**QUEEN'S
UNIVERSITY
BELFAST**

A Modeling Attack Resistant Deception Technique for Securing Lightweight-PUF based Authentication

Gu, C., Chang, C. H., Liu, W., Yu, S., Wang, Y., & O'Neill, M. (2020). A Modeling Attack Resistant Deception Technique for Securing Lightweight-PUF based Authentication. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. <https://doi.org/10.1109/TCAD.2020.3036807>

Published in:

IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

Document Version:

Peer reviewed version

Queen's University Belfast - Research Portal:

[Link to publication record in Queen's University Belfast Research Portal](#)

Publisher rights

Copyright 2020 IEEE. This work is made available online in accordance with the publisher's policies. Please refer to any applicable terms of use of the publisher.

General rights

Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.

A Modeling Attack Resistant Deception Technique for Securing Lightweight-PUF based Authentication

Chongyan Gu, *Member, IEEE*, Chip-Hong Chang, *Fellow, IEEE*, Weiqiang Liu, *Senior Member, IEEE*, Shichao Yu, Yale Wang, and Máire O'Neill, *Senior Member, IEEE*,

Abstract—Silicon physical unclonable function (PUF) has emerged as a promising spoof-proof solution for low-cost device authentication. Due to practical constraints in preventing phishing through public network or insecure communication channels, simple PUF-based authentication protocol with unrestricted queries and transparent responses is vulnerable to modeling and replay attacks. Although PUF itself is lightweight, the ancillary cryptographic primitives required to support secure handshaking in classical PUF-based authentication protocol is not necessarily so. In this paper, we present a modeling attack resistant PUF-based mutual authentication scheme to mitigate the practical limitations in applications where a resource-rich server authenticates a device with no strong restriction imposed on the type of PUF design or any additional protection on the binary channel used for the authentication. Our scheme uses an active deception protocol to prevent machine learning (ML) attacks on a device with a monolithic integration of a genuine Strong PUF (SPUF), a fake PUF, a pseudo random number generator (PRNG), a register, a binary counter, a comparator and a simple controller. The hardware encapsulation makes collection of challenge response pairs (CRPs) easy for model building during enrollment but prohibitively time-consuming upon device deployment through the same interface. A genuine server can perform a mutual authentication with the device using a combined fresh challenge contributed by both the server and the device. The message exchanged in clear cannot be manipulated by the adversary to derive unused authentic CRPs. The adversary will have to either wait for an impractically long time to collect enough real CRPs by directly querying the device or **the ML model derived from the collected CRPs will be poisoned** to expose the imposter when it is used to perform a spoofing attack. The false PUF multiplexing is fortified against prediction of waiting time by doubling the time penalty for every unsuccessful guess. Our implementation results on field programmable gate array (FPGA) device and security analysis have corroborated the low hardware overheads and attack resistance of the proposed deception protocol.

Index Terms—Physical Unclonable Function (PUF), machine learning attacks, authentication protocol, deception protocol, **poison attack**.

I. INTRODUCTION

WHILE Internet of things (IoT) revolutionizes our lives through remote healthcare, autonomous vehicles, smart

homes, it also brings security issues. The large number of devices open up new attack vectors, as exemplified by the IoT-based distributed denial-of-service (DDoS) attack on Dyn, that brought down Twitter, SoundCloud, Spotify, Reddit and a host of other sites [1]. Providing security to IoT devices is a major challenge as small devices tend to be limited in terms of resources and computing power. Conventional security approaches, based on provably secure cryptographic algorithms, are too resource intensive for implementation on these devices.

A physical unclonable function (PUF) is a security primitive that utilizes intrinsic manufacturing process variations to generate a unique digital fingerprint. A comprehensive review of PUF can be found in [2]. As this natural variation among silicon dies is outside the control of the manufacturer, PUFs are inherently difficult to clone, and possess additional tamper-evident properties [3], [4]. PUFs can produce unique keys on-the-fly, which reduces the risk of physical attacks and saves hardware resources. These properties open up interesting opportunities for higher level security protocols such as key generation and device authentication for both application-specific integrated circuit (ASIC) and FPGA based devices.

The initial proposal of using a Hamming distance (HD) threshold τ comparison for lightweight PUF-based authentication protocol was proposed in [5]. Since then, similar PUF-based authentication protocols have been derived to endow linearly-sized SPUF circuits with exponentially large CRP capacity for device authentication. Unfortunately, SPUFs used for device authentication with limited nonlinear mixing have been shown to be vulnerable to ML based modeling attacks. Masquerade attacks can be perpetrated through malicious nodes and unprotected communication links of ad hoc networks to efficiently collect a large number of CRPs to model a PUF. To prevent modeling attacks, many countermeasures [6]–[8] have been proposed, *e.g.*, increasing the circuit complexity of PUF. These approaches tend to degrade other quality such as reliability of the underlying PUF. They are not resilient against replay and man-in-the-middle (MITM) attacks without additional protection to track and prevent the used CRPs from being reused. To defend against brute-force query by using existing SPUF for authentication of lightweight devices, *e.g.*, IoT and wearable devices, the lockdown protocol [9] simply restricts the number of authentication events to limit the number of CRPs from being learned for modeling attack.

This paper is an invited extension of our preliminary work reported in [10] for this special issue. Unlike prior works, our proposed PUF-based authentication protocol does not

C. Gu and Y. Shichao and M. O'Neill are with the Centre for Secure Information Technologies (CSIT), Institute of Electronics, Communications & Information Technology (ECIT), Queen's University Belfast (QUB), U.K., BT3 9DT. e-mail: {cgu01, syu08}@qub.ac.uk and m.oneill@ecit.qub.ac.uk

C. H. Chang is with School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, 639798. e-mail: echchang@ntu.edu.sg.

W. Liu and Y. Wang are with College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics (NUAA), China, 211106. e-mail: {liuweiqiang, yalewang}@nuaa.edu.cn

Manuscript received xx xx, 2020; revised xx xx, 2020.

simplistically rely on expensive error correction code (ECC) or crypto-algorithm on the device side to passively increase its attack resistance. Instead, it adopts an offensive defense strategy to cautiously retaliate and frustrate the adversaries by poisoning their training data. This is achieved through a combination of hardware encapsulation and protocol-level approach to deceive or excessively delay the adversary attempting to collect the CRPs by brute-force queries made on the device enrollment protocol interface. Specifically, our contributions in this paper are summarized as follows:

- The proposed deception protocol has been improved to enable a resource-rich server to authenticate a device with no strong restriction imposed on the type of PUF design or any additional protection on the binary channel used for the authentication. Specifically, re-enrollment for model calibration is permitted by our model-based authentication. It can be made more secure against direct CRP collection through the enrollment protocol for model building attack.
- The proposed deception protocol has been re-evaluated under both the LR and CMA-ES modelling attacks using the most recent published experimental platforms [11].
- Experimental results previously conducted using Matlab RNG function to simulate TRNG outputs have been re-evaluated using TRNG in FPGA implementation.
- A comparison of heuristic modelling attack results for both XOR APUF and Feed-Forward APUF used for the proposed deception protocol has been presented.
- A hardware implementation for the proposed deception protocol is demonstrated on a Xilinx Artix-7 FPGA to validate its hardware efficiency.

The proposed deception protocol is implemented on a Xilinx Artix-7 FPGA to demonstrate its low overheads in comparison with previous works. Its resistance against modeling attacks is evaluated by linear regression (LR) and covariance matrix adaptation evolution strategies (CMA-ES), which are two of the widely utilized and powerful ML techniques. The effectiveness of the fake PUF in reducing the successful prediction rate is also evaluated.

The rest of this paper is organized as follows. Section II reviews the vulnerability of SPUFs to ML based modeling attacks, and identifies the gaps in existing PUF-based authentication schemes that motivate this research. Section III provides the preliminaries of PUF-based authentication and modeling attacks. The proposed deception protocol is presented in Section IV. It is evaluated against both LR and CMA-ES attacks in Section V. An FPGA-based implementation of the proposed deception protocol is presented in Section VI. A comparison between the proposed deception protocol and other lightweight authentication protocols is presented in Section VII. Finally, conclusions are drawn in Section VIII.

II. ATTACKS ON SPUFs

PUF architectures are broadly categorized into weak and strong in [12] based solely on their number of unique CRPs per bit cell. Weak PUFs have a limited CRP space. They are more suited as random key generator or for seeding a PRNG, where the response never leaves the chip and is only accessed as required. In contrast, SPUFs have a large number

of possible CRPs. The response bitstream returned from a random sequence of challenges is unique to each challenge sequence and the physical PUF device. By design, this implies the requirement for a much larger entropy pool such that related challenges should not lead to related responses on the same device. Hence, SPUFs are preferred to weak PUFs for device authentication.

Most SPUF architectures based on linear additive functions have been shown to be vulnerable to ML attacks [13]–[16]. Therefore, the focus on SPUF research nowadays has been directed towards preventing ML attacks. At circuit level, the modeling complexity of SPUF design has been increased by, *e.g.*, XOR Arbiter PUF (APUF) [6], feed-forward APUF [7], lightweight secure APUF [8] and multi-PUF [17]. Most of these approaches achieve increased resistance to ML attacks by using more complex PUF architectures, *e.g.*, [6], [8], while preserving a desirably huge number of CRPs. Even then, it has been shown more recently that the workload of ML attacks can be reduced and the success rate can be raised through exploiting the weakness in their protocols [13]–[16], [18]. Side channel information have also been demonstrated to aid PUF attack. PUF noise can be filtered to improve the signal-to-noise ratio for efficient side-channel attacks [19], [20]. A combination of side-channel analysis (SCA) and modeling attacks, exploiting the noise of PUFs, have been proposed to effectively crack the secret response of PUF [21], [22].

