Obfuscated Challenge-Response: A Secure Lightweight Authentication Mechanism for PUF-Based Pervasive Devices

Yansong Gao*†, Gefei Li†, Hua Ma†, Said F. Al-Sarawi*, Omid Kavehei‡, Derek Abbott* and Damith C. Ranasinghe†
*School of Electrical and Electronic Engineering, The University of Adelaide, SA 5005, Australia
{yansong.gao, said.alsarawi, derek.abbott}@adelaide.edu.au
†Auto-ID Labs, School of Computer Science, The University of Adelaide, SA 5005, Australia
gefei.li@student.adelaide.edu.au, {mary.ma, damith.ranasinghe}@adelaide.edu.au
‡School of Electrical and Computer Engineering, Royal Melbourne Institute of Technology, Victoria 3001, Australia
omid.kavehei@rmit.edu.au

***Abstract*—Low cost pervasive devices such as RFID (radio-frequency identification) tags and sensor nodes are increasingly becoming part of the fabric of life. Using these pervasive devices to store and collect data securely is becoming a challenge because stringent requirements on power and area constrain the implementation of standard cryptographic mechanisms used for providing essential security services such as authentication. In this paper, we propose a secure and lightweight authentication protocol for resource scarce pervasive devices built upon a physical unclonable function (PUF) primitive termed Obfuscated PUF (OB-PUF). We develop a variant of a parameter-based authentication protocol. This protocol sends obfuscating challenges to an OB-PUF, and guarantees the subsequent recovery of the obfuscated challenges by a server (verifier). In particular, our approach exploits server (verifer) aided computations to reduce the hardware complexity on the pervasive device while still maintaining a high level security, and takes advantage of the known vulnerability of PUFs to model building attacks. Most importantly, the unclonability of the OB-PUF and, consequently is reserved, OB-PUF based pervasive devices are resilient to cloning. We also show, through statistical analysis and implementing model building attacks, that constructing a model of the OB-PUF by an adversary is infeasible when our proposed challenge obfuscation method is employed.**

***Index Terms*—Physical Uncloanble Function, Pervasive device, Obfuscation, Model building attacks, Hardware security, Authentication.**

I. Introduction

Nowadays, computing is trending towards lower area-cost and power-cost devices such as wireless sensor nodes, RFID transponders, and smart dust. These secure and light computational platforms are the foundation for building the next generation pervasive and ubiquitous networks, often exemplified by the IoT (Internet-of-Things). Applications built upon such low cost devices range from monitoring bushfire to battlefields as well as securing pharmaceutical supply chains to prevent the trade in counterfeit medications. These applications rely on low cost pervasive devices to collect, store and send sensitive information, through wireless networks. Providing fundamental security services, such as authentication, for such devices is a significant challenge, especially taking into the requirements of small footprint and power consumption of these resource-constraint pervasive devices.

Providing authentication can be achieved by simply storing a secret (private key) on the device as a trust anchor. However, it has been shown that stored keys can be cloned by using, for example, physical attacks or side channel attacks [1], [2]. Further, implementation of standard cryptographic mechanisms is extremely challenging because stringent requirements on power and area limits their use in practice. For example, even minimalistic implementations of standard asymmetric key primitives such as AES (Advanced Encryption Standard) providing fast and low computational cost implementations in hardware used or hash functions such as SHA-1 (Secure Hash Algorithm) take thousands of gates and require thousands of clock cycles for operation [3], [4]. More importantly, they can not completely protect pervasive devices against being cloned [5]. Although tamper-resistance mechanisms can be implemented to prevent different types of attacks from extracting the stored keys, these mechanisms are too expensive to be applied to such cost-sensitive pervasive devices [6].

An emerging class of hardware security primitive—Physical Unclonable Functions (PUF)—offers a promising low cost solution for authenticating pervasive devices that cannot be physically cloned. Note that PUFs, especially *silicon or circuit based* PUFs, offer a simple alternative to acting as a 'finger-print' of a hardware device with a small hardware footprint. A silicon based PUF extracts secrets from a complex physical function by taking advantage of uncontrollable process variations resulting from fabrication processes. Therefore, a PUF is easy to build but practically impossible to duplicate even by the same manufacturer owing to the inevitable process variations. When a PUF is stimulated by a challenge (input), $\mathbf{C}$, a corresponding response (output), $\mathbf{R}$, will be generated and determined by $f(\mathbf{R})$, where $f(\cdot)$ is a physical function that is unique to each device. Given the same challenge, $\mathbf{C}$, different PUF instantiations built upon the same design will present a different response, $\mathbf{R}$. The challenge, $\mathbf{C}$, and its corresponding response, $\mathbf{R}$, are commonly referred to as a Challenge Response Pair (CRP). A set of CRPs can be used to identify/authenticate a PUF and hence the PUF integrated object, eg. ICs. Therefore, it is highly favorable that a PUF has an exponential number of CRPs as in the Arbiter PUF (APUF) [7], [8] and k-sum PUF [9].

