SLATE: A Secure Lightweight Entity Authentication Hardware Primitive

Wei-Che Wang, Yair Yona, Yizhang Wu, Suhas N. Diggavi, and Puneet Gupta

***Abstract*—Lightweight cryptography has become more and more important in recent years because of the rise of the Internet of Things (IoT) and usage of smart mobile devices. In this paper, we propose a novel secure lightweight entity authentication hardware primitive called SLATE, where its area is about 50% to more than 3X smaller than existing lightweight ciphers and strong physical unclonable functions (PUFs), respectively. Even though the authentication of SLATE is done through challenge response pair (CRP) verification similar to strong PUFs, the source of the key for SLATE must be coming from any existing secret key storage used for any ciphers. A main advantage of SLATE over most existing strong PUFs being an entity authentication primitive is that SLATE is resistant to known attacks to strong PUFs or logic obfuscations, such as model building attacks and Boolean satisfiability (SAT) attacks. Furthermore, we show that the implementation cost of SLATE with a 176-bit key and 2^{44} CRPs is only 663 gate equivalents (GEs). Compared with lightweight ciphers and existing secure strong PUFs, we show that SLATE is a practical security primitive for resource constrained systems for its extremely small footprint and security. Finally, we show that SLATE is information theoretically secure when valid CRPs are communicated through insecure channels.**

***Index Terms*—Authentication, hardware security, cryptography, ciphers, physical unclonable functions.**

I. Introduction

In recent years the demands of lightweight cryptographic solutions have been increasing for the growth applications enabled by the Internet of Things (IoT) and resource constrained low cost platforms [1]. Since in hardware the price of an application specific integrated circuit (ASIC) is roughly equivalent to the area in silicon it requires [2], a great deal of excellent work has already been done in the area of lightweight cryptography [3], [4]. It is known that lightweight ciphers can be used as authentication primitives, however, for the purpose of entity authentication only where confidentiality is not required during data transmission, the encryption/decryption circuits of these ciphers may not be fully utilized at all times. In other words, it is possible that the hardware cost of an entity authentication primitive can be even lower than the costs of existing lightweight ciphers.

Manuscript received November 3, 2018; revised February 28, 2019 and April 27, 2019; accepted May 14, 2019. Date of publication May 31, 2019; date of current version September 16, 2019. The work of S. N. Diggavi was supported in part by the NSF under Grant 1740047, in part by the Army Research Laboratory through the Cooperative Agreement Grant under Grant W911NF-17-2-0196, and in part by the UC-NL under Grant LFR-18-548554. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Chip-Hong Chang. *(Corresponding author: Wei-Che Wang.)*

The authors are with the Department of Electrical and Computer Engineering, University of California at Los Angeles, Los Angeles, CA 90095 USA (e-mail: weichewang@ucla.edu; yairyo99@ucla.edu; wuyizhang@ucla.edu; suhasdiggavi@ucla.edu; puneetg@ucla.edu).

Digital Object Identifier 10.1109/TIFS.2019.2919393

Other than lightweight ciphers, Physical Unclonable Functions (PUFs) have also been considered as promising security primitives that enable lightweight hardware implementations of identification [5], authentication [6], or secret key generation and storage [7]. The randomness of a PUF is extracted from random uncontrollable process variations, and its behavior, or Challenge Response Pair (CRP), is uniquely defined and is hard to predict or replicate. However, the stability of PUFs has always been a main issue that prevents PUFs from being put in a practical use. An attempt to overcome the instability of a PUF may result in large area or implementation overhead, which contradicts the purpose of PUF being used a lightweight authentication solution.

To address aforementioned potential improvements and concerns, we propose a Secure Lightweight Entity Authentication (SLATE) primitive. SLATE can be constructed from *any secure key storage* used in any existing ciphers or a model-based PUF [7], where the behavior of a strong PUF can be calculated from a compact model, therefore no CRP storage is needed. Most importantly, SLATE is more secure and hardware efficient (3.1× to 7.1× smaller) than existing strong PUFs from both empirical and theoretical perspectives. SLATE is also much more hardware efficient (more than 40% smaller) than existing lightweight ciphers, which can also be used as key-based entity authentication primitives. A 40% area reduction of SLATE compared to existing lightweight ciphers is a significant reduction, which translates to roughly 40% of cost reduction. Therefore, for the purpose of entity authentication, SLATE provides a much more cost efficient implementation than lightweight ciphers.

A. PUF-Based Authentication Protocols

Many PUF-based authentication protocols have been proposed since the PUF was introduced [8]. The simple authentication scheme proposed in [9] employs the CRP mechanism of strong PUFs, but the protocol is not practical because the PUF itself suffers from instability [10] and modeling attacks [11]. Recently, several secure strong PUFs are proposed and shown to be resistant to model building attacks [12], [13]. It has been shown that the use of XOR gates can effectively increase the difficulty of model building attacks. However, due to the number of XOR gates or parallel structures required, the hardware implementation costs of these secure strong PUFs are even higher than existing lightweight block ciphers, which can also be used as entity authentication primitives. Using an Optical PUF as stated in [14] is believed to be resistant to modeling attacks, however it may not adhere to the low-cost design principle of a PUF application. Most of the existing PUF-based authentication protocols aim to compensate the vulnerability to modeling attacks. Unfortunately, these approaches often undermine the benefits provided by the PUF technology, such as the lightweight implementation or the replacement of costly secure data storage, making most existing PUF-based protocols impractical [15].

B. Stability-Guaranteed PUF

One of the major research directions of PUF is the enhancement of its stability, and extensive effort has been devoted to the area (see [16]–[20] and references therein). It has been shown that stable PUFs can provide practical solutions including logic obfuscation [21], [22] and hardware metering [23]. Despite of the attractive benefits of a stable PUF, however, many applications require a guaranteed one-time secret key extraction, but to the best of our knowledge, the detailed physical mechanism of such key extraction has not been explained. A common one-time-read approach is to read the secret through eFuse and burn out the connections after reading out the values for characterization [22]. The approach assumes that the PUF is in a secure environment before reaching to the verifier, but this assumption may not be true. For example, the attacker can read out the secret *without* destroying the eFuse access to the secret at any point after the fabrication, and no one would know that the PUF has already been compromised. Another approach is to use asymmetric cipher framework, which comes with high design and implementation cost. Therefore there is still a need to develop a secure and efficient mechanism to extract secret from secure key storage.

C. Model-Based PUF

The concept of model-based PUF has been discussed in [7]. The advantages of a model-based PUF is that the verifier in an authentication protocol only stores a model of the PUF instead of storing a large CRP database. In [24], the delay signature of delay cells are extracted, so the verifier can emulate the response given the challenge. However, in addition to the complex delay characterization circuitry, the details of the one-time secret extraction mechanism is missing. In [25] the authors suggest that a strong PUF can be constructed from a weak PUF, however, the hardware cost of a digital Random Number Generator (RNG) or a stream cipher is commensurate to a cryptographic hash function [26], [27]. Similarly in [28], a strong Locally Enhanced Defectivity (LED)PUF is proposed by using a stable weak PUF as a key to a Hash-based Message Authentication Code (HMAC). However, the implementation cost of the strong LEDPUF is even higher than a block cipher due to the cryptographic hash function implementation. Therefore, a lightweight and secure authentication primitive is still yet to be discovered.

D. Main Contributions

The contributions of this paper include:

- A secure lightweight entity authentication primitive (SLATE) is proposed. We show that SLATE is 44% to 7.1× more hardware efficient than existing lightweight block/stream ciphers and secure strong PUFs.