Delvaux *et al.* in [23] presented a survey on entity authentication protocols using PUFs. It shows that most PUF protocols [24]–[28] are heavyweight or require complicated protocol operations that add to the hardware implementation area, power and performance overheads. Two protocols, the slender PUF [29] and noise bifurcation [30], require neither an ECC nor a strong cryptographic algorithm. A true random number generator (TRNG) has been used instead to provide heuristic security against modeling attacks, which is difficult to validate. The idea of slowing down the read out of PUF response was first raised in [31]. The SHIC PUF proposed in [31] was realized by an emerging high density $10^5 \times 10^5$ crossbar memory with an intrinsically slow access speed of 100 b/s. It is assumed that the amount of independent structural information of SHIC PUF cannot be fully modelled without a complete circuit characterization. Hence, it takes ≈ 3 years to collect the full set of CRPs with this access speed. However, if only 10^5 instead of the complete set of CRPs is required to learn a PUF by ML techniques, they can be acquired in less than 20 minutes. As SHIC PUF is implemented by emerging semiconductor technology, it does not integrate well with conventional CMOS designs [9]. The incompatibility also raises constraints on the “nano-micro” link [31]. Moreover, there is a constraint on intrinsic slowdown of SHIC PUF as it will impact the number of CRPs that can be practically enrolled. As SHIC is a weak PUF by the independent cell structure criterion, its CRP space grows only linearly as opposed to exponentially with array size. The cost of producing only $10^{10} \approx 2^{33}$ CRPs by SHIC PUF is significantly higher than a 128-stage APUF that can generate 2^{128} CRPs. Recently, Yu *et al.* [9] proposed two lockdown techniques to limit the number of authentication requests. The first lockdown protocol allows only unilateral

authentication while the second requires a TRNG on the device side to generate a device nonce, c_D . As the responses r_1 and r_2 in the second lockdown protocol are sent in clear, the adversary can query the device to obtain a new device-side challenge c'_D and then adjust the server-side challenge c'_S until the linear-feedback shift register (LFSR) output $\langle c \rangle$ is the same as one of the previously eavesdropped tuples (c_D, c_S, r_1) . The adversary can then reply with $c'_S \parallel r_1$ to gain successful authentication. The viability of this is limited by the previously eavesdropped tuples and LFSR seed length. The trade-off of lockdown protocol is that it can only support a limited number of authentication requests for the verifier. The protocol relies on the hardness of XOR PUF in the field to support model-based authentication. However, its hardware cost increases and reliability reduces commensurately with increasing number of XORs for higher ML resistance. Gao *et al.* [32] uses a reconfigurable latent obfuscation technique to conceal and distort the relationship between CRPs. The pattern vectors for challenge and response obfuscation are selected by a random number generator (RNG), and are made latent and reconfigurable per authentication session. Nevertheless, the most recent report shows that both methods [9], [32] are vulnerable to the protocol attack [33]. A PUF-based mutual authentication protocol called PHEMAP has been proposed recently [34]. However, it has been shown to be vulnerable to impersonate, desynchronization and traceability attacks [35]. Two multiplexer-based PUF (MPUF) variants (rMPUF and cMPUF) [36] were also proposed to resist reliability-based and cryptanalysis modeling attacks, respectively. However, they have been successfully attacked by approximation attack in [37] using an approximation algorithm based artificial neuron network. This advanced attack falls under the machine learning based approaches, which require the collection of sufficient number of valid CRPs for training.

III. PRELIMINARIES

This section provides an overview of a PUF-based authentication protocol, and the linear additive model that forms the basis of ML-based modeling attacks on PUF. A list of frequently used notations in this paper is provided in Table I.

A. Basic SPUF-based Authentication Protocol

The basic SPUF-based authentication method for unilateral authentication involves a verifier, usually a server, and a prover, which is a device embedded with only a single SPUF. The main operation is triggered by the server sending a challenge c to request for an authentication of the device. The device responds by activating its embedded SPUF to generate a response r to the server for verification. The complete process for this basic SPUF authentication protocol is depicted in Fig. 1. Before a device i is deployed, it will undergo an one-time enrollment process in a secure and control environment whereby d CRPs¹, (c_{ij}, r_{ij}) (j is the j -th CRP and $j \in [1, d]$) are collected from the device and stored along with the device identifier (ID) in the server secure database. The device

Table I
List of Frequently Used Notations

Notation	Description
ID_i	Identifier of device i
n	Length of PUF challenge string
m	Length of PUF response string
c	An n -bit challenge
$\langle c \rangle$	List of consecutive n -bit sub-challenges derived from c
r	Enrolled m -bit response
$\tilde{r}$	Reproduced m -bit response
(c, r)	A challenge-response pair with the response bits of r derived from $\langle c \rangle$
d	Number of CRPs stored in the server
$FHD(a, b)$	Fractional Hamming distance between two binary strings a and b of the same length
τ	Fractional Hamming distance threshold
N_{CRP}	Number of CRPs needed for training a SPUF model
$SPUF^G$	Genuine strong PUF enrolled in the server
$SPUF^F$	Fake strong PUF not enrolled in the server
$PRNG(a)$	Pseudo random number generator with seed a
$TRNG(a)$	True random number generator that outputs a random bits
$a \parallel b$	Concatenation of binary strings a and b
$a \oplus b$	Bitwise XOR of binary strings a and b of the same length
T_w	Waiting time
T_{os}	Dynamic component of T_w and P_{os} is its binary form
T_{min}	Static component of T_w and P_{min} is its binary form
σ_{noise}	Standard deviation of environmental noise
Q	Counter's output
Z	Comparator's output
S	SPUF ^G or SPUF ^F selection signal

Fig. 1. Basic PUF-based unilateral authentication protocol.

ID needs not be kept secret, and can be stored in an on-chip one-time programmable (OTP) memory. Upon device deployment, for each authentication process, a challenge, c , is sent to the device i , and a response, $\tilde{r}$, generated by a SPUF, is returned to the server. Authentication fails if the HD between the enrolled response r and the device response $\tilde{r}$ exceeds an acceptable noise threshold τ computed based on the reliability of the SPUF. The authenticated CRP is discarded after use to avoid a further authentication event from being replayed by an adversary. The protocol is assumed to be server-initiated typically for two reasons. Firstly, as the server is a master that serves multiple slave devices, this can avoid a denial-of-service (DoS) attack from preventing the master to serve other slave devices. Secondly, the server is assumed to be resource rich. The same protocol that is server-initiated can be easily adapted to device-initiated if necessary.

B. Linear Additive APUF Model

An APUF [24], as shown in Fig. 2, is one of the most widely studied SPUFs used in the abovementioned authentication protocol. It consists of two parallel n -stage multiplexer (MUX) chains that feed into an arbiter stage to produce one response bit from an n -bit challenge, $c_0, c_1, \dots, c_{n-1}$.

It has been shown that an APUF can be modeled by a linear additive model since its response bit to an input challenge can

¹ d should be sufficiently large to cater for all the authentication events throughout the service life of the device

Fig. 2. A basic arbiter PUF design [24].

be derived by summing the delay difference in each stage. This model has been used by several ML methods, *e.g.*, [13]–[16], to successfully attack practical delay-based SPUFs that are composed of linearly cascaded bit-slice circuits. Based on the additive delay model [6], the final delay difference between the top and bottom paths of the APUF, denoted by $\Delta(n)$, can be represented by

$$\Delta(n) = \mathbf{P} \cdot \boldsymbol{\omega}^T \quad (1)$$

where $\mathbf{P} = (p_0, p_1, \dots, p_n)$ is a parity vector and $\boldsymbol{\omega} = (\omega_1, \omega_2, \dots, \omega_{n+1})$ is a weight vector.

The parity $p_k \in \{-1, 1\}$ is defined by:

$$p_k = \prod_{i=k+1}^n 1 - 2 \cdot c_i \quad (2)$$

The elements of the weight vector $\boldsymbol{\omega}$ are given by:

$$\begin{aligned} \omega_1 &= \alpha_1, \\ \omega_i &= \alpha_i + \beta_{i-1}, \text{ for } 2 \leq i \leq n, \\ \omega_{n+1} &= \beta_n \end{aligned} \quad (3)$$

In (3), α_i and β_i can be calculated by

$$\alpha_i = \frac{p_i - q_i - r_i + s_i}{2} \quad (4)$$

$$\beta_i = \frac{p_i - q_i + r_i - s_i}{2} \quad (5)$$

where p_i, q_i, r_i and s_i denote the delays of signal passing through the upper straight, upper cross, lower straight and lower cross routes of the multiplexer switch, respectively, as shown in the bottom right subfigure of Fig. 2.

The elements of $\boldsymbol{\omega}$ are dependent on the multiplexer switching and routing delays. They are susceptible to manufacturing process variations even if the switch of each APUF stage is identically designed. The elements of $\mathbf{P}$, on the other hand, depends only on the challenge bits. If the delay difference at the output of the final stage, $\Delta(n)$, is greater than 0, the response bit r is 1; otherwise it is 0. The response bits of such a SPUF design can be predicted by building a software clone using this linear additive delay model. To succeed, a subset of CRPs needs to be collected to learn the weight vector $\boldsymbol{\omega}$ using ML algorithms.

Evolutionary Strategies (ES) is another powerful ML technique that uses primarily mutation and selection to model a SPUF. The main idea is to generate random PUF instances and find the instance that best matches the real PUF model iteratively. The reliability-based CMA-ES attack [15] utilizes repeated measurements to observe the reliability of the response bits. The reliability information is then fed into a fitness function to find the best fit delay parameters. It outperforms

traditional modeling attacks on XOR APUF with a large number of XORed APUFs. It is demonstrated in [16] that even highly obfuscated responses of, *e.g.*, Slender PUF [29] and reverse fuzzy extractor [26], can be attacked using CMA-ES.

In view of these advanced attacks on SPUF-based authentication, we propose a deception protocol that will immensely increase the time and cost of the adversary to successfully model a SPUF, and resist attacks that require repeated challenge-response measurements. Our adversary model assumes that the device's enrollment interface upon deployment and the communication channel of the authentication protocol are publicly accessible. In other words, attacker can freely query the device with selected challenges, and eavesdrop, manipulate and reply protocol messages between the server and the device.

IV. THE PROPOSED DECEPTION PROTOCOL

The problem with typical SPUF based authentication protocol in Fig. 1 is the adversary can unrestrictedly query the device to derive $(c, \tilde{r})$ and use them for training. It is also impractical for the resource-limiting device to keep track of the used challenges or use encryption in the authentication protocol to prevent replay and MITM attacks.

In addressing these problems, we would like to leverage on recent compact authentication protocol concepts [9], [31], [32] but avoid certain pitfalls in these and existing SPUF-based authentication schemes reviewed in Section II.

- First, a cheaper but safer way than using a monotonic counter to generate a device nonce is needed so that the device will never or rarely send a used challenge.
- Second, either the challenge or response of the SPUF should be encoded or obfuscated and decodable without requiring expensive cryptographic primitives at the device side.
- Third, to have the full CRP set at the disposal of the server with a constant size storage but avoid weakening reliability and security of the underlying SPUF after enrollment.
- Finally, resistance against ML attacks should be enhanced without limiting the number and frequency of authentication requests by the genuine server.

With these in mind, we propose an *active* deception technique to prevent model-building attack on SPUF^G by direct brute-force query on device enrollment interface. A timeout mechanism with fake PUF multiplexing technique is incorporated into the challenge-response exchange protocol of Fig. 1 after the enrollment phase. A deception-based mutual authentication protocol is also introduced to enable timeout free authentication by the server who has the real PUF model. In what follows, the device-side circuit architecture, enrollment protocol for model-based authentication, deception-based mutual authentication protocol, and attack scenarios of the proposed deception protocol are elucidated.

