Journal of Network and Computer Applications 217 (2023) 103693

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

DC-PUF: Machine learning-resistant PUF-based authentication protocol using dependency chain for resource-constraint IoT devices

Abolfazl Sajadi [a,1], Ahmad Shabani [a,1], Bijan Alizadeh [a,b,*]

[a] The Design, Verification and Debugging of Embedded Systems (DVDES) Laboratory, School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, 14399-57131, Iran
[b] The School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, 19538-33511, Iran

ARTICLE INFO

Handling editor: M. Atiquzzaman

Keywords:
Hardware security
PUF
IoT
Authentication
Modeling attack
Machine learning

ABSTRACT

The Physical Unclonable Function (PUF) is a hardware block which exploits the inherent process variations introduced during manufacturing to assign a unique fingerprint for each physical entity. This feature enables a low-cost authentication mechanism for connected devices, which is easy to evaluate but hard to predict. In this paper, we develop a low-cost PUF-based authentication mechanism called DC-PUF for resource-constrained IoT devices which can resist against machine-learning attacks. A new strategy known as the dependency chain (DC) is employed by which the response in each clock is dependent not only on the current challenge but also on the previous responses. This mechanism which takes advantage of both randomization and dependency chain hardens the ability to clone or predict the PUF instance by obfuscating the correlation between challenge-response pairs. Moreover, we propose a CNN-based attack scenario by which the existing PUF structures cannot resist while yielding more than 95% of prediction accuracy. The experimental results signify that unlike the existing PUFs, more than 58% of prediction accuracy cannot be reached when a sophisticated CNN-based attack is conducted to the DC-PUF, even with a large data-set. Also, the False-Negative rate is about 1%, in average, for three-bit mismatches in the receiving response in the presence of channel variations, whereas the False-Positive rate remains zero. Moreover, the proposed DC-PUF incurs lower hardware complexity and reasonable reliability and randomness compared to the existing PUF structures.

1. Introduction

Nowadays, different types of connected devices in modern systems communicate with each other without supervision, and these connections are expanding every day, which makes them the vulnerable targets to violate the privacy and integrity of the systems. Therefore, hardware security has become a major challenge and various types of attack and counteraction method have been introduced. Authentication, integrity, availability, and confidentiality are four fundamental security measurements that should carefully be taken into account by designers and engineers in different stages of the system's life time. However, the stringent limitations in hardware resources and power requirements restrict the hardware implementation of standard key primitives or hashing mechanisms, especially for resource-constraint IoT devices (Shabani and Alizadeh, 2020). This issue exposes a major security challenge in the existing application which employs cheap yet insecure devices. This situation is compounded by the fact that the underlying hardware can be maliciously modified by hardware Trojans which can be inserted by the untrusted entities in different IC design or manufacturing processes (Shabani and Alizadeh, 2021) (Sabri et al., 2021) (Khormizi et al., 2022). As IoT devices are typically limited in power and hardware resources, Physical Unclonable Functions (PUFs) have become very popular for realizing lightweight secure authentication mechanisms (Khormizi et al., 2022) (Trout and Betts, 1988). Unlike the conventional cryptographic primitives, the PUF mechanism no longer needs a pre-shared secret key to be hard-coded in target devices or secure key sharing between two communication parties. Therefore, the inherent vulnerability which originates from key-hacking or spoofing is not present when the PUF mechanism is employed.

The concept of PUF-based authentication which exploits the unique

* Corresponding author. The Design, Verification and Debugging of Embedded Systems (DVDES) Laboratory, School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, 14399-57131, Iran.
E-mail addresses: sajadi@ut.ac.ir (A. Sajadi), ah.shabani@ut.ac.ir (A. Shabani), b.alizadeh@ut.ac.ir (B. Alizadeh).
[1] Both authors contributed equally to this paper.

https://doi.org/10.1016/j.jnca.2023.103693
Received 6 February 2022; Received in revised form 15 May 2023; Accepted 15 June 2023
Available online 11 July 2023

and inherent device characteristics or the impairment of communication channel was the ground-breaking idea in system security. It is already found that the implementation of hash functions such as SHA-1 (Secure Hash Algorithm) takes thousands of gates and require thousands of clock cycles for operation (Maes and Maes, 2013) (Feldhofer et al., 2004), which is not tolerable for most of IoT devices and applications. The PUF mechanism promises a secure and lightweight authentication for resource-constraint devices and eliminates some drawbacks and vulnerability issues such as large footprint, key sharing, or side-channel attacks, which are already present in traditional authentication methods. The silicon or circuit-based PUFs structures offer a simple alternative acting as a lightweight fingerprint of a hardware device (Amsaad et al., 2021). A silicon-based PUF extracts the secrets from a complex physical function by taking advantage of uncontrollable process variations resulting from the fabrication processes (Gao et al., 2016). It benefits from the presence of the process variations to generate the unique responses corresponding to the incoming input challenges called the CRPs (Challenge Response Pairs). Therefore, due to the inevitable process variations, building a PUF is straightforward, but it is notably impossible to replicate even by the same manufacturer. In fact, as the PUF is firmed on the ground of inherent physical characteristics, an ideal PUF is inherently immune to the physical attacks that can be exploited on the underlying hardware. Nevertheless, most of the existing PUFs can be neutralized by simple Machine Learning (ML) algorithms with limited data-set due to the high correlation between its inputs and outputs (Gao et al., 2016) (Ruhrmair et al., 2013). Accordingly, many researches till date have given some idea to camouflage the relation between inputs and outputs of the PUF structures.

