Reliable and modeling attack resistant authentication of Arbiter PUF in FPGA implementation with trinary quadruple response

Zalivaka, Siarhei S.; Ivaniuk, Alexander A.; Chang, Chip Hong

2018

Zalivaka, S. S., Ivaniuk, A. A., & Chang, C. H. (2019). Reliable and modeling attack resistant authentication of Arbiter PUF in FPGA implementation with trinary quadruple response. IEEE Transactions on Information Forensics and Security, 14(4), 1109-1123. doi:10.1109/tifs.2018.2870835

https://hdl.handle.net/10356/143997

https://doi.org/10.1109/TIFS.2018.2870835

Reliable and Modeling Attack Resistant Authentication of Arbiter PUF in FPGA Implementation with Trinary Quadruple Response

Siarhei S. Zalivaka, *Student Member, IEEE,* Alexander A. Ivaniuk, *Member, IEEE,* and Chip-Hong Chang, *Fellow, IEEE*

***Abstract*—Field programmable gate array (FPGA) is a potential hotbed for malicious and counterfeit hardware infiltration. Arbiter based physical unclonable function (A-PUF) has been widely regarded as a suitable lightweight security primitive for FPGA bitstream encryption and device authentication. Unfortunately, metastability of flip-flop gives rise to poor A-PUF reliability in FPGA implementation. Its linear additive path delays are also vulnerable to modeling attacks. Most reliability enhancement techniques tend to increase the response predictability and ease machine learning attacks. This paper presents a robust device authentication method based on the FPGA implementation of a reliability enhanced A-PUF with trinary digit (trit) quadruple responses. A two flip-flop arbiter is used to produce a trit for metastability detection. By considering the ordered responses to all four combinations of first and last challenge bits, each quadruple response can be compressed into a quadbit that represents one of the five classes of trit quadruple response with greater reproducibility. This challenge-response quadruple classification not only greatly reduces the burden of error correction at the device but also enables a precise A-PUF model to be built at the server without having to store the complete challenge-response pair (CRP) set for authentication. Besides, the real challenge to the A-PUF is generated internally by a lossy, nonlinear and irreversible maximum length signature generator at both the server and device sides to prevent the naked CRP from being machine learned by the attacker. The A-PUF with short repetition code of length five has been tested to achieve a reliability of 1.0 over the full operating temperature range of the target FPGA board with lower hardware resource utilization than other modeling attack resilient strong PUFs. The proposed authentication protocol has also been experimentally evaluated to be practically secure against various machine learning attacks including evolutionary strategy covariance matrix adaptation.**

***Index Terms*—Arbiter PUF, reliability enhancement, machine learning attack resistance, authentication protocol.**

I. Introduction

According to Radiant Insights [1], the annual revenue of field programmable gate array (FPGA) market in 2014 was USD 3.92 billions and the revenue in 2022 is expected to double to USD 7.23 billions. One main reason for the rapid growth of FPGA market is the increasing fabrication cost and complexity of application specific integrated circuits (ASICs) escalated by the advanced manufacturing process technology. The chip design cost in 28 nm process node is estimated to be around USD 170 millions, which is twice as much as in 45 nm node [2]. On the contrary, the barrier for entrance into the FPGA market is low and many companies can afford to design their products with advanced process technology nodes without having to invest heavily into the reticle cost, yield lose and physical design.

In recent years, the integration of processor and FPGA architecture into a single device System-on-Chip (SoC) FPGA has further narrowed the performance gap between FPGA and ASIC. The same common practice of reusing third party Intellectual Property (IP) cores can also be adopted to improve its design productivity. Reconfigurability and field update are unique and competitive advantages of FPGA, but the communication of programmable bitstream from off-chip source also openly exposes the functional definition of the IP cores and increases the risks of loading and executing unauthorized or maliciously tampered IPs. Anti-counterfeiting methods, such as self-destruction [3], obfuscation [4], periodic licensing [5], bitstream encryption [6], IP watermarking [7] and fingerprinting [8], and hardware metering [9] have been proposed. Some of these countermeasures can be adapted to combat other security threats faced by FPGA SoC. Although modern FPGAs are able to process encrypted bitstreams, the device needs to be authenticated to avoid the installation of unauthorized bitstreams in authorized devices or authorized bitstreams in unauthorized devices. The binding of IP cores to target devices mandates the use of a unique device key or identifier for bitstream encryption and device authentication. Keeping the device key in non-volatile memory (NVM) is, however, less secure than generating it on-demand by embodying a physical disorder system into the FPGA fabrics.

Physical unclonable function (PUF) provides an efficient means to generate chip-dependent signatures. According to Tuyl's definition [10], a PUF is a characteristic of a physical (digital) system which cannot be cloned (copied or reproduced) using another physical system. A PUF can also be described mathematically as a mapping R = PUF(C), where C is an external stimulus (challenge) applied to the PUF and R is an output (response) generated by the PUF. A number of PUFs have been realized in ASIC and specifically in FPGA, including memory-based PUFs (e.g., SRAM-PUF [11], Set-Reset (SR)-latch PUF [12]), delay-based PUFs (e.g. Arbiter PUF (A-PUF) [13], digital PUF [14]), operational frequency based PUFs (ring oscillator PUF (RO-PUF) [15]), etc. Two major FPGA manufacturers Xilinx [16] and Intel (former Altera) [17] have recently announced PUF implementations in their mass product design for security purpose.

One major issue of silicon PUF is the sensitivity of its intrinsic parametric variations to changes in environmental

S. S. Zalivaka and C. H. Chang, School of Electrical and Electronic Engineering, Nanyang Technological University, 639798, Singapore, e-mail: zali0001@e.ntu.edu.sg, echchang@ntu.edu.sg.

A. A. Ivaniuk, Faculty of Computer Systems and Networks, Belarusian State University of Informatics and Radioelectronics, Minsk, 220013, Belarus, e-mail: ivaniuk@bsuir.by.

condition, especially temperature change, which is more difficult to control. This has led to the poor reproducibility of the enrolled responses during authentication. This problem is critical in security-sensitive applications that demand a highly reliable and robust key generation mechanism. Unfortunately, increasing the reliability of raw PUF response may reduce its uniqueness or randomness, and filtering out unreliable responses reduces the challenge-response space, which may be exploited to accelerate modeling attack [18], leading to compromised security of PUF based authentication scheme.

In general, A-PUF is more popular for FPGA implementation as it requires less hardware resources than RO-PUF [19] and has bigger challenge-response space than memory-based PUFs. Being a strong PUF [11], it does not require additional post-processing to prevent replay attacks on authentication protocol. However, FPGA implementation of classical A-PUF is known to have poor reliability [20] due primarily to the metastability of Delay Flip-Flop (DFF) arbiter and routing constraints of FPGA [13]. Error Correction Codes (ECC) is typically used to improve the reliability [21], with additional hardware overhead for ECC algorithm implementation and NVM for helper data storage. Maximum likelihood decoding scheme [22] is able to achieve the same reliability as ECC with lower hardware overhead, but it has the drawback of requiring more than one PUF instances for key generation. Since arbiter metastability is the main culprit of response instability, metastability detection technique utilizing either 4-DFF or SR-latch based arbiter has been proposed recently to significantly improve the average and minimum reliabilities [23]. However, this technique may lead to unbalanced response digits as well as detection and recoding overheads of metastable states. More arbiters can also be added to decrease the number of utilizable challenges and increase the reliability of the response at the expenses of hardware overhead [24]. Besides, the outputs of multiple arbiters are correlated, which may compromise the response unpredictability. Last but not least, hybrid PUF [25] combines the advantages of different kinds of PUF but this approach suffers from similar problem of high hardware resource requirement.

In this paper, a lightweight, reliable and robust A-PUF based device authentication method is proposed. The A-PUF is implemented in FPGA with enhanced reliability, i.e., its response to any challenge can be determined with 100% confidence over a temperature range from $-40°$C to $90°$. This is achieved by the following new contributions. First, metastability is detected by the binary encoded trinary digit (trit) of a SR-latch arbiter proposed by us in [23]. Secondly, the trinary quadruple response generated by different combinations of two end bits of a challenge is classified into one of the five different classes. Each response class can be uniquely represented by a quadbit that has better reproducibility than a single bit or trit response. Thirdly, the number of iterations applied to a Multiple Input Shift Register (MISR) circuit with reconfigurable seed and feedback polynomial coefficients is used for challenge obfuscation to thwart machine learning attacks. Last but not least, instead of storing the huge challenge-response pair (CRP) set in the server database, a precise software model of the A-PUF can be built based on the quadbit responses measured from the physical A-PUF implementation. The model can be utilized by the server to reproduce the entire challenge-response quadruple space without having to send the real challenges but uncorrelated random integers for MISR to regenerate these challenges on the device, and no helper data nor fuzzy extractor is required to recover the response at the server. The followings are new contributions extended from our preliminary works [23], [26], [27]: (1) A 100% accurate A-PUF model implemented with a three-stage classifier based on deep neural network. (2) The use of a trained A-PUF model at the server to efficiently reproduce all possible CRPs of the actual device. (3) New hardware architecture for the authenticable device with mirror software modules at the server to achieve highly reliable authentication with simpler error correction. (4) More rigorous security analysis, evaluation and comparison with competitive strong PUF-based authentication schemes.

This paper is organized as follows. Related works on modeling attack resistant A-PUFs are reviewed in Section II. Reliability enhancement technique based on metastability detection by SR-latch arbiter and trinary quadruple CRP classification is presented in Section III. Machine learning algorithm for precise A-PUF model building is described in Section IV. Device authentication using the A-PUF model at the server side and the FPGA based A-PUF at the device side is presented in Section V. Possible attack scenarios are analysed and discussed in Section VI. Quality and FPGA implementation cost of the proposed A-PUF are evaluated and compared with other obfuscated strong PUFs in Section VII. Section VIII concludes the paper.

II. Related works

There are two main ways to address the machine learning vulnerability of a strong PUF. The first approach involves modifications to the internal structure of a PUF to increase the complexity of each basic building block that is replicated to complete the mapping of an input challenge to an output response. One such example is the Double Arbiter PUF (DA-PUF) [28], which is designed to increase the number of possible symmetric routes. Although this PUF decreases the prediction accuracy of support vector machine (SVM) from 86.3 to 50%, its reliability also degrades from 0.95 to 0.89. Alternatively, the linearity of cascaded blocks can be broken by using non-linear tables to make the transfer function block nonlinear [18]. The lookup table can be designed using SRAM cells, which are similar to the SRAM PUF used to avoid the bias of zeros or ones. It was claimed in [18] that 20K of SRAM cells are sufficient to reduce the predictability of Bagging, SVM, Logistic Regression (LR) and Gradient Boosting (GB) to 70%. The obvious disadvantage of the latter approach is the unacceptably high hardware overhead, which is around 5000 times higher than the original A-PUF design. The non-linearity of the A-PUF circuit can also be increased by feed-forward loops [29]. All the modifications of this method have demonstrated an increase in randomness by 10 to 20% with a decrease in uniqueness and reliability by 5 to 10%. Thus, increasing the randomness by partially collapsing the linear addition effectively increases the resistance against machine learning attacks at a lower hardware cost. Owing to the trade-off between reliability and randomness, additional cost, which is dependent on the type of PUF and reliability enhancement technique used, is needed to restore the lost reliability. The transfer function for the CRP mapping of the PUF can be designed based on the inherently non-linear physical property of analog circuits such as the non-linear Voltage Transfer Characteristic PUF [30], nonlinear current PUF [31], Subthreshold Current Array PUF [32]), etc., but these analog voltage and current functions are difficult to be implemented efficiently on FPGA. As evaluated in [18], these analog transfer function based strong PUFs offer good resistance only against attacks by basic machine learning algorithms such as SVM or LR, but not to advanced attacks using Bagging, Boosting and Evolutionary methods.