A. Device Architecture

The device-side supporting architecture consists of a monolithic encapsulation of essentially a genuine SPUF (SPUF^G),

a fake SPUF (SPUF^F), a PRNG, a binary counter, a register and a comparator, as shown in Fig. 3. The PRNG is typically implemented by a maximum length LFSR. It is used to generate random sub-challenges from a seed challenge. The SPUF^F is used to generate random nonce or fake responses. The counter, register and comparator provide a practically realizable instantiation to turn a machine-learnable SPUF during enrollment prohibitively time-consuming to learn upon deployment.

Fig. 3. Proposed lockdown encapsulation of genuine SPUF for deception-based authentication.

With reference to the device hardware architecture of Fig. 3, when an n -bit input challenge c is applied to the device, m sub-challenges are derived from the PRNG using c as a seed. These sub-challenges are input to either SPUF^G or SPUF^F to generate an m -bit response. The time between two consecutive input challenges to the device can be tracked by a binary counter of length L . For an internal clock generator of period T_{clk} , the elapse time is determined by the product of the counter output Q and T_{clk} . The counter Q is cleared upon power-up reset and whenever a response is sent externally (via clr). The clock input to counter Q is gated by the comparator output Z . The comparator compares the output of the counter Q with the content of the register P and outputs $Z = 1$ if they are not equal, and outputs $Z = 0$ otherwise. The counter Q will count continuously until it is halt by $Z = 0$. The select enable strobe, $S = 1$ upon reset so that the response to an input challenge c is determined by the comparator output Z . The response will be output from SPUF^G if $Z = 0$ and from SPUF^F if $Z = 1$. If $S = 0$, the true response from SPUF^G will always be output. The output of P is the sum of two components, P_{min} and P_{os} . $P_{\text{min}} = 2^k$ is fixed after the enrollment phase and P_{os} is doubled upon the output of every fake response. A simple controller is used to manage the inputs and outputs to the external world when the device is queried.

B. Enrollment

A model-based authentication approach is considered. To make the full CRP space of SPUF^G available at the server without incurring exponential time-space complexity to read and store them, the pre-requisite is that SPUF^G must be learnable with polynomial resources in terms of training data and runtime. The enrollment process is performed once in a secure and controlled environment. The monolithic encapsulation is locked down to prevent the inputs and outputs of

Fig. 4. Enrollment protocol for model-based authentication.

CRPs of SPUF^G and SPUF^F from being accessed without going through the control logic. To enable model-based authentication, the enrollment protocol shown in Fig. 4 is needed to collect the CRPs of SPUF^G for model building after the device is manufactured. During enrollment, a sufficiently large number d_j of CRPs are extracted from SPUF^G of device j to train a model SPUF^G_{model} by machine learning algorithm for later reproduction of any CRPs of SPUF^G with a high accuracy. As the counter Q and the register P are both cleared to zero, and $S = 1$ upon reset, $Z = 0$ throughout the enrollment process. This will stop the counter and allow the response to be output from SPUF^G for each n -bit input challenge. After the authentication verification model SPUF^G_{model} has been successfully built and validated, one bit of register P is forced to stick at logic one permanently ($P_k : S@1$) by an irreversible antifuse or by using an OTP non-volatile memory (NVM) bitcell² to turn on the deception mechanism.

We assume a generic SPUF that has an acceptable response reliability for authentication application. If the raw responses of the SPUF do not meet the reliability expectation, several lightweight reliability enhancement techniques reported in the literature for commonly used SPUFs can be adopted. Examples of two widely adopted techniques with negligible overheads are spatial and temporal majority votings [40], [41]. The following additional measures are carried out during the enrollment phase to minimize the impact of SPUF reliability on the accuracy of SPUF^G_{model}. Firstly the temperature and supply voltage used for collecting the CRPs for model building are well regulated and monitored. Secondly, the responses to the same challenges are collected multiple times and the majority voted CRPs are used for the training. Thirdly, before shorted out the antifuse of P , SPUF^G_{model} is validated against the physical SPUF^G under varying operating conditions to determine the fractional Hamming distance (FHD) threshold τ required for authentication and the number of re-authentication required to reject or recall a device in the field.

Upon deployment, if the same protocol is used by the adversary to query the system, the response will only be output from SPUF^G if the challenge is input after a duration of $T_w = T_{\text{min}} + T_{\text{os}}$ from system reset or from the output of the previous response. The period of T_{clk} to the L -bit counter has been set to approximately $T_{\text{min}}/2^k$, where $k \in [0, L-1]$ is the position of the register output bit that has been forced into the stuck-at-one state after enrollment. Therefore, the range

²OTP based NVM with antifuse has been widely employed, e.g., the secret keys of automotive are stored in antifuse OTP to make them unattainable [38]. Embedded OTP NVM of high levels of security, high yields, low power and excellent reliability in standard CMOS processes are available in Synopsys IP library [39].

Fig. 5. Proposed deception-based mutual authentication protocol flow.

of waiting time T_w to apply the next challenge to SPUF^G can be varied from $T_{\min}$ to $\approx 2^{L-k} \times T_{\min}$ by loading the remaining $L-1$ bits of the register with a non-zero integer. The maximum offset time T_{os} that can be added to $T_{\min}$ ranges from $2^{L-1} \times T_{\text{clk}} \approx \frac{1}{2} T_{\min}$ for $k = L-1$ (i.e., the MSB of the register) to $(2^L - 2) \times T_{\text{clk}} = (2^L - 2) \times T_{\min}$ if $k = 0$ (i.e., the LSB of the register). The waiting time T_w can be changed by the server along with a bilateral authentication. Unlike existing model-based authentication protocol [9], this adaptive T_w allows the genuine server to make use of the direct query interface occasionally, for instance, to verify the SPUF^G of a recalled device or to retrain SPUF^G due to SPUF aging by resetting T_w to $T_{\min}$. Retraining a SPUF model usually takes much lower number of CRPs than training a model from scratch. $T_{\min}$ can be set higher for a SPUF that requires less training CRPs and lower for those requires more training CRPs to minimize the time cost for these exceptional situations that are not related to regular field authentication. Prior to re-deployment of the SPUF in the field, the waiting time T_w for the attackers can be increased by the genuine server by setting T_{os} with a mutual authentication.

C. Mutual Authentication

The genuine server can authenticate the deployed device by using the mutual authentication protocol shown in Fig. 5. The genuine server can concurrently change the waiting time T_w for each successful authentication. This can prevent an adversary from being able to collect enough valid responses by a brute-force query through the enrollment protocol within a practical time limit.

To initiate a bilateral authentication, the server randomly selects an unused half-length challenge c_s and sends it to the device. Upon receiving c_s , the device uses it as a *seed* to its PRNG to generate $n/2$ random sub-challenges $\langle c \rangle$. The device then clears the counter to ensure that $Z = 1$ so that the sub-challenges $\langle c \rangle$ are applied to the fake SPUF (SPUF^F) to produce a half-length challenge c_d to the server. The process is aborted by the server if the device ID is invalid or c_d has been

used. Otherwise, the server concatenates c_s and c_d to complete the full-length challenge $c_s || c_d$. This full-length challenge is used as a *seed* to the same PRNG as the device to generate $3m$ sub-challenges $\langle c \rangle$. These sub-challenges are used to produce three m -bit responses, r_1 , r_2 and r_3 from $\text{SPUF}_{\text{model}}^G$. Two helper data, $h_1 = P_{\text{os}} \oplus r_1$ and $h_3 = r_1 \oplus r_3$, are computed, where $P_{\text{os}} = T_{\text{os}}/T_{\text{clk}}$ is a $(L-1)$ -bit positive integer. If $m \geq L$, the m -bit response is truncated to match the length of P_{os} before the xor operation. h_1 and h_3 are concatenated into $h_1 || h_3$ and sent to the device.

Upon receiving the $(L+m)$ -bit string, using the same sub-challenges $\langle c \rangle$ with $c_s || c_d$ from its PRNG, the device generates three m -bit responses, $\tilde{r}_1$, $\tilde{r}_2$ and $\tilde{r}_3$ by setting $S = 0$ to select SPUF^G for the application of $\langle c \rangle$. Then it computes $\tilde{h}_3 = \tilde{r}_1 \oplus \tilde{r}_3$. If $\tilde{h}_3$ matches the received h_3 within acceptable FHD tolerance, the server is authenticated. The device then recovers P_{os} from the received h_1 by XORing it with $\tilde{r}_1$.

The recovered P_{os} is loaded into the register to change the waiting time to $T_w = T_{\min} + T_{\text{os}}$. The device will acknowledge the successful update by sending $\tilde{h}_2$ to the server, where $\tilde{h}_2 = P_{\text{os}} \oplus \tilde{r}_2$. Otherwise, if the FHD between $\tilde{r}_3$ and r_3 exceeds the acceptable tolerance, the device sets $S = 1$ to generate $\tilde{h}_2 = \text{SPUF}^F((c))$ and sends it to the server. The counter Q is cleared by setting clr upon transmission of $\tilde{h}_2$. The server can verify the authenticity of the device by checking the received $\tilde{h}_2$ against $h_2 = P_{\text{os}} \oplus r_2$. If they are equal within an acceptable FHD tolerance, the device is authenticated successfully. Otherwise, the authentication fails and the process is aborted.

It should be noted that the adversary cannot issue an unseen packet $h_1 || h_3$ that has not been issued by the server to obtain the corresponding response packet $\tilde{h}_2$. Consequently, the uniqueness of every authentication is assured by the fresh challenge. Because r_1 , r_2 and r_3 are generated from the same fresh starting challenge and “locked” to each other by design, replaying either half challenge will not produce new response bits. The random P_{os} helps to conceal r_1 and r_2 , and need not be precisely tracked. Even if $\tilde{r}_1 \neq r_1$ due to the reliability of SPUF^G , as long as the error due to the recovered P_{os} and the response $\tilde{r}_2$ generated by SPUF^G does not cause the FHD between $\tilde{h}_2$ against h_2 to exceed τ , the outcome of the authentication will not be affected. Since we do not limit the number of challenges like [9], the genuine server can authenticate the same device again by this protocol with a fresh challenge before rejecting the device.

D. Deception-based Technique Against Model-building Attack

Although the direct query protocol of Fig. 4 is used only once during device enrollment by the genuine server, this direct query may also be used by the adversary upon device deployment. Fig. 6 shows the device’s action when the enrollment protocol is utilized by the adversary to collect the CRPs for model building.

