large number of errors must be correctable.

This paper employs data retention voltage (DRV), the minimum supply voltage at which state is retained, as the basis for a new static random access memory (SRAM) PUF. Our previous work [1] has shown DRV fingerprints to be more informative than power-up SRAM PUFs [5], [7]. The physical characteristics responsible for DRV are imparted randomly to each cell during manufacturing, providing DRV with a natural resistance to cloning. DRVs are not only random across chips, but also have relatively little spatial correlation within a single chip and can be treated in analysis as independent [8]. The proposed technique has the potential for wide application, as SRAM cells are among the most common building blocks of nearly all digital systems.

In this paper, we extend the idea of DRV fingerprinting to create a PUF based on DRV. To overcome the temperature-sensitivity of DRV, we propose a DRV-based hashing (DH) scheme that is robust against temperature changes. The robustness of this hashing comes from its use of (reliable) DRV ordering instead of (less reliable) DRV values. The use of DRV ordering can be viewed as a differential mechanism at the logical level instead of the circuit level as in most PUFs. To help validate the DRV PUF, we propose a machine learning (ML) technique for simulation-free prediction of DRVs as a function of process variations and temperature. The ML model

Manuscript received July 2, 2014; revised November 5, 2014; accepted February 9, 2015. Date of publication April 2, 2015; date of current version May 20, 2015. This work was supported in part by the C-FAR, one of six centers of STARnet, the Semiconductor Research Corporation Program through Microelectronics Advanced Research Corporation and Defense Advanced Research Projects Agency, in part by the National Science Foundation under Grant CNS-1331652, Grant CNS-0923313, Grant CNS-0964641, and in part by the Semiconductor Research Corporation task under Grant 1836.074. This paper was recommended by Associate Editor R. Karri.

X. Xu, D. E. Holcomb, and W. Burleson are with the Department of Electrical and Computer Engineering, University of Massachusetts Amherst, Amherst, MA 01003 USA (e-mail: xiaolinx@umass.edu).

A. Rahmati and K. Fu are with the Department of Computer Science and Engineering, University of Michigan, Ann Arbor, MI 48109 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2015.2418288

enables the rapid creation of the large DRV datasets required for evaluating the DRV PUF approach. Our approach is furthermore supported using hardware measurement of DRV data.

The contributions of this paper are as follows.

1) We demonstrate that SRAM DRV can serve as a basis for reliable identification and key generation. This finding is supported by DRV characterizations of 20K SRAM cells measured three times at each of three different temperatures.
2) We present the first work that applies ML for simulation-free prediction of DRV as a function of temperature, process variation assignments, and transistor sizes. Once the ML model is trained, it can predict the DRV of a cell at a given temperature 2.2e6 faster than can circuit simulation, and its prediction error versus circuit simulation has a standard deviation of only 0.35 mV.

The remainder of this paper is structured as follows. Section II reviews related work on PUFs. Section III introduces DRV and some conventional methods of DRV prediction. Section IV explains how the DRVs of SRAM cells are characterized in hardware measurement and circuit simulation. Section V proposes the use of a neural network model for predicting DRV. Section VI presents our DH scheme (DH), secret key generation, and experimental evaluation thereof. Section VII presents the conclusion in this paper.

II. Related Work

A wide variety of PUFs and fingerprints based on custom or preexisting integrated circuit components have been proposed. The identifying features used by custom designs include MOSFET drain-current [9], timing race conditions [2], and the digital state taken by cross-coupled logic after a reset [10]. IC identification based on preexisting circuitry is demonstrated using SRAM power-up state [5], [7], and physical variations of flash memory [11]. Lee *et al.* [12] derived a secret key unique to each IC using the statistical delay variations of wires and transistors across ICs. Circuit-level techniques for increasing the reliability of SRAM PUFs are explored by Bhargava *et al.* [13]. An experimental evaluation of low-temperature data remanence on a variety of SRAMs is provided by Skorobogatov [14], and SRAM remanence in radio-frequency identification (RFID) has been studied by Saxena and Voris generation [15] as a limitation to SRAM-based true random number.

Previous works [16], [17] have used error correction to construct secret keys from noisy PUF sources; however, these approaches are expensive in their required number of gates. Suh *et al.* [18] used a BCH code to correct 21 errors among 127 raw bits to create a 64-bit key. Guajardo *et al.* [5] derived a 278-bit secret key from 1023 bits of power-up SRAM state using a BCH code that can correct up to 102 errors. Maes *et al.* [19] introduced an SRAM helper data algorithm to mask unreliable bits using low-overhead post-processing algorithms. Recently, Yu and Devadas [20] proposed the use of index-based syndrome (IBS) coding for deriving reliable key bits from PUF outputs. A notable feature of error correction using IBS coding in PUFs is that the syndrome does not leak information about the encoded bits. Hiller *et al.* [21] extend IBS coding for SRAM PUFs. Van Herrewege *et al.* [22] have designed a new lightweight authentication scheme using PUFs that does not require storage of a large number of PUF challenge-response pairs.

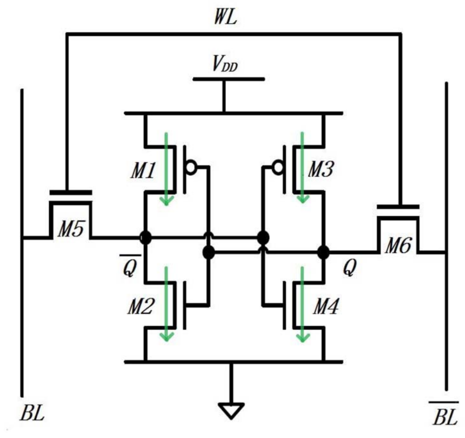


Fig. 1. Six transistor SRAM cell. Q and $\bar{Q}$ are the complementary state nodes that store a single bit value between cross-coupled inverters implemented by transistors $M1$–$M4$. WL is the word line, and controls access transistors $M5$ and $M6$. BL and $\overline{BL}$ are the complementary bitlines used to read and write the SRAM cell. Arrows denote the direction of current leakage.

Compared to the low cost of the SRAM used for DRV fingerprinting, a relatively significant practical cost may be associated with the generation of the test voltages for characterizing the DRVs. Emerging devices such as computational RFIDs [23] can use software routines to extract DRVs, but as contactless devices they must generate all test voltages on-chip. On-chip dynamic control of SRAM supply voltage is assumed in the low-power literature at least since work on drowsy caches [24]. Supply voltage tuning has also been applied with canary cells to detect potential SRAM failures, and as a post-silicon technique to compensate for process variation and increase manufacturing yields [25].

III. Data Retention Voltage of SRAM

An SRAM cell is commonly implemented in CMOS technology as a six-transistor circuit (Fig. 1). When an SRAM cell is in the standby condition, its word line (WL) is set low, and the two access transistors ($M5$ and $M6$) are shut off. If the supply voltage is sufficient, two inverters (composed of $M1, M2$ and $M3, M4$) use positive feedback to pull one complementary state node (Q or $\bar{Q}$) high, and the other low. If supply voltage is below DRV, then transistors operate in the sub-threshold (sub-V_{th}) region [26] where they are highly sensitive to variations and may lose state. Such a loss of state on account of insufficient supply voltage is termed a data retention failure. The voltage at which data retention failures occur in each SRAM cell depends on its asymmetric process variation. Because DRV is randomly assigned to each cell through process variation, the DRV fingerprint of SRAM is a physical fingerprint suitable for use in a PUF.

Since the DRV of SRAM signifies the minimum supply voltage at which cells can store arbitrary state, DRV is usually studied as a lower limit to supply voltage scaling. Most previous literature focuses on cases where the SRAM supply voltage remains safely above DRV. While remaining above DRV, the supply voltage can be adjusted to reduce leakage power [24], [27], compensate for manufacturing variability [25], or compensate for environmental variations [28]. This paper is not concerned with remaining above DRV, but instead with characterizing the DRV of each cell and using this unique variation-sensitive information as part of a PUF.

Fast and accurate DRV analysis is needed to evaluate DRV fingerprinting, and significant research effort has been spent on solving this problem. The default technique for DRV analysis is Monte Carlo circuit simulation. When searching for the DRV of an entire array instead of individual cells, improvements over basic Monte Carlo simulation include the use of importance sampling [29], adaptive sampling [30], [31], and boundary line searching [32]. An overview of several statistical techniques is given by Wang *et al.* [33]. In Section V, we propose a new technique that uses ML to predict DRV. This approach differs from the aforementioned statistical approaches in having the goal of predicting the DRV of individual cells, instead of just accurately estimating the failure rate of the entire SRAM using process variation statistics.

IV. DRV Characterization

We characterize the DRV of an SRAM cell at address i with a pair $\langle v_i^0, v_i^1\rangle$. Each v_i^w represents the highest voltage at which address i will have a retention failure after state w is written to it. In principle, v_i^0 and v_i^1 are real-valued; in practice, we approximate each one using N discrete voltages with a step size of Δv. Procedure 1 presents our characterization procedure. The implementation details of Procedure 1 vary slightly when applied in hardware measurement or simulation as explained in the next two sections.