The second approach to fortify the PUF against machine learning attacks is to reduce the leakage of challenge-response mapping by controlled strong PUF design [33]. The main idea of this technique is to randomize the challenge and/or response values with a hash function. This scheme was initially applied to weak PUFs to protect their challenge-response interface from being directly accessible by the attacker. Such protection was deemed unnecessary for strong PUFs due to the intractability to exhaustively measure and collect the exponentially huge CRPs, making it relatively safe to leave the CRP interface “as is” for lightweight authentication. As A-PUF has later been successfully modeled with accuracy of 95 to 98% by various machine learning methods with training set size ranges from a few hundred [34] to a few thousand CRPs [35], controlled implementation of challenge-response interface has been extended to strong PUF. However, secure hash function in hardware implementation consumes much more hardware resources than the PUF itself. The first and probably the most well-known implementation of controlled strong A-PUF was due to XOR-PUF [15]. A recent study [36] has shown that depending on the training set size, the prediction accuracy of A-PUF by LR has been reduced to 60-65% by n-input XOR ($n \geq 11$). This phenomenon can be explained by the exponentially decreasing reliability of A-PUF circuit with the number of XOR inputs n. If n = 10, the reliability becomes smaller than 0.1. Albeit resilient against basic A-PUF modeling attack, XOR-PUF has also found to be inadequate against modified mathematical model [35] and more advanced algorithms [18].

Challenge obfuscation can also be used to increase the modeling attack resistance of A-PUF. A pattern based challenge processing algorithm is proposed in [37]. The main idea is to apply part of the challenge to multiple A-PUF instances and choose a response based on a random number. This method is appealing as it does not affect A-PUF reliability. A robust and reverse-engineering resistant approach is made based on the comparison of the substrings of multiple responses [38]. The prover generates a random substring of an original response word and pads it with additional random bits to create a bit string of the required length. Then the verifier matches the transmitted padded string with the response obtained from a PUF model. Since the accuracy of the PUF model is very high (at least 90%), the verifier is able to find the correct substring based on their Hamming distance. This solution was tested to be secure against a number of different attacks, including modeling, replay, seed compromise, etc. However, this protocol can still be attacked by Covariance Matrix Adaptation Evolution Strategy (CMA-ES) with an accuracy of 96.9% using a training set of 4×10^5 CRPs and a total training time of around 30 hours.

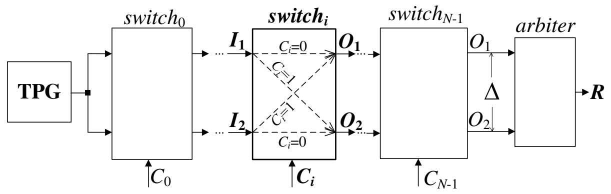


Fig. 1. Structure of an N-stage A-PUF.

In short, controlled strong PUF designs are by far a more practical approach to raise the barrier of modeling attack. It does not directly affect PUF reliability when the obfuscation is applied to the challenge and does not degrade uniqueness and randomness more than other methods, except that they are still attackable by advanced machine learning algorithms like CMA-ES and require non-trivial extra hardware resources. In this paper, we propose a lightweight A-PUF based device authentication method that is practically secure against CMA-ES, Bagging and Boosting techniques.

III. Classification of quadruple CRPs for reliability enhancement

Typically, an A-PUF is implemented with a test pulse generator (TPG), N connected path-swapping switches ($switch_i$, i = 0, 1, . . ., $N-1$) and an arbiter circuit ($arbiter$). The basic A-PUF structure is depicted in Fig. 1. Two symmetrical paths are selected by an N-bit input challenge C for the propagation of the test pulse. Each switch ($switch_i$) can operate in two modes, straight and cross. Fig. 1 shows both configurations for $switch_i$: I_1 to O_1 and I_2 to O_2 for the straight mode when the i-th bit C_i of C is 0, and I_1 to O_2 and I_2 to O_1 for the cross mode when C_i = 1. This operation can be implemented using a pair of multiplexers [39] or tri-state buffers [40]. An arbiter circuit compares the arrival time of the signal in these two paths at the last stage to generate a respose bit. The arbiter is classically implemented by a DFF, which outputs 0 when the bottom signal (O_2) is faster than the top signal (O_1) output from $switch_{N-1}$. One of the main flaws of this implementation is the low reliability due to the metastability of DFF. This issue is mitigated by using SR latch as the arbiter.

As a side effect, SR latch can be used to detect metastable states and these states can be assigned a unique trinary response digit X (i.e., $R \in \{0, 1, X\}$) [23]. If the delay difference between the two input signals is too small [41], a damped oscillation is produced at the output of SR latch when it transits from the forbidden state (S = 1, R = 1) to the hold state (S = 0, R = 0). This damped oscillation can be detected and characterized using a multibit synchronous counter. Fig. 2 shows a two-bit counter used for metastability detection, where the race of the two paths to the SR latch arbiter is triggered by a falling instead of rising edge. This circuit has three possible

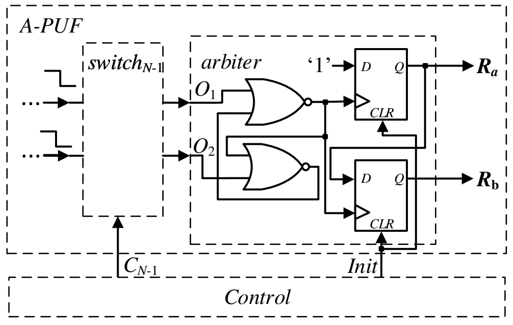


Fig. 2. Arbiter circuit for the detection of metastable response.

outputs, $\{R_a = 0, R_b = 0\}$ and $\{R_a = 1, R_b = 0\}$ represent stable zero and one states, respectively, and $\{R_a = 1, R_b = 1\}$ represents a metastable state. Different extent of metastability can be further categorized by using larger counter to estimate their rates of decay of damped oscillation.

The proposed metastability detector with trinary response significantly enhances the raw reliability of A-PUF circuit implemented in FPGA to 0.9985 at the nominal working condition. Additional enhancement is needed to make a more robust A-PUF with almost perfect reliability of 1.0 in order to build a precise ($\simeq$100% accurate) A-PUF model for authentication. According to [42], the probablility of response bit flip of an A-PUF is low ($\approx$0.05) by changing only the least significant bit (LSB) of a challenge. On the other hand, there is a high probability of flipping the response bit ($\approx$0.95) by inverting only the most significant bit (MSB) of the challenge. This observation can be explained with the well-known linear additive model of A-PUF [43]:

$$\Delta = W^T \Phi, \tag{1}$$

where Δ is the delay difference between the top and bottom paths at the input of the arbiter, and W and Φ are $N+1$ dimensional weight and sign vectors, respectively. The i-th element of W can be characterized by the propagation delays of four different paths (top-to-top, top-to-bottom, bottom-to-top and bottom-to-bottom) through the i-th switch independent of the challenge, whereas the i-element (except the last element, $\Phi_N = 1$) of Φ depends only on the parity of the i-th to $(N-1)$-th challenge bits. $\Phi_i = -1$ for odd parity and 1 for even parity.

If the LSB C_0 of a challenge C has changed, then the value of Δ will be changed to:

$$\Delta_{LSB} = \sum_{i=1}^{N} W_i \Phi_i - W_0 \Phi_0 = \Delta - 2W_0\Phi_0. \tag{2}$$

Similarly, flipping the MSB C_{N-1} changes Δ to

$$\Delta_{MSB} = \sum_{i=0}^{N} -W_i \Phi_i = -\Delta. \tag{3}$$

The latter challenge modification has a more significant impact on the response, as evident by:

$$|\Delta_{LSB} - \Delta| = 2W_0\Phi_0 \ll |\Delta_{MSB} - \Delta| = 2\Delta. \tag{4}$$

Consider an arbitrary challenge C^{Ω}, where Ω is the decimal representation of the N-bit challenge C and three other challenges, $C^{\Omega'}$, $C^{\Omega''}$ and $C^{\Omega'''}$, obtained by inverting only the LSB, only the MSB, and both the LSB and the MSB of Ω, respectively, i.e., $|\Omega - \Omega'| = 1$, $|\Omega - \Omega''| = 2^{N-1}$, $|\Omega''' - \Omega''| = 1$ and $|\Omega''' - \Omega'| = 2^{N-1}$. For each challenge C^{Ω}, a quadruple of responses $\{R_0, R_1, R_2, R_3\}$ is defined, where $R_0 = PUF(C^{\Omega})$, $R_1 = PUF(C^{\Omega'})$, $R_2 = PUF(C^{\Omega''})$ and $R_3 = PUF(C^{\Omega'''})$.

Equation (4) is valid at any voltage or temperature. Environmental variation affects W but not Φ of (1). Since W_0 depends only on the difference between the delay difference of the two parallel paths and the delay difference of the two swapped paths of the first switch, and ϕ_0 is independent of environmental variation, the compound probability of the independent events $R_0 \neq R_1$ and $R_2 \neq R_3$ is extremely low even if the operating condition has changed. On the other hand, Δ in (4) is dependent on the cumulative sum and difference of all elements of W, the probability of $R_0 \neq R_2$ escalates as the operating condition varies. By unifying the four challenges, C^{Ω}, $C^{\Omega'}$, $C^{\Omega''}$ and $C^{\Omega'''}$, as one single challenge with a quadruple response, the effective number of stages of an A-PUF is reduced by two but the quadruple response to a unified challenge can be classified by flipping either its LSB, MSB or both these bits simultaneously.

Based on the vanishingly low compound event probability of $R_0 \neq R_1$, $R_2 \neq R_3$ and $R_0 = R_2$, the quadruple response to a challenge C^{Ω} is classified as stable if it meets the following requirement:

$$R_0 = R_1 = \overline{R_2} = \overline{R_3}. \tag{5}$$

From (5), it is evident that all challenges that can produce one of the two possible types of quadruple response, i.e., $\{R_0, R_1, R_2, R_3\}$ = {0, 0, 1, 1} or {1, 1, 0, 0}, are desirable. All other challenges that have the quadruple $\{R_0, R_1, R_2, R_3\}$ different from {0, 0, 1, 1} and {1, 1, 0, 0} are undesirable from reliability perspective. Despite the quadruples, {0, 0, X, X}, {1, 1, X, X}, {X, X, 0, 0} and {X, X, 1, 1}, also agree with (5), strictly speaking, they are not considered "stable" since X is not a stable state but detectable as a metastable state using our proposed arbiter implementation. Thus, the quadruple response to each of the 2^{N-2} unique challenges can be classified accordingly.