Conventional PUF-based authentication—*CRP-based authentication*—exploits a challenge response protocol [8], [10]. In such a protocol, before the PUF is transferred to a prover, a CRP database of a PUF is established by the server (verifier) through measurements during *enrollment phase*. It is estimated that such an authentication mechanism can be implemented by using less than 1000 gates [5], [11]. With practically very large numbers of CPRs available, each pair is never used more than once to avoid replay attacks. This essentially serves as a low cost one-time pad for pervasive devices yielding a provably secure system where secret keys cannot be cloned and complex cryptographic operations are not required.

However, if an adversary can eavesdrop on CRPs or have access to a physical device for a short period to measure CRPs, it has already been shown that such a conventional CRP-based authentication protocol is vulnerable to model building attacks [7], [12]. It is shown that an adversary armed with only thousands of CRPs from, for example, a highly desirable silicon PUF architecture with exponential CRPs such as an APUF or a k-sum PUF built on linear additive blocks, and with access to a typical modern laptop computer can build a model of a PUF in less than one minute [13]. To increase the complexity of the task faced by an adversary to perform a model building attack, several solutions are proposed in the literature. One solution is to add nonlinearity, such as XOR-ing the responses of several PUF instantiations to generate a single bit response. Another solution is to alter the PUF architecture, for example, as in the Feed Forward Arbiter PUFs [7], [14] and lightweight secure PUFs [15]. Unfortunately, increasing the complexity of model building attacks through the integration of more non-linear elements also significantly reduces the reliability of the PUF [7], [12]. In addition, Such solutions only provide a modest protection on PUFs against model building attacks [12], [16].

Unlike the previous approaches, two recent proposals [17], [18] have demonstrated an innovative alternative that effectively hides the direct relationship between $\mathbf{C}$ and $\mathbf{R}$ from an adversary while still enabling successful authentication of a PUF-based device by a server. In these proposals, an adversary cannot obtain the exact pair of $\mathbf{C}$ and $\mathbf{R}$. However, the server (or verifier) is still able to discover the exact $\mathbf{C}$ and $\mathbf{R}$ pair to successfully authenticate a PUF (or prover). Because the server can take advantage of the vulnerability of the PUF to model building attacks, hence to build a model of the PUF during the enrollment phase to verify the response from the PUF [19], [17], [18]. In general, both of these approaches [17], [18] post-process the response generated on device, i.e. decimation [18] and sub-string padding [17]. Then only the decimated and expanded response is observable to prevent an adversary obtaining the direct relationship between $\mathbf{C}$ and $\mathbf{R}$.

In contrast, we propose a different approach—challenge obfuscation—that is suitable for authentication of low cost pervasive devices. We summarize the contributions from our study below:

- We propose a lightweight authentication protocol—obfuscated challenge-response authentication protocol—suitable for low cost pervasive devices and resilient to model building attacks. Only exposing an obfuscated PUF challenge—*partial challenge*, hence prevent model building attacks. The successful authentication by the server is ensured by first taking advantage of the ability to build a parameter model of a physical PUF expeditiously during the enrollment phase. Therefore, the server has the parameter model and the prover has the physical PUF during the authentication phase, while an adversary is faced with the massively difficult task of recovering the relationship between the obfuscated challenges and responses.
- We propose a method for a device to generate a full challenge from an obfuscated challenge to ensure resilience to model buildings attacks and provide a challenge recovery method for the server to discover the challenge used by the prover. We show that our proposal can be implemented at low cost. Then, we systematically evaluate the obfuscated challenge-response mechanism and propose a solution to reduce degradation in uniqueness (distinguishbility) of OB-PUF instances as a consequence of the multiple possible responses an OB-PUF can generate for a given obfuscated partial challenge.
- We perform statistical analysis to illustrate the exponentially increasing workload that an attacker will face to build a model of an OB-PUF after implementing the proposed obfuscated challenge-response mechanism. Further, we employ previously successful model building attack methodologies to break PUFs to evaluate the enhanced security provided by our approach.

II. Related Work

Over the years, a number of PUF structures have been proposed, built and analyzed. These include *time delay based* PUFs such as the Arbiter PUF [6], [7] (APUF), Feed-Forward APUF [7], and Ring-Oscillator PUF [8] (RO-PUF); *Memory-based* PUFs leveraging device mismatch such as SRAM PUF [20], Latch PUF [21], Flip-flop PUF [22]. A comprehensive review of different PUF architectures can be found in [23], [24]. In recent years, emerging PUFs with nanotechnology initially investigated aim to build PUFs beyond the aforementioned conventional silicon PUFs by taking advantage of prevalent process variations as a consequence of scaling devices and structures down to the nanoscale region, and other unique properties offered in emerging nanodevices [25], [26], [27]. A review of such nanotechnology-based PUFs can be found in [28].