The DRV characterization procedure is parameterized by maximum, minimum, and step size for test voltages ($v_{\max}$, $v_{\min}$, and Δv, respectively), and by the time (t_{test}) for which each test voltage is applied. Simulations by Nourivand *et al.* [25] using a procedure similar to Procedure 1 show that a value of 2 ms for t_{test} is sufficient to induce retention failures. The total time to characterize the DRV of an SRAM cell using Procedure 1 is given by t_{proc} in (1). In the case of simulation, t_{proc} is the simulated time, and the actual runtime for the circuit simulator is many orders of magnitude larger. The frequency of observing different DRVs in hardware measurements and simulation are shown in Fig. 4

$$t_{\text{proc}} = t_{\text{test}} \times \frac{v_{\max} - v_{\min}}{\Delta v}. \quad (1)$$

A. Hardware DRV Measurement

The target platform for DRV fingerprinting is an integrated SRAM block with an adjustable supply voltage, as is sometimes used to compensate for variation [34]. To simplify experiments, our platform mimics this configuration using a dedicated SRAM chip and a separate microcontroller. Fig. 2 presents the overview of our experimental system. SRAM supply voltages are generated using analog outputs of a Texas Instruments MSP430 F2618 microcontroller [35], and that same microcontroller also orchestrates the timing of the supply voltage changes (per Procedure 1). An op-amp configured as a voltage follower tracks the analog output voltage from the microcontroller and powers the SRAM at the same voltage; the op-amp is used because the analog output of the microcontroller cannot supply enough current to power the SRAM directly. All experiments use instances of SRAM chip AS6C6264 [36] and the DRV characterization parameters are $v_{\max} = 700$ mV, $v_{\min} = 0$ mV, $\Delta v = 2$ mV, and $t_{\text{test}} = 1$ s. Thermal tests are conducted inside of a Sun Electronics EC12 Environmental Chamber [37], and an OSXL450 infrared noncontact thermometer [38] with ±2 °C accuracy is used to verify the temperature.

Note that our experimental platform differs from that used in [1]. In our previous work, the DRVs of SRAM cells in a microcontroller memory are characterized by repeatedly lowering the microcontroller's supply voltage and observing the highest voltage that induces a retention failure in each cell. Because the microcontroller's SRAM shares a common supply node with the processing core, the low test voltages used for the characterization cause the core to reset and lose its state. As persistent state is required for the DRV characterization, our experiments used the microcontroller's nonvolatile memory to preserve state while the test voltages were applied.

Procedure 1 Characterize the DRV Fingerprint of a Set of SRAM Cells

```
Require: A set of bit-addressable SRAM cells
Ensure: $v_i^0, v_i^1$ {the DRV characterization of cell at address $i$.}
 1: Let $V_{nom}$ be the nominal supply voltage for the SRAM
 2: Let $s_i$ refer to the logical state of SRAM address $i$.
 3: for $w = 0, 1$ do
 4:     for $i \in SRAM$ do
 5:         $s_i \leftarrow w$    {write $w$ to SRAM address}
 6:         $v_i^w \leftarrow v_{min}$    {value used if no retention failure observed}
 7:     end for
 8:     $v_{test} \leftarrow v_{max}$    {initialize test voltage}
 9:     while $v_{test} > v_{min}$ do
10:         lower SRAM voltage from $V_{nom}$ to $v_{test}$
11:         remain at voltage $v_{test}$ for time $t_{test}$
12:         raise SRAM voltage from $v_{test}$ to $V_{nom}$
13:         for $i \in SRAM$ do
14:             if $(s_i \neq w) \wedge (v_i^w = v_{min})$ then
15:                 SRAM cell at address i did not retain state w after
                    applying $v_{test}$, and $v_{test}$ is the first and highest voltage
                    at which this retention failure occurred.
16:                 $v_i^w \leftarrow v_{test}$
17:             end if
18:         end for
19:         $v_{test} \leftarrow v_{test} - \Delta v$    {try a lower voltage next}
20:     end while
21: end for
```

B. DRV Measurement in SPICE Simulation

Circuit simulation is a second platform for DRV characterization (Procedure 1), and it complements hardware measurements by allowing for DRV exploration under controllable process variations and environmental conditions. In circuit simulation, we use transistor models from the 45 nm predictive technology model [39], [40]. To mimic the random process variations that give each cell its unique DRV, variations

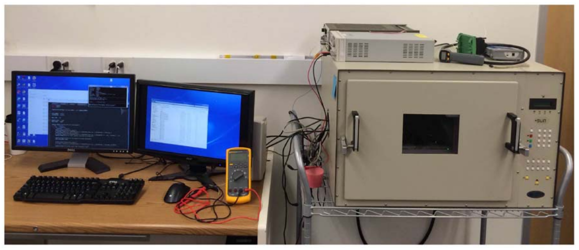
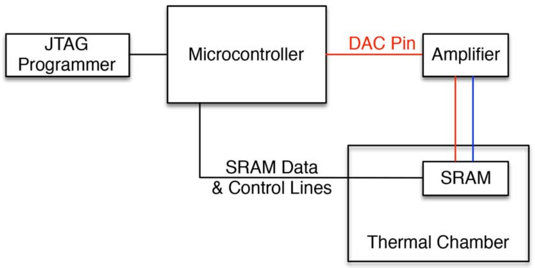



Fig. 2. Experimental platform used for determining SRAM chip DRV assignments.

are introduced for transistor width W, length L, and threshold voltage V_{th}. The International Technology Roadmap for Semiconductors indicates that transistor length should have a 3σ variation that is 10% of the nominal length L [41], [42]. Adopting the same guideline for transistor width, in our simulation the random components of both W and L are normally distributed with a standard deviation that is 3.33% of the nominal W or L value. The standard deviation of threshold voltage is given by (2); a value of 1.8 mV*μm is used for the matching constant $A_{V_{\text{th}}}$ [43]. The parameter values used when implementing the DRV characterization procedure (Procedure 1) in SPICE are $v_{\max} = 500$ mV, $v_{\min} = 0$ mV, $\Delta v = 0.1$ mV, and $t_{\text{test}} = 2$ ms

$$\sigma_{V_{\text{th}}} = \frac{A_{V_{\text{th}}}}{\sqrt{W * L}}. \tag{2}$$

C. Impact of Temperature Variations

DRVs generally increase with temperature [26], and this hinders the reliability of DRV-based fingerprinting. Recalling that each DRV is a point $\langle v_i^0, v_i^1 \rangle$ in 2-D space, an intuitive way to define the distance between two DRVs is to use their distance in this 2-D space (3). This distance metric is used as the basis for DRV fingerprint matching in [1]. To demonstrate the impact of temperature, we compute the average distance between two characterizations of the same cell, when one is taken at 28 °C and the other at 50 or 70 °C. As shown in Fig. 3, the average distance between the two characterizations increases with the temperature difference

$$d1(i,j) = \sqrt{\left(v_i^0 - v_j^0\right)^2 + \left(v_i^1 - v_j^1\right)^2}. \tag{3}$$

Given that DRV fingerprints are intended for use in real-world scenarios without precisely controlled temperatures, the temperature sensitivity shown in Fig. 3 indicates that the distance metric of (3) is prone to unreliability in real-world usage. In Section VI, we propose a new technique for extracting temperature-invariant information from DRV and demonstrate

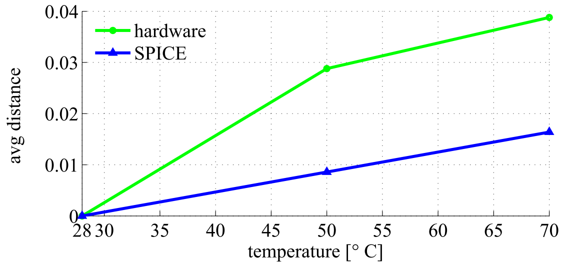


Fig. 3. Distance [per (3)] between two characterizations of the same cell increases as temperature changes.

that this new technique is highly reliable when temperature fluctuates.

V. Modeling the DRV of SRAM Cell

Although the SPICE simulation described in Section IV-B is a straightforward and highly accurate approach to characterize the DRV of SRAM cells, it is very time consuming for two reasons. The first reason is that, to find the maximum voltage that induces a failure in each cell, numerous test voltages must be applied (Procedure 1). The second reason is simply that simulating each test voltage is itself very slow. On our experimental machine, equipped with an Intel Xeon E5-2690 processor running at 2.90 GHz with 64 GB of RAM, simulating a single test voltage on a single SRAM cell for 2 ms has a runtime of 0.17 s.

An alternative to iterative SPICE analysis is to predict DRV using a model. Just as the DRV of each SRAM cell is ultimately determined by temperature and the process variations of its transistors, the DRV of an SRAM cell can be formulated as a function of its temperature T and transistor width, length, and threshold voltage (W, L, and V_{th}, respectively). Qin *et al.* [26] provided an analytical model for the DRV of an individual cell as in (4), where DRV_r is the DRV at room temperature, and DRV_f is defined in (5) with ΔT representing the temperature difference from room temperature. Terms a_i, b_i, and c in (5) are fitting coefficients and their values are determined empirically for each CMOS technology process [26]

$$\text{DRV} = \text{DRV}_r + \text{DRV}_f \tag{4}$$

$$\text{DRV}_f = \sum_{i=1}^{6} a_i * \frac{\Delta(W_i/L_i)}{W_i/L_i} + \sum_{i=1}^{6} b_i * \Delta(V_{\text{th}i}) + c * \Delta T. \tag{5}$$

Although this model can accurately estimate the DRV of a cell, it has two weaknesses that create the need for a more advanced model.