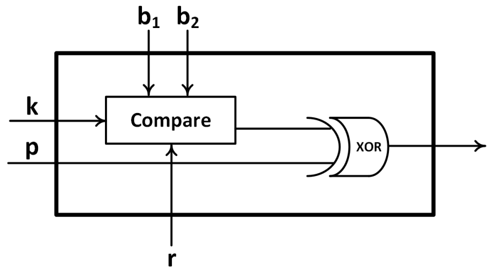

Fig. 1. Schematic of a Unit in the cascaded structure. All inputs and outputs of the Unit are 2-bit values.

- We show that SLATE is resistant to attacks that are effective to existing PUFs and digital logic obfuscations. Results of model building attacks show that the prediction rate is similar to random guessing, and the run time of Boolean Satisfiability (SAT) attack grows exponentially with the hardware size.
- From information theoretical perspective we show that SLATE is secure when the CRPs are communicated through an insecure channel.
- A physical tamper evident one-time-read secret extraction method is presented.

II. The Proposed Cascaded Architecture

In this section we first present the proposed cascaded architecture as the core design of the SLATE, and then we show that the cascaded architecture itself is resistant to model building attacks that are effective to existing strong PUFs. Following the model building attack results, we show that the cascaded structure is nevertheless not resistant to linear equation solving attack and single unit querying attack. To overcome these vulnerabilities, we present the complete secure SLATE structure in Section III-A.

A. Cascaded Unit Structure

The schematic of the Unit in the cascaded structure is given in Figure 1. All inputs and outputs of the Unit are 2-bit values. If k is equal to b_1 or b_2, then k is a *match*. The output of the Compare module in the Unit is defined as:

$$\text{Compare output} = \begin{cases} k, & \text{if } k \text{ is a match} \\ r, & \text{otherwise} \end{cases} \tag{1}$$

The proposed model building attack resistant structure is composed of cascaded Units as shown in Figure 2. b_{i1}, b_{i2} and r_i of the i^{th} Unit are secret values provided by a secret key, and k_i is the input challenge of the i^{th} Unit. For the first Unit, the output is simply the output of the Compare module; for all other Units, the output is k XOR p, where p is from the previous Unit.

For the cascaded structure, the entity authentication is done through CRP validation similar to a strong PUF where the challenge is all k_i combined and the response is the output of the last unit. The difference is that for the cascaded structure, a challenge is *valid* only when k_i is a match for

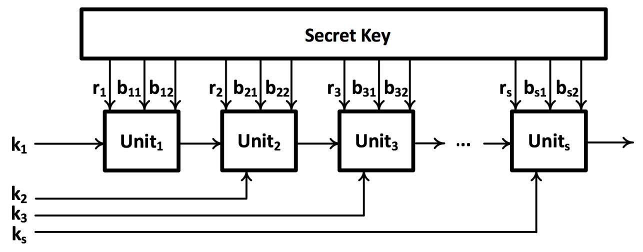

Fig. 2. Schematic of a secure cascaded structure.

TABLE I

PREDICTION RATE OF EACH OF THE OUTPUT BIT OF A 22-STAGE CASCADED UNIT STRUCTURE USING LR AND SVM ATTACKS. A VALUE CLOSE TO 50% INDICATES THAT THE PREDICTION RATE IS SIMILAR TO RANDOM GUESSING. INCREASING THE NUMBER OF TRAINING CRPS DOES NOT IMPROVE THE RATE. THESE VALUES SHOW THAT THESE ATTACKS ARE AS EFFECTIVE AS RANDOM GUESSING ONLY

Training CRPs	200	500	2,000	10,000	100,000
(LR) First Bit	50.0%	48.4%	49.1%	47.6%	49.4%
(LR) Second Bit	50.7%	50.0%	48.0%	49.9%	49.2%
(SVM) First Bit	48.3%	50.9%	50.3%	49.6%	51.1%
(SVM) Second Bit	48.2%	49.2%	46.8%	49.5%	50.5%

all Units. During authentication, only valid CRPs that have valid challenges are deployed. If any one of the Unit is not a match, the output of the Unit becomes r XOR p, which is a value that will not be used in any valid response. The only way to create a valid response for the attacker is to match every single Unit. For a cascaded structure with s Units and the length of b_1 and b_2 being 2-bit long, the probability of hitting a meaningful CRP is approximately $\frac{1}{2^s}$. Therefore, most CRPs collected by the attacker from the cascaded structure are not valid, making model building attacks ineffective.

B. Machine Learning Attack

Many strong PUFs with cascaded structure, such as Arbiter PUF [29], Feed Forward PUF [30], Lightweight secure PUF [31], or XOR-Arbiter PUF [7], [29], have been proven to be vulnerable to machine learning attacks [11], [32]. In [33], the authors even successfully break a commercial XOR PUF. In our model building attack model, we assume that the attacker has physical access and knows the complete design of the cascaded structure and can collect as many CRPs as possible for training. Given these assumptions we first try using Logistic Regression (LR) with sigmoid function with 44 independent variables because it has been shown to be effective to cascaded structures with well defined models [11] and Support Vector Machine (SVM) with Radial Basis kernel function to predict 1,000 unseen responses of a simulated 22-stage cascaded structure, assuming that the key bits are random. The input features of the attacks are k_i for all 22 stages (44 bits) and the training CRPs are generated randomly. Table I shows results of the prediction rate, which is defined as the probability of a correct prediction. We can see that the values are close to 50% for both attacks, which means that the prediction is ineffective and is similar to random guessing only, and there is no correlation between the prediction rate and the number of training CRPs.

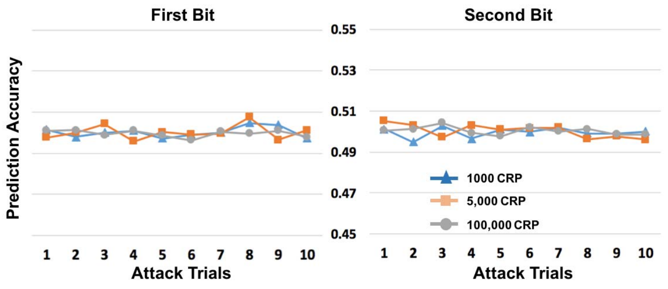

Fig. 3. NN attack results of 10 trials with different numbers of training CRPs. Prediction rates of both output bits are close to 50%, indicating that the attack is ineffective.

Next, we try to predict 1,000 unseen responses of the cascaded structure with a Multilayer Perceptron Neural Network (NN) using TensorFlow [34], which is another effective attack to existing PUFs [32]. Various numbers of hidden layers and units are tested, and the results are similar as shown in Figure 3, which is with 3 hidden layers, each has 128 units. For all training sizes (1,000 CRPs, 5,000 CRPs, and 100,000 CRPs) shown in the figure, the prediction rates of each of the response bit fall between 51% and 49% for all 10 attack trials (attempts), which means that the attack is not better than random guessing. Also, using a larger training set does not improve the prediction rate.

These results show that the proposed cascaded structure is resistant to machine learning attacks because it is difficult for the attacker to obtain valid or *meaningful* CRPs for the learning procedure and the use of XOR gates. A valid challenge must match either b_1 or b_2 for all Units in order to generate a valid response; otherwise, the response is not a valid response and therefore does not provide useful information to predict the response of an unseen valid challenge.