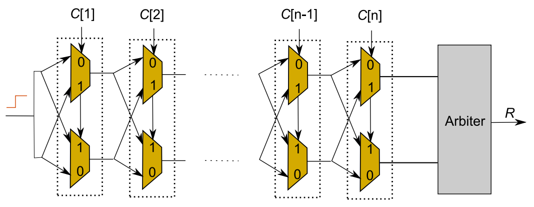

Fig. 1. An arbiter PUF (APUF) circuit.

Examples of PUFs that build upon linear additive blocks include the APUF [6], [29], [8] and the k-sum PUF [9], [24]. These PUFs are favorable, because they offer an exponential number of CRPs. However, they are vulnerable to model building attacks using machine learning techniques. From the model building attack perspective, both architectures have the same topology, therefore in this paper, the APUF is considered to demonstrate the proposed method, nonetheless this does not limit its applications to other PUF structures that can generate a large number of CRPs.

The APUF consists of k stages in sequence, each stage is composed of two 2-input multiplexers as shown in Fig. 1, or any other architectures that have two signal paths. To generate a response bit, a signal is applied to the first stage input, while the challenge $\mathbf{C}$ is used to determine the signal path to the next stage. The two electrical signals simultaneously race through each multiplexer path (top and bottom paths) in parallel. At the end of the APUF architecture, an arbiter, which can be implemented by a latch, can be used to determine whether the top or bottom signal arrives first and hence outputs a logic '0' or '1' accordingly.

It has been shown that an APUF can be modeled via a linear additive model since a response bit is generated based on the summation of each time delay segment in each stage (two, 2-input multiplexers) depending on the challenge $\mathbf{C}$, where $\mathbf{C}$ is made up of $(c_1, c_2, ...c_k)$ [7], [12], [16]. The final delay difference $\triangle$ between the two signals can be expressed as:

$$\triangle = \omega^T \mathbf{\Phi}, \tag{1}$$

where ω and $\mathbf{\Phi}$ are the delay determined vectors and the parity vector, respectively, of dimension $k+1$ as a function of $\mathbf{C}$. We denote $\sigma_i^{1/0}$ as the delay in stage i for the crossed $(c_i = 1)$ and uncrossed $(c_i = 0)$ signal path through the multiplexers, respectively. Hence σ_i^1 is the delay of stage i when $c_i = 1$, while σ_i^0 is the delay of stage i when $c_i = 0$. Then

$$\omega = (\omega^1, \omega^2, ...\omega^k, \omega^{k+1})^T, \tag{2}$$

where $\omega^1 = \frac{\sigma_1^0 - \sigma_1^1}{2}$, $\omega^i = \frac{\sigma_{i-1}^0 + \sigma_{i-1}^1 + \sigma_i^0 - \sigma_i^1}{2}$ for all $i = 2, ..., k$ and $\omega^{k+1} = \frac{\sigma_k^0 + \sigma_k^1}{2}$. Furthermore,

$$\mathbf{\Phi}(\mathbf{C}) = (\mathbf{\Phi^1}(\mathbf{C}), ..., \mathbf{\Phi^k}(\mathbf{C}), \mathbf{1})^{\mathbf{T}}, \tag{3}$$

where $\Phi^j(\mathbf{C}) = \Pi_{i=j}^k (1 - 2c_i)$ for $j = 1, ..., k$.

Here we can see that the ω encodes the delays for the subcomponents in the APUF stages, while the $\mathbf{\Phi}$ is a parity vector as a function of $c_1, ..., c_k$. The delay difference, $\triangle$, is the inner product of ω and $\mathbf{\Phi}$. If $\triangle$ is greater than 0, the response bit is '1', otherwise, the response bit is '0'. Then the task for an adversary is to find an estimate of ω that mimics the actual delay vector ω of a physical PUF structure. Notably this estimate will be based on both the full knowledge of $\mathbf{\Phi}$ and the corresponding response.