While machine learning can be used to attack a PUF, it can also be used in a positive way to build a precise model for secure authentication of a highly reliable strong PUF. In the experimental results shown in Fig. 3, six major classes of quadruple responses were generated from the 24-bit A-PUF implemented in Xilinx Artix-7 FPGA. This set contains 2^{22} unique quadruple challenges, which covers the complete CRP set of 24-bit A-PUF ($4 \times 2^{22} = 2^{24}$) quadruple responses. The distribution was obtained by repeating the experiment 100 times. All classes shown in Fig. 3 are stable except the category labelled "others", which contains the following

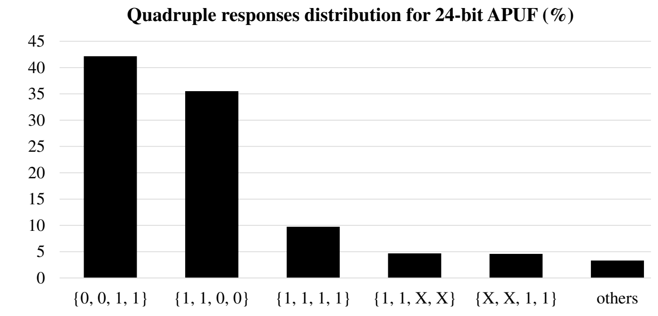


Fig. 3. Distribution of quadruple responses of a 24-bit A-PUF.

quadruples, {1, 1, 1, 0}, {1, 1, 0, 1}, {1, 0, 1, 1}, {0, 1, 1, 1}, {1, 1, 1, X}, {1, 1, X, 1}, {1, X, 1, 1}, and {X, 1, 1, 1}. Therefore, the complete CRP set can be merged into three main classes of "zero" ({1, 1, 0, 0} and {1, 1, X, X}), "one" ({0, 0, 1, 1} and {X, X, 1, 1}) and "unstable state" ({1, 1, 1, 1} and "others"). Every quadruple from the "others" category can be easily recoded into {1, 1, 1, 1}. These results agree with the experiments conducted on a 128-stage A-PUF implemented in the same FPGA chip as the response distribution is the same in terms of the percentage of complete CPR set (of 10^{10} quadruples) generated.

IV. Accurate A-PUF Model Based on Machine Learning Algorithm

From Fig. 3, the quadruple responses of an A-PUF can be divided into six different classes. In contrast to the existing models with accuracy of 95-98%, a more precise ($\simeq$100% accuracy) model can be built based on the stable quadruple responses defined in Section III and experimentally validated over allowable range of operating voltage and temperature variations specified by the FPGA device or board used for the A-PUF implementation. An accurate A-PUF model is built by using a tripartite classification algorithm to segregate the CRPs into five different classes, namely unstable, metastable zero, metastable one, stable zero and stable one. As illustrate in Fig. 4, the classes of response before processing in each stage are enclosed in rectangular blocks and the final classified responses are enclosed in circular blocks.

The first stage of classification is the most complicated one as the classifier has to differentiate between stable and metastable quadruple responses from the unstable quadruple responses. Therefore, a fully connected Deep Neural Networks (DNN) is used for the first classifier. The proposed DNN architecture is shown in Fig. 5. The input-layer contains an N-bit parity vector, which determines the sign of each delay element for Φ. These inputs are further propagated using 20 layers (L_1, …, L_{20}) of 2048 Rectified Linear Units (ReLU) followed by a softmax layer for binary classification, 0 for unstable quadruple response ({1, 1, 1, 1} and responses from the class "others") and 1 for stable and metastable quadruple responses ({0, 0, 1, 1}, {1, 1, 0, 0}, {X, X, 1, 1} and {1, 1, X, X})). There is no necessity to further segregate the responses {1, 1, 1, 1} from those under "others" as these unreliable responses can be dumped together with {1, 1, 1, 1} as one unreliable class to separate it from the other four response classes without loss of accuracy.

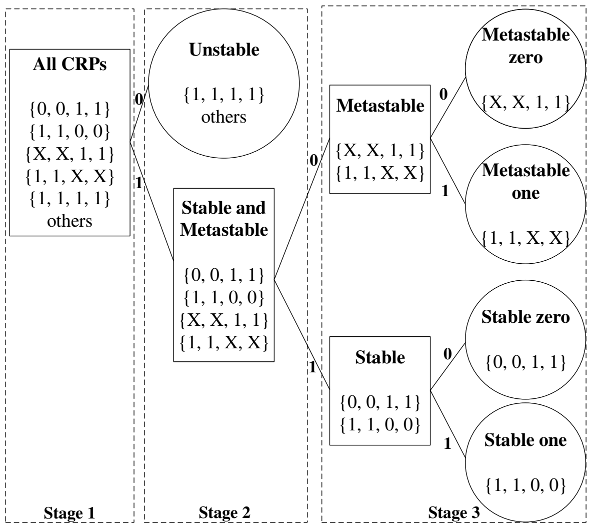


Fig. 4. Flow of response classification.

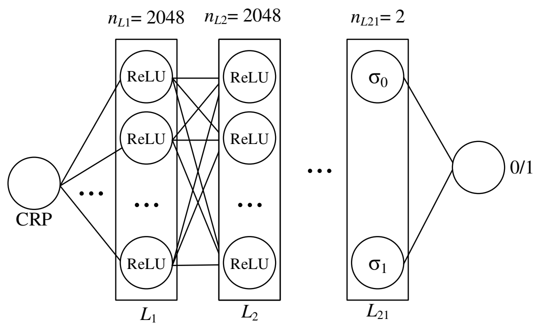


Fig. 5. Deep neural network architecture in Stage 1 of response classification.

The DNN is trained to model two different path lengths of A-PUF, N = 24 for short A-PUF and 128 for standard A-PUF. The quadruple CRP set generated is divided into three datasets, namely training (80% of the data), validation (or development – 10%) and test (10%). Each quadruple challenge is repeated 100 times on the physical PUF device to determine its response stability before feeding it into the DNN for training. If the response quadruple has the same value each time, it is labeled as stable and belongs to one of the followings, {0, 0, 1, 1}, {1, 1, 0, 0}, {X, X, 1, 1} or {1, 1, X, X}. Otherwise, it is labeled as "others" in the training set. The trained models are tested with temperature and voltage variations to determine its stability. To avoid overfitting, two techniques have been applied, L2-regularization and dropout (p = 0.5). The CRP sets containing 2^{22} and 10^{10} quadruples have been generated to build the models for N = 24 and 128, respectively. An accuracy score of 100% has been achieved consistently on training, validation and test datasets for both 24-stage and 128-stage A-PUF implementations.

The classifier in the second stage does not require to be as powerful because separating the stable quadruple responses from the metastable quadruple responses is a much simpler task. Therefore, the neural network in the previous stage is converted to a shallow version with the number of layers reduced to three including the softmax. This simplified classifier is reused and tested to be capable of providing 100% accuracy in classifying the quadruples as stable ({0, 0, 1, 1} and {1, 1,

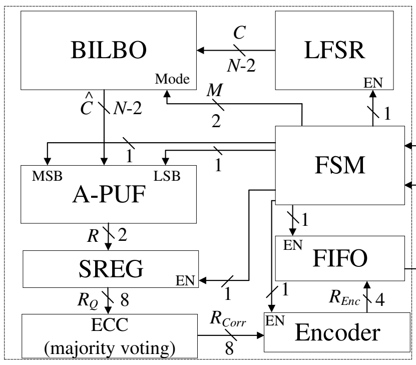


Fig. 6. Main components of authenticable device.

0, 0}) and metastable ({X, X, 1, 1} and {1, 1, X, X}).

The last stage utilizes the same model with different parameters to separate the one from zero for both stable and metastable classes of quadruples (see the classifiers "Metastable" and "Stable" in Fig. 4). The classifier in this level can be implemented by any linear classification algorithm such as LR or SVM. Since majority of the quadruples are stable (86.94%), it is reasonable to use a simple model for this classification.

The constructed precise A-PUF model is stored at the authentication server. Since all responses can be reliably generated internally by the server A-PUF model and the physical device, no helper data or "raw" CRP needs to be stored or transmitted in clear or accessible externally. Also, no quadruple CRP needs to be excluded or forfeited from being used by the protocol as every one of them can be reproduced on both the device and the server sides.

V. Proposed Authentication Protocol

The proposed protocol involves exchange of messages among a trusted party, an authentication server and an A-PUF embedded in the device to be authenticated.

A. *Auhtenticable device*

The proposed PUF is implemented based on an N-stage A-PUF block and a Built-In Logic Observer (BILBO) [44]. Its block diagram is shown in Fig. 6. The BILBO block is used primarily in Built-In Self-Test (BIST) applications. It can operate in four modes, which are determined by the value of M. When M = "00", it is used as a shift register in scan test; when M = "01", it is used as a Linear Feedback Shift Register (LFSR) for test pattern generation; when M = "10", it is used as a memory register; when M = "11", it is used as a Multiple Input Signature Register (MISR) for compact test signature generation. These functions can be used advantageously to aid the segregation and obfuscation of CRPs. BILBO in memory mode is used to feed the input challenges directly to the A-PUF for building the A-PUF model. MISR mode is used to obfuscate the true challenge, hence increases the complexity of machine learning attack. The functions of all components are summarized in Table I.

As shown in Fig. 7, the output of a MISR is determined by the initial value ($seed$) loaded into the BILBO in memory mode and a GF(2) feedback polynomial $\phi(X) = \oplus_{i=0}^{N}\alpha_i x^i$, where $\alpha_i \in \{0,\ 1\}$ are the polynomial coefficients, N is the degree of the characteristic polynomial $\phi(X)$ and X is an N-bit input vector. The coefficients α_i are fixed at the design phase but they can be reconfigured to generate different signatures. Changing α_i corresponds to applying a different non-linear transformation to the same challenge, which can greatly enhance its resistance against modeling attack. MISR converts an input challenge (C) to a new challenge ($\hat{C}$) based on its current state, making the actual challenge applied to the A-PUF dependent on the non-linear mixing of all the previously input and converted challenges. The actual challenge $\hat{C}(k)$ applied to the PUF instance can be expressed as:

$$
\begin{bmatrix} \hat{C}_0(k) \\ \hat{C}_1(k) \\ \hat{C}_2(k) \\ \vdots \\ \hat{C}_{N-1}(k) \end{bmatrix}
=
\begin{bmatrix} 0 & 0 & 0 & \dots & \alpha_0 \\ 0 & 1 & 0 & \dots & \alpha_1 \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & \dots & 1 & \alpha_{N-1} \end{bmatrix} \times
\times \begin{bmatrix} \hat{C}_0(k-1) \\ \hat{C}_1(k-1) \\ \hat{C}_2(k-1) \\ \vdots \\ \hat{C}_{N-1}(k-1) \end{bmatrix}
\oplus
\begin{bmatrix} C_0(k-1) \\ C_1(k-1) \\ C_2(k-1) \\ \vdots \\ C_{N-1}(k-1) \end{bmatrix},
\tag{6}
$$

where $C_i(k-1)$ is the i-th bit of the current input challenge and $\hat{C}_i(k-1)$ is the i-th bit of the current MISR state.

The $N-2$ bits of C (which are the effective middle bits of the quadruple challenge) input into the BILBO block are generated by a separate LFSR block. This $(N-2)$-bit word is fed into the BILBO circuit when M = "11" is sent from the Finite State Machine (FSM) to the BILBO block to set it into MISR mode. Each $(N-2)$-bit obfuscated challenge $\hat{C}$ generated by the BILBO is padded with four possible combinations of MSB and LSB provided by the FSM to form a quadruple challenge to the A-PUF block. The A-PUF outputs

TABLE I
Components of authenticable device.