C. Linear Equation Solving Attack

In this section we show that the cascaded structure is vulnerable to linear equation solving attack if the attacker can observe many valid CRPs and has no physical access to the cascaded structure. Let s be the number of stages in the cascaded structure. Define l be the number of valid CRPs observed by the attacker. By comparing two valid challenges, the attacker knows whether the match of each Unit is changed or not. For example, when comparing 2 valid challenges 110101 and 110001 of a 3-stage cascaded structure, the attacker knows that the match of the second Unit is changed from 01 to 00, say from b_{21} to b_{22}. Let the match of the first challenge of the i^{th} Unit be b_{i1} for all i, and the responses of the first and second challenges be y_1 and y_2, respectively. The attacker now knows $y_1 = b_{11} \oplus b_{21} \oplus b_{31}$ and $y_2 = b_{11} \oplus b_{22} \oplus b_{31}$, where the $\oplus$ operation is a GF(2) operation that is essentially the XOR operation. The attacker can then construct a matrix multiplication formula $K_{valid} \cdot X = Y_{valid}$ from the two CRPs observed as:

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} b_{11} \\ b_{12} \\ b_{21} \\ b_{22} \\ b_{31} \\ b_{32} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix},$$

where K_{valid} is a $l \times 2s$ matrix representing the selection of matches, X is a $2s \times 2$ matrix representing b_{i1} and b_{i2} of all Units, and Y_{valid} is a $l \times 2$ matrix representing the 2-bit responses.

Whenever the attacker observes a new valid CRP, the number of rows in K_{valid} and Y_{valid} can be incremented by 1 following the same strategy. If adding the new row to K_{valid} does not increase the rank of K_{valid}, it means that the new CRP is a linear combination of observed CRPs, which indicates that the new response can be predicted. If adding the new row to K_{valid} increases its rank, the new CRP cannot be predicted. However, since K_{valid} has only $2s$ columns, the rank of it cannot exceed $2s$, therefore the number of CRPs that cannot be predicted is at most $2s$. In other words, during the life time of the cascaded structure, the number of secure valid CRP is at most $2s$, which is only linear to the number of stages.

D. Single Unit Querying Attack

In this section we show that the cascaded structure is vulnerable to single unit querying attack if (1) the attacker can only observe limited valid CRPs and (2) has physical access to the cascaded structure. Assume that the attacker observes a valid challenge $K = (k_1, k_2, \cdots, k_i, \cdots, k_s)$ and the response Y. For any i^{th} Unit, it is now known to the attacker that k_i is equal to either b_{i1} or b_{i2}. The attacker can then do the following steps:

1) Assume that k_i is equal to b_{i1} for all Units, the response can be represented as $Y = b_{11} \oplus b_{21} \oplus \cdots \oplus b_{i1} \oplus \cdots \oplus b_{s1}$.
2) For the i^{th} Unit, since k_i is a 2-bit value, the attacker can query all other 3 possible input values k_{i1}, k_{i2}, and k_{i3} to observe the 3 responses Y_1, Y_2, and Y_3.
3) At least 2 of the responses Y_1, Y_2, and Y_3 will be the same because 2 of the k_{i1}, k_{i2}, and k_{i3} correspond to non-match inputs, where the output of the module is r_i.
4) Let Y_1 and Y_2 be the same responses, a non-valid response $Y_1 = Y_2 = b_{11} \oplus b_{21} \oplus \cdots \oplus r_i \oplus \cdots \oplus b_{s1}$ is obtained, and an unseen valid response $Y_3 = b_{11} \oplus b_{21} \oplus \cdots \oplus b_{i2} \oplus \cdots \oplus b_{s1}$ with challenge $(k_1, k_2, \cdots, k_{i3}, \cdots, k_s)$ corresponds to the matched b_{i2} is obtained.

In fact, the attacker can further obtain $b_{i1} \oplus b_{i2}$ for the i^{th} Unit by performing $Y \oplus Y_3$. The information can be further exploited to calculate more unseen valid CRPs.

To prevent the linear equation solving attack, the matrix K_{valid} must be hidden to the attacker. In other words, when any 2 valid challenges are observed by the attacker, the selection of the matched b_1 or b_2 of any Unit should not be revealed, therefore the attacker cannot construct the matrix K_{valid}. To prevent the single unit querying attack, the cascaded structure needs to cause confusion to the attacker when different k_{i1}, k_{i2}, and k_{i3} are applied to the i^{th} Unit. The attacker should not be able to figure out which one is a real match from the responses. We propose SLATE architecture in the following section as a cascaded structure that is resistant to all aforementioned attacks.

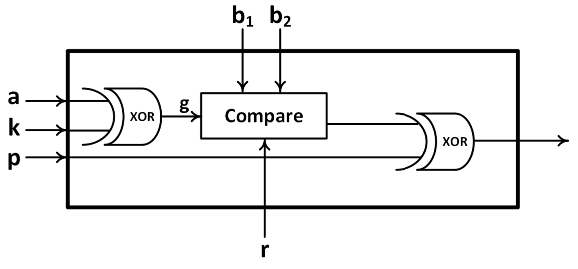

Fig. 4. Schematic of the S-Unit for SLATE.

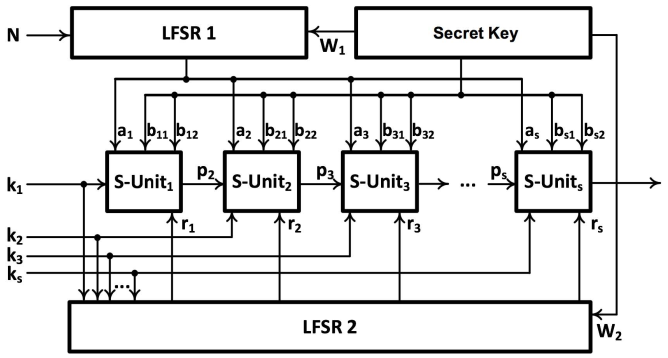

Fig. 5. Structure of the proposed cascaded SLATE.

III. The Proposed SLATE Architecture

A. Secure SLATE Structure

The secure SLATE is composed of cascaded S-Units as shown in Figure 4. The Compare module is the same as defined in Section II-A. However for a S-Unit, $g = k \oplus a$ is the input to the Compare module.

The complete SLATE structure with s S-Units is given in Figure 5. Define $A = (a_1, a_2, \cdots, a_s)$ and $R = (r_1, r_2, \cdots, r_s)$ the outputs of the two Linear Feedback Shift Registers (LFSRs). $G = (g_1, g_2, \cdots, g_s)$ is calculated from $K \oplus A$. The initial state of the first LFSR is determined by (N, W_1), and the initial state of the second LFSR is determined by (K, W_2), where W_1 and W_2 are secret bits of length N obtained directly from the secrete key. After the initial bits are loaded, the LFSRs are "warmed-up" by running a fixed F cycles to randomize the outputs of LFSRs sufficiently and to make sure that the outputs are depending on both C and the key. A and R, which depend on (N, W_1) and (K, W_2), respectively, are the inputs to the cascaded S-Units. For each of the i^{th} S-Unit, k_i is XORed with a_i to get g_i. If g_i is a match, then $g_i \oplus p_i$ is propagated to the next S-Unit; otherwise, $r_i \oplus p_i$, which will not exists in any valid challenge, is propagated to the next S-Unit. The input, or challenge, to the SLATE is $C = (K, N)$, and the response of the SLATE is the output of the last S-Unit.

B. Authentication Protocol

The enrollment and authentication phases are described as follows:

Enrollment phase: Similar to existing ciphers, the verifier shares the secret key with the hardware entity. No CRP collection is required.

Authentication phase:

1) The verifier generates a *valid* challenge $C = (K, N)$, which is a challenge that matches either b_1 or b_2 for all S-Units of the SLATE module.
2) The first LFSR and second LFSR take (N, W_1) and (K, W_2) as initial seeds, respectively. Both LFSRs run a "warm-up" phase for a fixed number F of clock cycles to generate inputs of the S-Unit.
3) The final response generated from the cascaded S-Units is sent to the verifier.
4) The verifier calculates the corresponding response of C and examines the response from the SLATE. If the response is a complete match, the entity is authenticated; otherwise, the entity is not authenticated.