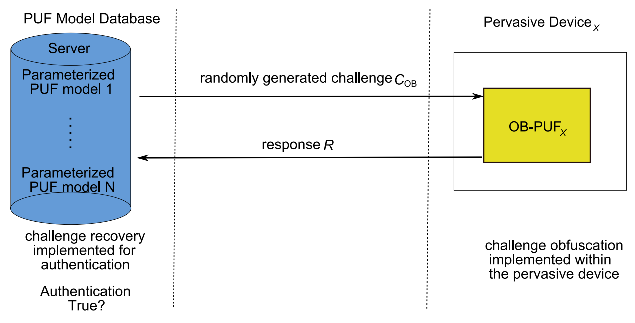
PUF Model Database
Server
Parameterized PUF model 1
Parameterized PUF model N
challenge recovery implemented for authentication
Authentication True?
randomly generated challenge C_{OB}
response R
Pervasive Device$_x$
OB-PUF$_x$
challenge obfuscation implemented within the pervasive device

Fig. 2. Obfuscated challenge-response authentication protocol. Note the server has parameter models of the underlying PUFs used in OB-PUFs.

III. Obfuscated Challenge-Response Authentication Mechanism

Our goal is to obfuscate the direct relationship between challenges and responses from an adversary by revealing only partial challenges or obfuscated challenges. It is the partial challenge, $\mathbf{C}_{\text{OB}}$, that will be applied to an OB-PUF integrated within a pervasive device that generates a response (see Fig. 2). The obfuscated challenge prevents an adversary building a model of the OB-PUF through modeling attacks. However, the server can still successfully authenticate the device by using a challenge recovery method. Hence, the server has *mathematical models* of internal PUFs in the PUF block—see Fig. 3—within the OB-PUF and the device has a physical OB-PUF while the adversary faces a challenge to build a model of OB-PUF. First we describe OB-PUF structure at the heart of the authentication mechanism.

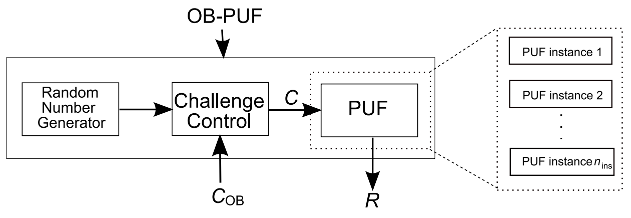

Fig. 3. Ofuscated PUF (OB-PUF) structure. Note $n_{\text{ins}} = 3$ is used for case study in this paper.

A. Obfuscated PUF Primitive: OB-PUF

The Fig. 3 shows the OB-PUF structure. It comprises a random number generator (RNG), challenge control block and the PUF block. In our article we analyse an OB-PUF built upon APUFs. As can be seen, a PUF block consists of a number of APUFs.

The challenge applied to the OB-PUF is the *partial challenge*, $\mathbf{C}_{\text{OB}}$. Here a $\mathbf{C}_{\text{OB}}$, of $k - m$ bits, is applied to the

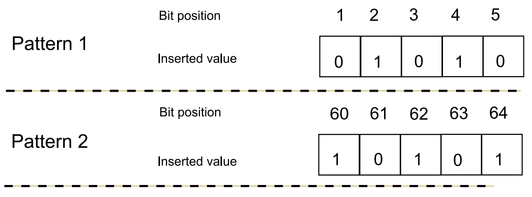

Fig. 4. Example of one latent challenge set. The positions and values that can be inserted into a partial challenge $\mathbf{C}_{\mathrm{OB}}$ of four *Patterns* to produce four full length challenges $\mathbf{C}$s, The *Patterns* are public information. $k = 64, m = 5$.

challenge control block, where k is the length of a full length challenge, $\mathbf{C}$ and m is the length of eliminated bits. The random number generator (RNG) determines which *Pattern*—see the example case shown in Fig. 4—will be chosen to determine the position of the m bits and the values of the m bits that will be inserted into the $\mathbf{C}_{\mathrm{OB}}$ to form a $\mathbf{C}$. Note that in our example case we use two *Patterns* ($p = 2$) to determine two possible full length challenges, $\mathbf{C}$s, corresponding to a $\mathbf{C}_{\mathrm{OB}}$. Subsequently, the challenge control block is incharge of insertion operation to output a full length challenge $\mathbf{C}$ based on $\mathbf{C}_{\mathrm{OB}}$ and the randomly selected *Pattern*. Core idea of OB-PUF is that the *Pattern*, hence a corresponding $\mathbf{C}$, is randomly chosen by a RNG and that the selection is hidden from an adversary as well as the server (verifier).

B. Uniqueness of OB-PUF

In the authentication scheme, only $\mathbf{C}_{\mathrm{OB}}$ and one possible $\mathbf{R}$ are exposed. This can obfuscate the exact relationship between the possible $\mathbf{C}$ and the exposed $\mathbf{R}$. This obfuscation method, however, introduces a large collision probability among responses from different OB-PUF instances given the same $\mathbf{C}_{\mathrm{OB}}$ and degrades our ability to uniquely identify or distinguish OB-PUFs instances.