Component	Description
LFSR	Generates $(N-2)$-bit challenge with low correlated M-sequence.
BILBO	As memory register (M = "10") during enrollment to feed C from LFSR directly to A-PUF module. As MISR in authentication mode to obfuscate (lossy data encoding) C into $(N-2)$-bit $\hat{C}$.
A-PUF	Arbiter PUF with SR-latch based arbiter, which takes N bits of MSB + $\hat{C}$ + LSB as input and outputs a 2-bit response R for three states, "00" (logical zero), "10" (logical one), "11" (metastable state).
SREG	Collect four responses from A-PUF to form an 8-bit quadruple response R_Q.
ECC (majority voting)	Stabilize quadruple response R_Q to R_{Corr}.
Encoder	Compress R_{Corr} by extracting the first four bits into R_{Enc} as the remaining bits of R_{Corr} can be decoded from R_{Enc}.
FIFO	Store R_{Enc} $\frac{K}{4}$ times to make up R_{Final}.
FSM	Control LFSR, BILBO, SREG, FIFO and Encoder blocks, and padding response $\hat{C}$ with MSB and LSB.

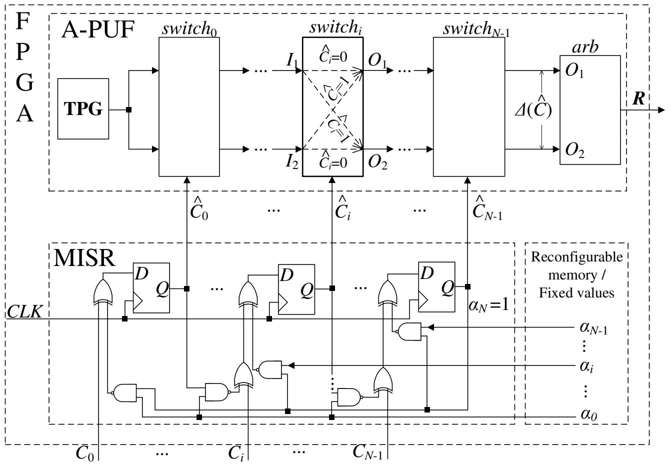


Fig. 7. MISR for obfuscation of input challenge.

a two-bit binary word R to encode the three trits ({0, 1, X}) as shown in Fig. 2. The quadruple response due to $\hat{C}$ is stored in an 8-bit shift-register SREG in Fig. 6. Although the reliability of the A-PUF has been significantly enhanced by considering quadruple response, there is still a very small probability of bit error. Since every CRP has been repeated 100 times to build an A-PUF model at the server side, it has been experimentally verified that repeating every quadruple challenge k = 5 times at the device side to vote for the trits of the quadruple response is sufficient to achieve a reliability of 1.0 over the specified operating temperature and voltage range. The k = 5 collected quadruple responses R_Q to the same $\hat{C}$ are fed to the ECC block that implements the majority voting. If the result of majority voting of R_Q is equal to {0, 0, 1, 1}, {1, 1, 0, 0}, {X, X, 1, 1}, {1, 1, X, X} or {1, 1, 1, 1}, the output R_{Corr} of ECC will be one of these quadruples according to the majority votes. Otherwise, R_{Corr} will be corrected to {1, 1, 1, 1} which falls into the unstable quadruple response class. The five unique quadruples of R_{Corr} is symmetrical, i.e., the second and third trits are the same as the first and fourth trits. This undesirable redundancy is removed by an Encoder block to compress R_{Corr} to only four binary bits that uniquely encode the two non-redundant trits for each quadruple response class. The quadbit R_{Enc} to each $\hat{C}$ is stored in a first-in-first-out (FIFO) buffer. The final K-bit response R_{Final} is produced by making the BILBO performing N_{Rounds} of iterations $\frac{K}{4}$ times to generate $\frac{K}{4}$ different obfuscated quadruple challenges to the A-PUF.

B. Authentication Server

Unlike the physical device, the server contains a software implementation of BILBO, LFSR, FSM and A-PUF model built using the algorithm described in Section IV. The blocks labeled LFSR and BILBO perform exactly the same functions as those described for the authenticable device in Fig. 6. ECC or majority voting is not needed as the A-PUF model has already been trained with CRP quadruples with known stability status. It is capable of accurately predicting the class of quadruple response based on the $N-2$ middle bits of a challenge. The FSM model in the server also generates a random 8-bit number N_{Rounds}, which is sent to the device to be authenticated. The response analyzer in the server compares the response R_{Final} received from the device, and the response R_{Model} generated by it's A-PUF model for the same N_{Rounds} in each authentication session to determine if the device is authentic. The descriptions of server's software modules are summarized in Table II.

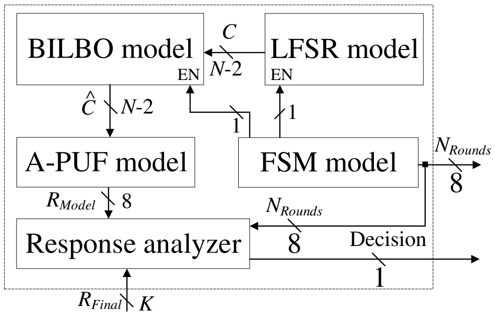


Fig. 8. Building blocks of server authentication model.

TABLE II
SOFTWARE COMPONENTS OF AUTHENTICATION SERVER BLOCK.

Component	Description
LFSR	Python script for generating pseudorandom $(N-2)$-bit challenge C using the same seed as the device to be authenticated.
BILBO	Python script for lossy compression of C to $\hat{C}$, using the same seed and polynomial coefficients as the device to be authenticated.
A-PUF	Deep neural network modeled in TensorFlow (Python) for generating an 8-bit R_{Model}. As a precise software model, SREG and ECC is not required.
FSM	Python script for generating an 8-bit N_{Rounds} for both authenticable device and authentication server.
Response analyzer	Python script to encode and assemble $\frac{K}{4}$ R_{Model}. It also compares R_{Final} received from the device with R_{Model} generated by the server.

C. Authentication Protocol

As shown in Fig. 9, the communication protocol consists of three phases, which are enrollment, authentication and update.

The first phase involves a trusted party, who provides the initialization data for both the device and the server. During this phase, the device (BILBO block) is set to memory mode (M = "10") to enable the FSM to feed any challenge C directly to the A-PUF block without obfuscation. In this mode, the FSM can feed an arbitrary quadruple challenge to the A-PUF to produce an 8-bit response R_Q. Each N-bit quadruple challenge C is repeated many times to gather a majority voted quadruple response R_Q. This way a large number of reproducible quadruple CRPs can be collected from the physical device. This will enable the machine learning algorithm presented in Section IV to create an accurate A-PUF model to precisely predict the encoded quadruple response (one of the five response classes) to any input quadruple challenge specified by their middle bits $\hat{C}$. The number of quadruple CRPs required to build an accurate A-PUF model varies from few millions for N = 24 to few billions for

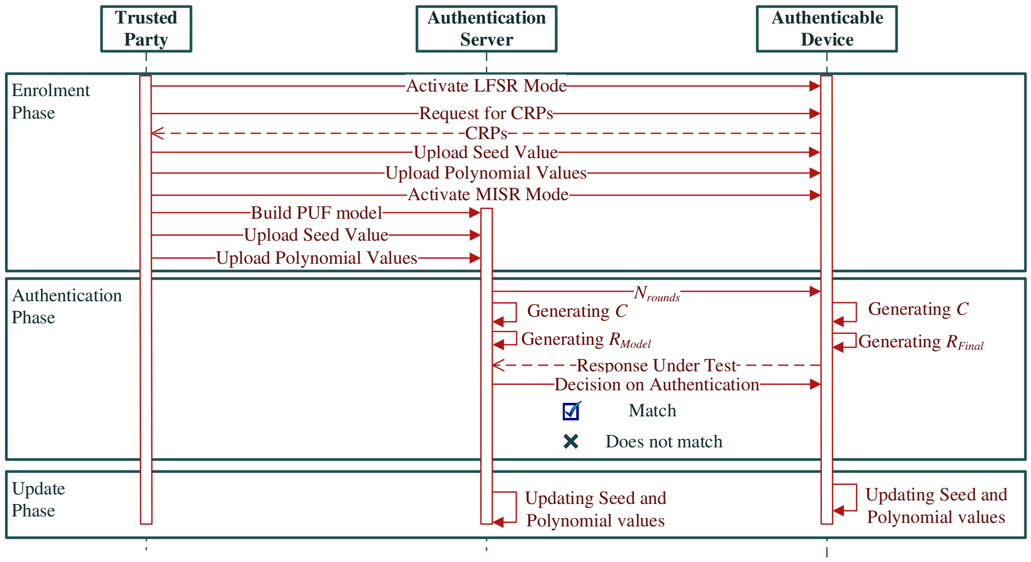


Fig. 9. Proposed authentication protocol.

N = 128. Therefore, this phase can take a significant amount of time (from hours to weeks). The built model is uploaded to the server.

The device and server share some common parameters of BILBO, which determine the actual challenges to the A-PUF and its final responses. These parameters are namely the $seed$ value or initial value of MISR circuit and polynomial coefficients, which are used to generate the real challenge to the physical A-PUF or its mathematical model at the server. The feedback polynomial and seed value of the LFSR block are different from those of the BILBO block and they are set once during the design phase. As shown in our preliminary work [27], the key parameters of MISR circuit can be fixed or stored in the reconfigurable memory for further updating. The last step of this phase is the MISR mode activation after all the parameters have been set and the A-PUF model has been created. The memory mode of MISR will be permanently disabled by the FSM after the enrollment phase to prevent the attacker from accessing the “raw” CRPs during other phases.

The authentication phase is the main mode of operation that is invoked most frequently. The server responds to the device’s request for service by sending a random number N_{Rounds} to initiate a request to authenticate the device before the service can be granted. Based on the value of N_{Rounds}, the device uses the LFSR block to generate the challenge C, stores it and feeds it to the BILBO in MISR mode. N_{Rounds} of iterations are executed in MISR mode to generate an obfuscated challenge $\hat{C}$ to the PUF to produce one quadruple response. During the authentication phase, $\frac{K}{4}$ of N_{Rounds} values are generated at the server and sent to the device. A K-bit R_{Model} is computed at the server to compare with the K-bit R_{Final} received from the device. R_{Model} is calculated in almost the same manner as that on the device except that the computations are performed in software by the A-PUF and BILBO models instead of physically by the hardware. The authenticity of the device is confirmed by $R_{Final} = R_{Model}$ and denied by $R_{Final} \neq R_{Model}$.

If the parameters of BILBO are stored in reconfigurable memory, then an update phase can also be invoked in the proposed protocol. The parameters can be updated to new values from the memory regularly or after certain number of successful authentications. The total number of available polynomials can be estimated using the Euler’s totient function. For N = 128, the number of GF(2) polynomials is $\approx 1.3\times 10^{36}$. The number of possible seed values is also exponentially large (2^N). Thus, these values can be randomly chosen and the number of different combinations will not be exhausted throughout the device life cycle for N bigger than 64. The choice of exact number of parameters to be stored in the memory depends on the application and on the trade-off between hardware overhead and system security.

VI. Security Analysis

A. Attacker Model

According to Kerckhoffs’ principle [45], attacker knows everything about our system except the keys. In our proposed authentication protocol, the following five secrets are shared between the authenticable device and authentication server: seed value of LFSR, polynomial values of LFSR, seed value of BILBO, polynomial values of BILBO, challenge-response map of A-PUF.