Over the last few decades, many studies have focused on the PUF concepts, and several PUF structures were proposed (e.g., Arbiter PUF (Ruhrmair and Solter, 2014) (Gassend et al., 2004), SRAM PUF (Holcomb et al., 2007), RF-PUF (Holcomb et al., 2007) (Chatterjee et al., 2018), and Ring Oscillator (RO) PUFs (Suh and Devadas, 2007)). The existing PUF structures can be categorized into strong (Lee et al., 2004) (Vijayakumar and Kundu, 2015) (Sahoo et al., 2014) and weak PUFs (Majzoobi et al., 2009) (Ashtari et al., 2019) (Sahoo et al., 2014) (Tuyls et al., 2006). Weak PUFs display only a few challenge-response pairs (CRPs) that can be used as a unique key or seed of the traditional encryption systems. Whereas, the strong PUFs (e.g., Arbiter PUF) can offer a large number of unique CRPs, which makes them a suitable candidate for low-cost authentication of devices with limited hardware resources. However, the existing strong PUFs have demonstrated some vulnerability factors to ML attacks, where attackers can exploit a certain number of CRPs from the communication channel to clone the structure of the PUF (Ye et al., 2018). In (Rührmair and Sölter, 2014), the authors examined the most relevant and comprehensive ML algorithms to model the PUF structures and proposed the primary conditions to fulfill the need for the practical attacks. Also, the authors in (Ashtari et al., 2022) employed different ML algorithms for PUF-based authentication and provided the comprehensive recommendations by inspecting the application of different classifiers.

In this paper, we present a new low-cost and scalable PUF structure for authentication of the resource-constrained IoT devices offering a strong resilience and reliability against the ML attacks. The key idea lies in the fact that the correlation between CRPs of a strong PUF can be significantly weakened by increasing the dependency between consecutive responses. That is, the generated response in a specific clock cycle not only depends upon the current challenge, but also on the previous responses while the previous response itself depends upon the previous CRPs. This chain of dependency binds a strong PUF structure, which can stand out against ML attacks while obfuscating the correlation. Therefore, the ability to predict or classify the responses based on the incoming challenges is hardened. In contrast to the previous studies focusing on the PUF structures, where the utmost two or four responses are associated with each challenge, our proposed PUF can associate 2^n responses to each challenge, where n is the number of bits that are related to the previous response. As the number of related bits from the previous response increases, the dependency chain in our PUF structure tightens, albeit at the expense of minor area overhead. In fact, this dependency chain increases the possible candidates for responses while hardening the prediction or replication processes using ML algorithms. In the early days, the simple LR and SVM classifier were able to predict the simple PUF structures. However, as the PUFs are getting stronger and more complicated, the more sophisticated neural networks are required to predict the behavior and clone their functionality. To the best of our knowledge, none of the available attack scenarios using ML algorithms can effectively classify the nonlinear data with a limited data-sets. Accordingly, in this paper, we also propose a deep convolutional neural network (CNN) which is able to deliver higher information from the intermediate layers to the dense layers for classification. This enables us to better predict the responses corresponding to the incoming challenge. The major contributions of this work are as follow:

- Developing a new low-cost and scalable PUF-based authentication protocol called DC-PUF which is able to resist against existing ML attacks by employing the dependency chain.
- Embracing the fact that increasing the dependency between consecutive responses in a PUF structure will lead to weaken the correlation between CRPs while hardening the prediction or cloning process. We called this feature the CRPs' dependency chain. In fact, to predict the response for each challenge, a partial history of the previous responses will be required.
- Presenting a new CNN-based attack scenario that can classify the existing PUF structures with high accuracy though our proposed PUF structure is relatively immune to be cloned or predicted by this attack scenario.

The rest of this paper is organized as follows. Section II provides a detailed background about different PUF structures and their resistance against different types of attacks. Section III introduce the proposed DC-PUF structure, component, and its application in authentication. The experimental results are provided in Section IV, and finally, a comprehensive conclusion will be presented in the last section.

2. Overview of the PUF structures and resiliency against ML attacks

Resistance and reliability to the ML attacks are two major concerns for the overall viability of the PUF-based authentication protocols. In existing PUF structures, a large number of CRPs can be obtained by sharing the path segments between different CRPs. This high correlation present between CRPs can be used by attacker as the training sets of ML method. So, the pre-trained model can predict the expected responses with high accuracy. It is found in (Ruhrmair and Solter, 2014)that the CRP of the *Arbiter PUF* yields more than 99% of prediction accuracy. They achieved a 99% prediction rate by the Logistic Regression (LR (to model the *Arbiter PUF* with a maximum of 39×10^3 CRPs in a few seconds. For *XOR Arbiter PUF,* 99% accuracy is achieved with 500×10^3 CRPs in 19 h. On the other hand, the research presented in (Wen and Lao, 2018) offered a new attack scenario by using the active learning techniques. They showed that the active learning could notably improve the learning efficiency of PUF under attack. By taking advantage of the support vector machine (SVM) method, they were able to use a 64-stage MUX-PUF with 200 CRPs to model the attack. In (Mispan et al., 2018), a challenging permutation is offered to increase the ML attack resistance of Strong PUFs, which can decrease the prediction of *Arbiter-PUF* responses to less than 70%. In (Delvaux, 2019), the authors conducted a detailed study by applying an efficient impersonation attack to five candidate Arbiter-based PUF authentication protocols. Following that, the authors in (Delvaux et al., 2017) analyzed the practicality and security measurement of 21 PUF-based authentication protocols, which

was finally concluded that only six candidates could survive from the impersonation attacks. As a result, the majority of the existing PUF designs are prone to modeling attacks with the aim of ML techniques.