1) To predict a specified DRV value with (5), the user needs to know the DRV_r for each SRAM cell. This value is not expressed as a function of transistor parameters and can only be calculated through hardware measurement or computationally expensive circuit simulation.
2) Using the same coefficients a_i, b_i, and c for different SRAM cells creates estimation errors. In reality, the DRV of different cells increase according to different coefficients depending on their unique process variations. This distinction is especially important in

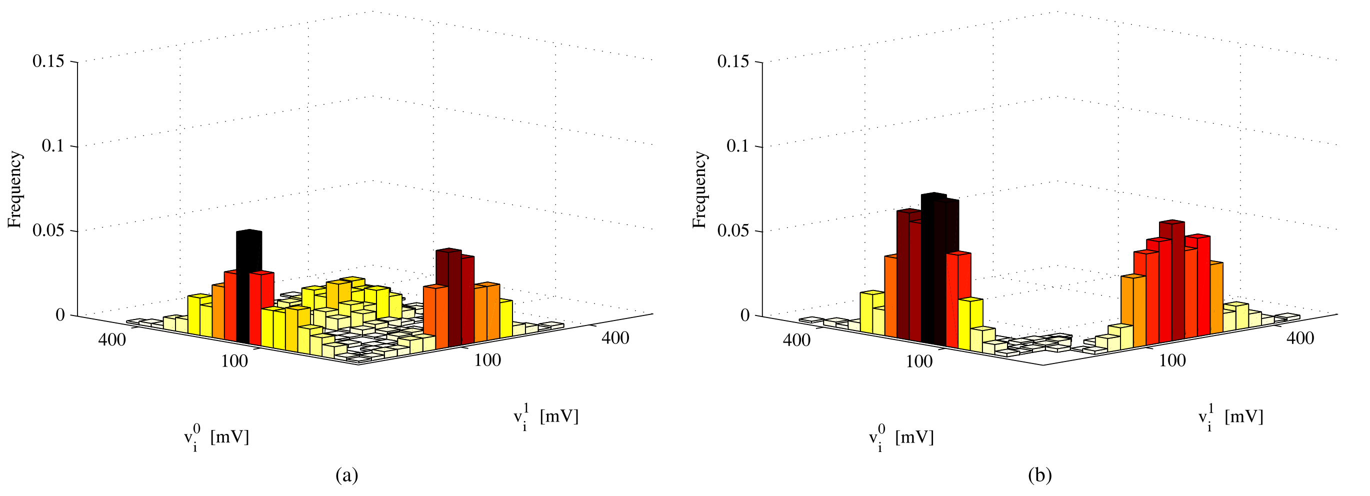


Fig. 4. Joint probability distribution function over all cells of the two variables (v_i^0 and v_i^1) comprising a DRV characterization. The distribution is determined experimentally using Procedure 1, and shows that a large fraction of cells have the minimum possible value ($v_{\min}$) for either v^0 or v^1, indicating a cell that retains one written state across all test voltages. (a) DRV joint PDF from hardware measurements. (b) DRV joint PDF from SPICE.

this paper, where the unique impact of process variations across cells is critical to the overall work.

A. Predicting DRV Using Artificial Neural Networks

Our approach addresses the two aforementioned weaknesses in Qin *et al.*'s work [26] by using ML for DRV prediction. The ML algorithm predicts the DRV directly from the parameters that are responsible for determining it, without using expensive circuit simulation. Given that the values of process variation parameters vary over bounded ranges, it follows that the DRVs too fall within a bounded range $[\mathrm{DRV}_{\min}, \mathrm{DRV}_{\max}]$. The range of DRVs is manually divided into K classes, each with size $\Delta\mathrm{DRV}$ (6). The use of K classes makes DRV prediction a "multiclassification" problem: for any given feature pattern $\{W_i, L_i, \ldots, V_{\mathrm{th}\_i}, T\}$, there is exactly one among K classes corresponding to the correct DRV output

$$\begin{aligned}[\mathrm{DRV}_{\min}, \mathrm{DRV}_{\max}] = \{&[\mathrm{DRV}_{\min}, \mathrm{DRV}_{\min} + \Delta\mathrm{DRV})\\ &\cup [\mathrm{DRV}_{\min} + \Delta\mathrm{DRV}, \mathrm{DRV}_{\min}\\ &\quad + 2 * \Delta\mathrm{DRV}) \cup \cdots\\ &\cup [\mathrm{DRV}_{\max} - \Delta\mathrm{DRV}, \mathrm{DRV}_{\max}]\}. \quad (6)\end{aligned}$$

Artificial neural network (ANN) is a well known ML method [44], [45] that is widely used to solve multiclassification problems. Our DRV prediction method specifies the DRV of an SRAM cell as an ANN output class, and indicates the class to which the corresponding SRAM parameter pattern should be assigned to. Our approach collects a group of samples from SPICE, including physical parameters of SRAM circuitry as input and corresponding DRV values (which can be viewed as golden value) as output. An ANN model is later trained based on this data to match input with output. In this process, neurons learn to classify the examples from each class (Fig. 5). Finally, the hidden neurons dealing with the same class will be combined as one group, so the number of groups corresponds to the number of output classes. Each class has a corresponding surface, which is approximated by the combined neuron groups.

To get data for ANN model training, we use SPICE simulator as the infrastructure for collecting DRV statistics. DRV values are extracted from simulations of 2000 cells across temperature 25–100 °C, with a step size of 1 °C. A common problem with ML models is overfitting, where a model is trained to perform well on training data but fails to yield similar results upon seeing new data. To avoid overfitting in building the DRV model, we reordered the samples and divided them into three subsets: 1) training (60%); 2) validation (20%); and 3) test (20%). Training set is the dataset used for computing the gradient and updating the network weights and biases. The validation set is used to monitor errors during the training process. The validation and training set errors usually decreases during the initial phase of training. The test set error is not used during training, but is used to validate the model performance and compare different models.

B. Evaluating Accuracy of DRV Prediction

The prediction results of the test subsets are shown in Fig. 6. The regression plots display the ANN-predicted DRVs with respect to the golden DRV values collected from SPICE simulation. In Fig. 6, R denotes the correlation between model outputs and golden values. For a perfect fit, the predicted outputs should be equal to the golden values (the data should fall along a 45° line). For our DRV model, there is a high correlation between prediction and output for all datasets.

Before our ANN model in this paper, the linear model described in (5) was widely used to model the DRV value of SRAM designs, which can be optimized with linear regression (LR) method. LR fits a data model that is linear in the model coefficients. The most common type of LR is a "least-squares fit," which can find an optimal line to represent the discrete data points. In an LR model, the same physical parameters of ANN models are defined as input training features $p = \{p_i, \mid p_i \in \{W_1/L_1, W_2/L_2, \ldots, T\}\}$. By denoting the linear coefficients with $\theta = \{\theta_0, \theta_1, \ldots, \theta_n\}$, we get

$$h_\theta(p) = \theta_0 + \theta_1 * p_1 + \cdots + \theta_n * p_n \qquad (7)$$

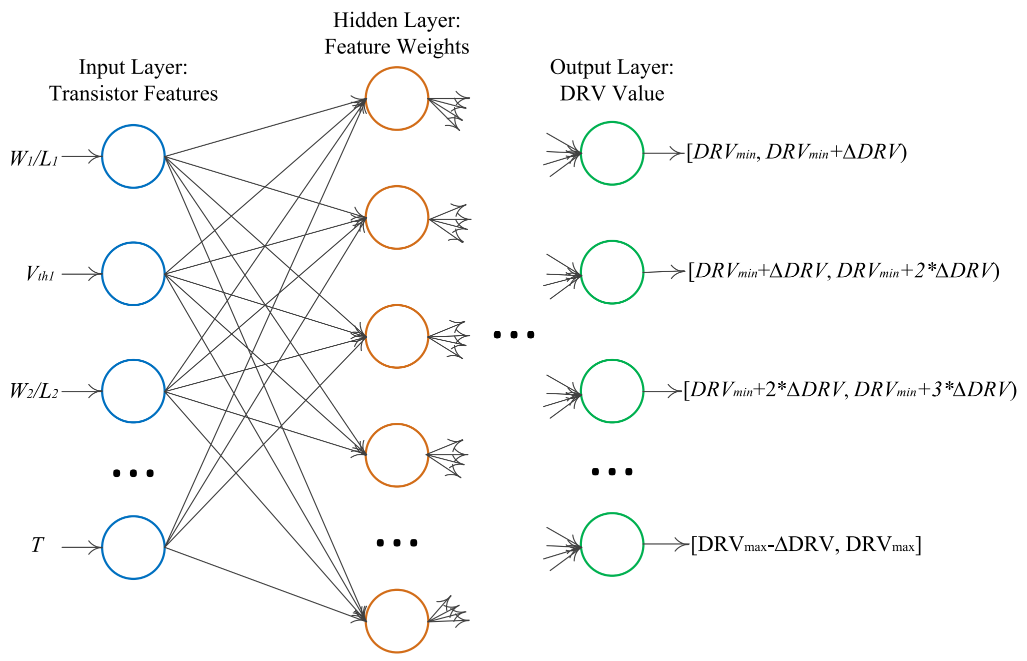


Fig. 5. ANN for DRV classification and prediction.