As a motivating example, firstly, we consider the collision probability in a conventional authentication scheme built upon a PUF primitive using two different PUFs, PUF$_A$ and PUF$_B$, stimulated by the same full length challenge $\mathbf{C}$, the collision probability between $\mathbf{R_A}$ and $\mathbf{R_B}$, here $n_{\mathrm{ins}} = 1$ and hence the response is only 1-bit, is expected to be 50%. Therefore, the server has a 50% probability of differentiating PUF$_A$ from PUF$_B$.

Next, we consider a partial challenge $\mathbf{C}_{\mathrm{OB}}$ and two *Pattern* implementation with $n_{\mathrm{ins}} = 1$. Here, one $\mathbf{C}_{\mathrm{OB}}$ will give two possible responses $\mathbf{R}_{\mathrm{A1}}$ & $\mathbf{R}_{\mathrm{A2}}$ for OB-PUF$_A$ and $\mathbf{R}_{\mathrm{B1}}$ & $\mathbf{R}_{\mathrm{B2}}$ for OB-PUF$_B$, respectively. As for the server, it is able to differentiate OB-PUF$_A$ from OB-PUF$_B$ only if $\{\mathbf{R}_{\mathrm{A1}} = \mathbf{R}_{\mathrm{A2}}\} \neq \{\mathbf{R}_{\mathrm{B1}} = \mathbf{R}_{\mathrm{B2}}\}$. Therefore the probability of being able to distinguish two given OB-PUFs, P_{dist}, is $\frac{1}{2^4}$, as a consequence, the collision probability $P_{\mathrm{collision}}$ between possible responses $\mathbf{R_A}$ and $\mathbf{R_B}$ is $1 - \frac{1}{2^4}$. We can now see that the server faces a challenge in distinguishing between OB-PUF$_A$ and OB-PUF$_B$.

To lower the collision probability or to increase the distinguishability, simultaneous generation of a multiple bit response $\mathbf{R}$ should be deployed. In this case, n_{ins} basic PUF instances should be implemented in parallel within the PUF block shown in Fig. 3. The parallel PUF instances will share the same challenge to generate n_{ins}-bit response $\mathbf{R}$. Then, the relationship between the collision probability and n_{ins} is derived as:

$$P_{\mathrm{collision}} = 1 - (1 - (\frac{1}{2^{n_{\mathrm{ins}}}}))^{2p}, \tag{4}$$

where p is the number of $Pattern$s used. As a consequence, the distinguishability or uniqueness is given as:

$$P_{\mathrm{dist}} = (1 - (\frac{1}{2^{n_{\mathrm{ins}}}}))^{2p}. \tag{5}$$

It can be seen that the higher $P_{\mathrm{collision}}$ introduced by our latent challenge method can be greatly decreased through implementing basic PUF instances in parallel sharing the same challenge. When $n_{\mathrm{ins}} = 3$ and $p = 2$, the uniqueness is 58.62%.

C. Obfuscated Challenge Pattern Control

The CRPs from an APUF have correlations, which means that the hamming distance (HD) between $\mathbf{C_1}$ and $\mathbf{C_2}$ of an APUF has a positive correlations with the HD between $\mathbf{R_1}$ generated by $\mathbf{C_1}$ and $\mathbf{R_2}$ generated by $\mathbf{C_2}$. For binary strings $\mathbf{X}$ and $\mathbf{Y}$ of same length n, the HD between them is defined as:

$$\mathrm{HD}(\mathbf{X}, \mathbf{Y}) = \sum \mathbf{X} \oplus \mathbf{Y}. \tag{6}$$

In the OB-PUF, one $\mathbf{C}_{\mathrm{OB}}$ will give p possible $\mathbf{C}$s. Therefore, there will be p possible $\mathbf{R}$s. The $\mathbf{R}$ exposed is randomly determined by the RNG. If the HDs among p possible $\mathbf{C}$s are small, as a consequence, the HDs among p possible $\mathbf{R}$s has a high chance to be small. In other words, if the HDs among p possible $\mathbf{C}$s are too small, it does not matter which *Pattern* is chosen by RNG, the adversary can simply select one possible $\mathbf{C}$ and the exposed $\mathbf{R}$ to train the model. Because it is reasonable to use $\mathbf{R_1}$ to replace $\mathbf{R_2}$ as the response corresponding to $\mathbf{C_2}$ as long as the HD between $\mathbf{C_1}$ and $\mathbf{C_2}$ are very small.

We can see that having a small HD between possible $\mathbf{C}$s is a security latency that can be utilized by the adversary to carry out model building attacks on the OB-PUF since it is unnecessary for an adversary to guess which *Pattern* is chosen. This issue—if it is not addressed appropriately—will yield the design of OB-PUF vulnerable to model building attacks despite our attempts to obfuscate the relationship between $\mathbf{C}$ and $\mathbf{R}$. To eliminate this flaw, we must maximize the HDs among p possible $\mathbf{C}$s.