When the device is queried by a challenge c , the device reads the comparator output Z . If $Z = 0$, the controller will apply the sub-challenges $\langle c \rangle$ derived from c to the genuine SPUF (SPUF^G) to produce a response $\tilde{r}$ to the query. If $Z \neq 0$,

Fig. 6. Deception against brute-force model-building attack.

the response $\tilde{r}$ will be output from the fake SPUF (SPUF^F) by applying $\langle c \rangle$ to it. The device clears the counter Q by setting $\overline{clr} = 0$ and doubles P_{os} upon transmitting $\tilde{r}$.

$Z = 0$ when the time lapse T of the current query from the last output response is at least $T_{\min}$. This frequency of authentication events is perceived to be normal. $Z \neq 0$ when $T = Q \times T_{\text{clk}} < T_w$, where $T_w = P \times T_{\text{clk}}$. Therefore, $Z \neq 0$ signifies that the authentication events are unusually frequent. As each consecutive query within the waiting time T_w will double P_{os} , the waiting time will extend rapidly. Hence, an adversary who uses the enrollment protocol to brute-force attack the SPUF system will receive fake responses from the SPUF^F . Using the incorrect CRPs to train the model will result in either non-convergence or convergence with highly inaccurate prediction results. This will be further demonstrated in Section V. Unknowingly using such an incorrect SPUF model to mount an attack on the target device will easily expose the adversary.

The appropriate value of T_w depends on the use-case. Even with the same number of CRPs required for a successful attack on a SPUF, $T_{\min}$ and T_{os} can be initialized differently according to the application risk. Different applications has different service span and diminishing profitability of a successful attack with time while the device is still in operation. For example, T_w for a disposable near-field communication (NFC) wristband for a month or multiday convention can be shorter than that of a RFID for supply chain tracking. The dynamic penalty component of the waiting time P_{os} is controllable by the genuine server. It helps to quickly contaminate the CRPs collected by the adversary who attempts to model SPUF^G in a practical amount of time. As opposed to alerting the attacker of incorrect attempt by inaction, the interplay between false PUF multiplexing and timeout mechanism exponentially increases the attacker's time and exhausts the attacker's resources to build an incorrect model. It causes the attacker to miss the opportunity cost of attacking another target or changing their attack strategy for a better chance of success. A timely toy example is an electronic wristband used for contact tracing during the pandemic. Its frequency of use rules out authentication protocols against ML attack that limit the number of authentications. Our proposed mutual authentication protocol will detect imposer who wears a counterfeit wristband, as the SPUF clone trained from multiplexed responses of SPUF^G and SPUF^F of a genuine wristband will fail the authentication.

E. Attack Scenarios

1) *Replay or Spoofing Attack*: The easy way of preventing replay attack is to ensure freshness of challenge so that the adversary is unable to produce a new response to any not yet

used challenge. To establish a trust transmission of sensitive data from the sender to the receiver, it is sufficient to ensure that the used CRPs do not allow the adversary to successfully impersonate the recipient (prover) to obtain subsequent sensitive data transmitted from the sender (verifier). As long as the challenge is sent by the verifier and never been reused, then replay attack cannot be used to impersonate the prover. For the mutual authentication protocol of Fig. 5, since the server can keep track of all the used half challenges c_d from the device, the adversary cannot impersonate the device by replaying any used half challenge c_d in response to c_s from the server. Neither can the adversary be able to produce a valid h_3 to an unseen new challenge $c_s || c_d$ or be authenticated as a legitimate server by replaying an eavesdropped $h_1 || h_3$ to a new challenge $c_s || c_d$. Without the genuine device SPUF^G or its model $\text{SPUF}_{\text{model}}^G$, it is impossible for the adversary to modify an intercepted c_s and c_d to match r_3 . Due to the onewayness of SPUF^G , it is also impossible to modify the intercepted $h_1 || h_3$ to obtain new $c_s || c_d$, h_1 , h_3 and h_2 that will pass authentication at both sides. The server will be alerted of any abnormalities by its inability to receive an acceptable $\tilde{h}_2$ in a given time after sending out $h_1 || h_3$. Hence, it is impossible for the adversary to spoof the protocol by the MITM attack.

2) *Strong Knowledge Attack*: In the proposed deception based mutual authentication protocol, neither the real responses, r_1 , r_2 and r_3 , to the challenge $c_s || c_d$ nor the threshold time offset parameter P_{os} are transmitted in clear between the server and the device. Without the genuine PUF or its model, the secrecy of r_1 , r_2 and r_3 will not be compromised by divulging h_1 , h_2 and h_3 . The device authenticates the server by checking h_3 computed from its generated $\tilde{r}_1$ and $\tilde{r}_3$ against h_3 . It is computationally intractable for an adversary to determine a new $c'_s || c'_d$ with the correct r_1 and r_3 that will map to h_3 transmitted by the server without $\text{SPUF}_{\text{model}}^G$. Even if this is possible, the adversary cannot force SPUF^F of the device to produce this specific c'_d from his chosen challenge c'_s .

The server authenticates the device by checking h_2 against $\tilde{h}_2$ returned by the device. The probability of correctly guessing the output of XORing an unknown random binary bitstream and a known binary bitstream of length $L - 1$ is 2^{1-L} . Since the adversary needs to correctly predict P_{os} from h_1 without knowing r_1 and then predict h_2 from P_{os} without knowing r_2 , the probability of an adversary being successfully authenticated by the server is $2^{2(1-L)}$. By making the register length at least 64 bits, this probability is at most 2^{-126} .

3) *Modeling Attack*: The adversary needs to collect enough CRPs from a device to train a PUF model before it can be used to accurately predict any unused CRPs. In the proposed deception protocol, with the k -th bit of P permanently stuck-at-one after enrollment, the adversary can only obtain one real SPUF response after a minimum waiting time of $T_w = (2^k + P_{os}) \times T_{\text{clk}}$. The stuck-at-one bit position k of the register and the counter clock frequency can be fixed at design time. T_{os} can also be initialized so that T_w upon deployment can be made long enough to prevent the adversary from collecting enough valid CRPs within a practical time

span through the direct query protocol. The offset component T_{os} of T_w is not static but changed randomly upon each successful authentication by the genuine server and doubled for every unsuccessful authentication attempt by the adversary. To be able to collect the CRPs from the SPUF^G at the shortest possible time, the adversary will have to first determine T_w by comparing the responses obtained from sending the same challenge at different time intervals apart. If SPUF^F is used to generate the fake responses, when the same challenge is sent consecutively within a short time, all responses are identical subjected to a small probability of error due to the reliability of SPUF^F, which may allure the attacker to progressively extend the waiting interval to apply the same challenge until a different response is obtained. As each trial within the current T_w will double T_{os} , the waiting time will blow up to months or years after just a few incorrect attempts depending on the T_{os} before the first incorrect guess. The adversary can hardly collect enough true CRPs within a practical amount of time to model SPUF^G.

V. RESULTS AND ANALYSES

A. Test Setup for Modeling Attacks

CMA-ES attack is an effective ML based modeling attack against PUF even in a blackbox setting [15], [16]. In particular, reliability-based CMA-ES attack demonstrates higher efficiency at breaking an l -XOR APUF than LR attack, where it is comparatively more efficient when the number of parallel APUFs l is higher. Both CMA-ES and LR attacks are used to evaluate the proposed protocol. We follow the approaches in [13], [15], [16], [20], [42] to perform the attacks on APUF. Specifically, we utilize the most recent published platform in [11] for the experiments in this work. The delay parameters, p_i, q_i, r_i and s_i , of each stage in an APUF design described in Section II are randomly generated using a standard normal distribution $\sim \mathcal{N}(0, 1)$. The experimental results reported in [6] showed that 4.57% of noise were introduced into its response when the temperature was varied from 27°C to 70°C, and the responses were 2.16% noisier when the voltage was deviated from the rated voltage of 1.8V by $\pm 2\%$. To test the impact of noise on the proposed deception protocol, in all our experiments, a random variable $\sim \mathcal{N}(0, \sigma_{noise}^2)$ is inserted into each delay path, $\Delta(n)$, where $\sigma_{noise} = 0.5$ is derived from the practical noise level obtained in [6]. The challenges are generated randomly using Python. All CRPs are equally divided into two sets, a training set for modeling and a testing set for prediction.

The program of [42] is implemented to build an adversarial model for evaluating the effectiveness of false PUF multiplexing against LR attack. In LR based modeling attacks, the PUF is modeled using the response values. When a PUF cannot generate the same response all the time for a given challenge, the PUF response r can be modeled by (6) [21], where the delay difference $\Delta(n)$ is contributed by various sources of noise $\Delta_{noise}(n)$ in addition to the actual delay difference of the PUF $\Delta_{actual}(n)$.

$$\begin{aligned} \Delta(n) &= \Delta_{actual}(n) + \Delta_{noise}(n) = \mathbf{P} \cdot \boldsymbol{\omega}^T + \Delta_{noise}(n) \\ r &= \begin{cases} 1, & \text{if } \Delta(n) > 0 \\ 0, & \text{if } \Delta(n) < 0 \end{cases} \end{aligned} \quad (6)$$

This is used by the reliability-based CMA-ES attack [15] to find the best fit delay difference, $\boldsymbol{\omega}$, at each stage in (3). To compute the reliability H_i , the same challenge $\mathbf{c}_i$ is sent to the PUF t times. H_i is 1 if the t responses, $r_{i1}, r_{i2}, \dots, r_{it}$, are all matched and 0 otherwise. The reliabilities of the responses to n different challenges are grouped into $\mathbf{H} = \{H_1, H_2, \dots, H_n\}$. A group of p hypothetical reliabilities, $H_{i1} \dots H_{ip}$, for the challenge $\mathbf{c}_i$ is then computed by testing all p possible absolute delay differences, $|\mathbf{P} \cdot \boldsymbol{\omega}^T|$, generated by the CMA-ES algorithm. The fitness metric f_i used to evaluate the quality of the candidate solution $\tilde{\boldsymbol{\omega}}$ is computed by the Pearson correlation coefficient between $\mathbf{H}_i$ and $\mathbf{H}_{ij}$. However, this reliability-based CMA-ES attack does not help to accelerate or improve the accuracy of attacking our proposed deception protocol. It is difficult to determine the actual reliability $\mathbf{H}$ with the fake responses or a mixture of real and fake responses. As it requires applying the same challenge multiple times to obtain the true response reliability data for each challenge, more time is required to collect the CRPs even if the static T_{min} of T_w is known by the attacker. T_w has a dynamic component T_{os} that is changeable by the server. Due to its multiplying effect to penalize wrong prediction, if the attacker is misled by using the same waiting time predicted upfront to collect the training data, the repeated challenges will be worse off as most data collected after the first incorrect waiting time are likely to be fake. For this reason, only the original CMA-ES attack will be considered. Its fitness function is given by (7).

$$f_i = \max_{j=1, \dots, p} \left\{ \rho(\mathbf{R}_i, \tilde{\mathbf{R}}_{ij}) \right\} \quad (7)$$

where $\mathbf{R}_i$ is the response obtained from the training set and $\tilde{\mathbf{R}}_{ij}$ is the response generated by the CMA-ES algorithm. The higher the correlation coefficient between $\mathbf{R}_i$ and $\tilde{\mathbf{R}}_{ij}$, the larger the f_i . The fitness metric f used to evaluate the best candidate $\tilde{\boldsymbol{\omega}}$ is derived by adding up all f_i for all the CRPs. The larger the value of f , the more accurate the PUF model.