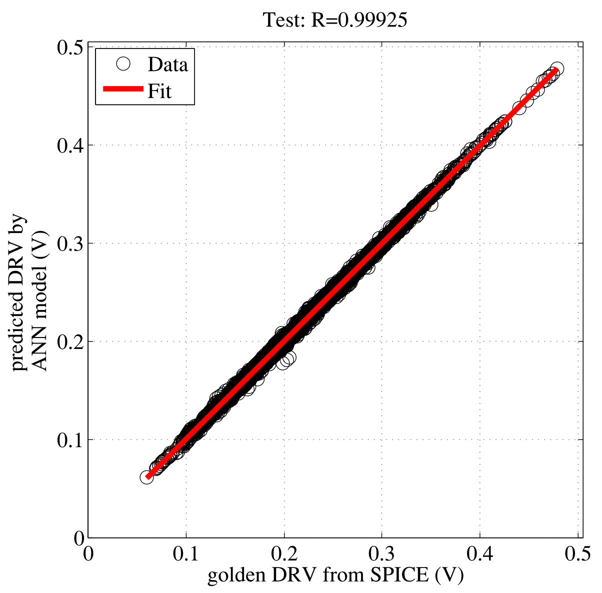


Fig. 6. Training results based on neural network model, across three datasets. R denotes the correlation between golden DRV data from SPICE simulation, and predicted DRV value from our model.

where θ stands for the set of coefficients [e.g., a_i and b_i as shown in (5)]. Each training sample is composed of transistor feature set p and the corresponding golden DRV value $\text{DRV}_{\text{golden}}$ from SPICE simulation. Based on least-squares fit rule, the cost function of m training examples can be expressed as

$$J(\theta) = \frac{1}{2m}\sum_{k=1}^{m}\left(h_\theta\left(p^{(k)}\right) - \text{DRV}_{\text{golden}}^{(k)}\right)^2 \tag{8}$$

where $p^{(k)}$ corresponds to the training features of kth training sample, like the transistor sizes and temperature. To obtain

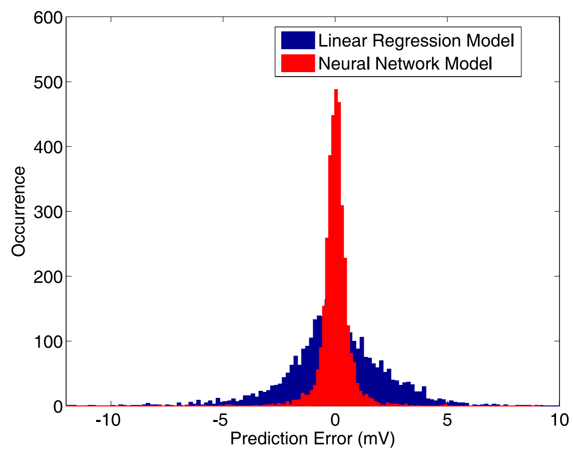


Fig. 7. DRV prediction error for the ANN model and LR model. In both cases, error is determined by comparison to SPICE simulated results.

the optimal θ, we applied "gradient descent" simultaneously on each coefficient θ_j, $j \in (1, 2, \ldots, n)$

$$\text{Repeat}\left\{\begin{aligned}\theta_j &:= \theta_j - \alpha\frac{\partial J(\theta)}{\partial\theta_j}\\ &= \theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(p^{(k)}\right) - \text{DRV}_{\text{golden}}^{(k)}\right)p_i^{(k)}\end{aligned}\right\} \tag{9}$$

α is the learning rate of LR model and $p_i^{(k)}$ is the ith feature of the kth training sample.

To further evaluate the effectiveness of our ANN model, we compare its prediction error to that of the LR model[1] on a randomly chosen dataset of size 3500. Fig. 7 presents the prediction error of these models. The neural network model achieves smaller prediction errors than the LR model.

[1]The LR model is trained and optimized with the same three datasets as the neural network model.

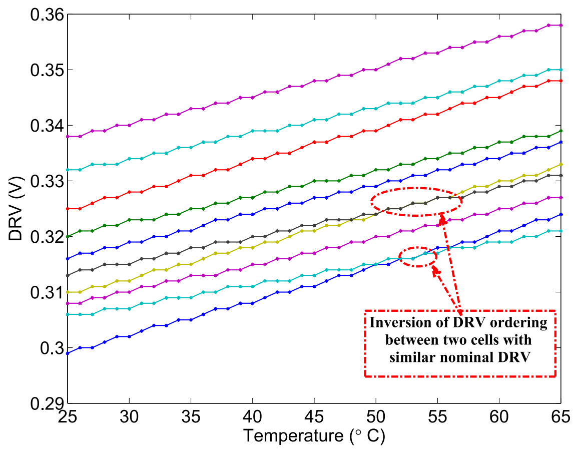


Fig. 8. DRV of SRAM cells increase linearly with temperature, but the slope varies across cells. The ordering of DRV is largely preserved across temperatures, except for a few cells that switch ordering.

The mean μ and standard deviation σ of prediction error for the neural network model are −0.01 and 0.35 mV, respectively, while those of the LR model are 0.041 and 0.9 mV. The neural network model outperforms the linear model because the neural network model assigns varied weights and bias to different feature patterns,[2] whereas the linear model formulates all input features with the same optimized θ.

VI. DRV-Based PUF

DRV-based identification or authentication schemes must consider the impact of temperature changes. The DRV of each SRAM cell increases approximately linearly with temperature [26], and the coefficient relating DRV to temperature varies only slightly across cells. Accordingly, the relative ordering of DRVs across cells is more reliable than the values themselves; in other words, the cell with the ith highest DRV will remain roughly the ith highest when temperature changes, even though all DRVs will change in absolute terms. Fig. 8 shows the relationship of DRV and temperature, according to ML prediction, for ten randomly chosen SRAM cells. The DRV ordering is preserved across temperature values, except for two pairs of cells that flip their ordering. Any two cells with sufficiently different nominal DRVs have a DRV ordering that does not change with temperature.

A. *DRV-Based Hashing With* DH *and* DH-PREIMAGE

To utilize the robustness of DRV ordering, we propose a hashing scheme with the mapping between challenges and responses defined by the DRV ordering within SRAM address pairs. In this scheme, a challenge C of length m is a sequence of address pairs $(\langle \bar{c}_0, c_0\rangle, \ldots, \langle \bar{c}_{m-1}, c_{m-1}\rangle)$, a response R is a bit string $(r_0, \ldots, r_{m-1})$. A DRV measurement D assigns values to each address of an SRAM such that $D(c) = \max(v_c^0, v_c^1)$, where v_c^0 and v_c^1 are the minimum retention voltages after writing the 0 and 1 states to the cell at address c (Procedure 1). Note that a DRV D is a single

[2]This also validates our finding in Fig. 8, that different SRAM cells have different DRV growth slopes while temperature is increasing.

Procedure 2 $R = \mathrm{DH}(D, C)$: Use DRV Assignment D to Hash Challenge C to Response R

```
Require: $D$ {DRV assignments for a set of SRAM addresses}
Require: $C$ {sequence of addr pairs $(\langle \bar{c}_0, c_0\rangle, \ldots, \langle \bar{c}_{m-1}, c_{m-1}\rangle)$}
Ensure: $R$ {bit string response $(r_0, \ldots, r_{m-1})$ to challenge}
1: for $\langle \bar{c}_i, c_i\rangle \in C$ do
2:     $r_i \leftarrow (D(c_i) \geq D(\bar{c}_i))$
3: end for
4: return $R$
```