Over the past decade, different PUF structures have been presented, all of which sought to improve the security in the authentication debate. In the following, some of the best PUF models such as *Slender PUF* (Majzoobi et al., 2012), *Poly-PUF* (Konigsmark et al., 2016), *R–PUF* (Ye et al., 2016) and *OB-PUF* (Gao et al., 2016) are examined. Fig. 1 shows the circuit diagram of basic *Arbiter PUF* with *n*-bit challenge $[C_1, C_2, \ldots, C_n]$ and 1-bit response *r*. The transition from challenge to response may propagate in two different paths. The *Arbiter PUF* determines how the incoming challenge propagates through these two pathways using multiplexers until it reaches to the destination flip-flop. If the transition reaches D earlier, the response is 1; otherwise, it will be 0. Since the pathway delays in the multiplexer are affected by the presence of process variations during the manufacturing process, predicting and cloning them before and after manufacturing is too tricky. Generally, there are 2^n CRPs for the basic arbiter PUF while different parts of multiplexers are shared between different CRPs, resulting in a high correlation between CRPS. This issue provides a great opportunity for attackers to perform the modeling attacks, which are able to predict the unknown CRPs from the known ones. In 2012, another PUF model known as the *Slender-PUF* was proposed using the random bits to empower PUF resiliency as shown in Fig. 2 (d). Unlike the *Arbiter-PUF*, where there exists *m* bits of response for each challenge, the response of *slender PUF* incorporates 2*m* bits, leading to hardening the prediction process. However, in 2014, the slender PUF was found to be vulnerable against an ML-based evolution strategy (Becker and Kumar, 2014).

In 2016, another PUF structure called *Poly-PUF* was presented in (Konigsmark et al., 2016) which uses the TRNG unit to generate random bits. The different random bits generated by the TRNG unit are XORed with the incoming challenge and the generated response as shown in Fig. 2(a). Unfortunately, in this strategy, no attention has been paid to the hamming distance (HD) of consecutive challenges. This weakness led to the successful attack in (Delvaux, 2019)with more than 90% accuracy by using a neural network with only single neuron. During the same period, the OBPUF model was presented by (Gao et al., 2016) which incorporates a challenge control unit to inject two different pre-determined challenge patterns into the incoming challenge. According to the TRNG output, one of the 5-bit challenge patterns is randomly injected into the incoming 64-bit challenge, and its results are fed to the parallel arbiter PUFs, as shown in Fig. 2 (b). The main weakness in this protocol is that there exist some cases where the same 3-bit response (nominal value) is generated for different challenge patterns. In fact, the response does not react to the change in the challenge patterns. Accordingly, in 2019 authors in (Delvaux, 2019) extracted all the existing nominal values and used them to train the LR model with 85% of accuracy. In the next step, they achieved more than 90% of accuracy by using the previously trained LR model as the De-obfuscation tool, which gathers more data with low HD in respect to the predicted outputs.

In 2017, a new protocol known as RPUF was presented which includes two operation modes controlled by a TRNG unit as shown in Fig. 2(c). The incoming challenge may be entirely inverted or directly supplied to the linear-feedback shift register (LFSR) block according to the random bit generated by TRNG unit. The LFSR block can help a single arbiter PUF to generate n-bit responses from a single challenge which eliminates the need for duplicating the arbiter PUFs. In order to empower the resistance, the RPUF can use two random bits resulting in four different operation modes. The authors claimed that the RPUF could not be attacked with more than 75% accuracy, although an alternate learning strategy could break down the resistance of this method by around 90% of accuracy in 2019 (Delvaux, 2019). The majority of the methods presented here were trying to minimize the correlation between CRPs as well as camouflaging the hyperplanes by using the different fixed challenge patterns which are randomly selected by some random bits. The history of research reveals that they eventually have failed to resist against the sophisticated neural networks. Nevertheless, our approach follows a different path in lowering the CRPs' correlation by exploiting the temporal correlation between consecutive responses in addition to the randomization of challenge patterns. This strategy enables us to employ a secure and lightweight authentication protocol which can resist even against deep CNN-based attack scenarios.

3. The proposed PUF-based authentication protocol (DC-PUF)

According to the case studies, there is a pressing need to design a strong and lightweight PUF-based authentication protocol resistant to the modeling attacks conducted by the ML algorithms. The main purpose of presenting the current PUF model is to prevent the learning process to predict the correlation between challenges and responses; thus, preventing from an unauthorized malicious node to exploit the identity replication attack. In this way, the attacker fails to clone the legitimate nodes and the authentication session is denied by the server. The proposed PUF-based authentication flow includes two phases of enrollment and deployment. The enrollment phase is performed only once for each legitimate node. In this phase, the server collects a large number of challenge-response pairs corresponds to the target node to train its authentication model. Afterwards, in the deployment mode, the server regenerates the possible responses related to the incoming chal-