A valid challenge can be calculated easily by the verifier because the verifier knows W_1, W_2, b_1 and b_2 of all S-Units. Once a N is decided, the verifier calculates the states of the LFSRs after F cycles and find K that matches all S-Units. The number of valid CRPs is 2^N because every N is used only once to prevent the linear equation solving attack. Since the key of SLATE must be completely stable like any existing ciphers, every response must be completely matched to authenticate the entity. Therefore, inter- and intra-Hamming distances [7] are not proper metrics for the SLATE. Please note that multiple CRPs may be used (similar to Arbiter PUF with only 1-bit response) to prevent the attacker from impersonating SLATE with high success probability since the response of SLATE contains only 2 bits. For example, if only one CRP is used for authentication, the probability of guessing the correct response is 2^{-2}; however if 64 CRPs are used, the probability becomes 2^{-128} since the attacker needs to impersonate all 128 bits. The throughput comparisons between SLATE and other lightweight ciphers are presented in Table VI.

The SLATE structure used in the proposed authentication protocol is resistant to model building attack because its underlying cascaded structure is secure. Also, it is resistant to linear equation solving attack because when comparing 2 valid challenges, the attacker cannot figure out which one of the b_1 and b_2 is the match as the output of the first LFSR changes for every valid challenge. Therefore the attacker cannot construct K_{valid} given that a new N is used for every new valid challenge. It is resistant to single S-Unit querying attack because even if the attacker can fix N and try different K, the r of each S-Unit also changes because of the second LFSR, therefore the attacker will not be able to distinguish the r from a match.

C. Boolean Satisfiability (SAT) Attack

In this section we show that the SLATE is resistant to SAT attack proposed in [35], which has been shown to be an effective attack to retrieve the correct key of many logic obfuscation schemes. The SAT attack algorithm allows an attacker to decipher an obfuscated circuit using a small number of carefully selected input patterns and their corresponding outputs (distinguishing input/output pairs or DIPs) which can be observed from an activated functional chip. The algorithm finds such DIPs ($\boldsymbol{X}^d$/$\boldsymbol{Y}^d$) iteratively and formalizes them as a sequence of SAT formulas which can be solved by SAT solver. Each DIP rules out a subset of wrong key combinations and the algorithm terminates when all wrong keys have been removed. The SAT attack algorithm guarantees to find the equivalent class of the correct key upon the termination [36].

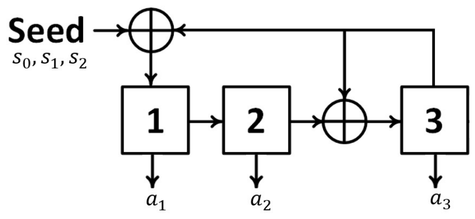

Fig. 6. 3-bit LFSR with polynomial $x^3 + x^2 + 1$.

TABLE II
EXPANSION OF 3-BIT LFSR

Cycle	Seed	a_1	a_2	a_3
0	s_0	0	0	0
1	s_1	s_0	0	0
2	s_2	s_1	s_0	0
3	0	s_2	s_1	s_0
4	0	s_0	s_2	$xor(s_1, s_0)$
5	0	$xor(s_1, s_0)$	s_0	$xor(s_2, s_1, s_0)$
6	0	$xor(s_2, s_1, s_0)$	$xor(s_1, s_0)$	$xor(s_2, s_1)$

The objective of an attacker is to obtain the correct key of the obfuscated circuit. In the case of SLATE, the attacker wishes to find the value of $\boldsymbol{B} = (b_{11}, b_{12}, b_{21}, b_{22}, \ldots, b_{s1}, b_{s2})$, and $\boldsymbol{W} = (W_1, W_2)$. We hereby note these key values as $\boldsymbol{WB}$. According to Kerckhoff's principle and Shannon's maxim [37], a cryptographic system should be secure even if everything about the system, except the key, is public knowledge. Therefore, the attack model assumes that the attacker has access to the following two components:

1) The gate-level netlist of the SLATE, which can be represented as $\boldsymbol{Y} = f(\boldsymbol{X}, \boldsymbol{WB})$, where $\boldsymbol{X} = (N, K)$ is the primary input and $\boldsymbol{Y}$ is the primary output of the circuit. The SAT formula of the netlist in conjunction normal form (CNF) is represented as $C(\boldsymbol{X}, \boldsymbol{WB}, \boldsymbol{Y})$.
2) The physical victim SLATE module, which is used to observe the correct output given an input $\boldsymbol{Y} = eval(\boldsymbol{X})$.

When modeling SLATE into CNF form, we first expand both LFSRs in Figure 5 into XOR networks. Given the structure, seeds and number of warm-up cycles of a LFSR, the attacker can obtain the final expression of each bit and model LFSR as a combinational network with only XOR gates. For example, given the sequence of a 3-bit seed, s_0, s_1, s_2, the expression for each bit a_1, a_2 and a_3 of a 3-bit LFSR after 6 warm-up cycles as shown in Figure 6 can be computed as in Table II. Therefore, this LFSR can be unrolled to a XOR network with inputs s_0, s_1, s_2 and outputs a_1, a_2, a_3, where $a_1 = xor(s_2, s_1, s_0)$, $a_2 = xor(s_1, s_0)$, and $a_3 = xor(s_2, s_1)$.

The goal of the attacker is to infer the fixed secret key of $\boldsymbol{B} = (b_{11}, b_{12}, b_{21}, b_{22}, \ldots, b_{s1}, b_{s2})$ and $\boldsymbol{W} = (W_1, W_2)$

Algorithm 1 SAT Attack Algorithm [35]

```
Input: CNF and $eval$
Output: $\boldsymbol{WB}_C$
1: $i = 1$;
2: $G_i = True$;
3: $F_i = C(\boldsymbol{X}, \boldsymbol{WB}_1, \boldsymbol{Y}_1) \wedge C(\boldsymbol{X}, \boldsymbol{WB}_2, \boldsymbol{Y}_2) \wedge (\boldsymbol{Y}_1 \neq \boldsymbol{Y}_2)$;
4: while $sat[F_i]$ do
5:     $\boldsymbol{X}_i^d = sat\_assignment_X[F_i]$;
6:     $\boldsymbol{Y}_i^d = eval(\boldsymbol{X}_i^d)$;
7:     $G_{i+1} = G_i \wedge C(\boldsymbol{X}_i^d, \boldsymbol{WB}, \boldsymbol{Y}_i^d)$;
8:     $F_{i+1} = F_i \wedge C(\boldsymbol{X}_i^d, \boldsymbol{WB}_1, \boldsymbol{Y}_i^d) \wedge C(\boldsymbol{X}_i^d, \boldsymbol{WB}_2, \boldsymbol{Y}_i^d)$;
9:     $i = i + 1$;
10: $\boldsymbol{WB}_C = sat\_assignment_{\mathbf{WB}}(G_i)$;
```

TABLE III

SAT Attack on SLATE With Different Number of S-Units. The Required Time Grows Exponentially With the Number of S-Units

S-Units	6	8	10	12	14	22
Time (s)	36	228	1,125	27,237	N/A	N/A

using the SAT attack algorithm shown in Algorithm 1 to attack SLATE.