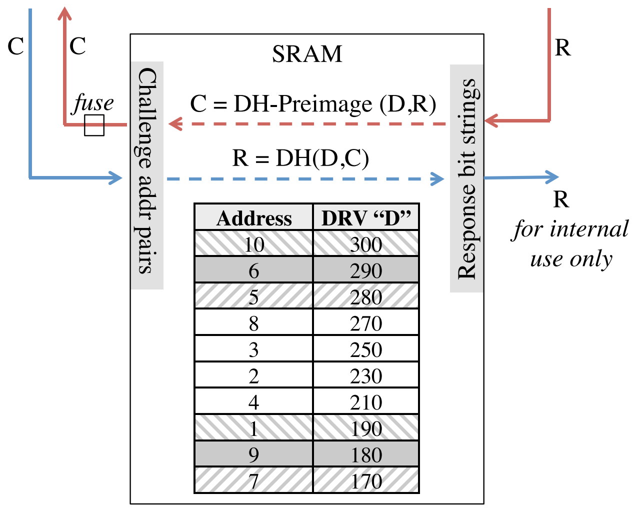
Procedure 3 $C = \mathrm{DH\text{-}PREIMAGE}(D, R)$: Map Response R to Challenge C Using DRV Assignment D

```
Require: $D$ {SRAM DRVs. $D(a)$ is DRV of cell at address $a$.}
Require: $R$ {the desired response bit string $(r_0, \ldots, r_{m-1})$}
Ensure: $C$ {sequence of addr pairs $(\langle \bar{c}_0, c_0\rangle, \ldots, \langle \bar{c}_{m-1}, c_{m-1}\rangle)$}
    {sort addresses by DRV. Let $a_i$ denote address such that $D(a_i)$ is $i$th highest among all addresses}
1: for $i \in 0..(|R| - 1)$ do
2:     if $r_i = 1$ then
3:         $\langle \bar{c}_i, c_i\rangle \leftarrow \langle a_{i+|D|-m}, a_i\rangle$ {$c_i$ gets addr with higher DRV}
4:     else
5:         $\langle \bar{c}_i, c_i\rangle \leftarrow \langle a_i, a_{i+|D|-m}\rangle$ {$\bar{c}_i$ gets addr with higher DRV}
6:     end if
7: end for
8: return $C$
```

imprecise observation, and two DRVs produced by the same chip will only match approximately. Procedure $\mathrm{DH}(D, C)$ (Procedure 2) hashes a challenge C to a response R. Procedure $\mathrm{DH\text{-}PREIMAGE}(D, R)$ (Procedure 3) computes a challenge C that reliably hashes to response R on a particular chip. For any DRV assignment D and response R, the relationship $R = \mathrm{DH}(D, \mathrm{DH\text{-}PREIMAGE}(D, R))$ holds. Procedures DH and DH-PREIMAGE are the building blocks for key generation and identification applications in Sections VI-B and VI-D.

The DH procedure is designed to be resilient to small fluctuations in DRV, and to common-mode DRV shifts such as those caused by temperature. The steps for DH are given in Procedure 2. For each address pair $\langle \bar{c}_i, c_i\rangle$ in the challenge, the corresponding response bit r_i is assigned a 1 if address c_i has the higher or equal DRV, and 0 if address $\bar{c}_i$ has a higher DRV. Procedure DH is made resilient by applying challenges for which the addresses in each pair have vastly different DRVs, so that the inequality at line (2) of Procedure 2 consistently resolves in the same way despite small variations.

Given a DRV D and a desired response R, the role of procedure DH-PREIMAGE is to create a challenge C that will reliably generate R whenever it is applied to the same SRAM that produced D. The steps for DH-PREIMAGE are shown in Procedure 3. For each bit r_i of the desired response, a pair of challenge addresses $\langle \bar{c}_i, c_i\rangle$ is chosen. If the desired response bit is 0 (1), then $\bar{c}_i$ (c_i) is assigned the address of the cell with the higher DRV. Note that the two addresses chosen for each pair have markedly dissimilar DRVs that are separated by $|D| - m$ positions in DRV ordering (see lines 3 and 5 of Procedure 3), where $|D|$ is the SRAM size and m is the response length. Stated differently, one address in each pair has one of the m highest DRVs in the SRAM, and the other has one of the m lowest DRVs. The DRV dissimilarity within



Address	DRV "D"
10	300
6	290
5	280
8	270
3	250
2	230
4	210
1	190
9	180
7	170

Fig. 9. Example of DRV-hashing. According to the depicted DRV assignment D, and letting challenge C be $(\langle 1, 10\rangle, \langle 6, 9\rangle, \langle 7, 5\rangle)$, procedure DH$(D, C)$ produces response $R = (1, 0, 1)$. Similarly, procedure DH-PREIMAGE(D, R), given this response R, would produce as output the same challenge C.

each address pair ensures that the higher DRV can be reliably determined when the challenge is applied in a subsequent call to DH.

A demonstration of the DH is given in Fig. 9. According to the depicted DRV assignment D, procedure DH hashes challenge $C = (\langle 1, 10\rangle, \langle 6, 9\rangle, \langle 7, 5\rangle)$ to response $R = (1, 0, 1)$: the first response bit is 1 because address 10 has a higher DRV than address 1, the second response bit is 0 because address 6 has a higher DRV than address 9, and the third response bit is 1 because address 5 has a higher DRV than address 7.

A necessary condition for obtaining a wrong response bit for a challenge address pair is that, when some test voltage is applied, the cell with nominally lower DRV fails, and the cell with the nominally higher DRV does not. In the toy example of Fig. 9, given that the DRVs within each pair have a gap of 110 mV, this will only happen in the case of extreme noise or if the supply voltage differs by 110 mV from one cell location to the other. As the cells of an SRAM are powered by the same supply, and given that supply nodes are already designed to avoid local voltage droop, such a large supply voltage difference across cells is uncommon.

The DRV hashing scheme in this paper is related to IBS coding [20] and ordering-based encoding schemes applied to PUFs with real-valued outputs [46]. Given a group of indexed objects with real-valued measurements, IBS coding encodes each bit to a syndrome that is the index of the maximum or minimum value in the group depending on the bit's polarity. Given noisy measurements of the same indexed objects, the syndrome is decoded by determining whether the measurement it indexes is closer to maximal or minimal in the group. A comparison of the reliability and security of IBS versus other approaches is given by Yu *et al.* [47]. Variants of IBS coding are also applied to SRAM power-up state PUFs [21]. Procedures DH-PREIMAGE and DH are analogs for IBS encoding and decoding, respectively. In addition to using a hashing scheme related to IBS coding, a second reliability enhancing feature is that the SRAM cell pairs are selected to maximize the discrepancy between the values in each pairing. The idea of configuring real-valued PUFs to utilize large discrepancies for enhanced reliability has been proposed previously for ring oscillator PUFs [48], [49], where the identifying feature is oscillator frequency instead of minimum retention voltage.

B. Secret Key Generation

Cryptographic keys must be fully reliable, and this is in conflict with the inherent imprecision of PUFs in sensing the effects of small physical variations. Error correcting codes can bridge the gap from noisy PUFs to reliable keys. With error correction, some number of raw response bits are transformed by helper data into a noisy codeword that is decoded into a reliable key. One example of error correction in weak PUFs is the use of BCH codes with power-up fingerprints [5]. The physical nature of PUFs also allows for circuit-level reliability mechanisms that enable lighter-weight[3] error correcting codes, or in some cases supplant them entirely. Examples of reliability-enhancing circuit techniques are reinforcing variation tendencies with directed aging [6], [13], and using helper data to mark on each device the response bits that are precharacterized as unreliable [20].

Our key generation using DRV-hashing uses circuit-level reliability enhancement and (optionally) error correcting codes. The steps to implement DRV-based secret key generation for a given SRAM instance are shown in Procedure 4. Lines 1–5 comprise the enrollment process to occur at the manufacturer immediately after fabrication. First, an arbitrary secret key K is chosen and encoded into codeword R; R is the value that should be the response of the DRV PUF in the field later. Next, DH-PREIMAGE is called (line 3) to generate a challenge that will reliably hash to response R on this PUF instance. The challenge is then stored to a one-time-programmable on-chip memory (line 4). Finally, the enrollment process is completed by blowing a fuse to disable the DH-PREIMAGE functionality (line 5). After the enrollment process is completed, the PUF can be used in the field as a secret key. When the stored challenge C is applied to the PUF in the field, it hashes to response R' according to DRV D' (line 6). If D' is similar to the enrollment DRV D (as it will be for the same chip), then the inherent robustness of DH should produce a response R' that exactly matches or closely approximates R. Response R', a possibly noisy version of the original codeword R, is decoded to correct errors and regenerate the enrolled key K (line 7). This key is a secret, chosen by the party that enrolled the DRV PUF, and known only to them. To maintain secrecy of key K, it must only be used as an input to a cryptographic hash, and never be revealed in plain text.

Note that the secrecy of the generated key requires that an attacker cannot apply arbitrary challenges to the DRV PUF. If an attacker can apply chosen challenges, then helper data manipulation attacks from other pairing-based PUFs [50] can also expose the secret key from the DRV PUF. It is therefore

[3]That is, codes that are cheaper to implement but cannot correct as many errors.

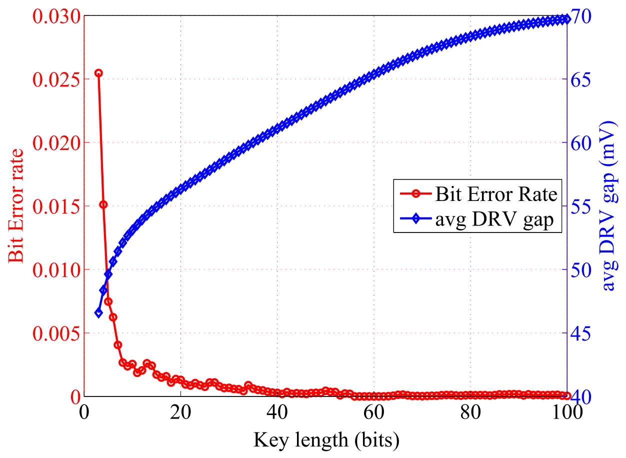


Fig. 10. When implementing a key of length m using a DRV PUF in SRAM of size $2m$ (per Procedure 4), the response BER decreases as m increases. The decrease in BER results from an increase in the DRV gap, where "avg DRV gap" represents the absolute DRV difference between $\bar{c}_i$ and c_i, averaged over challenge pairs.