To maximize the HDs among p full length challenges, the public known $Pattern$s must be designed according to a set of rules. The two example *Pattern*s listed in Fig. 4, i.e where $p = 2$, is one of the strategies to maximize the minimum HDs among four possible $\mathbf{C}$s given $m = 5$ eliminated bits. Based on our simulation, the two *Pattern*s shown in Fig. 4 can give near 32 bits HD on average among two possible $\mathbf{C}$s given $\mathbf{C}_{\mathrm{OB}}$. Therefore, such *Pattern* design ensures a maximum HDs among two possible $\mathbf{R}$s generated by the same $\mathbf{C}_{\mathrm{OB}}$s.

Moreover, the illustrated *Pattern*s in Fig. 4 are always inserted in the partial challenge $\mathbf{C}_{\mathrm{OB}}$ as a contiguous bit string.

Therefore, in a practical design, *Pattern*s and the received $\mathbf{C}_{\text{OB}}$ string can be directly applied to the PUF block with simple control logic circuits that are implemented in the challenge control block.

D. Challenge Recovery

The server has the task of recovering the full length challenge $\mathbf{C}$ used by the OB-PUF once a response $\mathbf{R}$ is received. During the challenge recovery stage, the server firstly emulates all possible responses corresponding to p possible $\mathbf{C}$s based on publicly known information on $Patterns$. Then the server compares each emulated response $\mathbf{R}_{\text{emu}}$ with the response $\mathbf{R}$ sent from the prover (device). Only the $\mathbf{R}_{\text{emu}}$ generated based on the same $Pattern$ that is used in the OB-PUF device to generate received $\mathbf{R}$ will be same. Otherwise, they will not match if $\mathbf{R}_{\text{emu}}$ and received $\mathbf{R}$ are not generated based on the same $Pattern$ from the same OB-PUF. Therefore, the server has the ability to discover the *Pattern* with the help of the parameter model of OB-PUF.

E. Obfuscated Challenge-Response Authentication Protocol

Now we present the obfuscated challenge-response authentication protocol—a variant of a parameter-based authentication protocol [17], [18]—based on the OB-PUF primitive, and briefly illustrate in Fig. 2, below:

1: A server collects a number of CRPs from the underlying PUF instance in PUF block of OB-PUF and obtain a parameter model of the OB-PUF. Then the extraction of CRPs is destroyed, eg. fuse the extraction wires.

2: Whenever authentication is requested, the server randomly selects and sends a partial challenge $\mathbf{C}_{\text{OB}}$ to the pervasive device which applies $\mathbf{C}_{\text{OB}}$ to the OB-PUF to obtain a response $\mathbf{R}$ (see Fig. 3). The prover sends the obtained $\mathbf{R}$ back to the server.

3: The server attempts to authenticate the device by implementing the challenge recovery method to select a candidate emulated response $\mathbf{R}_{\text{emu}}$ to compare with the $\mathbf{R}$. If the candidate emulated response for the given $\mathbf{C}_{\text{OB}}$ same to the received $\mathbf{R}$, the authenticity of the pervasive device is established; otherwise, the authenticity is rejected.

Notably, a specific partial challenge $\mathbf{C}_{\text{OB}}$ is only used once. Multiple rounds of authentication can be carried out following the aforementioned three steps to ensure both false rejection rate and false acceptance rate to be low.

IV. Analysis

A. Statical Analysis

The number of CRPs needed to train a machine learning model is a function of the prediction accuracy $1-\epsilon$ (ϵ is the prediction error rate of the machine learning model) and the number of stages, k, in an APUF [12], [16], [17]

$$N_{\text{CRP}} = O(\frac{k}{\epsilon}). \tag{7}$$

TABLE I
Prediction Rate Comparison

	XOR2-APUF		OB-PUF	
CRPs ($\times 10^3$)	P_{pred}	Time	P_{pred}	Time
10	71.11%	0:34 min	52.28%	0:11 min
20	84.89%	0:53 min	63.27%	0:23 min
30	92.85%	1:31 min	70.12%	0:35 min
50	96.09%	2:43 min	71.24%	1:03 min
100	97.38%	6:10 min	71.62%	9:57 min
200	97.93%	12:59 min	71.92%	15:32 min
1,000	—	—	71.99%	1:22:32 hrs