We apply SAT attack on SLATE with different number of stages with scaled key size to evaluate the effectiveness of the SAT attack. The execution time required to solve the correct key is presented in Table III. We use the SAT Solver developed in [38] and implement the attack in C++. The attack is run on a Quad Intel Xeon 2.8GHz CPU server, and the run time limit for the attack is set to 10 hours. The results show that the execution time of SAT attack algorithm grows exponentially with the number of SLATE stages. For SLATE with 14 or more S-Units, the SAT attack fails to find the correct key within the 10-hour time limit. These results indicate that SLATE is resistant to the SAT attack.

D. Hardware Implementation

The hardware implementation of SLATE shown in Fig. 5 is not unique but rather depending on the choice of trade-off between area and the number of CRPs. The main deciding factor is the selection of LFSRs. The output of SLATE could be any number of bits but the size of LFSRs grow linearly with the number of bits. For example, for a 22-stage SLATE with n-bit output, the number of CRPs is 2^{22n}, and the length of LFSR is 22n. We propose to use 2-bit output and compare it to existing authentication primitives as it results in the most effective trade-off point between implementation size and the number of CRPs.

A maximum-length N-bit LFSR along with W_1 provides 2^N different outcomes to mask input K, therefore the number of valid CRPs is 2^N. Since the length of b_{i1} and b_{i2} for the i^{th} unit are 2-bit each, a N-bit LFSR after each warm-up can support $\frac{N}{2}$ units, which determines the probability of a CRP being valid. Since each unit requires $2\times$2-bit keys for b_{i1} and b_{i2} and the length of W_1 and W_2 are both N, the total length of the key is $4N$. Table IV shows three version of possible SLATE implementations.

TABLE IV

Bit Specifications of a Variety of SLATE Implementations. Last Column Shows the Number of Valid CRPs

	$N(K)$	$W_1(W_2)$	LFSR	Key	CRP
176-bit key SLATE	44	44	44	176	2^{44}
128-bit key SLATE	32	32	32	128	2^{32}
104-bit key SLATE	26	26	26	104	2^{26}

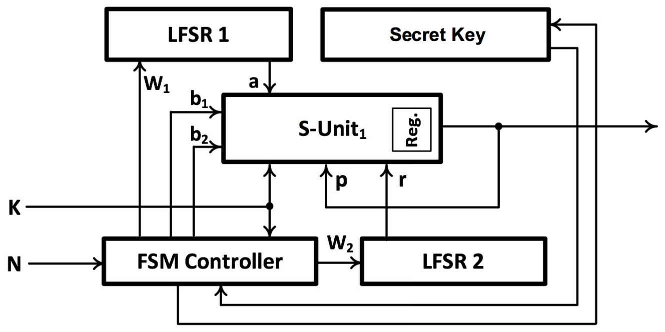

Fig. 7. Compact SLATE implementation.

In this section we present the detailed hardware implementation cost of the 176-bit key SLATE with 44-bit LFSR and 22-stages, where the probability of a CRP being valid is $\frac{1}{2^{22}}$ for the number of stages. The Finite State Machine (FSM) controller module takes N, K, W_1, and W_2 to initialize the LFSRs. Since the values of the secret key are not directly accessed but one-time padded before the real matching, the attacker learns nothing about the secret by observing CRPs as proven in Section IV.

The SLATE design is implemented in Verilog and synthesized using a commercial 65nm CMOS technology using Cadence Genus. We use Gate Equivalents (GE) for area evaluation, where one GE is equivalent to the area of a NAND2 gate with the lowest driving strength of the corresponding technology. The SLATE implementation with minimized area is presented in Figure 7. Instead of implementing all 22 S-Units, a 2-bit register is added to the S-Unit and the cascaded structure is implemented as a sequential loop. The FSM controller reads K, N, and sends the address to the key storage to request the key one bit per clock cycle. A counter in the FSM controller is used to start the initialization (warm-up) of LFSRs. After the initialization, b_1 and b_2 are requested by the FSM controller and the evaluation of the S-Unit starts.

The LFSR implemented is a 44-bit maximum-length internal LFSR with primitive polynomial $x^{44}+x^6+x^5+x^2+1$ [39] as shown in Figure 8. The initial bits of the LFSR is loaded one bit per clock cycle from the feedback XOR loop. Once all bits are loaded, the LFSR runs $4 \times 44 = 176$ cycles in the warm-up phase before evaluating the S-Unit.

The implementation cost of the S-Unit including the 2-bit register is 33.75 GE. It takes 205.5 GE to implement the 44-bit LFSR using pules-latch structure [40]. The FSM controller would require 218.25 GE. Therefore the total GE required

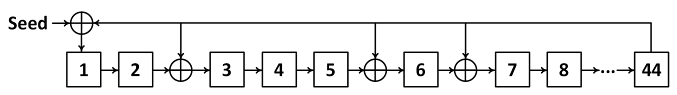

Fig. 8. 44-bit LFSR with polynomial $x^{44} + x^6 + x^5 + x^2 + 1$.

TABLE V
GE COMPARISONS BETWEEN 176-BIT SLATE AND SECURE STRONG PUFs WITH 2^{44} CRPs

PUF	GE	Ratio to SLATE
60-Stage Secure XOR PUF [12]	3,369	3.10
Strong LEDPUF [41]	3,517	3.23
44 x 44 LRR-DPUF [13]	7,744	7.11
SLATE + OTR module	1,088	1.00

TABLE VI
AREA (GE) AND THROUGHPUT COMPARISONS BETWEEN SLATE AND LIGHTWEIGHT CIPHERS WITH 128-BIT KEY. SLATE WITH SHORTER KEY SIZE HAS HIGHER THROUGHPUT BUT SMALLER CRP SPACE. THE AREAS SHOWN DO NOT INCLUDE THE KEY STORAGE FOR FAIRNESS COMPARISONS AS REPORTED IN THE LITERATURE

Cipher	GE	GE Ratio to 176/128/108 SLATE	kbps
* PRESENT [43]	1,339	2.0 / 2.4 / 2.7	12.1
SIMON [3]	958	1.4 / 1.7 / 1.9	4.2
SPECK [3]	996	1.5 / 1.8 / 2.0	3.6
PICOLLO [44]	1,362	2.1 / 2.5 / 2.8	12.1
Grain [27]	1,857	2.8 / 3.4 / 3.8	0.9
* Trivium [27]	2,599	3.9 / 4.7 / 5.3	0.4
176-bit SLATE	663	1.0 / 1.2 / 1.3	1.01
128-bit SLATE	550	0.8 / 1.0 / 1.1	1.38
104-bit SLATE	494	0.8 / 0.9 / 1.0	1.71

* 80-bit key cipher

to implement the 176-bit key SLATE is $33.75 + 205.5 \times 2 + 218.25 = 663$ GE, which is about 955 μm^2 for 65nm technology. If a 176-bit stable weak LEDPUF [41] is used as the key storage for SLATE, the cost of the PUF and the one-time-read (OTR) implementation in Section V is about 425 GE, including an AntiFuse (AF), a current comparator, an address decoder at the key storage output and an eFuse cell connected to the output of the decoder.