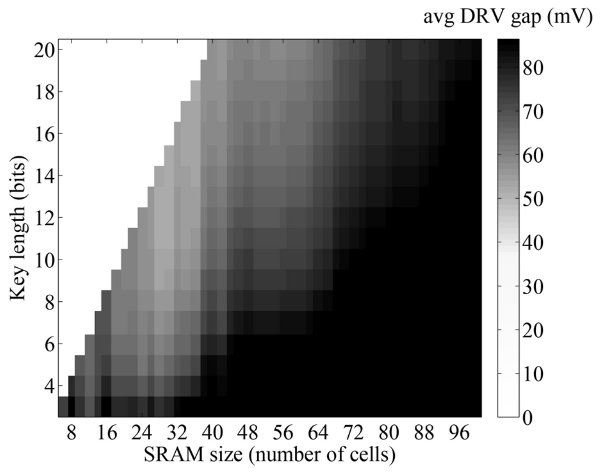


Fig. 11. When implementing a key of length m using a DRV PUF in SRAM of size $\geq 2m$ (per Procedure 4), there is a clear increase in the average DRV gap as SRAM size increases. Because a larger DRV gap equates to a lower BER, changing the size of SRAM therefore represents a reliability knob for the DRV PUF.

necessary in the secret key application that the challenge addresses are supplied exclusively from a memory that is not overwritable in the field. Note that nonoverwritable challenge addresses need not be secret. The addresses do not leak information about the key under the assumption that DRVs are independent and identically distributed, as shown in work on IBS coding [47].

An adversary possessing a chip may somehow modify the test voltages prescribed by Procedure 1 for DRV characterization. This could induce a flawed DRV characterization; for example, if the voltage is never lowered at all, the characterization would wrongly conclude that all cells have a DRV of 0 V since none ever had a retention failure at any test voltage. However, voltage manipulation does not provide any useful information about the DRV-based secret key. Even though it may be possible to learn that a retention failure happens at some particular voltage, no information is leaked unless it can be determined which address in a pair failed, and this is not observable because the output of the DRV PUF is never revealed in the clear.

C. Reliability

The response bit error rate (BER) in key generation experiments depends on the key size and the size of the SRAM used for the DRV PUF. The result of Fig. 10 shows the BER[4] of an m-bit key generated by a $2m$-cell SRAM.[5] The response BER decreases as m increases, and does not exceed 1e−5 for any key size larger than 60 bits. The BER decreases as the SRAM size increases, because a larger SRAM tends to have a larger difference between the DRVs of the addresses within each challenge pair. We refer to the DRV difference between the two addresses in each challenge pair as the "DRV gap" of a DRV PUF. Larger DRV gaps indicate more reliable DRV PUF responses, because the determination in DH of which challenge address has the higher DRV will be less error prone.

The BER can be further reduced by increasing the size of the SRAM to beyond the minimum of twice the key length m. In this case, the cells with DRVs near the median DRV of the SRAM are not among the m highest nor m lowest, and are therefore not selected by DH-PREIMAGE to be used in the challenge. This further increases the DRV gap to reduce BER. Fig. 10 shows experimentally the average DRV gap as a function of SRAM size and key length. The areas in Fig. 11 with the darkest coloring correspond to the most reliable scenarios for key generation. Therefore, arbitrary robustness can be added directly to the hashing scheme, creating a second reliability knob to be used in concert with, or instead of, error correcting codes.

[4]In BER analysis, error correcting codes are not used so that circuit-level reliability of the proposed hashing scheme can be observed. Error correcting codes would serve to correct the errors that contribute to BER.

[5]An SRAM with $2m$ cells is the smallest SRAM capable of generating an m-bit response, because each response bit is generated using two addresses.

Procedure 4 Use DRV PUF as Reliable Secret Key. Lines 1–4 Enroll the PUF and Personalize it With Key K. Lines 5 and 6 Occur in the Field to Regenerate K From Challenge C

```
1: Choose a secret key $K$
2: $R = \text{ECC-ENCODE}(K)$ {for error correction}
3: $C \leftarrow \text{DH-PREIMAGE}(D, R)$ {challenge $C$ is public}
4: Store $C$ to one-time-programmable on-chip memory
5: Disable DH-PREIMAGE {Blow fuse. See Fig. 9}

6: $R' \leftarrow \text{DH}(D', C)$ {$D' \approx D \implies R' \approx R$}
7: $K \leftarrow \text{ECC-DECODE}(R')$ {Regenerated secret key inside chip}
```

D. Circuit Identification

Hashing functions DH and DH-PREIMAGE can be used for reliable chip identification, in a way that is similar to their use in key generation. In this application, the DRV of each SRAM is not considered secret, and arbitrary challenges can be applied to the SRAM. The goal of chip identification is to determine whether two DRV characterizations $D1$ and $D2$ are generated by the same SRAM cells. Using a distance

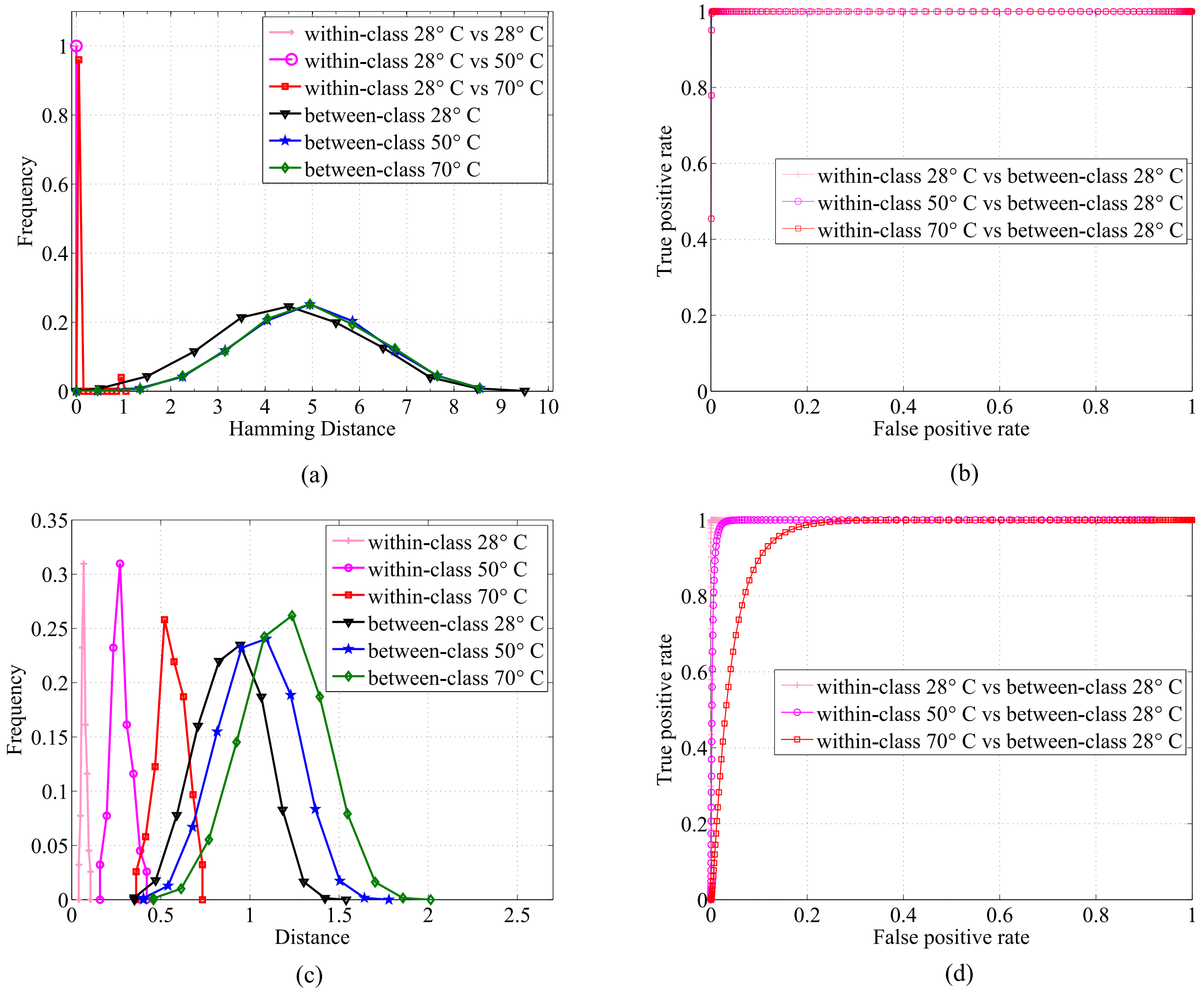


Fig. 12. Results showing identification performance using the distance metric RESPONSE-DISTANCE (Procedure 5) in upper plots, and performance using VOLTAGE-DISTANCE (10) in the lower plots. The temperature resilience of DH and DH-PREIMAGE causes RESPONSE-DISTANCE to outperform VOLTAGE-DISTANCE, as indicated by lower false positive rates for equivalent true positive rates. (a) Distribution of between- and within-class distances according to the metric RESPONSE-DISTANCE (Procedure 5). (b) ROC curves from RESPONSE-DISTANCE data at left. (c) Distribution of between- and within-class distances according to the metric VOLTAGE-DISTANCE (10). (d) ROC curves from VOLTAGE-DISTANCE data at left.

metric to quantify dissimilarity between two DRV characterizations, a determination of "same identity" is made whenever the distance between $D1$ and $D2$ is below some matching threshold. Distances between two DRVs from the same cells are denoted within class, and distances between two DRVs from different cells are denoted between class. A true positive identification occurs when a within-class distance is below the matching threshold, and a false positive identification occurs when a between-class distance is below the matching threshold. Perfect identification is possible when all within-class distances are smaller than all between-class distances, as it is then possible to choose a matching threshold that will produce a true positive identification for all within-class distances without any false positives.