We assume that the attacker has access to all data transmitted between the device and the server. The attacker can exploit the collected data to produce correct responses to the N_{Rounds} initiated by the server to gain successful authentication. The attacker is assumed to have access to only the device external IO ports to directly apply authentication request or measure any side-channel signals by direct wired connection in attempt to recover the secret parameters of LFSR or BILBO. The incentive of the attack vanishes if the device is damaged or the tamper can be conspicuously detected before the attack is successful. The protocol is considered secure if the attacker is not able to predict the correct responses under known approaches notwithstanding the full access to the transmitted data and IO pins of authenticable device. We further assume

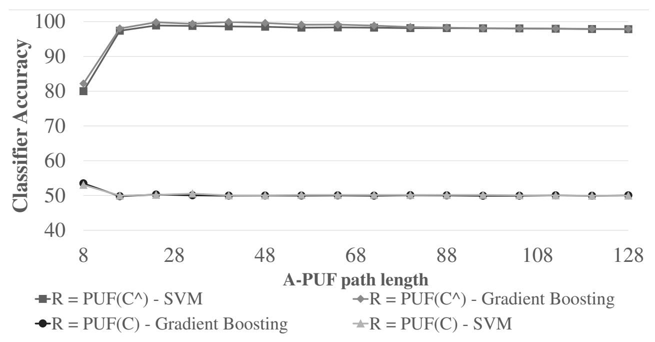


Fig. 10. Prediction accuracies of SVM and GB based modeling attacks trained with naked and obfuscated CRP data sets.

that the profitability of an attack diminishes rapidly if the attacker has to continuously deploy significant computing power to perform the attack and analysis beyond a realistic time span. Hence the protocol can be considered secure if the time required for a successful attack grows exponentially with the number of stages N of A-PUF.

Side-channel attacks require expensive equipment, and well-timed and accurate measurements. They rely on known exploitable implementation weaknesses of LFSR and BILBO, which can be easily circumvented. Since the ultimate goal of the attacker is to successfully predict the response to unknown and unused challenge of the embedded A-PUF, which can be more effectively achieved by modeling the propagation delays of the parallel and swapped paths of each switch of the A-PUF than side-channel attacks, machine learning attacks [43] are emphasized in our security analysis.

B. Modeling Attack

This attack requires the attacker to collect a reasonable subset of CRPs to build an accurate model of the PUF. One of the most efficient methods to perform this kind of attack is machine learning. Since an A-PUF can be approximated by a linear classifier (e.g., LR or SVM), the proposed MISR based challenge obfuscation creates a non-linear dependency of the response on the naked challenge.

To estimate the effectiveness of the proposed challenge transformation algorithm, a 128-stage A-PUF with SR latch based multi-arbiter [23] is implemented in Xilinx Artix-7 FPGA. This circuit has 16 arbiters uniformly allocated to sections of 8 multiplexer pairs to emulate different lengths of A-PUF path. Meantime, a MISR circuit with feedback polynomial $\phi(X) = x^{128} \oplus x^{127} \oplus x^{126} \oplus x^{121} \oplus 1$ is implemented on the same FPGA chip to transform an arbitrary 128-bit input challenge C to an obfuscated challenge $\hat{C}$. Both CRP (C, R) and $\widehat{CRP}$ ($\hat{C}$, $\hat{R}$) are split into 16 data sets according to the emulated A-PUF path length to generate a 16-bit response from each arbiter instance. A host computer is used to generate a uniformly distributed sequence of 128-bit challenges, i.e., $C \in [0,\ 2^{128} - 1]$, for input into the MISR of the FPGA chip. To test the resistance of the proposed MISR augmented A-PUF against modeling attacks, SVM with radial basis function (RBF) kernel and gradient boosting with 10,000 trees and learning rate of 0.1 are implemented on the host computer using Sklearn library of Python. 10% of the randomly chosen CRPs are used as the training set, which is cross-validated using 5 blocks K-fold method. The remaining 90% are used as the test set. Both algorithms are capable of discriminating two classes of responses using non-linear surface. The attack resistance is evaluated by an accuracy score in terms of the percentage of successful classifications.

The results are shown in Fig. 10. If the machine learning model is trained by the $\widehat{CRP}$ data set, which is inaccessible to the attacker, high accuracy scores of [97.38, 98.34]% for SVM and [97.88, 99.83]% for gradient boosting are obtained. The only exception is found in the 8-stage A-PUF emulation, where the scores are 80% for SVM and 82.17% for GB due to the limited number of CRPs (only 256). This is because A-PUF with less than 16 stages has very unstable behavior [23]. For path longer than 8 stages, the prediction accuracy reduces slightly from 98.89% to 97.83% for SVM and from 99.83% to 97.88% for GB and for 128 stages, the predication accuracy is almost the same for both attack algorithms. Since the attacker can only access to the CRP data set before the challenge C is transformed into $\hat{C}$, the prediction accuracy fluctuates around 50% for both SVM and GB, as shown in Fig. 10.

Recent research [46] shows that secure designs based on A-PUF [38] can be successfully attacked using powerful multi-core CPU and GPU servers even if the attacker does not know the system, i.e., by treating the PUF system as a black box. Therefore, we have simulated the CMA-ES attack using 24-core Intel Xeon CPU server with 32 GBytes of RAM on both 24 and 128-bit A-PUF circuits with MISR based challenge obfuscation algorithm. CMA-ES algorithm was implemented using PyBrain library. The results are shown in Fig. 11 (solid circle and triangular points are measured experimental results and the hollow points are extrapolated results). The extrapolation is justifiable as the longest physical experiment (the fourth solid triangle point in Fig. 11) conducted on 128-bit A-PUF took more than two months. It is made based on the four physically conducted experiments for each of the 24-bit and 128-bit A-PUFs. When the size of the training CRP set is increased by tenfold in each experiment, the accuracy of machine learning algorithm increases only marginally by 2.79% to 2.97% for the 24-bit A-PUF and 0.89% to 1.08% for the 128-bit A-PUF, but the training time increases exponentially by 2.07x, 5.17x to 10.76x for the 24-bit A-PUF and 1.43x, 5.38x and 10.96x for the 128-bit A-PUF. For larger training sets, we deliberately underestimate the training time and overestimate the training accuracy by conservatively assuming a maximum of 3% and 1% in our extrapolation of the increase in accuracy for 24- and 128-bit A-PUFs, respectively, as well as a lower than expected flat 10x growth rate in training time for every decade increase in the size of the training CRP set.

Although the classifier accuracy increases with training set size, this attack cannot succeed for two practical reasons. First, the number of CRPs required to build a reasonably accurate model is significantly greater than that required for authentication throughout the device lifetime. To achieve an accuracy of 60% for 128-stage A-PUF, the number of required CRPs exceeds 16 millions. If the training data is to be collected within a year, then 44943 CRPs have to be extracted per

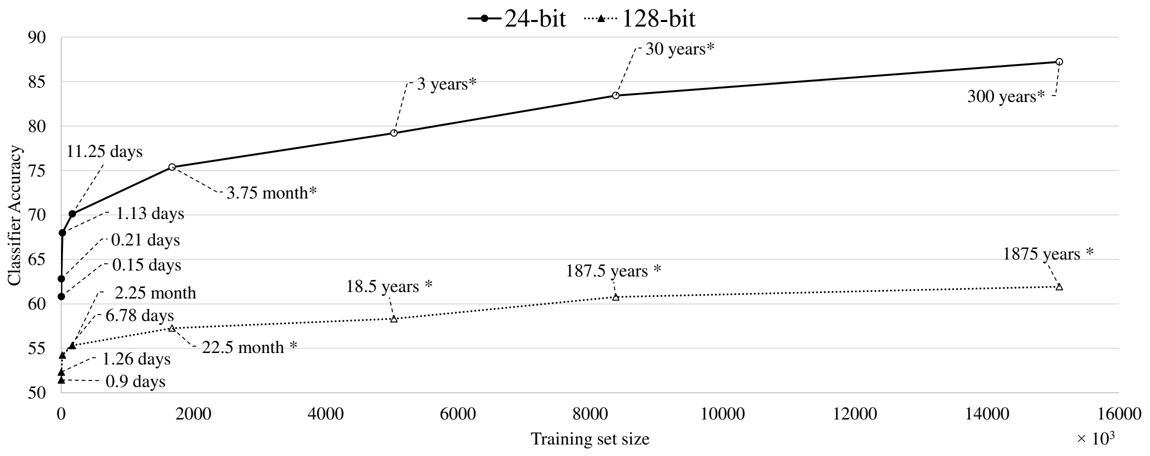


Fig. 11. Results of evolution strategy attack on 24-bit and 128-bit A-PUFs.

day. If each authentication requires a response of 64 bits or less for validation, then no more than 32 CRPs (as each challenge produces a two-bit trinary response) can be collected with every authentication request. The device will have to make at least 1404 authentications a day, which is close to one authentication per minute. This has yet taken into consideration the training time. If the prediction accuracy of the model is to be increased by just another 5%, the attacker will have to continuously query the device at sub-millisecond rate for one year just to collect enough CRPs for training. It is hard not to burnout the device or remain stealthy at this rate of authentication. Secondly and most importantly, the time required for a successful attack has exceeded the lifetime of an ordinary person. Even if the computing power can be increased by 100 times, it will still require a couple of years to achieve a poor prediction accuracy of only 60%.

Thus, the proposed algorithm is practically secure against different machine learning algorithms as it requires huge subset of CRPs with training time exceeding the device lifetime. If any of the BILBO parameters is reconfigured, the attack will have to start all over again as the successfully learned parameters have been changed.

C. Random Guessing

There are two parameters in the proposed design that the attacker may determine by trial and error. They are the $seed$ value and the number of possible polynomial $\phi(X)$ coefficients. The probability of guessing both values correctly in any one attempt can be estimated by:

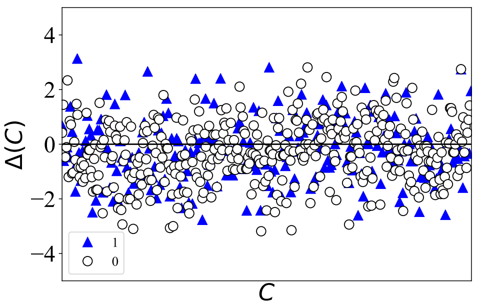
$$P_{Guess} = \frac{N}{2^{2\times N} - 2^N}. \quad (7)$$

If N = 24, P_{Guess} = 8.52×10^{-14}, which is equivalent to more than billion years of computation to break the system assuming that it takes one second to make and apply a guess. It is probably easier to perform a brute force attack by exhausting the whole challenge response set (2^{24} CRPs) to compromise these parameters. Since the actual CRP set is inaccessible to an attacker, this attack is not possible.