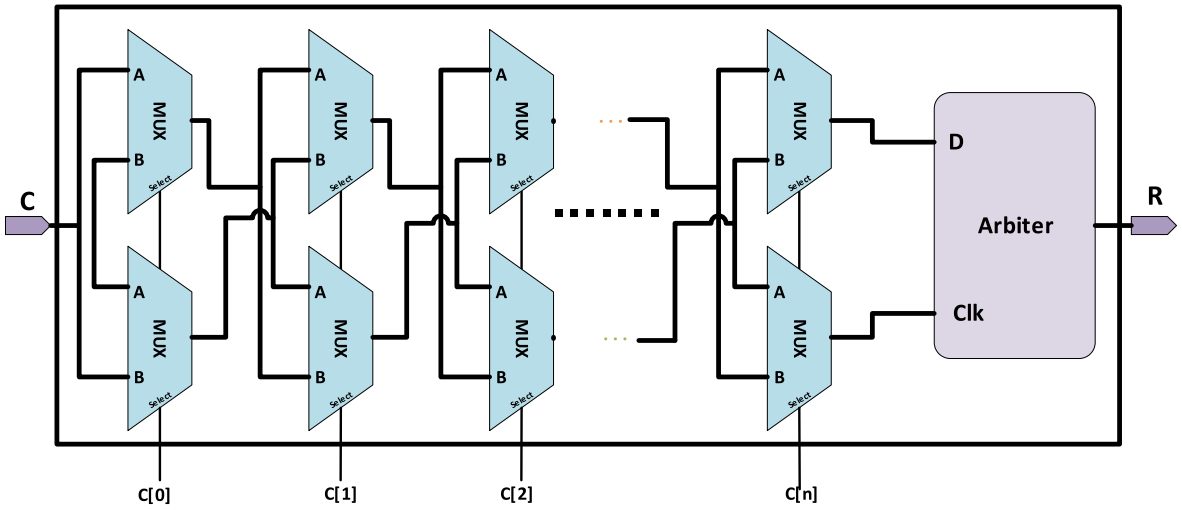


Fig. 1. General structure of basic Arbiter PUF.

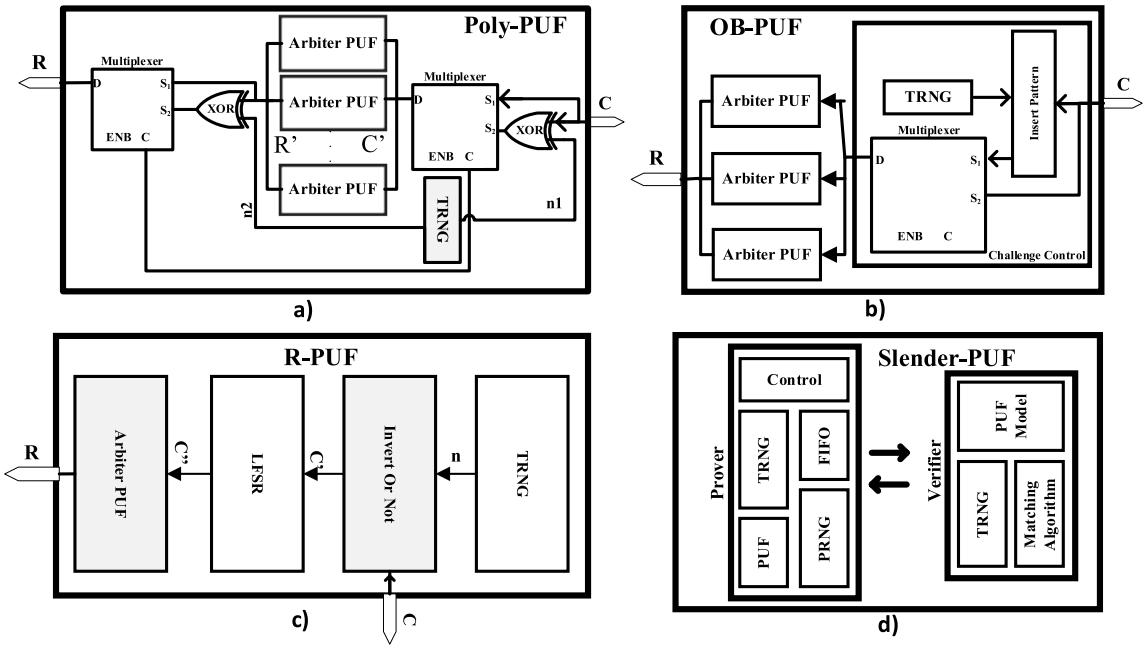


Fig. 2. Architecture of different PUFs; a) Poly-PUF, b) R–PUF, c) OB-PUF, and d) Slender-PUF.