In order to impersonate a physical PUF, the model should achieve a prediction rate higher than the reliability of the PUF. To achieve a prediction accuracy of $1-\epsilon$, the adversary needs $N_{\text{CRP},1-\epsilon}$ CRPs for training a model to break a k-stage APUF. However, in our case, the adversary has no knowledge of full length challenges and hence has to guess it based on guessing the *Pattern* that is randomly selected in the OB-PUF. Nevertheless, an adversary may try to achieve the same prediction accuracy of $1-\epsilon$ from machine learning using a challenge-replication strategy [18] in which the adversary substitutes p possible CRPs one by one given a $\mathbf{C}_{\text{OB}}$ to their model. Now, for each possible CRP, the adversary needs to build up a new model based on the previous models. Then the number of models the adversary has to build is [17]:

$$N_{\text{Model}} = p^{N_{\text{CRP},1-\epsilon}}, \tag{8}$$

where N_{Model} is the number of models the adversary needs to build up, and then one of the models will pass the authentication or, in other words, impersonate the underlying physical APUF used in the OB-PUF. In the above equation, it can be seen that by only using $p=2$ *Patterns*, the adversary already faces a monumental challenge, i.e. attack complexity of the order of 2^{640} for an APUF stimulated with a 64 bit challenge, to build a valid model with misclassification lower than 1% [13]. Moreover, the number of *Pattern*s can be increased to an even larger number, which will further increase the computational load of the adversary.

B. Model Building Attack Results

The CRPs used are generated using simulations that can effectively model a physical APUF architecture as in [14], [12], [16].

As for OB-PUF, there are p possible responses for a given $\mathbf{C}_{\text{OB}}$. If the response predicted by the OB-PUF model is same to one of the possible responses from a physical OB-PUF, then the model of an OB-PUF can impersonate the physical OB-PUF. Therefore, P_{pred} of OB-PUF is the probability of the response $\mathbf{R}$ predicted by the model matching one possible response $\mathbf{R}$s from a physical OB-PUF for a given partial challenge $\mathbf{C}_{\text{OB}}$.

We employed LR (logistic regression) machine learning algorithm to build a model, because this algorithm is the most successful algorithm for modeling attacks [13]. The P_{pred} of

OB-PUF is shown in Table. I. As a comparison, we also show that the P_{pred} for XOR2-APUF is 96% using only 50,000 CRPs. In [7], it shows that the worse-case reliability of an APUF is 95.18%. Therefore, the reliability of a XOR2-APUF under worst-case is 90.82%. This indicates that the XOR2-APUF is broken already as the P_{pred} of XOR2-APUF is already 5% higher than its reliability [29].

To keep an OB-PUF stable, all $n\text{ins} = 3$ APUF must generate stable response bits simultaneously. Considering the worst-case 95.18% reliability of an APUF, the reliability of an OB-PUF is $(95.18\%)^3 = 86.24\%$. It can be seen that the P_{pred} is still 14% lower than the reliability, 86.24%, of the OB-PUF under worst-case. Therefore, the OB-PUF cannot be broken even after using more than one million CRPs—as machine learning did not converge when using 1 million CRPs.

V. Conclusion

In this paper, we proposed an obfuscated challenge-response authentication mechanism suitable for authenticating pervasive device at low cost, which is lightweight, secure and cannot be cloned. Such an authentication mechanism can prevent an adversary from exploiting model building attacks to break the authentication mechanism and yet allows the server (or verifier) to successfully authenticate the OB-PUF based pervasive device (or prover). Statistical analysis demonstrates exponentially increased computational load to the adversary. In addition, results of model building attacks tests indicates the enhanced security of the proposed OB-PUF. As a consequence, our results confirm the enhanced security provided by our lightweight authentication mechanism.

VI. Acknowledgment

This research was supported from both Australian Research Council Discovery Program (DP140103448) and Department of State Development-Collaborative Pathway program. Also we acknowledge support from China Scholarship Council (201306070017).

References

[1] R. Anderson, *Security engineering*. John Wiley & Sons, 2008.

[2] O. Kömmerling and M. G. Kuhn, “Design principles for tamper-resistant smartcard processors,” in *USENIX Workshop on Smartcard Technology*, vol. 12, 1999, pp. 9–20.

[3] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, “Strong authentication for RFID systems using the AES algorithm,” in *Cryptographic Hardware and Embedded Systems*-CHES. Springer, 2004, pp. 357–370.

[4] M. Feldhofer and C. Rechberger, “A case against currently used hash functions in RFID protocols,” in *On The Move to Meaningful Internet Systems*. Springer, 2006, pp. 372–381.

[5] S. Devadas, E. Suh, S. Paral, R. Sowell, T. Ziola, and V. Khandelwal, “Design and implementation of PUF-based” unclonable” RFID ICs for anti-counterfeiting and security applications,” in *IEEE International Conference on RFID*. IEEE, 2008, pp. 58–64.

[6] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, “Silicon physical random functions,” in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, 2002, pp. 148–160.