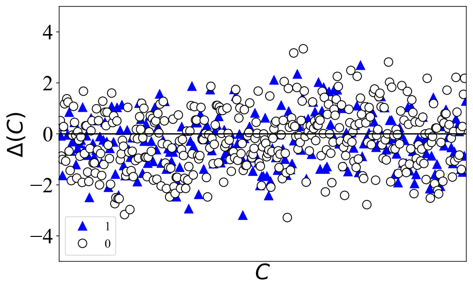
D. Compromising Seed and Polynomial Coefficients of BILBO

Performant physical and side channel attacks on strong PUFs have been reported [47]. These attacks assume that the attacker has access to the device to perform controlled measurements to collect a sufficient number of side-channel traces without being discovered. The attacker is also assumed to have complete understanding of the device protection to determine the most appropriate analysis for the exploitable side-channel signal, provided that such side-channel leakage can be accurately measured. Side channel attacks, even if feasible, help only to derive the approximate but not the exact values of the key parameters of BILBO circuit. It can at best reduce the search space for machine learning algorithm to boost modelling attack. According to [47], the training time can be reduced by 200 to 300 times, which means that in the worst case, the training time for the proposed 128-bit A-PUF design will be shortened to around 5-6 years to achieve an accuracy of 60% assuming that the attacker is able to measure exploitable side-channel leakage with similar accuracy. The attacker may, however, independently attack LFSR and MISR by side channel analysis [48] and then use the information to model the A-PUF. To successfully deduce four of the five secret parameters (seed and polynomial values of LFSR and MISR), without access to the internal nets, a large number of accurate power measurements is required due to the interplay between these components and PUF. As the computations are controlled by the internal FSM and triggered by authentication request, the power traces have to be collected during authentication with proper synchronization between the device and assumed server, which make the attack challenging.

To demonstrate the demand of accurate measurements required for a successful side channel attack, two different seed values, $00\ldots0$ and $00\ldots1$ are used to simulate a 16-stage A-PUF with its challenges pre-processed by a MISR with feedback polynomial $\phi(X) = x^{16} \oplus x^{15} \oplus x^{13} \oplus x^4 \oplus 1$. In both cases, M = 65534 challenges are applied to the MISR in the same order. The distribution of the two sets of CRP obtained are shown in Fig. 12, where C denotes the challenge and $\Delta(C)$ denotes the delay difference in ns between the two paths selected by C. Around 99% of the path delay values have changed, even with only a LSB change in the MISR $seed$. As a small error in the estimation of BILBO



(a) *seed* is $0\cdots00$



(b) *seed* is $0\cdots01$

Fig. 12. Distribution of differential delays (in ns) for obfuscated challenges generated by MISR with different seeds.

parameters has an avalanche effect on the CRP set, the cost of side-channel analysis increases significantly with the increased resolution of measurements required for an accurate modeling while countermeasures against side-channel attacks are made easier by reducing the magnitude of measurable side-channel leakage. For example, additional T Flip-Flop register can be used to mirror the LFSR. Each T Flip-Flop toggles when the corresponding D Flip-Flop in the LFSR is not switching and vice versa to eliminate any measurable difference in switching power. Changing the polynomial coefficients in reconfigurable memory will have similar effect. The collected traces are useless when the coefficients of the MISR has been reconfigured. Taking into consideration the training time required, 2-3 updates of polynomial coefficients or *seed* of MISR is adequate during the device lifetime to thwart combined modeling and side-channel attacks. A simple secure protocol to refresh these parameters by making use of the existing strong A-PUF and reliable PUF model is described as follows. The device sends a challenge C to the authentication server upon request to update the secret parameter. The server applies C and the value p_{New} of the secret parameter to be updated to its A-PUF model to obtain a response $R = \text{PUF}_{Model}(C)$ and $h_1 = \text{PUF}_{Model}(p_{New})$. Then, the server calculates $h_2 = p_{New} \oplus R$. The values h_1 and h_2 are then transmitted to the device. The device recovers p'_{New} by PUF(C) $\oplus h_2$ and authenticates the recovered parameter $p'_{New} = p_{New}$ by checking if h_1 is equal to PUF(p'_{New}) before it is updated on the device. To acknowledge the successful update of p_{New}, the device generates another challenge C' and calculates $h' = p'_{New} \oplus$ PUF(C'). The device then sends C' and h' to the server. The server verifies if p_{New} has been correctly updated on the device by checking if $p_{New} = h' \oplus \text{PUF}_{Model}(C')$ before switching to the new parameter for future authentication. Since the APUF is a strong PUF and the server can keep track of the used challenges, the attacker cannot reuse the response to a used challenge to generate h_1 without a genuine PUF nor create a valid h_2 to change the device secret parameter.

E. Spoofing attacks

During the authentication phase, the only values transmitted between the server and the device are N_{Rounds} from the server and R_{Final} from the device. Knowledge of these two information does not allow the attacker to obtain the real CRPs of the PUF. The attacker cannot falsify authentication by eavesdropping the communications to replay R_{Final} to N_{Rounds}. This is because the state of MISR depends not only on the current input challenge but also the past obfuscated challenges. A different output will be generated by the BILBO block each time even for the same N_{Rounds}. Knowing only the number of BILBO loops required to obfuscate the challenge from each authentication session does not allow the attacker to generate the correct response similar to a real device. Even if the attacker manages to apply the same challenge to the BILBO block, the responses from the device will be vastly different as the MISR transformation is lossy and cannot be uniquely recovered. Therefore, the only chance for an attacker to spoof the server is to build a mathematical clone of the device, which amounts to the same cost and difficulty as a successful machine learning attack. Any other manipulation of the original messages passing between the server and device may compromise the device from being successfully authenticated but cannot gain illegitimate privilege in accessing the services offered by the server as the authentic device.

F. Exploiting PUF instability

Recent research [49] has pointed out that the use of only selective stable CRPs for device authentication can be hazardous [37], [50], [51], [52]. These protocols validate the response based on the Hamming distance between the received and enrolled responses. This may allow the attacker to correlate the distance between the noisy response and the private parameters of the protocol. For example, FSM PUF [51] relies only on 1% of the CRP set to avoid the reliability issues of PUF but this gives the attacker an opportunity to deduce the path delay difference from the transmitted response. The proposed protocol utilizes the entire CRP set but corrects the unstable responses internally according to their response classification to achieve a reliability of 1.0 without having to transmit helper data that will reveal sensitive information. Furthermore, the MISR transformation is lossy and after a few iterations, it is irreversible and non-reproducible from the same input challenge. Thus, the proposed authentication protocol is practically secure against reliability-based side-channel modeling attacks. Its ability to refresh configurable private parameters makes it even harder for the attacker.

VII. Experimental Results and Discussions

A. Reliability test

The reliability [53] of a PUF can be evaluated by the Bit Error Rate (BER) of its responses, which is the average number of response bit errors obtained in a series of repetitive experiments. It was confirmed by our earlier experiments [23] that FPGA implementation of classical 128-stage A-PUF has average reliability of only 0.577. This is far from the acceptable BER requirement of 10^{-6} based on the standard of BER [54] required for a 128-bit key generator.

Assume that n_0 of logic zeros, n_1 of logic ones and n_X of logic Xs are obtained as the response bits to the same challenge in E tests ($n_0 + n_1 + n_X = E$). The trit X is due to the metastable state detectable by the arbiter circuit of Fig. 2. Let f_0, f_1 and f_X denote the frequency of occurrences of 0, 1 and X, respectively. Then,

$$f_\alpha = \frac{n_\alpha}{E}, \alpha \in \{0, 1, X\}. \tag{8}$$

Let R_{ref} be the most frequently appeared response of an A-PUF to a particular challenge C during E tests. R_{Ref} is defined by majority voting as:

$$R_{Ref} = \begin{cases} 0 & \text{if } \max(f_0, f_1, f_X) = f_0, \\ 1 & \text{if } \max(f_0, f_1, f_X) = f_1, \\ X & \text{if } \max(f_0, f_1, f_X) = f_X. \end{cases} \tag{9}$$

Then the A-PUF reliability $S(C)$ can be computed by:

$$S(C) = 1 - BER(C) = \\ = 1 - \frac{1}{E}\sum_{i=1}^{E} HD(R_{Ref}, R_i), \tag{10}$$

where R_i is the response to C at the i-th test and HD is the Hamming distance between two trinary responses [23].

If $K \in [1, 2^N]$ challenges C^{Ω_i} (i = 1, 2, …, K) are applied to an N-stage A-PUF, then the average reliability can be calculated by:

$$S_{avg} = \frac{1}{K}\sum_{i=1}^{K} S(C^{\Omega_i}). \tag{11}$$

The minimum reliability can be calculated by:

$$S_{min} = min(C^{\Omega_1}, C^{\Omega_2}, \ldots, C^{\Omega_K}). \tag{12}$$

To achieve a minimum reliability of 1.0, error correction is required. In our current A-PUF implementation, simple repetition code is used to decide the class of a quadruple response. The responses were further tested under varying operational conditions. The vendor specified allowable temperature range for Nexys-4 evaluation board is [−40° C, +90° C]. All temperature tests were performed with the Thermotron ®8800 temperature chamber. The results are shown in Fig. 13. It shows that repetition code of length k = 5 with majority voter is sufficient to achieve the reliability of 1.0.

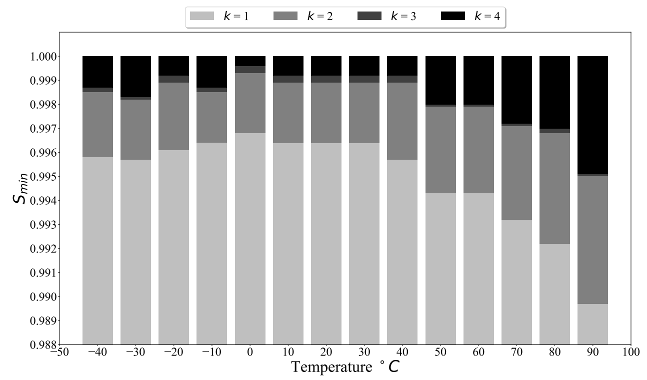


Fig. 13. Minimum reliability of A-PUF with repetition code length k at different operating temperature.

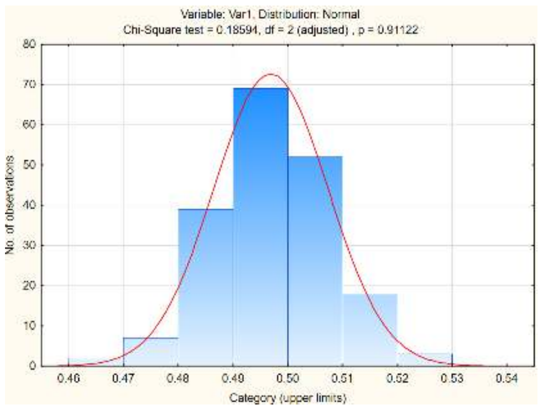


Fig. 14. Uniqueness of proposed A-PUF design.

B. Uniqueness test

Uniqueness is a measure of the average number of bit differences among the responses of different PUF instances to the same challenge. It is usually estimated by the inter-die Hamming distance (HD) [53] as follows:

$$U = \frac{2}{m(m-1)}\sum_{u=1}^{m-1}\sum_{v=u+1}^{m}\frac{HD(R_u, R_v)}{n}. \tag{13}$$

where m is the number of PUF instances. R_u and R_v are two n-bit responses to the same challenge generated by two different PUF instances, u and v, respectively.

10,000 challenges were generated for the uniqueness test. All challenges were generated in a MISR mode and repeated in the memory mode. m = 20 Xilinx Nexys 4 FPGA chips were used for the implementation of different PUF instances. The results are shown in Fig. 14. The inter-chip HD exhibits a normal distribution, with a mean of 0.497 and a standard deviation of 0.011. This is approximated to the ideal uniqueness of 0.5 with an accuracy of up to one decimal place. This implies that the zeros and ones in the MISR based challenge transformation has no negative effect on the uniqueness of the original A-PUF design.

C. Randomness test

The cryptographic randomness of a True Random Number Generators (TRNG) can be validated by 15 statistical tests

TABLE III
NIST RANDOMNESS TEST RESULTS.