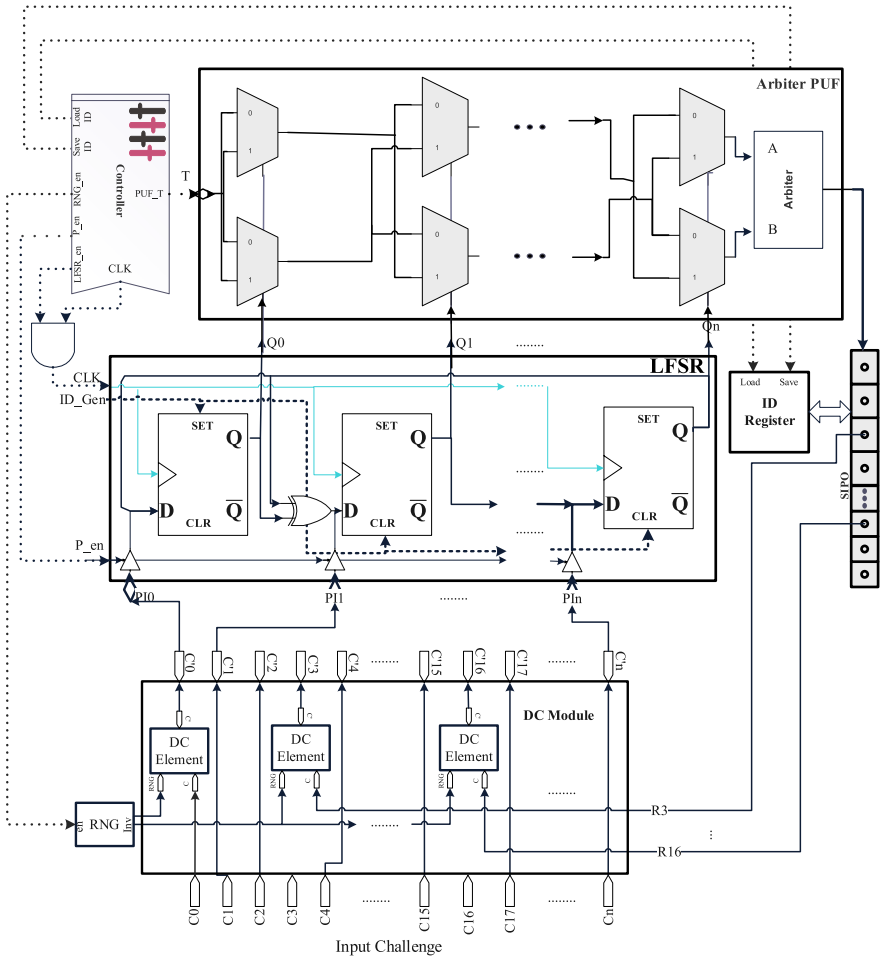


Fig. 3. The block diagram of the proposed PUF-based authentication.

lenges and checks whether the receiving response from the requested node relatively matches with the predicted one. Fig. 3 shows the block diagram of the proposed PUF-based authentication method. The diagram consists of four main blocks including DC-module, arbiter PUF, LFSR, and the controller unit. Unlike the basic arbiter PUFs by which only one bit of response can be generated, many applications for secure authentication require multi-bit responses. There exist two solutions to extend the responses of basic arbiter PUF, one is to use multiple concurrent arbiter PUFs (e.g., OBPUF, Poly-PUF) and the second one is to use the LFSR block to generate multiple sub-challenges from a single challenge and use a single arbiter PUF to generate multiple 1-bit responses for each sub-challenge (Suh and Devadas, 2007) (Majzoobi et al., 2008a). In our design, we used the second solution as it incurs less design overheads. Before any request, the device needs to introduce its identity to the server. The device identity (ID) in our scheme is generated by the PUF instance as shown in Fig. 3. To generate the ID, the controller issues the *ID_Gen* signal and a pre-specific pattern is applied to the LFSR module using the *Set* and *Clr* inputs. Then, the PUF instance is fed with the specific pattern of LFSR, and it generates a unique *n*-bit ID which can be hardcoded in the device before installation. The generated ID will be stored and used for the future request to the server. Even with the same input patterns, different IDs will be obtained for different devices using proposed scheme. In the following, a detailed description about constituent blocks of the proposed DC-PUF is provided.

3.1. The proposed DC-PUF architecture

3.1.1. DC-module

The proposed DC-module is responsible for camouflaging the hyperplane with modifying some selected bits of an incoming challenge as shown in Fig. 4. The modification process on the challenge's bits is performed in two steps. The first step is to inject a random bit which inverts some bits of the incoming challenge (i.e., randomization step) and the second step is to skip some challenge's bits and replace them with some bits of the previous response (dependency step). In the DC-module, some bits of the incoming challenge are supplied to the DC-elements while the rest of them are bypassed the DC module and are directly connected to the LFSR module in a parallel manner as shown in Fig. 4. The challenge's bits which are connected to the DC-elements may be inverted or intact according to the status of a random bit which is generated by the RNG module (randomization step). The multiplexer unit in DC-element decides whether to pick up the challenge's bit itself or the inverted bit according to the status of *Sel* signal. If the *Sel* signal is activated, all the challenge's bits are directly passed to the LFSR module without any modification (i.e., DC-elements are skipped). In fact, this condition activates the enrollment phase, where the server needs to acquire the original responses to the plain challenge. In our scheme, some challenge's bits connected to the DC-elements are substituted by some bits of the previous response (dependency step). It is up to the designer to decide which and how many of the challenge's bits to connect to the DC-elements and which ones need to be replaced by the previous response.

3.1.2. Random number generator (RNG)

The RNG block is responsible for injecting randomness into the incoming challenges. The single random bit from the RNG generates two states, resulting in two possible responses for each challenge. Note that the server does not have any clue about the random logic generated by the RNG block, so it will consider two possible responses for each challenge and check which of the possible responses have the most similarity with the receiving response. The SPIO register in the proposed DC-PUF serially stores the 1-bit responses for each sub-challenge in every clock cycle and sends all the m-bit responses in parallel to the server at once. Moreover, using the SPIO register, some response's bits are feedbacked to the DC-module to create the dependency chain. In summary, the server only needs to generate two possible responses for each challenge according to the logic of a single random bit (one for bit '0' and the other for bit '1') and compare them with the receiving response. The comparison process repeats once for each generated response and the response with the least difference in range of predefined $\mathscr{E}$ will be regarded as the acceptable answer, the previous response is updated in both server and client sides, and the authentication is passed. This matching process is sufficient to decide whether the authentication is passed or rejected. The identification accuracy highly depends upon the range of $\mathscr{E}$. As the $\mathscr{E}$ range increases, the less false-negative rate will be achieved, but the increase in $\mathscr{E}$ increases the false-positive rate as there may be some malicious nodes that are wrongly identified as the legitimate due to the presence of channel