A common adversarial model [15] is used for the evaluation. The program used to mount the CMA-ES attack is executed in MATLAB R2016b, by adopting the code for the core CMA-ES algorithm from [43]. For SPUF design such as APUF, one m -bit response is derived from m n -bit sub-challenges generated from a PRNG to form one CRP, where m is the bit length of a response and n is the bit length of a challenge or the number of stages of an APUF. $k \times m$ n -bit sub-challenges are used to produce k m -bit responses. It should be noted that an XOR-APUF can be made more difficult to model if a different n -bit challenge is used for each of its parallel APUFs [15].

B. Effect of False PUF Multiplexing on Modeling Attack Results

The fake response generator is implemented by a SPUF design with a different circuit architecture from the genuine SPUF to produce vastly different CRPs. A TRNG [44] is

implemented to generate random responses for comparison. The resistance of the deception protocol, $\text{SPUF}^G + \text{SPUF}^F$ are evaluated by both LR and CMA-ES attacks and compared against that of $\text{SPUF}^G + \text{TRNG}$ in this section.

1) Percentage of Valid/Invalid Responses on LR Attack:

Two conventional APUFs, one for SPUF^G and the other for SPUF^F , are employed to produce the CRPs for training. Three different sizes of CRP sets, $N_{CRP} = 640$ and $N_{CRP} = 24,000$ (similar to [13], [42]) are used for training. Depending on the percentage of fake information, x_f ($x_f \in \{0, 100\}$), a group of mixed response bits from both SPUF^G and $\text{SPUF}^F/\text{TRNG}$ is derived and used for the LR attack.

Fig. 7. Comparison of LR attack results for the proposed deception protocol utilizing either a TRNG or a XOR PUF as fake response generator. The y-axis shows the achieved correct prediction rate P_{pred} of the LR attacks based on different percentages of fake information mixed with the training responses.

Fig. 7 depicts the LR attack results of the proposed deception protocol by using either SPUF^F or TRNG as its fake response generator. A varying number of fake CRPs is produced according to x_f . The prediction rate of a modeling attack is calculated by $P_{pred} = \frac{N_{correct}}{N_{total}} \times 100\%$, where $N_{correct}$ and N_{total} are the number of correctly predicted response bits and the total number of response bits, respectively. For random guess of a binary variable, the correct prediction of zero and one should be equally probable. Hence the worst result is $P_{pred} = 50\%$. It can be seen that P_{pred} decreases proportionally with the percentage of fake responses x_f in the training samples. For the same x_f , the prediction rate is only slightly higher for a larger number of training samples, e.g., 24,000, than a smaller number, e.g., 640. For the same number of training CRPs and the same percentage of fake responses, the prediction rates are the same for both fake APUF and TRNG. The relationship between the percentage of fake responses and the correct prediction rate by the LR attack for the training set of 24,000 CRPs, obtained by a curve fitting function, is

$$P_{pred} = a \cdot e^{b \cdot x_f} + c \cdot e^{d \cdot x_f} \quad (8)$$

where $a = 0.115$, $b = -3.472$, $c = 0.8784$, $d = -0.3351$.

2) Percentage of Valid/Invalid Responses on CMA-ES Attack: The CRPs ($N_{CRP} = 4,000$) used for training consist of a mixed combination of responses collected from the real PUF and fake PUF/TRNG designs, depending on the percentage of

fake responses. The proposed deception protocol is evaluated for two different challenge bit lengths, 64 and 128 bits.

Fig. 8. The CMA-ES attack results for the proposed deception protocol by applying different challenge bit lengths, 64-bit and 128-bit, as well as utilizing different fake response generators (TRNG and XOR PUF). The number of training samples used for this experiment, $N_{CRP} = 4,000$, is the same as that used in [15].

Fig. 8 shows the CMA-ES attack results on the proposed deception protocol for different percentages of responses from the fake PUFs or TRNGs with different challenge bit lengths. The prediction rates for both lengths of challenges decrease proportionally with x_f from approximately 100% with all real responses to approximately 50% with 90% of responses generated from the fake PUFs/TRNGs in the 4,000 training samples. It can be seen that P_{pred} decreases proportionally with the percentage of fake responses x_f in the training samples. Moreover, there is no appreciable difference in the prediction rates of the CMA-ES attack by using either a fake APUF or a TRNG to generate the fake responses for the proposed protocol.

From the above experiments, the percentage of fake responses has a greater impact on the prediction rate of both CMA-ES and LR attacks. Since the fake APUF or TRNG may generate the same response as that of the real APUF, we also evaluate the prediction rate for a given percentage of erroneous CRPs, i.e., a given percentage of errors occurred in the responses of training samples. Fig. 9 illustrates the impact on the prediction rate of LR attack by using varying percentages of erroneous responses (without adding extra noise) of the real APUF for training. The erroneous responses for real APUF is generated by flipping its response bits. If a fake APUF is obtained from a real APUF with errors, it is more difficult for the attackers to predict the real APUF response using a LR attack. The results show that correct prediction rate under this scenario drops rapidly to about 50% with slightly more than 50% bit error rate.

C. Effect of Dynamic and Static Components of Waiting Time on Modeling Attack

For every authentication attempt made by the adversary before the waiting time T_w , the offset time T_{os} will be doubled. Hence, the waiting time after applying N_f challenges with waiting time shorter than the current T_w is given by:

$$T_w = (P_{min} + P_{os} + 2P_{os} + \dots + 2^{N_f}P_{os}) T_{clk} \\ = \{2^k + P_{os}(2^{N_f} - 1)\} T_{clk} \quad (9)$$

Fig. 9. The LR attack results for the proposed deception protocol in mixing different fake information, including responses from fake APUF, TRNG and real APUF with error injection.

For $T_{clk} = 1ms$, $L = 32$, $k = 21$ and $P_{os} = 1000$, T_{min} is less than an hour. After only $N_f = 25$, the waiting time T_w exceeds 388 days. If P_{os} is updated by the server to 2^{25} , it takes only $N_f = 10$ adversarial attempts for the waiting time T_w to exceed 397 days. As analyzed in Section IV-E, the more the attacker attempts to close in their estimate of T_w , the faster T_w will blow up to years. The attacker can derive a full set of valid training samples by always waiting for a time longer than T_w before sending the next challenge, provided that the genuine server did not authenticate the device to change the P_{os} during this period. If the attacker sends a challenge every t , where $t > T_w$, the minimum overall time, T , to obtain the training set can be calculated by (10), which includes the time to derive all training samples, N_{CRP} , the training time for modeling the PUF, T_{train} , and the trial-and-error time for determining the T_w , $T_{t\&e}$.

$$T = N_{CRP} \times T_w + T_{train} + T_{t\&e} \quad (10)$$

Even if we omit $T_{t\&e}$ and assume that T_w is constant, the best attack time T without considering the delay penalty of T_{os} can still be made impractically long by varying T_w according to the number of training samples N_{CRP} required to successfully model the chosen genuine SPUF. To simulate the optimistic attack time for this scenario in subsequent experiments of this section, instead of fixing T_{min} and changing T_{os} to achieve a given T_w , we set $T_{os} = 0$ and change T_{min} to avoid the multiplying delay penalty for incorrect waiting time. Under this premise, Fig. 10 shows that the efficiency of modeling attacks on existing unfortified ($T_w = 0$ hr) 64-stage APUF reduces with different T_{min} . For example, a 64-stage APUF design can be predicted with 95% accuracy in approximately 0.01 seconds (the time for sending and reading one CRP is neglected) with a training sample size of $N_{CRP} = 640$ using the above LR based experiment. Then using the proposed deception protocol with a threshold time of 1 hour, it will require 25.2 days for the same N_{CRP} to achieve the same prediction rate. Similarly, the overall attack time, T , is also longer when more training samples, N_{CRP} , are needed if a larger APUF with more number of stages or a more complex XOR-PUF is used. The original attack time increases to 0.13 seconds and the prediction rate hits 99% when the number

of training samples required is increased to $N_{CRP} = 2,555$. By using the same T_{min} of 1 hour with our protocol, without compromising the success rate of prediction, the overall attack time T will increase to 109 days. If T_{min} is set to 24 hours, the attacker will require approximately 7.1 years to collect $N_{CRP} = 2,555$ from a 64-stage APUF to achieve a prediction rate of 99%.

Table II shows the original training time and brute-force attack time with known T_{min} of 1 hour and zero T_{os} . They are computed based on the number of CRPs required by LR/ES modeling attack on n -XOR APUF and Feed-Forward APUF [42], as well as CMA-ES attack on n -XOR APUF [15]. The original training time of modeling attack for a 128 stage Feed-Forward Arbiter PUF is approximately 3.15 hours, but increases to almost $\frac{50,000}{24 \times 365} = 5.7$ years by merely delaying the CRP collection with T_{min} of 1 hour. The original training time for the 128 stage 8-XOR APUF is 3.3 hours, but the attack time increases to 34.3 years with T_{min} of 1 hour.

Fig. 10. The overall time, T , taken by modeling attacks to predict a 64-bit APUF with respect to the minimum number of training samples N_{CRP} and static T_{min} of the proposed deception protocol. The typical training time, T_{train} , is assumed to be 1 second.

VI. HARDWARE IMPLEMENTATION

To demonstrate the proposed deception protocol using FPGA, it is important to choose an FPGA based APUF design that has a high uniqueness and reliability. To this end, the lightweight Flip-flop based Arbiter PUF (FF-APUF) design [45] is adopted since it has a higher uniqueness ($\sim 40\%$) compared to the conventional APUF ($\sim 9\%$) on Xilinx Artix-7 FPGA implementation. Moreover, a 64-stage FF-APUF achieves good reliabilities of 97.10% and 93.90% over a temperature range of $0^\circ C \sim 70^\circ C$ and $\pm 10\%$ voltage variations, respectively. The uniformity of the FF-APUF design is $\sim 47\%$. It uses 44 slices, 130 look up tables (LUTs) and 172 flip flops (FFs).