Test Description	Passed/Total	P-value
Frequency	98/100	0.04
Frequency Block	100/100	0.53
Runs	100/100	0.74
The Longest Run	100/100	0.53
Binary Matrix Rank	100/100	0.35
FFT	100/100	0.46
Non-overlap. Template	98/100	0.57
Overlap. Template	96/100	0.35
Universal	100/100	0.05
Approx. Entropy	97/100	0.05
Serial	100/100	0.73
Cumulative Sums	98/100	0.04
Random exc.	10/10	—
Random exc. var.	10/10	—

of the National Institute of Standard and Technology (NIST) statistical test package [55]. A sequence of 10 million bits was generated in the MISR mode. This bit sequence was divided into 100 blocks, each of 100,000 bits. The NIST test results in Table III show that the response bits generated from the obfuscated challenges passed all the NIST tests successfully.

Besides, the bit sequence produced by the A-PUF with enhanced reliability of 1.0 still possesses good entropy, which matches the results obtained earlier with multiple A-PUF using MISR circuit [25]. Hence, it can also be post-processed by hash function similar to what has been done with the raw responses of RO-PUF, SRAM PUF and other designs for use as a true random sequence.

D. Hardware overhead

The proposed protocol can be viewed as an obfuscated strong PUF based authentication. Therefore, it is compared with several approaches from this category. Table IV compares the FPGA resources (LUT and DFF blocks) consumed by different 64- and 128-stage A-PUF based protocols with the authenticable device implemented on Xilinx Artix-7 FPGA chip.

TABLE IV
COMPARISON OF FPGA RESOURCE USAGES.

Approach	N	#LUT (% of available)	#FF (% of available)
PolyPUF [50]	64	213 (1.34)	450 (2.84)
Slender PUF [38]	64	652 (4.11)	1400 (8.83)
OB-PUF [37]	64	680 (4.29)	360 (2.27)
CMOS PUF [56]	64	395 (2.49)	176 (1.12)
RPUF [52]	128	350 (2.21)	389 (2.45)
PUF-FSM [51]	128	960 (6.06)	1500 (9.46)
This approach	64	192 (1.19)	132 (0.84)
This approach	128	380 (2.39)	264 (1.67)
This approach (with secure update)	128	419 (2.64)	264 (1.67)

For the same number of stages N, the proposed design is more efficient in terms of the overall utilization rate of LUT blocks and FFs. It uses slightly more LUTs but significantly less FFs than RPUF. As indicated in the last row, the secure parameter update protocol only incurs $\sim$0.25% additional LUTs. Since the proposed A-PUF leads to a more secure protocol with increased resistance against modeling attack by both basic and advanced machine learning algorithms, the sacrifice of around 2% of hardware resources of a FPGA chip is well justifiable. Moreover, if the BILBO block already exists in the system for BIST, it can be reused for this purpose to make the overhead twice smaller.

As machine learning attack can also be prevented by encrypting challenge and/or response values as stated in [57], the hardware resources required for the implementation of two standard cryptographic techniques, namely symmetric encryption (fast and compact versions of 128-bit Advanced Encryption Standard (AES)) and Secure Hash Algorithm (SHA-256 of SHA-2 mentioned in [57]) on the same Xilinx Artix-7 FPGA are also evaluated and compared. The number of LUTs and Flip-Flops used, and accuracies of SVM and CMA-ES based modeling attacks on these implementations are compared in Table V. The results show that existing cryptographic approaches to enhance ML-resistance of PUF require around twice to ten times more FPGA fabrics than the proposed MISR obfuscation method.

TABLE V
COMPARISON WITH STANDARD CRYPTOGRAPHIC TECHNIQUES.

Technique	#LUT	#FF	SVM	CMA-ES
This	380	264	50%	56%
SHA-256 [58]	3756	2609	50%	50%
AES (small) [59]	888	444	50%	50%
AES (fast) [60]	1642	820	50%	50%

VIII. CONCLUSION

A new approach to generate highly reliable trinary response with SR latch based A-PUF in FPGA implementation has been proposed. Robust authentication is achieved by classification of quadruple responses based on the detected trits of logic zero, logic one and metastable states from four challenges differ only in MSB and LSB. It has been demonstrated that this quadruple response classification augmented by majority voting with simple repetition code of length five can achieve a perfect reliability of 1.0 over the temperature range from $-40°$C to $90°$. To break the linear dependency between the challenge and response mapping, a BILBO module is used to obfuscate the input challenge and the quadruple response classes are compressed into five unique quadbits. To avoid maintaining a large CRP database at the server, a DNN is trained by the measured quadruple CRPs to construct a precise A-PUF model. The proposed authentication protocol does not require the server to send raw challenge or helper data to authenticate the A-PUF device, which significantly increases its resistance against modeling attacks. It has been shown that more than billions of CRPs and hundreds of years of training are required by CMA-ES to break the proposed A-PUF authentication system. Thus, the proposed authentication method has fulfilled multiple challenging desiderata, such as high reliability, high modeling attack resistance, high uniqueness, low predictability and low hardware resource overhead, in using strong PUF for FPGA security.

ACKNOWLEDGMENT

This research is supported by the Singapore Ministry of Education Academic Research Fund (AcRF) Tier II Grant No. MOE 2015-T2-2-013.

REFERENCES

[1] R. Insights. (2015, Oct) FPGA market size, share, analysis, research report, 2020. [Online]. Available: http://www.radiantinsights.com/research/fpga-market

[2] I. Bolsens, "All programmable SoC FPGA for networking and computing in big data infrastructure (Keynote I)," in *Proc. Asia and South Pacif. Des. Automat. Conf. (ASP-DAC'14)*, Singapore, Jan 21 2014, p. 1. [Online]. Available: http://img.deusm.com/eetimes/zob-3pi-0023-01-lg.jpg

[3] R. Das, V. Markovich, J. McNamara, and M. Poliks, "Anti-tamper microchip package based on thermal nanofluids or fluids," USA Patent US20 120 068 326 A1, Mar 22, 2012. [Online]. Available: http://www.google.com/patents/US20120068326

[4] R. S. Chakraborty and S. Bhunia, "RTL hardware IP protection using key-based control and data flow obfuscation," in *Proc. Int. Conf. on VLSI Des. (VLSI'10)*, Bangalore, India, Jan 2010, pp. 405–410.

[5] N. Couture and K. B. Kent, "Periodic licensing of FPGA based intellectual property," in *Proc. IEEE Int. Conf. on Field Progr. Techn. (ICFPT'06)*, Bangkok, Thailand, Dec 2006, pp. 357–360.

[6] E. Peterson, "Developing tamper resistant designs with xilinx virtex-6 and 7 series FPGAs," Xilinx Inc., Tech. Rep., Oct 2013.

[7] W. Liang *et al.*, "A chaotic IP watermarking in physical layout level based on FPGA," *Radioeng.*, vol. 20, no. 1, pp. 118–125, Apr 2011.

[8] C. H. Chang and L. Zhang, "A blind dynamic fingerprinting technique for sequential circuit intellectual property protection," *IEEE Trans. Comput.-Aided Des. Integr. Circ. and Syst.*, vol. 33, no. 1, pp. 76–89, Jan 2014.

[9] F. Koushanfar, "Hardware metering: A survey," in *Introduction to Hardware Security and Trust*, M. Tehranipoor and C. Wang, Eds. Switzerland: Springer, 2011, pp. 103–122.

[10] P. Tuyls *et al.*, *Security with Noisy Data*, P. Tuyls *et al.*, Eds. Switzerland: Springer, 2007.

[11] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "FPGA intrinsic PUFs and their use for IP protection," in *Proc. Int. Worksh. Crypt. Hardw. and Emb. Syst. (CHES'07)*, Vienna, Austria, Sep 2007, pp. 63–80.

[12] D. Yamamoto *et al.*, "Uniqueness enhancement of PUF responses based on the locations of random outputting RS latches," in *Proc. Int. Worksh. Crypt. Hardw. and Emb. Syst. (CHES'11)*, Nara, Japan, Sep 2011, pp. 390–406.

[13] S. Morozov, A. Maiti, and P. Schaumont, "An analysis of delay based PUF implementations on FPGA," in *Proc. Int. Symp. Reconfig. Comput.: Architect., Tools and Applic. (ARC'10)*, Berlin, Germany, Mar 2010, pp. 382–387.

[14] T. Xu and M. Potkonjak, "Robust and flexible FPGA-based digital PUF," in *Proc. Int. Conf. on Field Program. Logic and Applic. (FPL'14)*, Munich, Germany, Sep 2014, pp. 1–6.

[15] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. Des. Autom. Conf. (DAC'07)*, San Diego, USA, Jun 2007, pp. 9–14.

[16] G. Prophet. (2016, Oct.) Xilinx to add PUF security to ZYNQ devices. http://www.eenewseurope.com/news/xilinx-add-puf-security-zynq-devices-0.

[17] (2015, Jun.) Altera reveals stratix 10 with intrinsic-ID's PUF technology. https://www.intrinsic-id.com/altera-reveals-stratix-10-with-intrinsic-ids-puf-technology/.

[18] A. Vijayakumar, V. C. Patil, C. B. Prado, and S. Kundu, "Machine learning resistant strong PUF: Possible or a pipe dream?" in *Proc. Hardw. Orient. Secur. and Trust (HOST'16)*, McLean, USA, May 2016, pp. 19–24.

[19] J.-L. Zhang *et al.*, "Techniques for design and implementation of an FPGA-specific physical unclonable function," *J. of comp. sci. and technol.*, vol. 1, no. 31, pp. 124–136, Jan 2015.

[20] M. Majzoobi, F. Koushanfar, and S. Devadas, "FPGA PUF using programmable delay lines," in *Proc. IEEE Int. Worksh. Info. Forens. Secur. (WIFS'10)*, Seattle, USA, Dec 2010, pp. 1–6.

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

[22] M.-D. Yu, M. Hiller, and S. Devadas, "Maximum-likelihood decoding of device-specific multi-bit symbols for reliable key generation," in *Proc. Hardw. Orient. Secur. and Trust (HOST'15)*, Washington, USA, May 2015, pp. 38–43.

[23] S. S. Zalivaka *et al.*, "Multi-valued arbiters for quality enhancement of PUF responses on FPGA implementation," in *Proc. Asia and South Pacific Des. Automat. Conf. (ASP-DAC'16)*, Macau, China, Jan 2016, pp. 533–538.

[24] V. P. Klybik and A. A. Ivaniuk, "Use of arbiter physical unclonable function to solve identification problem of digital devices," *Autom. Contr. Comput. Sci.*, vol. 49, no. 3, pp. 139–147, 2015.

[25] S. S. Zalivaka *et al.*, "Design and implementation of high-quality physical unclonable functions for hardware-oriented cryptography," in *Secur. System Des. and Trust. Comput.*, C. H. Chang and M. Potkonjak, Eds. Switzerland: Springer, 2016, pp. 39–81.

[26] S. S. Zalivaka, A. A. Ivaniuk, and C. H. Chang, "FPGA implementation of modeling attack resistant arbiter PUF with enhanced reliability," in *Proc. IEEE Int. Symp. on Quality Electron. Des. (ISQED'17)*, Santa Clara, USA, Mar 2017.

[27] S. S. Zalivaka, A. A. Ivaniuk, and C. H. Chang, "Low-cost fortification of arbiter PUF against modeling attack," in *Proc. IEEE Int. Symp. on Circ. and Syst. (ISCAS'17),*, Baltimore, USA, May 2017, pp. 1600–1603.