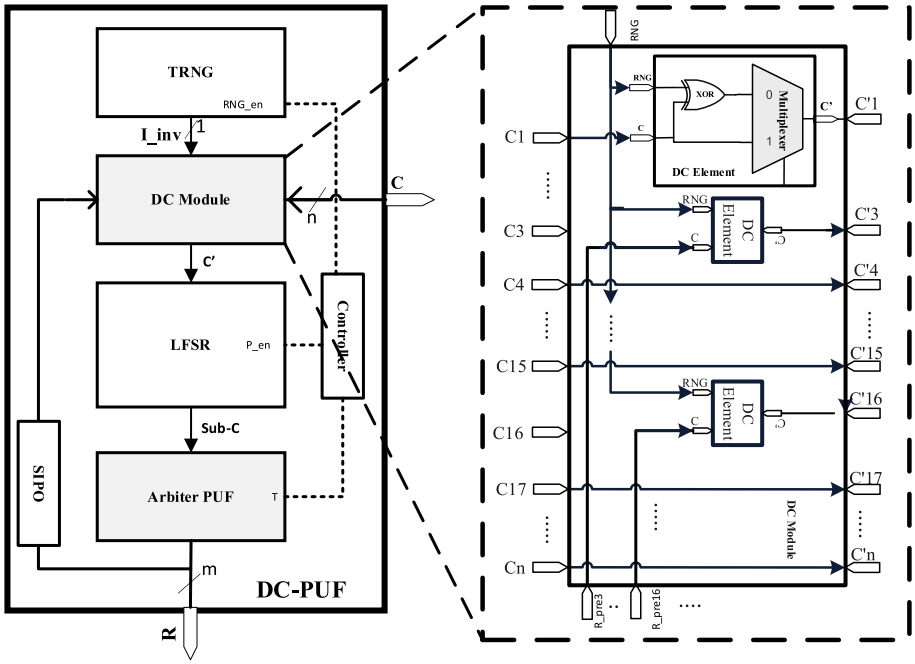


Fig. 4. The proposed DCT-PUF and DC-Elements.

disturbance.

3.1.3. Arbiter PUF

In the arbiter PUF, an input signal T can pass one of the two different pathways according to the status of the incoming challenges. In fact, the input challenge is connected to the multiplexers that decide which path is selected for the propagation of input signal T. At the end, the arbiter unit which is a D-type flip-flop determines the output signal based on the arrival time of its inputs. If the transition on D comes first the output will be one; otherwise, the output will be zero. Since every path-delays related to the arbiter PUF are highly affected by the presence of process variations, the output prediction and cloning of the model will be difficult.

3.2. The proposed enrollment mechanism

In the enrollment phase, the server generates N number of n-bit challenges and sends them to the target device. We recommend to perform this operation in a secure local network since there is a risk of man-in-the-middle (MitM) attacks in a remote access. To enroll a given device, the *Sel* signal of multiplexers in DC-elements is set for the first time to enable the n-bit challenges directly pass through the DC-elements without any modification. In this case, the arbiter-PUF has a direct access to the unmodified sub-challenges, and the server can capture the plain responses. Next, the *P_en* signal is asserted by the controller to let the LFSR module picks up the plain challenge in parallel. Then, the *P_en* is de-asserted, and the LFSR generates m sub-challenges in m clock cycles for every plain challenge and delivers them to the arbiter-PUF in parallel.

The arbiter-PUF generates 1-bit response for every n-bit sub-challenge and transfers the generated response's bits serially to the SPIO register. This process is repeated for all the challenges, and the generated responses are sent to the server. After this process, the *Sel* is de-asserted to disable the direct access to the arbiter-PUF. Next, a new ω model for every device in the server is trained by using the challenge-response pairs (CRPs) to predict the relation between CRPs. Before that, the internal structure of the DC-module of the target device is defined for the model in the form of a specific modification function to let the model simulate the modified challenge. This function specifies which of the challenge's bits are modified by the DC-elements and which of the response's bits are feedbacked to the input challenge. Finally, an interface for direct access to the CRPs have to be irreversibly disabled. In fact, the physical access to the plain CRPs should be permanently disabled before device installation. This can be done by burning the irreversible fuse, so an attacker is not able to replicate the trained model (Majzoobi et al., 2012).

3.3. The proposed authentication/deployment mechanism

Algorithm 1 describes the client-server authentication protocol. At the beginning of a session, the device sends an authentication request to the server, including the Device-ID. The server receives the request and sends back n-bit random challenge ($\mathscr{C}$) to the device. In both client and server sides, the device ID is regarded as the previous response (Res_{pre}) when the first session is performed, or the connection resets. The client receives the challenge $\mathscr{C}$ and modifies it by inverting some challenges bits according to the random number ($I_{i\text{-}rv}$) while some of the challenges' bits are replaced by the Res_{pre}, resulting in a modified challenge $\mathscr{C}'$ at the client side. Next, the *P_en* of the LFSR block is enabled to let the LFSR captures the modified challenge in parallel, then at the next clock it will be disabled. The LFSR block generates m sub-challenges ($\mathscr{C}_{xj}$) for every modified challenge and delivers them to the arbiter-PUF to generate m-bit response R. The generated response R is sent to the server. The server receives the response while it generates two possible challenges ($\mathscr{C}'_0$, $\mathscr{C}'_1$) for $I_{i\text{-}rv}=0$ and $I_{i\text{-}rv}=1$. Note that the server knows the modification function (DC-Module-Sim ($\mathscr{C}, Res_{pre}, 0/1$)) of the client from the enrollment phase. Therefore, it can modify the challenge as the client already did. However, the server does not know the random bit ($I_{i\text{-}rv}$); thus, it requires to generate two possible challenges, resulting in predicting two responses (R'_0, R'_1). In order to decide whether the authentication is granted or denied, the server calculates the minimum hamming distance (HD) of two predicted responses in respect to the received response R. If the minimum HD is lower than a pre-specified fault tolerance (ε), the authentication is granted and the predicted response with minimum HD is selected as the Res_{pre}; otherwise, the request is denied. To prevent from the noise propagation in the next authentication phases, we select the predicted response with minimum HD rather than the received response to be replaced with the previous response. To clarify the authentication process, the server-client communication flowchart is depicted in Fig. 5.