From the analysis of Section V-B, the fake response generator implemented by another SPUF produces no significant difference in ML attack results as a TRNG fake response generator. Since uniqueness is not a major concern for the fake response generator, a 64-bit APUF design is considered here so that there is no difference in the number of cycles required to generate a response bit from the application of an input challenge between the real and fake SPUFs. This will prevent the attacker from exploiting the timing difference to discriminate the responses between SPUF^F and SPUF^G. The

Table II

A List of Heuristic Modeling Attack Results for n -XOR APUF and Feed-Forward APUF. (Time in Bracket Refers to Attack Time in Years with $T_{\min} = 1$ hour and $T_{\text{os}} = 0$.)

PUF type	Evaluation	No. of stages	No. of XORs, n									
			1	4	5	6	7	8	9	10	16	32
n -XOR APUF (LR) [42]	Train time	64	0.13 sec. (0.3 yrs)	3.5 mins (1.3 yrs)	2.1 hrs (9.1 yrs)	1.29 days (23.0 yrs)	-	-	-	-	-	-
		128	0.5 sec. (0.64 yrs)	2.6 hrs (2.8 yrs)	16.3 hrs (57.2 yrs)	-	-	-	-	-	-	-
	No. of CRPs	64	2555	12,000	80,000	200,000	-	-	-	-	-	-
		128	5,570	24,000	500,000	-	-	-	-	-	-	-
n -XOR APUF (CMAES) [15]	Train time	128	0.9 hrs (2.3 yrs)	1.8 hrs (17.1 yrs)	-	-	-	3.3 hrs (34.3 yrs)	-	-	30.5 hrs (57.1 yrs)	60 hrs (228 yrs)
	No. of CRPs	128	20,000	150,000	-	-	-	300,000	-	-	500,000	2,000,000
Feed-Forward APUF(ES) [42]	Train time	64	-	-	-	7.51 mins (5.7 yrs)	47.07 mins (5.7 yrs)	47.07 mins (5.7 yrs)	47.07 mins (5.7 yrs)	47.07 mins (5.7 yrs)	-	-
		128	-	-	-	3.15 hrs (5.7 yrs)	3.15 hrs (5.7 yrs)	3.15 hrs (5.7 yrs)	3.15 hrs (5.7 yrs)	3.15 hrs (5.7 yrs)	-	-
	No. of CRPs	64	-	-	-	50,000	50,000	50,000	50,000	50,000	-	-
		128	-	-	-	50,000	50,000	50,000	50,000	50,000	-	-

Table III

Hardware Resources of 64-bit FF-APUF, APUF and Ancillary Components on Xilinx Artix-7 FPGA for the Proposed Deception Protocol.

Type	FF-APUF	LFSR	APUF(TRNG)**	Counter	UART	Others*	Total	Percentage (%)
LUTs	130	65	134/(164)	28	51	170	578	0.91%
FFs	172	67	2/(139)	63	54	426	784	0.62%
Clk Cycles	192	132	132/(1)	6	-	-	324	-
Slices	44	25	134/(58)	21	23	141	388	2.45%

*Controller + Comparator + 64-bit Register + glue logic.

**One of two options, APUF or TRNG, for fake response generation.

The available hardware resources on Xilinx Artix-7 FPGA: LUT(63,400), FF(126,800), slice(15,850).

APUF consumes only 134 slices, 134 LUTs and 2 FFs. Fig. 11 shows the floorplan of the hardware component implementation for the proposed deception protocol on a Xilinx Artix-7 FPGA. The multiplexers in both FF-APUF and APUF are placed and routed by minimizing the skew between the top and bottom delay paths. To generate a 64-bit response from a 64-bit challenge, a 64-bit maximum length LFSR with a feedback polynomial of $x^{64} + x^{63} + x^{61} + x^{60} + 1$ is used to generate 64 random internal challenges with the input challenge as seed. The number of slices required to implement this LFSR is 25.

Fig. 11. The floorplan of implementing the proposed deception protocol components on Xilinx Artix-7 FPGA.

Table III shows the hardware resource consumption of the

device with 64-bit CRPs for the proposed deception protocol in terms of the numbers of LUTs, registers, clock cycles and slices on the Xilinx Artix-7 FPGA. The clock frequency is set to 100 MHz. The propagation delays of FF-APUF and APUF can both be completed well within one clock cycle, *i.e.*, 10 ns. Three clock cycles are consumed to load a 64-bit challenge, generate a 1-bit response and store it into a register. To generate a 64-bit response, 3×64 clock cycles are required. FF-APUF occupies much fewer configurable fabrics of Xilinx Artix-7 FPGA than Slender PUF [29] and System-of-PUF [46] designs, which use 128 LUTs and 130 LUTs, respectively, for generating a 1-bit response. In total, 388 slices are consumed, which is only 2.45% of the resources of Xilinx Artix-7 FPGA. From Table III, the total time taken from authentication request to response generation by the device is $10 \text{ ns} \times 264 = 2,640 \text{ ns} = 2.64 \mu\text{s}$, which is fast enough for most authentication protocols. The speed bottleneck of authentication event is due to the data rate used for the serial communication between the server and the device. For our prototype experiment, this is limited by the UART with a baud rate of 115,200 bit per second for communication with the PC through RS232. It takes the device approximately $694 \mu\text{s} \times 2 = 1,388 \mu\text{s}$ to receive a 64-bit challenge and transmit a 64-bit response, including the start and stop bits required for the transmission of each byte. The total time for one authentication event is thus $2.64 \mu\text{s} + 1,388 \mu\text{s} = 1,390.64 \mu\text{s}$. This can be substantially reduced by using faster serial link with much higher baud rate. The total power consumption of the proposed protocol implemented on Xilinx Artix-7 FPGA is 101 mW.

Since TRNG can also be utilized for fake response generation, for comparison, we have implemented the coherent sampling ring oscillator based TRNG (COSO-TRNG) from among the low-cost TRNGs suitable for FPGA implementation recently surveyed in [47]. As shown in Table III, this TRNG consumes 164 LUTs and 139 FFs, which are more than the 134 LUTs and 2 FFs of APUF.

Fig. 12 compares the prediction rate and attack time by LR attack on using 64- and 256-stage FF-APUFs against conventional APUF of similar hardware complexity. For the same number of stages, FF-APUF requires significantly more number of training data to achieve the same prediction rate. Fig. 13 shows the attack time T for using FF-APUF as real

Fig. 12. A comparison of LR attack on both conventional APUF and FF-APUF designs. ‘BitLen’ represents the number of stages of the real PUF.

Fig. 13. The overall attack time, T , taken by modeling attacks to predict FF-APUF designs with respect to the minimum number of training samples N_{CRP} and static T_{min} of the proposed deception protocol.

APUF in our proposed deception protocol with different T_{min} . Compared with using conventional APUF for the same fixed $T_w = T_{min}$ with $T_{os} = 0$ in Fig. 10, T is comparable, if not slightly higher, for the same number of training samples N_{CRP} . Although the training time for each sample increases only negligibly, N_{CRP} increases significantly for the same prediction accuracy. Hence, the overall attack time increases substantially by replacing the conventional APUF with equally lightweight FF-APUF as SPUG^G in the proposed deception protocol.

The prediction rate of a 64-bit FF-APUF model trained with 18,050 CRPs during enrollment is 99.5%, as shown in Fig. 12. The worst reliability of the FF-APUF implemented on Xilinx Artix-7 FPGA is 93.9% at 10% underrated core voltage [36]. Since the average bit error rate due to the accuracy of FF-APUF model is negligibly small (0.5%) compared with the worst reliability of the physical FF-APUF in the field, the fraction of response bit differences between the model and the real FF-APUF will be kept below 7% for the device operating at temperature between 0°C and 70°C with no more than $\pm 10\%$ rated voltage variations. This is confirmed experimentally by the 94.17% prediction rate of the 64-bit FF-APUF model tested with 6% errors injected into the CRPs generated by the device PUF. Hence, the FHD tolerance τ can be conservatively set to 0.1 to reduce the false rejection rate to virtually zero. The barrier to modeling attack is still high as the model built by the attacker must have at least 90% accurate prediction rate in order for its response to be authenticated successfully.

Table IV
A Comparison of Proposed Deception Protocol against Lightweight Protocol Finalists from [23] and Two Recent Published Protocols, Lockdown Protocol from [9] and Latent Obfuscation [32].

Protocol	PUF-independent	Modeling resistance	Number of auth.	HW implement	TRNG
Basic authentication [5]	✓	×	d	×	×
Slender PUF [29]	×	×	∞	✓	×
Noise bifur. [30]	×	×	∞	✓	×
Lockdown Ia [9]	✓	✓	d	✓	×
Lockdown Ib [9]	✓	✓	d	✓	×
Lockdown IIa [9]	✓	✓	d	✓	×
Lockdown IIb [9]	✓	✓	d	✓	×
Latent obfus. [32]	×	✓	d	✓	×
System-of-PUF [46]	×	×	d	×	×
Proposed protocol	✓	✓	∞	✓	×

The symbol $\times$ denotes ‘no’. The symbol $\checkmark$ denotes ‘yes’.

The symbol ∞ denotes the number of authentications is not preset but limited only by the CRP space of the SPUF.

The symbol d denotes the number of authentications has a prespecified limit.

The symbol $\sim$ denotes result not available.

VII. PROTOCOL COMPARISON

The survey in [23] divides the PUF-based authentication protocols into two groups, *heavyweight* and *lightweight*. Most of the *heavyweight* protocols require either a strong cryptographic algorithm for privacy amplification and an ECC for response reconciliation [24]–[28]. For example, the *controlled PUF* [48] applies hashing to obfuscate its CRPs and requires ECC to correct the noisy responses. The ECC helper data has to be securely stored in order to prevent helper data manipulation attack [49]. A detailed review and comparison of the *heavyweight* authentication protocols have already been done in [23]. In this work, we focus on the *lightweight* group. Three *lightweight* protocols, including the *slender PUF* [29], *noise bifurcation* [30] and *system-of-PUF* protocols [46], as well as two recently published authentication protocols, *lockdown* [9] and *latent obfuscation* are considered.

Table IV compares the proposed deception protocol against the basic authentication protocol and five *lightweight* authentication protocols. The evaluation metrics used in the comparison are elaborated as follows:

- *PUF-independent* relates to whether or not the protocol requires a specific PUF design.
- *Modeling resistance* refers to the protocol assisted resistance to model building attacks.
- *Number of auth.* refers to the quantity of CRPs that can be authenticated.
- *HW implement* evaluates the physical feasibility to implement these protocols on an FPGA or other hardware devices.
- *TRNG* refers to any TRNG component used in these protocols.