The distance between DRVs $D1$ and $D2$ is computed as RESPONSE-DISTANCE($D1, D2$) (Procedure 5). The first step of Procedure 5 is to choose a random response $R1$ (line 1) and generate for $D1$ a challenge C that is the preimage of $R1$ (line 2). Response $R2$ is then obtained by hashing C using $D2$ (line 3). If $D1$ and $D2$ are from the same chip, then the

Procedure 5 RESPONSE-DISTANCE($D1, D2$): Compute Distance Between Responses of Two SRAMs When the Same Challenge is Applied to Both

```
Require: $D1$ {DRVs for chip 1. $D1(a)$ is chip 1 DRV at address $a$.}
Require: $D2$ {DRVs for chip 2. $D2(a)$ is chip 2 DRV at address $a$.}
Ensure: $x$ {the distance between $D1$ and $D2$}
  1: Choose randomly $R1 \in \{0,1\}^{|D1|/2}$
  2: $C \leftarrow$ DH-PREIMAGE($D1, R1$) {note: $R1 = \mathrm{DH}(D1, C)$}
  3: $R2 \leftarrow \mathrm{DH}(D2, C)$
  4: $x \leftarrow$ HAMMING-DISTANCE($R1, R2$)
  5: return $x$
```

BER analysis of the previous section shows that $R1$ and $R2$ will also be similar. To perform identification in a challenging (high-BER) scenario, we use a short key of length 10 and a small SRAM with 20 cells (see Fig. 10). The distribution of within- and between-class distances is shown in Fig. 12(a), and the receiver operating characteristic (ROC) plot in Fig. 12(b) shows the identification performance for the data. Each ROC curve traces tradeoffs between true positive and false positive identification that can be achieved by changing the matching threshold. The three ROC curves in Fig. 12(b) compare identification of across-temperature within-class distances against between-class distances at a single temperature; these three comparisons are chosen because they are the most challenging identification scenarios, as indicated by the largest overlaps between the two distributions [Fig. 12(a)].

We evaluate the performance of RESPONSE-DISTANCE for circuit identification (Procedure 5) by comparing its ROC curves [Fig. 12(b)] against the ROC curves obtained for the same data by our prior identification scheme [1]. In our prior work, letting the characterization of address i in $D1$ and $D2$ be denoted $\langle v1_i^0, v1_i^1\rangle$ and $\langle v2_i^0, v2_i^1\rangle$, the distance between $D1$ and $D2$ is given by VOLTAGE-DISTANCE$(D1, D2)$ (10). The within- and between-class distances according to VOLTAGE-DISTANCE have a larger overlap [Fig. 12(c)], and the corresponding ROC curves [Fig. 12(d)] have inferior performance because they admit in all cases a larger false positive identification rate for the same true positive identification rate

$$\text{VOLTAGE-DISTANCE}(D1, D2) = \sum_i \sqrt{\left(v1_i^0 - v2_i^0\right)^2 + \left(v1_i^1 - v2_i^1\right)^2}. \quad (10)$$

VII. CONCLUSION

This paper has demonstrated a collection of techniques that allow the DRV of SRAM cells to be used as the basis of a reliable PUF. We propose an ML approach for fast and accurate simulation-free prediction of DRV values. We present procedures DH and DH-PREIMAGE for reliable hashing based on DRV measurements, and demonstrate that the reliability of these approaches stems from their use of DRV ordering instead of the absolute DRV values that were previously proposed. We use a large dataset of DRVs from circuit simulation to train and analyze our DRV-prediction scheme, and use a large dataset of DRV measurements from SRAM chips to quantify the reliability of DH and DH-PREIMAGE in key generation and identification applications. Future work will reduce the runtime of DRV characterization (Procedure 1) by decreasing t_{test} and increasing Δv (1), and will explore techniques for learning DRV ordering when the voltages applied to the circuit are unknown and must be inferred from the number of failures induced.

ACKNOWLEDGMENT

Any opinions, findings, conclusions, and recommendations expressed in these materials are those of the authors and do not necessarily reflect the views of the sponsors.

REFERENCES

[1] D. E. Holcomb, A. Rahmati, M. Salajegheh, W. P. Burleson, and K. Fu, "DRV-fingerprinting: Using data retention voltage of SRAM cells for chip identification," in *Radio Frequency Identification. Security and Privacy Issues*. Berlin, Germany: Springer, 2013, pp. 165–179.

[2] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. 9th IEEE Comput. Commun. Soc.*, Washington, DC, USA, 2002, pp. 148–160.

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

[4] B. Gassend, "Physical random functions," Master's thesis, Comput. Sci. Artif. Intell. Lab., Massachusetts Inst. Technol., Cambridge, MA, USA, 2003.

[5] J. Guajardo, S. Kumar, G. Schrijen, and P. Tuyls, "FPGA intrinsic PUFs and their use for IP protection," in *Cryptographic Hardware and Embedded Systems*. Berlin, Germany: Springer, 2007.

[6] S. K. Mathew *et al.*, "16.2 a 0.19 pJ/b PVT-variation-tolerant hybrid physically unclonable function circuit for 100% stable secure key generation in 22nm CMOS," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers (ISSCC)*, San Francisco, CA, USA, 2014, pp. 278–279.

[7] D. E. Holcomb, W. P. Burleson, and K. Fu, "Power-up SRAM state as an identifying fingerprint and source of true random numbers," *IEEE Trans. Comput.*, vol. 58, no. 9, pp. 1198–1210, Sep. 2009.

[8] A. Kumar, H. Qin, P. Ishwar, J. Rabaey, and K. Ramchandran, "Fundamental data retention limits in SRAM standby—Experimental results," in *Proc. 9th Int. Symp. Qual. Electron. Design (ISQED)*, San Jose, CA, USA, 2008, pp. 92–97.

[9] K. Lofstrom, W. Daasch, and D. Taylor, "IC identification circuit using device mismatch," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, San Francisco, CA, USA, 2000, pp. 372–373.

[10] Y. Su, J. Holleman, and B. Otis, "A digital 1.6 pJ/bit chip identification circuit using process variations," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 69–77, Jan. 2008.

[11] P. Prabhu *et al.*, "Extracting device fingerprints from flash memory by exploiting physical variations," in *Proc. 4th Int. Conf. Trust Trustworthy Comput.*, Pittsburgh, PA, USA, 2011, pp. 188–201.

[12] J. Lee *et al.*, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *Proc. Symp. VLSI Circuits Dig. Tech. Papers*, Honolulu, HI, USA, Jun. 2004, pp. 176–179.

[13] M. Bhargava, C. Cakir, and K. Mai, "Reliability enhancement of bistable PUFs in 65nm bulk CMOS," in *Proc. Int. Symp. Hardw.-Orient. Secur. Trust*, San Francisco, CA, USA, 2012, pp. 25–30.

[14] S. Skorobogatov, "Low temperature data remanence in static RAM," Comput. Lab., Univ. Cambridge, Cambridge, U.K., Tech. Rep. UCAM-CL-TR-536, 2002.

[15] N. Saxena and J. Voris, "We can remember it for you wholesale: Implications of data remanence on the use of RAM for true random number generation on RFID tags," in *Proc. Conf. RFID Secur.*, Leuven, Belgium, 2009, pp. 1–13.

[16] P. Tuyls and L. Batina, "RFID-tags for anti-counterfeiting," in *Topics in Cryptology—CT-RSA* (LNCS 3860). Berlin, Germany: Springer, 2006, pp. 115–131.

[17] A.-R. Sadeghi, I. Visconti, and C. Wachsmann, "Enhancing RFID security and privacy by physically unclonable functions," in *Towards Hardware-Intrinsic Security* (Information Security and Cryptography). Berlin, Germany: Springer, Sep. 2010, pp. 281–307.

[18] G. Suh, C. O'Donnell, and S. Devadas, "Aegis: A single-chip secure processor," *IEEE Des. Test Comput.*, vol. 24, no. 6, pp. 570–580, Nov./Dec. 2007.

[19] R. Maes, P. Tuyls, and I. Verbauwhede, "Low-overhead implementation of a soft decision helper data algorithm for SRAM PUFs," in *Cryptographic Hardware and Embedded Security*. Berlin, Germany: Springer, 2009.