Table V shows the comparisons between stable and secure strong PUFs and SLATE with the OTR 176-bit key. The 425 GE key storage cost for SLATE is included in the comparisons because the secure storage is required for SLATE, while for strong PUFs the secret randomness is from intrinsic process variation. In [12] the authors suggest that a XOR PUF is secure when no less than 10 delay-based strong PUFs are XORed in parallel. However, only 0.0028% of the CRPs are stable after more than 10 XORs, therefore, to provide 2^{44} CRPs, 60-stage delay-based PUFs are required, and its area is about 3.1× of the SLATE. In [41] the authors proposed to use a cryptographic hash function with a weak PUF to construct a strong PUF. However, the combined area of a lightweight SHA-3 [42] and the weak PUF decoder is more than 3.2× of the SLATE. Another secure strong LRR-DPUF uses XOR network to generate secure CRPs [13]. To provide 2^{44} CRPs, one possible structure of LRR-DPUF is with a 44 x 44 XOR network as suggested in [13], which comes with an area of 7.1× of the SLATE. For a LRR-DPUF with similar GE as SLATE, the number of CRPs is approximately 2^{17}, which is too small and can be broken by simply collecting all possible CRPs. Please note that these comparisons are based on LRR-DPUFs with same number of rows and columns of the XOR network. Results of other LRR-DPUFs may vary depending on the design of the XOR network. One advantage of the LRR-DPUF is the throughput for its parallel architecture. More detailed comparisons and discussions about throughput of entity authentication primitives are presented in the following paragraph. Please note that these comparisons are based on estimated values given the design of the secure strong PUFs.

Since entity authentication can also be accomplished by ciphers, in Table VI we compare the implementation costs and throughputs (kbps) at a 100kHz clock frequency of three versions of SLATE to existing lightweight block ciphers and stream ciphers excluding the key storage. From the table we can see that the hardware cost in GE of existing ciphers are 44% to 5.26× larger than SLATE. These results show that for the purpose of entity authentication, SLATE is a much more compact primitive than existing solutions. The throughput of SLATE is higher than the two stream ciphers (Grain and Trivium) but not as high as block ciphers due to the nature of these ciphers. In the case of multiple CRPs of SLATE may be used to authenticate the device, SLATE could be slower than block ciphers. SLATE with shorter key size has smaller CRP space as shown in Table IV but higher throughput because of the shorter LFSR. In fact, the impact of throughput to an authentication primitive is not as significant as to a cipher because authentication is usually done only once on a small size bit stream, for example a 128-bit response, while encryption/decryption can be done on much larger message or data for the entire information transmission. Therefore, for the purpose of entity authentication for resource constrained systems, such as the Internet of Things (IoT), SLATE provides an overall better option than lightweight ciphers with significant area reduction and limited runtime overhead on the response time.

IV. THEORETICAL GUARANTEES

In this section we show that under idealized assumption SLATE is secure from information theoretical perspective [45] when the attacker can observe valid CRPs. We begin by describing a random matrix multiplication problem, we then show that it is information theoretically secure, and then connect it to the scheme of SLATE.

First let us define the following problem of random matrix multiplication.

Definition 1: Consider the following set of equations

$$Y = G \cdot X \tag{2}$$

where Y consists of l rows and two columns, G is a $l \times s$ matrix, and X has s rows and two columns. Furthermore, each entry of G as well as the entries of X are independent and identically distributed (i.i.d.) Bernoulli$(1/2)$. Finally, $\mathcal{B}$ represents the set of invertible matrices with s rows and two columns.

The following lemma shows that the mutual information [46] between X and Y is equal to zero as long as X is invertible, and also shows that the probability that X is not full rank decreases exponentially with s.

Lemma 1: When $X \in \mathcal{B}$ the mutual information between X and Y is equal to zero, that is,

$$I(X; Y|X \in \mathcal{B}) = 0. \tag{3}$$

Furthermore, $P(X \notin B) \leq 3 \cdot 2^{-s}$.

Proof: All entries of G are i.i.d. Bernoulli(1/2). When X is invertible it means that its columns are different and both are not equal to zero, and so the entries of Y are also i.i.d. Bernoulli(1/2). Therefore, the entropy of Y and $X|Y$ is the same and so the mutual information in this case is zero. In the case when X is not full rank, the mutual information between X and Y is no longer zero, and this happens when either the columns are equal or one of them is equal to zero. The probability of this event is upper bounded by $3 \cdot 2^{-s}$, that is, $P(X \notin \mathcal{B}) \leq 3 \cdot 2^{-s}$. □

Theorem 1: Assume that G is one-time-padded such that $K = G \oplus A$ is observed, where K corresponds to the partial challenge and A corresponds to the first LFSR output of the SLATE (i.e., l responses as those are matrices with l rows). The matrix X represents the secret values of all S-Units, that is, b_1 and b_2, and Y represents l observed responses of SLATE. When both K and A are drawn i.i.d. Bernoulli(1/2), the amount of information which is exposed when observing K along with l responses (Y), is zero with probability which is upper bounded by $3 \cdot 2^{-s}$, where the number of S-Units is equal to s. Therefore, as s increases this scheme becomes information theoretically secure.

Proof: For simplicity let us assume that the possible values in the S-Units are either zero or two random bits that are drawn i.i.d. Bernoulli(1/2). This simplified assumption also holds when the challenge chooses between two pairs of random bits. As long as G is XORed with A, it can be assumed that no information on the matrix G is exposed when observing K (i.e., the mutual information between G and the K is equal to zero) and so the attacker does not know the matrix G in equation (2). In this case, based on Lemma 1 we get that the amount of information, which is revealed when observing valid values of K along with l responses (Y), is equal to zero as long as X is invertible, and that the probability that this is not the case is upper bounded by $3 \cdot 2^{-s}$. This proves the theorem. □

Remark 1: When the number of S-Units in the scheme $s = 22$ we get that the probability that the mutual information is not equal to zero is upper bounded by $3 \cdot 2^{-22}$.

Remark 2: In practice a challenge is XORed with the output of the first LFSR of the scheme presented in Section III-A and not with a "pure" source of randomness.

Remark 3: It is assumed that the attacker observes both Y and $K = G \oplus A$, where A is i.i.d. Bernoulli(1/2). Therefore, one may be interested in $I(X; G \oplus A, Y)$. However, in our setting $I(X; G \oplus A, Y) = I(X; Y)$. This is because $I(X; G \oplus A|Y) = 0$ as Y, X and G are statistically independent of A, which is drawn i.i.d. Bernoulli(1/2). This is the reason why in theorem 1 we focus on $I(X; Y)$.

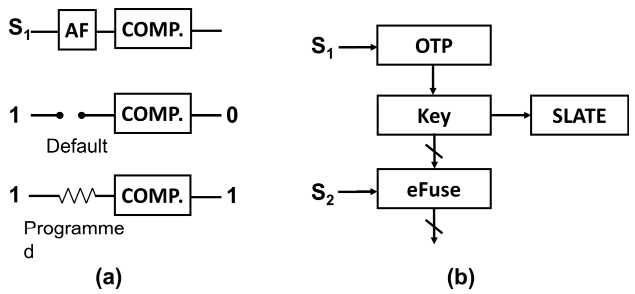

Fig. 9. (a) OTP module implementation. An AF cell is connected to a current comparator to generate different outputs. (b) One-time-read secret extraction. The OTP activates the key after the AF cell is irreversibly programmed. Once the key is accessed, the eFuse connections is burnt so only the SLATE module has access to the key.

V. Tamper Evident One-Time-Read Method

We propose a tamper evident one-time-read method which can be applied to any applications where one-time access to sensitive data is required.