Algorithm 1
Client-Server Authentication Process

Client Side:

1 **Checks** whether the connection is the first session? If yes:
Res_{pre} = *Device-ID*//Res_{pre}: previous response
Sending connection request with *Device-ID* to server ▶▶▶
3 **Receiving** $\mathscr{C}$ Challenge from Server ◀◀◀
$I_{i\text{-}rv}$ = **Generate** 1-bit random number by **RNG** block
$\mathscr{C}'$ = ***DC-Module*** ($\mathscr{C}$; $I_{i\text{-}rv}$, Res_{pre})//generate modified challenge from incoming challenge, previous response, and RNG output//**Set** *P_en*←1; //LFSR captures the modified challenge $\mathscr{C}'$
Reset *P_en*←0//LFSR disables the parallel capturing **foreach** bit j of m-bit response R **do**:
$\mathscr{C}_{xj}$ = **Generate** Sub-Challenge using LFSR block
R_j = **Generate** Response bit of $\mathscr{C}_{xj}$ using *Arbiter-PUF*
Res_{pre} = R//update the previous response with a new response
Sending generated response R to server ▶▶▶

Server Side:

2 **Receiving** connection request with Device ID ◀◀◀
Checks whether the connection is the first session? If yes:
Add Res_{pre} = *Device-ID*
$\mathscr{C}$ = Random number Generation (n)//generate n-bit challenge
Send $\mathscr{C}$: n-bit challenge to Device ▶▶▶
4 **Receiving** generated m-bit response R from client◀◀◀
$[\mathscr{C}'_0, \mathscr{C}'_1]$ = ***DC-Module-Sim*** ($\mathscr{C}, Res_{pre}, 0/1$)//generate two possible challenges based on 1-bit random number
$[R'_0, R'_1]$ = **Predict** ($\mathscr{C}'_0, \mathscr{C}'_1$)//predicts two possible responses R'_0, R'_1 from trained model
η = min (**HD** (R, R'_0), **HD** (R, R'_1))//calculate minimum hamming distance **if** $\eta < \varepsilon$ **then**//ε: a user-defined fault tolerance
Authentication granted ▶▶▶
Res_{pre} = **HD** (R, R'_0) < **HD** (R, R'_1)) ? R'_0 : R'_1 **else**
Authentication denied ▶▶▶

Fig. 6 shows the timing diagram of the proposed DC-PUF respect to the main clock. After the connection request, the input data (i.e., challenge) to the DC module will be available. Then, the LFSR block accepts the modified challenge in parallel by enabling the *P_en* signal for one clock cycle. Next, the *LFSR_en* is set to let the LFSR's registers serially shift the modified input challenge. Every shift process in LFSR generates a new sub-challenge for arbiter-PUF, resulting in one-bit response. Meanwhile, the arbiter-PUF generates a transition on *PUF_T* input (e.g., 0 → 1, 1 → 0) for every clock cycle. Therefore, one-bit response is generated every clock cycle after the first transition on *PUF_T* input. The m-bit response is obtained after $m+1$ clock cycles when the input challenge receives.

3.4. The CNN-based attack scenario

In this paper, a new attack scenario based on CNN is proposed, which can successfully predict the responses with high accuracy when it is