As analyzed previously, the proposed deception protocol demonstrates good robustness to different ML based attacks. Any type of SPUF can be used as the genuine SPUF or fake SPUF. Among the previous works, only *lockdown* protocols have no restriction on the PUF design. However, its implemen-

tation cost is not reported and its model-based authentication is designed based on system-level instantiation of hard-to-learn XOR-PUF instead of generic machine learnable SPUF.

Except *lockdown*, *latent obfuscation* and our proposed deception protocols, other authentication protocols listed in Table IV are vulnerable to modeling attacks. Typically, a *TRNG* is used to generate a random substring to hide the response, e.g., in slender PUF [29]. In our work, a temporal control is used to delay the attacker from collecting enough correct CRPs from the real SPUF within a practical time. Overall, the proposed deception protocol is the only authentication protocol that has achieved all the desirable metrics. It is the only scheme that allows practical realization of secure model-based authentication with generic machine-learnable SPUF. The number of authentications is limited only by the CRP space of the SPUF and the attacker is time-restricted in making consecutive queries to collect the unused CRPs.

VIII. CONCLUSION

Security solutions today focus mostly on blocking attacks. Deception as a defense strategy provides greater delay, confusion and disruption than rejecting sessions to the attacker's onslaught. It can drive preventive countermeasures to delay an attack, causing the adversary economic harm to figure out what is real and what is not, and hesitant to proceed. In this paper, we propose a novel deception authentication protocol by deceiving the adversary to use a training set dominated by fake/invalid responses for ML. This will prevent their PUF clone from correctly predicting the response to the unknown challenge. The proposed deception-based challenge-response interface works with a real SPUF, a fake SPUF, a counter, a register and a comparator to make modeling of the real SPUF easy during enrollment but infeasible upon device deployment by delaying the collection of the required number of correct training data by an unpredictable and impractically long time. Attempts to shorten this data collection time will increase the number of fake CRPs used for training. Every incorrect attempt to guess the waiting time will be met by multiplying time penalty. The rapid drop of prediction accuracy with increasing fraction of fake responses is demonstrated using two of the most widely known modeling attack techniques, LR and CMA-ES. The enrolled software model of real SPUF is used by the server to perform authentication requests through the proposed mutual authentication protocol. Each successful mutual authentication initiated by the server can also alter the dynamic component of the waiting time required by the adversary to query the real SPUF consecutively. The protocol has been analyzed to be secure against replay and MITM attacks. The device side components required to support the deception protocol are implemented on a Xilinx Artix-7 FPGA to validate its hardware efficiency. The entire authentication system for a prover consumes only 1.12% of LUTs and 0.62% of FFs. 388 slices are consumed overall, representing 2.45% of the resources of a Xilinx Artix-7 FPGA.

ACKNOWLEDGMENTS

This work was partly supported by the Singapore MOE Tier 1 Grant No. MOE2018-T1-001-131 RG87/18(S), by the

Engineering and Physical Sciences Research Council (EPSRC) (EP/N508664-CSIT2), the National Natural Science Foundation China (61871216 and 62022041) and the Nature Science Foundation of Jiangsu Province (BK20151477).

REFERENCES

- [1] KrebsSecurity. DDoS on dyn impacts twitter, spotify, reddit. Accessed: 08-11-2016. [Online]. Available: <https://krebsonsecurity.com/2016/10/ddos-on-dyn-impacts-twitter-spotify-reddit/>
- [2] C. H. Chang, Y. Zheng, and L. Zhang, "A retrospective and a look forward: Fifteen years of physical unclonable function advancement," *IEEE Circuits Syst. Mag.*, vol. 17, no. 3, pp. 32–62, Aug. 2017.
- [3] C. Gu, J. Murphy, and M. O'Neill, "A unique and robust single slice FPGA identification generator," in *Proc. Int. Symp. on Circuits and Syst. (ISCAS'14)*. Melbourne, Australia: IEEE, Jun. 2014, pp. 1223–1226.
- [4] C. Gu and M. O'Neill, "Ultra-compact and robust FPGA-based PUF identification generator," in *Proc. Int. Symp. on Circuits and Syst. (ISCAS'15)*. Lisbon, Portugal: IEEE, May 2015, pp. 934–937.
- [5] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Sci.*, vol. 297, no. 5589, pp. 2026–2030, 2002.
- [6] D. Lim, J. W. Lee, B. Gassend, G. E. Suh, M. Van Dijk, and S. Devadas, "Extracting secret keys from integrated circuits," *IEEE Trans. VLSI Syst.*, vol. 13, no. 10, pp. 1200–1205, 2005.
- [7] B. Gassend, D. Lim, D. Clarke, M. van Dijk, and S. Devadas, "Identification and authentication of integrated circuits: Research articles," *Concurr. Comput. : Pract. Exper.*, vol. 16, no. 11, pp. 1077–1098, Sep. 2004.
- [8] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure PUFs," in *Proc. Int. Conf. on Comput.-Aided Des. (ICCAD'08)*. San Jose, CA, USA: IEEE/ACM, Nov. 2008, pp. 670–673.
- [9] M.-D. Yu, M. Hiller, J. Delvaux, R. Sowell, S. Devadas, and I. Verbauwhede, "A lockdown technique to prevent machine learning on PUFs for lightweight authentication," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 2, no. 3, pp. 146–159, 2016.
- [10] C. Gu, C. H. Chang, W. Liu, S. Yu, Q. Ma, and M. O'Neill, "A modeling attack resistant deception technique for securing PUF based authentication," in *Proc. Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, Dec 2019, pp. 1–6.
- [11] P. H. Nguyen, D. P. Sahoo, C. Jin, K. Mahmood, U. Rührmair, and M. van Dijk, "The interpose PUF: Secure PUF design against state-of-the-art machine learning attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2019, no. 4, pp. 243–290, Aug. 2019.
- [12] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, *FPGA Intrinsic PUFs and Their Use for IP Protection*, P. Paillier and I. Verbauwhede, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.
- [13] U. Rührmair, J. Sölter, F. Sehnke, X. Xu, A. Mahmoud, V. Stoyanova, G. Dror, J. Schmidhuber, W. Burleson, and S. Devadas, "PUF modeling attacks on simulated and silicon data," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 11, pp. 1876–1891, 2013.
- [14] J. Tobisch and G. T. Becker, "On the scaling of machine learning attacks on PUFs with application to noise bifurcation," in *Proc. Int. Workshop on Radio Frequency Identification Security and Privacy Issues*. Springer, 2015, pp. 17–31.
- [15] G. T. Becker, "The gap between promise and reality: On the insecurity of xor arbiter PUFs," in *Cryptographic Hardware and Embedded Syst. (CHES'15)*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 535–555.
- [16] —, "On the pitfalls of using arbiter-PUFs as building blocks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 8, pp. 1295–1307, 2015.
- [17] 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 *Proc. 23rd Asia and South Pacific Des. Automation Conf. (ASP-DAC'18)*, Jeju Island, Korea, Jan. 2018, pp. 97–104.
- [18] S. Tajik, E. Dietz, S. Frohmann, J.-P. Seifert, D. Nedospasov, C. Helfmeier, C. Boit, and H. Dittich, "Physical characterization of arbiter PUFs," in *Proc. Int. Workshop on Cryptographic Hardware and Embedded Syst. (CHES'14)*. Springer, 2014, pp. 493–509.
- [19] U. Rührmair, X. Xu, J. Sölter, A. Mahmoud, M. Majzoobi, F. Koushanfar, and W. Burleson, "Efficient power and timing side channels for physical unclonable functions," in *Proc. Int. Workshop on Cryptographic Hardware and Embedded Syst. (CHES'14)*. Springer, 2014, pp. 476–492.