In the method both eFuse and AF cells are used. The implementation of an One-Time-Programmable (OTP) module is given in Figure 9 (a). It is composed of an AF cell and a common current comparator as mentioned in the AF memory design in [47]. Since the current difference of AF between default (open) and programmed (close) states are larger than 100X, when one (VDD) is applied to the AF, the comparator can effectively distinguish the two states to generate different outputs as demonstrated in [47] with AF cells fabricated using 0.18μm technology. When applied with one, the current of a default AF is less than $100pA$, and for a programmed AF, the current is larger than $20\mu A$. Therefore, with $2\mu A$ reference current the comparator can distinguish and generate different outputs accordingly. As shown in Figure 9 (b), the output of the OTP is used to activate the secret key, such as the enable signal or the power gating signal to any storage modules. Once the AF cell is permanently programmed, the key that provides secret values to the proposed SLATE module can be evaluated through eFuse connections. The steps of the method is given in the following:

Tamper Evident One-Time-Read Method:

1. Set_1 is set to one (VDD) and the outputs of the eFuse connections are evaluated. If the values of key can be accessed, it means that the AF connections have been programmed to the closed states, which implies that an unexpected read is detected and the primitive should be discarded. If the output is floating, continue to step 2.
2. S_1 programs the AF cell to the closed state.
3. OTP activates the key and the secrets are read through eFuse connections.
4. Set_2 programs the eFuse connections to the open states after the reading.

By examining the states of the AF connections in step 1, the user knows the reading history of the secret key and can discard the primitives that have been read before. Therefore, the proposed method is tamper evident. The method also guarantees one-time-read because the eFuse connections are programmed to open states in step 4, where the read channel is destroyed permanently. The AF cell cannot be replaced by eFuse because otherwise the default states of eFuse is closed and an attacker does not have to program the eFuse to activate the key. The eFuse connections cannot be replaced by AF cells either because once the AF cell is programmed, it cannot be reversed. Please note that the SLATE module in Figure 9 (b) can be any applications.

VI. Conclusions

In this paper we propose SLATE as a lightweight and secure entity authentication primitive. We show that SLATE is resistant to model building attacks and SAT attacks that are known to be effective to most existing strong PUFs or logic obfuscation. We compare the hardware implementation area of SLATE to secure strong PUFs and ciphers to show that existing entity authentication solutions are at least 44% to 7.1× larger than SLATE. Also, we show that ideally SLATE is information theoretically secure in terms of the amount of information revealed by observing valid CRPs. Finally, we propose a tamper evident one-time-read method to ensure the unpredictability and the unclonability of SLATE. Our future work is focused on the hardware fabrication of SLATE to study attacks utilizing side channel information and protecting strategies.

Acknowledgment

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] S. Surendran, A. Nassef, and B. D. Beheshti, "A survey of cryptographic algorithms for IoT devices," in *Proc. IEEE LISAT*, May 2018, pp. 1–8.
[2] C. Rolfes, A. Poschmann, G. Leander, and C. Paar, "Ultra-lightweight implementations for smart devices—Security for 1000 gate equivalents," in *Smart Card Research and Advanced Applications*. Berlin, Germany: Springer, 2008.
[3] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK families of lightweight block ciphers," IACR Cryptol. ePrint Arch., Tech. Rep. 2013/404, 2013.
[4] R. Beaulieu, S. Treatman-Clark, D. Shors, B. Weeks, J. Smith, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," in *Proc. DAC*, Jun. 2015, pp. 1–6.
[5] A. R. Krishna, S. Narasimhan, X. Wang, and S. Bhunia, "MECCA: A robust low-overhead PUF using embedded memory array," in *Proc. CHES*, Oct. 2011, pp. 407–420.
[6] W. Che, F. Saqib, and J. Plusquellic, "PUF-based authentication," in *Proc. ICCAD*, Nov. 2015, pp. 337–344.
[7] C. Herder, M.-D. Yu, F. Koushanfar, and S. Devadas, "Physical unclonable functions and applications: A tutorial," *Proc. IEEE*, vol. 102, no. 8, pp. 1126–1141, Aug. 2014.
[8] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. CCSC*, 2002, pp. 148–160.
[9] S. Devadas, E. Suh, S. Paral, R. Sowell, T. Ziola, and V. Khandelwal, "Design and implementation of PUF-based 'unclonable' RFID ICs for anti-counterfeiting and security applications," in *Proc. Int. Conf. RFID*, Apr. 2008, pp. 58–64.
[10] Q. Ma, C. Gu, N. Hanley, C. Wang, W. Liu, and M. O'Neill, "A machine learning attack resistant multi-PUF design on FPGA," in *ASP-DAC*, Jan. 2018, pp. 97–104.
[11] U. Rührmair *et al.*, "PUF modeling attacks on simulated and silicon data," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 11, pp. 1876–1891, Nov. 2013.
[12] C. Zhou, K. K. Parhi, and C. H. Kim, "Secure and reliable XOR arbiter PUF design: An experimental study based on 1 trillion challenge response pair measurements," in *Proc. DAC*, Jun. 2017, pp. 1–6.
[13] J. Miao, M. Li, S. Roy, and B. Yu, "LRR-DPUF: Learning resilient and reliable digital physical unclonable function," in *Proc. ICCAD*, Nov. 2016, pp. 1–8.
[14] R. Maes and I. Verbauwhede, "Physically unclonable functions: A study on the state of the art and future research directions," in *Towards Hardware-Intrinsic Security*. Berlin, Germany: Springer, 2010.
[15] J. Delvaux, D. Gu, D. Schellekens, and I. Verbauwhede, "Secure lightweight entity authentication with strong PUFs: Mission impossible?" in *Proc. CHES*, Sep. 2014, pp. 451–475.
[16] W.-C. Wang, Y. Yona, Y. Wu, S.-Y. Hung, S. Diggavi, and P. Gupta, "Implementation of stable PUFs using gate oxide breakdown," in *Proc. IEEE AsianHOST*, Oct. 2017, pp. 13–18.
[17] M.-Y. Wu *et al.*, "A PUF scheme using competing oxide rupture with bit error rate approaching zero," in *Proc. ISSCC*, Feb. 2018, pp. 130–132.
[18] M. Gao, K. Lai, and G. Qu, "A highly flexible ring oscillator PUF," in *Proc. DAC*, Jun. 2014, pp. 1–6.
[19] T. Xu and M. Potkonjak, "Stable and secure delay-based physical unclonable functions using device aging," in *Proc. IEEE ISCAS*, May 2015, pp. 33–36.
[20] R. Liu, H. Wu, Y. Pang, H. Qian, and S. Yu, "A highly reliable and tamper-resistant RRAM PUF: Design and experimental validation," in *IEEE HOST*, May 2016, pp. 13–18.
[21] M. Yasin, B. Mazumdar, J. J. V. Rajendran, and O. Sinanoglu, "SAR-Lock: SAT attack resistant logic locking," in *IEEE HOST*, May 2016, pp. 236–241.
[22] J. B. Wendt and Mi. Potkonjak, "Hardware obfuscation using PUF-based logic," in *Proc. ICCAD*, 2014, pp. 270–271.
[23] M. T. Rahman, D. Forte, Q. Shi, G. K. Contreras, and M. Tehranipoor, "CSST: Preventing distribution of unlicensed and rejected ICs by untrusted foundry and assembly," in *IEEE Int. Symp. DFT*, Oct. 2014.
[24] M. Majzoobi and F. Koushanfar, "Time-bounded authentication of FPGAs," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 1123–1135, Sep. 2011.
[25] U. Ruhrmair, J. Sölter, and F. Sehnke, "On the foundations of physical unclonable functions," IACR Cryptol. ePrint Arch., Tech. Rep. 2009/277, 2009.
[26] V. van der Leest, E. van der Sluis, G.-J. Schrijen, P. Tuyls, and H. Handschuh, "Efficient implementation of true random number generator based on SRAM PUFs," in *Cryptography and Security: From Theory to Applications*. Berlin, Germany: Springer, 2012.
[27] T. Good and M. Benaissa, "Hardware results for selected stream cipher candidates," in *Proc. Workshop Rec. SASC*, 2007, pp. 1–14.
[28] W.-C. Wang, Y. Yona, S. N. Diggavi, and P. Gupta, "Design and analysis of stability-guaranteed PUFs," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 4, pp. 978–992, Apr. 2018.
[29] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. DAC*, 2007, pp. 9–14.
[30] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *Proc. IEEE Int. Symp. VLSI Circuits*, Jun. 2004, pp. 176–179.
[31] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure PUFs," in *Proc. ICCAD*, Nov. 2008, pp. 670–673.
[32] G. Hospodar, R. Maes, and I. Verbauwhede, "Machine learning attacks on 65 nm Arbiter PUFs: Accurate modeling poses strict bounds on usability," in *Proc. IEEE WIFS*, Dec. 2012, pp. 37–42.
[33] G. T. Becker, "The gap between promise and reality: On the insecurity of XOR arbiter PUFs," in *Proc. CHES*, Sep. 2015, pp. 535–555.
[34] *TensorFlow*. Accessed: Aug. 2017. [Online]. Available: https://www.tensorflow.org
[35] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *Proc. IEEE HOST*, May 2015, pp. 137–143.