[7] D. Lim, “Extracting secret keys from integrated circuits,” Ph.D. dissertation, Massachusetts Institute of Technology, 2004.

[8] G. E. Suh and S. Devadas, “Physical nclonable functions for device authentication and secret key generation,” in *Proceedings of the 44th Annual Design Automation Conference*. ACM, 2007, pp. 9–14.

[9] M.-D. M. Yu, D. M’ Raihi, R. Sowell, and S. Devadas, “Lightweight and secure PUF key storage using limits of machine learning,” in *Cryptographic Hardware and Embedded Systems*–CHES. Springer, 2011, pp. 358–373.

[10] D. C. Ranasinghe, D. Engels, and P. Cole, “Security and privacy: Modest proposals for low-cost RFID systems,” in *Auto-ID Labs Research Workshop,* Zurich, Switzerland, 2004.

[11] D. C. Ranasinghe, S. Devadas, and P. H. Cole, “A low cost solution to cloning and authentication based on a lightweight primitive,” in *Networked RFID Systems and Lightweight Cryptography*. Springer, 2008, pp. 289–309.

[12] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, “Modeling attacks on physical unclonable functions,” in *Proceedings of the 17th ACM Conference on Computer and Communications Security*, 2010, pp. 237–249.

[13] U. Ruhrmair, J. Solter, 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 Transactions on Information Forensics and Security*, vol. 8, no. 11, pp. 1876–1891, 2013.

[14] M. Majzoobi, F. Koushanfar, and M. Potkonjak, “Testing techniques for hardware security,” in *IEEE International Test Conference* – ITC, 2008, pp. 1–10.

[15] M. Majzoobi, F. Koushanfar, and Potkonjak, “Lightweight secure PUFs,” in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, 2008, pp. 670–673.

[16] U. Ruhrmair and M. Van Dijk, “PUFs in security protocols: Attack models and security evaluations,” in *IEEE Symposium on Security and Privacy (*SP), 2013, pp. 286–300.

[17] M. Rostami, M. Majzoobi, F. Koushanfar, D. S. Wallach, and S. Devadas, “Robust and reverse-engineering resilient PUF authentication and key-exchange by substring matching,” *IEEE Transactions on Emerging Topics in Computing*, vol. 2, no. 1, pp. 37–49, 2014.

[18] M.-D. Yu, D. M’Raihi, I. Verbauwhede, and S. Devadas, “A noise bifurcation architecture for linear additive physical functions,” in *IEEE International Symposium on Hardware-Oriented Security and Trust–(*HOST), 2014, pp. 124–129.

[19] 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 *IEEE Symposium on Security and Privacy Workshops–(*SPW), vol. 6917, 2012, pp. 33–44.

[20] D. E. Holcomb, W. P. Burleson, and K. Fu, “Initial SRAM state as a fingerprint and source of true random numbers for RFID tags,” in *Proceedings of the Conference on RFID Security*, vol. 7, 2007.

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

[22] R. Maes, P. Tuyls, and I. Verbauwhede, “Intrinsic PUFs from flip-flops on reconfigurable devices,” in *3rd Benelux Workshop on Information and System Security (*WISSec), vol. 17, 2008.

[23] M. Roel, “Physically unclonable functions: Constructions, properties and applications,” Ph.D. dissertation, University of KU Leuven, 2012.

[24] C. Herder, M.-D. Yu, F. Koushanfar, and S. Devadas, “Physical unclonable functions and applications: A tutorial,” *Proceedings of IEEE*, vol. 102, pp. 1126–1141, 2014.

[25] Y. Gao, D. C. Ranasinghe, S. F. Al-Sarawi, O. Kavehei, and D. Abbott, “mrPUF: A novel memristive device based physical unclonable function,” in *13th International Conference on Applied Cryptography and Network Security*, 2015.

[26] Y. Gao, D. C. Ranasinghe, S. F. Al-Sarawi, O. Kavehei, and D. Abbott, “Memristive crypto primitive for building highly secure physical unclonable functions,” *Scientific Reports*, vol. 5, art. no. 12785, 2015.

[27] L. Zhang, X. Fong, C.-H. Chang, Z. H. Kong, and K. Roy, “Highly reliable spin-transfer torque magnetic RAM based physical unclonable function with multi-response-bits per cell,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 8, pp. 1630–1642, 2015.

[28] Y. Gao, D. C. Ranasinghe, S. F. Al-Sarawi, O. Kavehei, and D. Abbott, “Emerging physical unclonable functions with nanotechnology,” IEEE Access, 2015, DOI: 10.1109/ACCESS.2015.2503432.

[29] D. Lim, J. W. Lee, B. Gassend, G. E. Suh, M. Van Dijk, and S. Devadas, “Extracting secret keys from integrated circuits,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 10, pp. 1200–1205, 2005.