- [20] G. T. Becker, R. Kumar *et al.*, "Active and passive side-channel attacks on delay based PUF designs," *IACR Cryptology ePrint Archive*, vol. 2014, p. 287, 2014.
- [21] J. Delvaux and I. Verbauwhede, "Side channel modeling attacks on 65nm arbiter PUFs exploiting CMOS device noise," in *Proc. Int. Conf. on Hardware-Oriented Security and Trust (HOST'13)*. IEEE, 2013, pp. 137–142.
- [22] —, "Fault injection modeling attacks on 65 nm arbiter and RO sum PUFs via environmental changes," *IEEE Trans. Circuits Syst. I*, vol. 61, no. 6, pp. 1701–1713, 2014.
- [23] J. Delvaux, R. Peeters, D. Gu, and I. Verbauwhede, "A survey on lightweight entity authentication with strong PUFs," *ACM Comput. Surv.*, vol. 48, no. 2, pp. 26:1–26:42, Oct. 2015.
- [24] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. 9th Int. Conf. on Comput. and Commun. Security (CCS'02)*. ACM, 2002, pp. 148–160.
- [25] S. Katzenbeisser, Ü. Koçabaş, V. van der Leest, A.-R. Sadeghi, G.-J. Schrijen, H. Schröder, and C. Wachsmann, "Recyclable PUFs: Logically reconfigurable PUFs," in *Cryptographic Hardware and Embedded Syst. (CHES'11)*, B. Preneel and T. Takagi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 374–389.
- [26] A. Van Herreweghe, S. Katzenbeisser, R. Maes, R. Peeters, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann, "Reverse fuzzy extractors: Enabling lightweight mutual authentication for PUF-enabled RFIDs," in *Financial Cryptography and Data Security*, A. D. Keromytis, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 374–389.
- [27] Ü. Koçabaş, A. Peter, S. Katzenbeisser, and A.-R. Sadeghi, "Converse PUF-based authentication," in *Proc. Int. Conf. on Trust and Trustworthy Computing*. Springer, 2012, pp. 142–158.
- [28] Y. Jin, W. Xin, H. Sun, and Z. Chen, "PUF-based RFID authentication protocol against secret key leakage," in *Proc. Asia-Pacific Web Conference*. Springer, 2012, pp. 318–329.
- [29] M. Majzoobi, M. Rostami, F. Koushanfar, D. S. Wallach, and S. Devadas, "Slender PUF Protocol: A Lightweight, Robust, and Secure Authentication by Substring Matching," in *Proc. IEEE Symposium on Security and Privacy Workshops*, May 2012, pp. 33–44.
- [30] M. D. Yu, D. M'Raihi, I. Verbauwhede, and S. Devadas, "A noise bifurcation architecture for linear additive physical functions," in *Proc. IEEE Int. Symp. on Hardware-Oriented Security and Trust (HOST'14)*, May 2014, pp. 124–129.
- [31] U. Rührmair, C. Jaeger, M. Bator, M. Stutzmann, P. Lugli, and G. Csaba, "Applications of high-capacity crossbar memories in cryptography," *IEEE Trans. Nanotechnol.*, vol. 10, no. 3, pp. 489–498, May 2011.
- [32] Y. Gao, S. F. Al-Sarawi, D. Abbott, A.-R. Sadeghi, and D. C. Ranasinghe, "Modeling attack resilient reconfigurable latent obfuscation technique for PUF based lightweight authentication," *arXiv preprint arXiv:1706.06232*, 2017.
- [33] J. Delvaux, "Attacks on the PUF-based authentication protocols YeHL16 and GaoMAAR17," *Cryptology ePrint Archive*, Report 2017/1134, 2017, <https://eprint.iacr.org/2017/1134>.
- [34] M. Barbareschi, A. De Benedictis, E. La Montagna, A. Mazzeo, and N. Mazzocca, "A puf-based mutual authentication scheme for cloud-edges iot systems," *Future Generation Computer Systems*, vol. 101, pp. 246–261, 2019.
- [35] M. Adeli and N. Bagheri, "Cryptanalysis of two recently proposed puf based authentication protocols for iot: Phemap and salted phemap," *IACR Cryptol. ePrint Arch.*, vol. 2019, p. 1461, 2019.
- [36] D. P. Sahoo, D. Mukhopadhyay, R. S. Chakraborty, and P. H. Nguyen, "A multiplexer-based arbiter puf composition with enhanced reliability and security," *IEEE Transactions on Computers*, vol. 67, no. 3, pp. 403–417, 2018.
- [37] J. Shi, Y. Lu, and J. Zhang, "Approximation attacks on strong pufs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2138–2151, 2020.
- [38] P. Piacentini, "The black box in auto vehicles," Jun. 2017. [Online]. Available: <https://semiengineering.com/black-box-auto-vehicles/>
- [39] Synopsys, "DesignWare OTP NVM IP," Jan. 2019. [Online]. Available: https://www.synopsys.com/dw/ipdir.php?ds=nvm_otp
- [40] P. Koerberl, J. Li, and W. Wu, "A spatial majority voting technique to reduce error rate of physically unclonable functions," in *Trusted Systems*, R. Bloem and P. Lipp, Eds. Cham: Springer International Publishing, 2013, pp. 36–52.
- [41] A. Mills, S. Vyas, M. Patterson, C. Sabotta, P. Jones, and J. Zambreno, "Design and evaluation of a delay-based fpga physically unclonable function," in *2012 IEEE 30th International Conference on Computer Design (ICCD)*, Sep. 2012, pp. 143–146.
- [42] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proc. 17th Conf. on Comput. and Commun. Security (CCS'10)*. ACM, Oct. 2010, pp. 237–249.
- [43] N. Hansen, "The CMA evolution strategy: A tutorial," *arXiv preprint arXiv:1604.00772*, 2016.
- [44] F. Mei, L. Zhang, C. Gu, Y. Cao, C. Wang, and W. Liu, "A highly flexible lightweight and high speed true random number generator on FPGA," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, July 2018, pp. 399–404.
- [45] C. Gu, Y. Cui, N. Hanley, and M. O'Neill, "Novel lightweight FF-APUF design for FPGA," in *Proc. 29th Int. Conf. on System-on-Chip (SOCC'16)*. Seattle, WA, USA: IEEE, Sep. 2016, pp. 75–80.
- [46] S. C. Konigsmark, L. K. Hwang, D. Chen, and M. D. Wong, "System-of-PUFs: multilevel security for embedded systems," in *Proc. Int. Conf. on Hardware/Software Codesign and Syst. Synthesis (CODES+ISSS)*. IEEE, 2014, pp. 1–10.
- [47] O. Petura, U. Mureddu, N. Bochard, V. Fischer, and L. Bossuet, "A survey of AIS-20/31 compliant TRNG cores suitable for FPGA devices," in *Proc. 26th Int. Conf. on Field Programmable Logic and Appl. (FPL'16)*. IEEE, Aug. 2016, pp. 1–10.
- [48] B. Gassend, M. V. Dijk, D. Clarke, E. Torlak, S. Devadas, and P. Tuyls, "Controlled physical random functions and applications," *ACM Transactions on Information and System Security*, vol. 10, no. 4, p. 3, 2008.
- [49] J. Delvaux, D. Gu, D. Schellekens, and I. Verbauwhede, "Helper data algorithms for puf-based key generation: Overview and analysis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 6, pp. 889–902, June 2015.

Chongyan Gu (S'14–M'16) received the Ph.D. degree from Queen's University Belfast, Belfast, U.K., in 2016. She is currently a Lecturer (Assistant Professor) in the School of EEECS at Queen's University Belfast, and a member of Center for Secure Information Technologies (CSIT) with in the Institute of Electronics Communications and Information Technologies (ECIT). Dr. Gu is an expert in hardware security. Her research into physical unclonable function (PUF) has been utilised as part of a security architecture for electronic vehicle (EV) charging systems, licensed by LG-CNS, South Korea, and was also licensed for evaluation by Thales, U.K.. Her team was the overall winner of INVENT 2015, a competition to accelerate the commercialisation of innovative ideas. She has co-authored two research book chapters on the topics of "Lightweight Cryptographic Identity Solutions for the Internet of Things" published by IET in 2016 and "Approximate Computing and Its Application to Hardware Security" published by IET in 2016 and Springer in 2018, respectively. She has successfully organised two special session conferences (IEEE APCCAS in 2018 and ACM GLSVLSI in 2020). She was invited to give tutorial/talks to international conferences, such as, IEEE ASP-DAC 2020 on the topic of practical PUF design on FPGA. Her current research interests include physical unclonable functions (PUFs), security in/for approximate computing, true random number generator (TRNGs), hardware Trojan detection and machine learning attacks.

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) of Singapore in 1993 and 1998, respectively. He joined the School of Electrical and Electronic Engineering (EEE) of NTU in 1999, where he is currently an Associate Professor. He held joint appointments with the university as Assistant Chair of Alumni of the School of EEE 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 five books, published 13 book chapters, more than 100 international journal papers (70 are IEEE) and more than 180 refereed international conference papers (mostly in IEEE), and delivered over 40 colloquia. His current research interests include hardware security and trustworthy computing, low-power and fault-tolerant computing, residue number systems, and application-specific digital signal processing algorithms and architectures.

Dr. Chang currently serves as the Senior Area Editor of IEEE Transactions on Information Forensic and Security, and Associate Editor of the IEEE Transactions on Circuits and Systems-I and IEEE Transactions on Very Large Scale Integration (VLSI) Systems. He also served as the Associate Editor of the IEEE Transactions on Information Forensic and Security and IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems from 2016 to 2019, IEEE Access from 2013 to 2019, IEEE Transactions on Circuits and Systems-I from 2010 to 2013, Integration, the VLSI Journal from 2013 to 2015, Springer Journal of Hardware and System Security from 2016 to 2020 and Microelectronics Journal from 2014 to 2020. He guest edited several journal special issues and served in the organizing and technical program committees of more than 60 international conferences (mostly IEEE). He is an IEEE Fellow, IET Fellow, and 2018-2019 IEEE Circuits and Systems Society Distinguished Lecturer.

Weiqiang Liu (M'12–SM'15) received the B.Sc. degree in Information Engineering from Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China and the Ph.D. degree in Electronic Engineering from the Queen's University Belfast (QUB), Belfast, UK, in 2006 and 2012, respectively. In Dec. 2013, he joined the College of Electronic and Information Engineering, NUAA, where he is currently a Professor and the Vice Dean of the college. He has published one research book by Artech House and over 100 leading journal and

conference papers. His paper was selected as the Feature Paper of IEEE TC in the 2017 December issue. He has two Best Paper Candidates in IEEE ISCAS 2011 and ACM GLSVLSI 2015. He has been awarded the prestigious Outstanding Young Scholar Award by National Natural Science Foundation of China in 2020. He serves as the Associate Editors for IEEE Transactions on Circuits and System I: Regular Papers (2020.1-2021.12), IEEE Transactions on Emerging Topics in Computing (2019.5-2021.4) and IEEE Transactions on Computers (2015.5-2019.4), an Steering Committee Member of IEEE Transactions on Multi-Scale Computing Systems (2018.1-2019.12). He is the program co-chair of IEEE ARITH 2020, and also technical program committee members for ARITH, DATE, ASAP, ISCAS, ASP-DAC, ISVLSI, GLSVLSI, SiPS, NANOARCH, AICAS and ICONIP. He is a member of CASCOM and VSA Technical Committee of IEEE Circuits and Systems Society. His research interests include approximate computing, hardware security and VLSI design for digital signal processing and cryptography.

Shichao Yu received the B.S degree in electrical and information engineering from Hangzhou Normal University, Hangzhou, China, in 2014 and his M.S degree in electronic circuit and system from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2017. He is currently pursuing the Ph.D. degree in Electrical and Electronic with Queen's University Belfast. His research interests mainly include secure hardware architecture, hardware Trojan detection and its software/hardware implementation.

Yale Wang received the B.S degree in Automation from Luoyang Institute of Science and Technology, Luoyang, China, in 2013 and his M.S degree in Information Engineering from Henan University of Science and Technology, Luoyang, China, in 2016. He is currently pursuing the Ph.D. degree in Electrical and Information Engineering at Nanjing University of Aeronautics and Astronautics. His research interests mainly include physical unclonable functions (PUFs), machine learning based modeling attacks on PUFs and countermeasures.

Máire O'Neill (M'03–SM'11) is currently Director of the UK Research Institute in Secure Hardware and Embedded Systems (RISE). She is Chair of Information Security and is Research Director of Data Security Systems at the Centre for Secure Information Technologies (CSIT), Queen's University Belfast. She also leads the EU H2020 SAFECrypto (Secure architectures for Future Emerging Cryptography) project (www.safecrypto.eu). She previously held an EPSRC Leadership Fellowship (2008-2014) and was a former holder of a Royal Academy of

Engineering research fellowship (2003-2008). She has received numerous awards for her research work which include a 2014 Royal Academy of Engineering Silver Medal and British Female Inventor of the Year 2007. She has authored two research books and has over 130 peer-reviewed conference and journal publications. She is an Associate Editor for IEEE TC and IEEE TETC and is an IEEE Circuits and Systems for Communications Technical committee member. She is a member of the Royal Irish Academy and a Fellow of the Irish Academy of Engineering. She is also a senior member of the IEEE and a member of the IET and IACR. Her research interests include hardware cryptographic architectures, lightweight cryptography, side channel analysis, physical unclonable functions, post-quantum cryptography and quantum-dot cellular automata circuit design.