[36] Y. Xie and A. Srivastava, "Mitigating SAT attack on logic locking," in *Proc. CHES*, Aug. 2016, pp. 127–146.

[37] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, no. 4, pp. 656–715, Oct. 1949.

[38] T. Balyo and M. Heule, "CHBR_glucose," in *Proc. SAT Competition*, 2016. Accessed: Aug. 2017. [Online]. Available: http://www.satcompetition.org/

[39] L.-T. Wang, C.-W. Wu, and X. Wen, *VLSI Test Principles and Architectures: Design for Testability*. San Mateo, CA, USA: Morgan Kaufmann, 2006.

[40] K. H. Kishore *et al.*, "Power and area efficient LFSR with pulsed latches," *Int. J. Pure Appl. Math.*, vol. 115, no. 7, pp. 447–452, 2017.

[41] W.-C. Wang, Y. Yona, S. Diggavi, and P. Gupta, "LEDPUF: Stability-guaranteed physical unclonable functions through locally enhanced defectivity," in *Proc. IEEE HOST*, May 2016, pp. 25–30.

[42] P. Pessl and M. Hutter, "Pushing the limits of SHA-3 hardware implementations to fit on RFID," in *Proc. CHES*, Aug. 2013, pp. 126–141.

[43] H. Yap, K. Khoo, A. Poschmann, and M. Henricksen, "EPCBC—A block cipher suitable for electronic product code encryption," in *Cryptology and Network Security*. Berlin, Germany: Springer, Dec. 2011.

[44] K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, "*Piccolo*: An ultra-lightweight blockcipher," in *Proc. CHES*, Sep. 2011, pp. 342–357.

[45] J. L. Carter and M. N. Wegman, "Universal classes of hash functions (Extended Abstract)," in *Proc. ACM STOC*, New York, NY, USA, 1977, pp. 106–112.

[46] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: Wiley, 2006.

[47] L. Xian, Z. Huicai, J. Cheng, and L. Xin, "A 4-kbit low-cost antifuse one-time programmable memory macro for embedded applications," *J. Semicond.*, vol. 35, no. 5, pp. 5, May 2014.

Wei-Che Wang received the B.S. degree in computer science and the M.S. degree in electronics engineering from National Taiwan University (NTU), Taiwan, in 2007 and 2009, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of California at Los Angeles (UCLA), in 2018. From 2009 to 2013, he was with the Standard Cell Library Department, Taiwan Semiconductor Manufacturing Company (TSMC), as an Engineer. He joined the NanoCAD Laboratory, UCLA, led by Prof. P. Gupta in 2013. In 2018, he joined Cadence Design Systems, San Jose, CA, USA. His research interests include the development of computational techniques for optimizing semiconductor technologies and hardware security.

Yair Yona received the B.Sc. and M.Sc. degrees *(magna cum laude)* in electrical engineering and the Ph.D. degree in electrical engineering from Tel-Aviv University, Israel, in 2005, 2009, and 2014, respectively.

From 2003 to 2008, he served as a DSP and Algorithms Engineer for several companies, including Intel Corporation, Marvell Ltd., and Amimon Ltd. From 2015 to 2017, he was a Post-Doctoral Scholar with the Department of Electrical Engineering, University of California at Los Angeles, Los Angeles, CA, USA. From 2017 to 2019, he was a Research Scientist with Intel Labs, Santa Clara, CA, USA. He is currently a Staff Engineer at Qualcomm, San Jose, CA, USA.

Dr. Yona was a recipient of the Intel Award for excellence in academic studies and research in 2009, a Motorola Scholarship in the field of advanced communication in 2009, and the Weinstein Prize in 2010 and 2014, for research in the area of signal processing.

Yizhang Wu received the B.S. degree in electrical engineering and the M.S. degree in electrical and computer engineering from the University of California at Los Angeles (UCLA) in 2017 and 2019, respectively. He joined the NanoCAD Laboratory, UCLA, in 2016, where he is involved in research on hardware security and 2.5D integration. In 2019, he joined Micron Technology as a Design Engineer.

Suhas N. Diggavi received the B.Tech. degree in electrical engineering from IIT Delhi, New Delhi, India, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, USA, in 1998. He was a Principal Member Technical Staff with the Information Sciences Center, AT&T Shannon Laboratories, Florham Park, NJ, USA. He was on the faculty of the School of Computer and Communication Sciences, EPFL, where he directed the Laboratory for Information and Communication Systems (LICOS). He is currently a Professor with the Department of Electrical and Computer Engineering, University of California, Los Angeles, where he directs the Information Theory and Systems Laboratory. He holds eight issued patents. His research interests include wireless networks information theory, wireless networking systems, networks data compression, and networks algorithms; more information can be found at http://licos.ee.ucla.edu. He has received several recognitions for his research, including the 2013 IEEE Information Theory Society and Communications Society Joint Paper Award, the 2013 ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc) Best Paper Award, and the 2006 IEEE Donald Fink Prize Paper Award. He has been serving on the program committees of several IEEE conferences. He has also helped to organize IEEE and ACM conferences, including serving as the Technical Program Co-Chair for the 2012 IEEE Information Theory Workshop (ITW), the Technical Program Co-Chair for the 2015 IEEE International Symposium on Information Theory (ISIT), and the General Co-Chair for Mobihoc 2018. He is currently a Distinguished Lecturer and also serves on the board of governors for the IEEE Information Theory Society. He has been an Associate Editor of the IEEE TRANSACTIONS ON INFORMATION THEORY, the ACM/IEEE TRANSACTIONS ON NETWORKING, and the IEEE COMMUNICATION LETTERS and a Guest Editor of the IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING.

Puneet Gupta received the B.Tech. degree in electrical engineering from IIT Delhi in 2000 and the Ph.D. degree from the University of California, San Diego, in 2007. He co-founded Blaze DFM Inc. (acquired by Tela Inc.) in 2004 and served as its Product Architect until 2007. He is currently a Faculty Member and a Vice-Chair of the Electrical and Computer Engineering Department, University of California, Los Angeles (UCLA). He also leads the C-DEN Center, which focuses on future semiconductor technologies. He has authored over 100 papers, a book, and a book chapter. He holds 16 U.S. patents. His research interests include building high-value bridges across application-architecture-implementation-fabrication interfaces for lowered cost and power and increased yield and improved predictability of integrated circuits and systems. He was a recipient of the NSF CAREER Award, the ACM/SIGDA Outstanding New Faculty Award, the SRC Inventor Recognition Award, and the IBM Faculty Award.