[20] M.-D. Yu and S. Devadas, "Secure and robust error correction for physical unclonable functions," *IEEE Des. Test Comput.*, vol. 27, no. 1, pp. 48–65, Jan./Feb. 2010.

[21] M. Hiller, D. Merli, F. Stumpf, and G. Sigl, "Complementary IBS: Application specific error correction for PUFs," in *Proc. Int. Symp. Hardw.-Orient. Secur. Trust*, San Francisco, CA, USA, 2012, pp. 1–6.

[22] A. van Herrewege *et al.*, "Reverse fuzzy extractors: Enabling lightweight mutual authentication for PUF-enabled RFIDs," in *Financial Cryptography Data Security* (LNCS 7397). Berlin, Germany: Springer, Feb. 2012.

[23] B. Ransford, S. Clark, M. Salajegheh, and K. Fu, "Getting things done on computational RFIDs with energy-aware checkpointing and voltage-aware scheduling," in *Proc. USENIX Workshop Power Aware Comput. Syst. (HotPower)*, San Diego, CA, USA, Dec. 2008, pp. 1–6. [Online]. Available: http://www.cs.umass.edu/~kevinfu/papers/ransford-CRFIDs-hotpower08-30112008.pdf

[24] K. Flautner, N. Kim, and S. Martin, "Drowsy caches: Simple techniques for reducing leakage power," in *Proc. Int. Symp. Comput. Archit.*, Anchorage, AK, USA, 2002, pp. 148–157.

[25] A. Nourivand, A. J. Al-Khalili, and Y. Savaria, "Postsilicon tuning of standby supply voltage in SRAMs to reduce yield losses due to parametric data-retention failures," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 1, pp. 29–41, Jan. 2012.

[26] H. Qin, Y. Cao, D. Markovic, A. Vladimirescu, and J. Rabaey, "SRAM leakage suppression by minimizing standby supply voltage," in *Proc. 5th Int. Symp. Qual. Electron. Design*, San Jose, CA, USA, 2004, pp. 55–60.

[27] A. C. Cabe, Z. Qi, and M. R. Stan, "Stacking SRAM banks for ultra low power standby mode operation," in *Proc. 47th IEEE Design Autom. Conf.*, Anaheim, CA, USA, Jun. 2010, pp. 699–704.

[28] J. Wang and B. H. Calhoun, "Techniques to extend canary-based standby V_{DD} scaling for SRAMs to 45 nm and beyond," *IEEE J. Solid-State Circuits*, vol. 43, no. 11, pp. 2514–2523, Nov. 2008.

[29] L. Dolecek, M. Qazi, D. Shah, and A. Chandrakasan, "Breaking the simulation barrier: SRAM evaluation through norm minimization," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2008, pp. 322–329.

[30] C. Dong and X. Li, "Efficient SRAM failure rate prediction via Gibbs sampling," in *Proc. 48th Design Autom. Conf.*, San Diego, CA, USA, 2011, pp. 200–205.

[31] J. Jaffari and M. Anis, "Adaptive sampling for efficient failure probability analysis of SRAM cells," in *Proc. Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2009, pp. 623–630.

[32] F. Gong *et al.*, "QuickYield: An efficient global-search based parametric yield estimation with performance constraints," in *Proc. 47th ACM/IEEE Design Autom. Conf. (DAC)*, Anaheim, CA, USA, 2010, pp. 392–397.

[33] J. Wang, A. Singhee, R. A. Rutenbar, and B. H. Calhoun, "Two fast methods for estimating the minimum standby supply voltage for large SRAMs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 29, no. 12, pp. 1908–1920, Dec. 2010.

[34] Y.-C. Lai, S.-Y. Huang, and H.-J. Hsu, "Resilient self-V_{DD}-tuning scheme with speed-margining for low-power SRAM," *IEEE J. Solid-State Circuits*, vol. 44, no. 10, pp. 2817–2823, Oct. 2009.

[35] T. I. Inc. (Apr. 2015). *Texas Instruments Application Report*. [Online]. Available: http://www.ti.com/lit/ds/symlink/msp430f2418.pdf

[36] *8K X 8 BIT low power CMOS SRAM*, Alliance Memory Inc., San Carlos, CA, USA, 2007.

[37] *I. Model EC1X Environmental Chamber User and Repair Manual*, Sun Electron. Syst., Titusville, FL, USA, 2011.

[38] *I. OSXL450 Infrared Non-Contact Thermometer Manual*, Omega Eng., Stamford, CT, USA. [Online]. Available: http://www.omega.com/Manuals/manualpdf/M4415.pdf

[39] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45 nm early design exploration," *IEEE Trans. Electron Devices*, vol. 53, no. 11, pp. 2816–2823, Nov. 2006.

[40] Y. Cao, T. Sato, D. Sylvester, M. Orshansky, and C. Hu. (2002). *Predictive Technology Model*. [Online]. Available: http://ptm.asu.edumodelcard/2006/45nm_bulk.pm

[41] S. I. Association. (Oct. 2014). *International Technology Roadmap for Semiconductors (ITRS)—Front End Process, 2004 Update*. [Online]. Available: http://www.itrs.net/links/2004update/2004_06_FEP.pdf

[42] M. Anis and M. H. Aburahma, "Leakage current variability in nanometer technologies," in *Proc. 5th Int. Workshop Syst. Chip Real-Time Appl.*, Banff, AB, Canada, 2005, pp. 60–63.

[43] M. Qazi, M. Tikekar, L. Dolecek, D. Shah, and A. Chandrakasan, "Loop flattening & spherical sampling: Highly efficient model reduction techniques for SRAM yield analysis," in *Proc. Design Autom. Test Europe Conf. Exhibit.*, Dresden, Germany, 2010, pp. 801–806.

[44] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*. Boston, MA, USA: PWS, 1996.

[45] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, 1989.

[46] R. Maes, A. van Herrewege, and I. Verbauwhede, "PUFKY: A fully functional PUF-based cryptographic key generator," in *Cryptographic Hardware and Embedded Systems*. Berlin, Germany: Springer, 2012.

[47] M. Yu, D. M'Raïhi, S. Devadas, and I. Verbauwhede, "Security and reliability properties of syndrome coding techniques used in PUF key generation," in *Proc. GOMACTech Conf.*, Las Vegas, NV, USA, 2013, pp. 1–4.

[48] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. 44th Annu. Design Autom. Conf. (DAC)*, San Diego, CA, USA, 2007, pp. 9–14. [Online]. Available: http://doi.acm.org/10.1145/1278480.1278484

[49] C. Yin and G. Qu, "LISA: Maximizing RO PUF's secret extraction," in *Proc. IEEE Int. Symp. Hardw.-Orient. Secur. Trust*, Anaheim, CA, USA, 2010, pp. 100–105.

[50] J. Delvaux and I. Verbauwhede, "Key-recovery attacks on various RO PUF constructions via helper data manipulation," in *Proc. Conf. Design Autom. Test Europe*, Dresden, Germany, 2014, pp. 1–11.

Xiaolin Xu (S'15) received the B.S. and M.S. degrees in electrical engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2008 and 2011, respectively. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Massachusetts Amherst, Amherst, MA, USA.

His current research interests include embedded system security with a special focus on physical unclonoble function, machine learning aided design, and side-channel analysis.

Amir Rahmati received the B.S. degree in computer engineering from the Sharif University of Technology, Tehran, Iran, in 2011. He received the M.S.E. degree in computer science and engineering from the University of Michigan, Ann Arbor, MI, USA, in 2014, where he is currently pursuing the Ph.D. degree.

His current research interests include mobile, embedded, and pervasive systems, computer networks, security, and reliability.

Daniel E. Holcomb (M'07) received the B.S. and M.S. degrees in electrical and computer engineering from the University of Massachusetts Amherst, Amherst, MA, USA, and the Ph.D. degree in electrical engineering and computer sciences from the University of California Berkeley, Berkeley, CA, USA.

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, University of Massachusetts Amherst. His current research interests include span formal verification, very large-scale integration, embedded systems, and hardware security.

Kevin Fu (SM'13) received the Ph.D. in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA, where his doctoral research pertained to secure storage and Web authentication.

He is an Associate Professor of Electrical Engineering and Computer Science with the University of Michigan, Ann Arbor, MI, USA, where he directs the Archimedes Center for Medical Device Security and the Security and Privacy Research group. His current research interests include how to achieve trustworthy computing on embedded devices with application to health care, commerce, and communication.

Prof. Fu is a member of ACM Committee on Computers and Public Policy and the National Institute of Standards and Technology Information Security and Privacy Advisory Board.

Wayne Burleson (M'84–SM'01–F'11) received the B.S. and M.S. degrees from the Massachusetts Institute of Technology, Cambridge, MA, USA, and the Ph.D. degree from the University of Colorado, Boulder, CO, USA.

He is a Professor of Electrical and Computer Engineering with the University of Massachusetts Amherst, Amherst, MA, USA, where he has been a faculty member since 1990. He currently directs and conducts research in security engineering. He has substantial industry experience in Silicon Valley. Although his primary focus is hardware security, building on 20 years in the microelectronics field, he also studies higher level issues of system security and security economics, with applications in payment systems, radio-frequency identification and medical devices. He teaches courses in microelectronics, embedded systems, and security engineering.