[28] T. Machida, D. Yamamoto, M. Iwamoto, and K. Sakiyama, "Implementation of double arbiter PUF and its performance evaluation on FPGA," in *Proc. Asia and South Pacific Des. Automat. Conf. (ASP-DAC'15)*, Chiba/Tokyo, Japan, Jan. 2015, pp. 6–7.

[29] Y. Lao and K. K. Parhi, "Statistical analysis of MUX-based physical unclonable functions," *IEEE Trans. on Comp.-Aided Des. of Int. Circ. and Syst.*, vol. 33, no. 5, pp. 649–662, 2014.

[30] A. Vijayakumar and S. Kundu, "A novel modeling attack resistant PUF design based on non-linear voltage transfer characteristics," in *Proc. Des., Autom. and Test in Europe Conf. (DATE'15)*, Grenoble, France, Mar. 2015, pp. 653–658.

[31] R. Kumar and W. Burleson, "On design of a highly secure PUF based on non-linear current mirrors," in *Proc. Hardw. Orient. Secur. and Trust (HOST'14)*, Arlington, USA, May 2014, pp. 38–43.

[32] M. Kalyanaraman and M. Orshansky, "Novel strong PUF based on nonlinearity of MOSFET subthreshold operation," in *Proc. Hardw. Orient. Secur. and Trust (HOST'13)*, Austin, USA, Jun. 2013, pp. 13–18.

[33] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Controlled physical random functions," in *Proc. ACM Annual Comp. Secur. Appl. Conf. (ACSAC'02)*, Washington, DC, USA, Dec. 2002, pp. 149–161.

[34] U. Chatterjee, R. S. Chakraborty, H. Kapoor, and D. Mukhopadhyay, "Theory and application of delay constraints in arbiter PUF," *ACM Trans. on Embed. Comp. Syst.*, vol. 15, no. 1, pp. 1–20, Jan 2016.

[35] U. Ruhrmair *et al.*, "PUF modeling attacks on simulated and silicon data," *IEEE Trans. on Inf. Forens. and Secur.*, vol. 8, no. 11, pp. 1876–1891, Aug. 2013.

[36] 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. Des. Autom. Conf. (DAC'17)*, Austin, USA, Jun. 2017, pp. 101–106.

[37] Y. Gao *et al.*, "Obfuscated challenge-response: A secure lightweight authentication mechanism for PUF-based pervasive devices," in *Proc. IEEE Int. Worksh. on Secur., Priv. and Trust for IoT*, Sydney, Australia, Mar. 2016, pp. 1–6.

[38] M. Rostami *et al.*, "Robust and reverse-engineering resilient PUF authentication and key-exchange by substring matching," *IEEE Trans. on Emerg. Topics in Comput.*, vol. 2, no. 1, pp. 37–49, Jan. 2014.

[39] J. Lee *et al.*, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *Proc. Int. Symp. VLSI Circ. (VLSI'04)*, Honolulu, USA, Jun. 2004, pp. 176–179.

[40] E. Ozturk, G. Hammouri, and B. Sunar, "Physical unclonable function with tristate buffers," in *Proc. IEEE Int. Symp. on Circ. and Syst. (ISCAS'08)*, Seattle, USA, May 2008, pp. 3194–3197.

[41] T. Kacprzak, "Analysis of oscillatory metastable operation of an RS flip-flop," *IEEE J. of Solid-State Circ.*, vol. 23, no. 1, pp. 260–266, Feb 1988.

[42] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Testing techniques for hardware security," in *Proc. IEEE Int. Test Conf. (ITC'08)*, Santa Clara, USA, Oct 2008, pp. 1–10.

[43] U. Ruhrmair *et al.*, "Modeling attacks on physical unclonable functions," in *Proc. ACM Conf. on Comp. and Comm. Secur. (CCS'10)*, Oct. 2010, pp. 237–249.

[44] V. Agrawal, C. Kime, and K. Saluja, "A tutorial on built-in self-test. 2. applications," *IEEE Des. Test Comput.*, vol. 10, no. 2, pp. 69–77, Jun. 1993.

[45] H. C. A. van Tilborg and S. Jajodia, Eds., *Encyclopedia of Cryptography and Security*. Springer, 2011. [Online]. Available: https://link.springer.com/referenceworkentry/10.1007%2F978-1-4419-5906-5_487
[46] G. T. Becker, "On the pitfalls of using arbiter-PUFs as building blocks," *IEEE Trans. on Comp.-Aided Des. of Integr. Circ. and Syst.*, vol. 34, no. 8, pp. 1295–1307, Apr. 2015.
[47] A. Mahmoud, U. Ruhrmair, M. Majzoobi, and F. Koushanfar. (2013) Combined modeling and side channel attacks on strong PUFs. [Online]. Available: https://eprint.iacr.org/2013/632.pdf
[48] S. Burman, D. Mukhopadhyay, and K. Veezhinathan, "LFSR based stream ciphers are vulnerable to power attacks," in *Int. Conf. on Cryptol. in India (Indocrypt'07)*, Chennai, India, Dec. 2007, pp. 384–392.
[49] J. Delvaux. (2017, Dec.) Machine learning attacks on PolyPUF, OB-PUF, RPUF, and PUF–FSM. https://eprint.iacr.org/2017/1134.pdf.
[50] S. T. C. Konigsmark, D. Chen, and M. D. F. Wong, "PolyPUF: Physically secure self-divergence," *IEEE Trans. on Comp.-Aided Des. of Integr. Circ. and Syst.*, vol. 35, no. 7, pp. 1053–1066, Oct. 2015.
[51] Y. Gao *et al.*, "PUF-FSM: A controlled strong PUF," *IEEE Trans. on Comp.-Aided Des. of Integr. Circ. and Syst.*, vol. 1, no. 99, pp. 1–5, 2017.
[52] J. Ye, Y. Hu, and X. Li, "RPUF: Physical unclonable function with randomized challenge to resist modeling attack," in *Proc. IEEE Asian Hardw.-Orient. Secur. and Trust (AsianHOST'16)*, 2016.
[53] Y. Hori, T. Yoshida, T. Katashita, and A. Satoh, "Quantitative and statistical performance evaluation of arbiter physical unclonable functions on FPGAs," in *Proc. Int. Conf. on Reconfig. Comput. and FPGAs (ReConFig'10)*, Quintana Roo, Mexico, Dec 2010, pp. 298–303.
[54] M. Bhargava and K. Mai, "An efficient reliable PUF-based cryptographic key generator in 65nm CMOS," in *Proc. Des., Autom. and Test in Europe Conf. (DATE'14)*, Dresden, Germany, Mar 2014, pp. 1–6.
[55] L. E. Bassham *et al.*, "Sp 800-22 rev. 1a. A statistical test suite for random and pseudorandom number generators for cryptographic applications," National Institute of Standards & Technology, Gaithersburg, USA, Tech. Rep., 2010.
[56] J. Zhang and L. Wan. (2018, Jun.) CMOS: Dynamic multi-key obfuscation structure for strong PUFs. Cornell University Library. Ithaca, USA. [Online]. Available: https://arxiv.org/pdf/1806.02011.pdf
[57] C. Herder *et al.*, "Trapdoor computational fuzzy extractors and stateless cryptographically-secure physical unclonable functions," *IEEE Trans. on Depend. and Secur. Comput.*, vol. 14, no. 1, pp. 65–82, Mar. 2016.
[58] Z. Shi, C. Ma, J. Cote, and B. Wang, *Introduction to Hardware Security and Trust*. Springer, 2012, ch. Hardware Implementation of Hash Functions, pp. 27–50.
[59] P. Chodowiec and K. Gaj, "Very compact FPGA implementation of the AES algorithm," in *Proc. Int. Worksh. Crypt. Hardw. and Emb. Syst. (CHES'03)*, Cologne, Germany, Sep. 2003, pp. 319–333.
[60] T. Good and M. Benaissa, "AES on FPGA from the fastest to the smallest," in *Proc. Int. Worksh. Crypt. Hardw. and Emb. Syst. (CHES'05)*, Edinburg, UK, Sep. 2005, pp. 427–440.

Siarhei S. Zalivaka Siarhei S. Zalivaka received the B. Eng (Hons.) degree and M.Eng. degree in Belarusian State University of Informatics and Radioelectronics (BSUIR) in 2012 and 2013, respectively. His thesis is pending approval for Ph.D. degree in the School of Electrical and Electronic Engineering of Nanyang Technological University (NTU). His area of research is Active Metering Schemes for Digital Rights Management, Physical Unclonable Functions.

Alexander A. Ivaniuk Alexander A. Ivaniuk received the M.Eng. degree in Belarusian State University of Informatics and Radioelectronics (BSUIR), Minsk, Republic of Belarus, in 1999 he received the Ph.D. degree from BSUIR and in 2010 the Dr.S. (Doctor of Science) degree from BSUIR. He served as an Assistant of Professor and as an Associated Professor at the department of Software for Information Technologies in BSUIR form 1998 to 1999 and from 1999 to 2008 respectively. He served as a head of the Computational Methods and Programming Department in BSUIR from 2008 to 2013 and as a head of the Computer Science Department in BSUIR from 2013 to 2014. Since 2014, he is a professor at the Computer Science Department in BSUIR. His areas of research interest are Design and Testing of Digital Devices and Systems, Fault-tolerant Computing, Hardware Synthesis and Simulations, Embedded Systems Design and Programming, Hardware Cryptography. He is a coauthor of 8 books, one book chapter, more than 40 journal papers and more than 60 international conference papers. He is a member of the IEEE Circuits and Systems Society and IEEE Computer Society.

Chip-Hong Chang Chip-Hong Chang (S'92–M'98–SM'03–F'18) received the B.Eng. (Hons.) degree from the National University of Singapore in 1989, and the M. Eng. and Ph.D. degrees from Nanyang Technological University (NTU), Singapore in 1993 and 1998, respectively. He served as a Technical Consultant in industry prior to joining the School of Electrical and Electronic Engineering (EEE) of NTU in 1999, where he is currently an Associate Professor. He holds joint appointments with the university as Assistant Chair of Alumni from 2008 to 2014, Deputy Director of the Center for High Performance Embedded Systems from 2000 to 2011, and Program Director of the Center for Integrated Circuits and Systems from 2003 to 2009. He has coedited four books, published ten book chapters, around 100 international journal papers (two-thirds are IEEE) and more than 170 refereed international conference papers (mostly in IEEE). His current research interests include hardware security, residue number systems, and application-specific digital signal processing algorithms and architectures.

Dr. Chang serves as the Associate Editor of IEEE Transactions on Very Large Scale Integration (VLSI) Systems since 2011, IEEE Access since 2013, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems and IEEE Transactions on Information Forensic and Security since 2016, IEEE Transactions on Circuits and Systems-I from 2010-2013, Integration, the VLSI Journal from 2013-2015, Springer Journal of Hardware and System Security since 2016 and Microelectronics Journal since 2014. He was the editorial advisory board member of Open Electrical and Electronic Engineering Journal from 2007 to 2013 and Journal of Electrical and Computer Engineering from 2008 to 2014. He guest edited several special issues and served in the organizing and technical program committee of more than 50 international conferences (mostly IEEE). He is an IET Fellow, IEEE Fellow, and 2018-2019 IEEE Circuits and Systems Society Distinguished Lecturer.