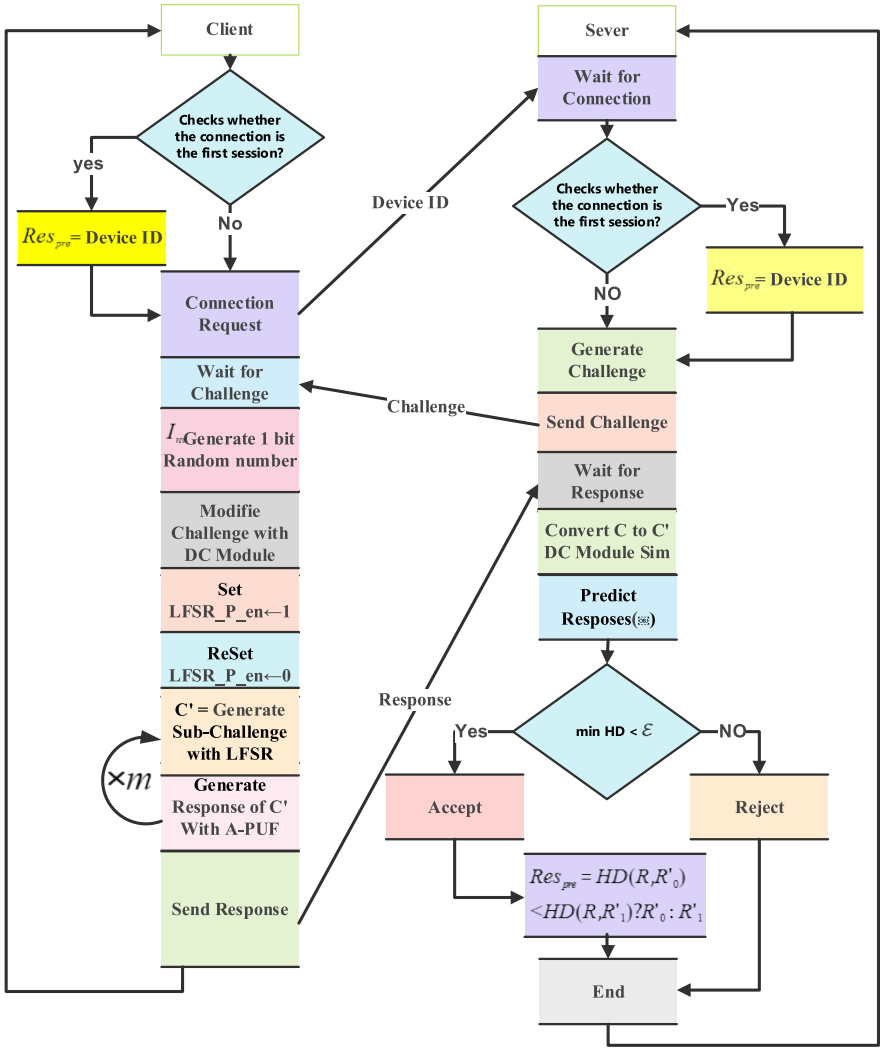


Fig. 5. Client-Server authentication flowchart in DC-PUF.

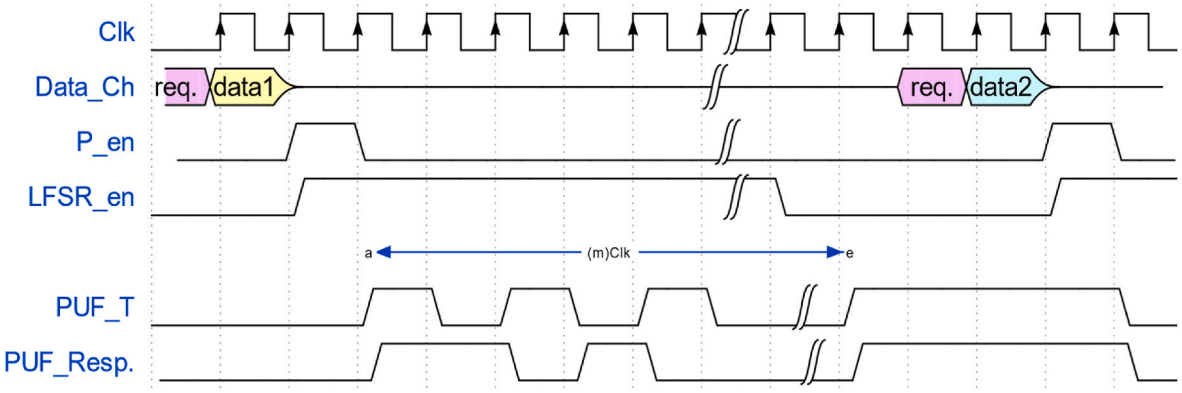


Fig. 6. Timing diagram of the proposed deployment (authentication) phase.

applied to the existing PUF structures. We evaluated various ML techniques such as Support Vector Machines (SVMs), Logistic Regression (LR), Artificial Neural Network (ANN), and Convolutional Neural Network (CNN) in our experiments. However, the CNN-based model has been considered in our study as the basis of the attack strategy since it is found to have the superior potential of modeling highly non-linear data. Due to the higher prediction accuracy and fast convergence time, a resilient back-propagation is considered as the best training algorithm. Although the CNN-based attack engine incurs high complexity and computational power compared to that of the other classification engines, it can be deployed in a strong server where the trained model is located. So, an adversary can mount the malicious nodes by replicating the authorized nodes. These malicious nodes can send multiple requests to the main server by replying the genuine requests. When the challenge

is sent by the server, the malicious node resends the challenge to the malicious server and receives the predicted response from the trained model, and sends it back to the main server to gain unauthorized access. When the authentication is granted, the malicious nodes can exploit the Daniel-of-Service (DoS) attacks, or they can be used as the suitable infrastructure to acquire vital information and other attack models. To achieve higher prediction accuracy from the existing PUFs, we used our proposed CNN model, as shown in Fig. 7. The network consists of three convolution layers by which the 1-D input data is preprocessed, and the unique features are extracted. Next, the extracted features are classified by five fully-connected layers. In the first 1-D convolutional layer, 200 kernels with size 2 are employed using Rectified Linear Unit (ReLU) activation function. The input data can be challenges or the pre-processed challenges. In the next stages, 1-D convolutional network with 180 kernels and 90 kernels of size 3 and ReLU activation function are used, respectively. Next, the intermediate data is supplied to the fully-connected layers to be classified. In the first fully-connected layer, the Tanh activation function is used to consider the negative values, while the next three layers use the ReLU function as the activation function.

The last fully-connected layer which has an equal size with response uses the sigmoid function. In the learning stage, we used the *Binary_Crossentropy* as the loss function and *Gradient Descent* (SGD) as the optimizer engine. The results of applying the proposed CNN-based attack model on existing PUF structures showed that it can yield higher prediction accuracy compared with the LR, SVM and ANN attack models. We have achieved more than 95% of prediction accuracy when the proposed CNN-based attack model accompanied by the proposed preprocessing technique is applied to the OB-PUF. The detailed pre-processing method and the experimental results will be discussed in the next section.

4. Experimental results

4.1. Experimental setup

We have implemented the proposed DC-PUF on Xilinx Zynq7000 SoC chip using Verilog description language. To extract the delay of path segments within multiplexers of arbiter PUF, we have used the Montecarlo algorithm and specified the delay variation corners using the Gaussian distribution. To do so, the circuit model of the arbiter-PUF in transistor-level using H-SPICE software was simulated 1000 different times with Montecarlo algorithm to obtain the delay corners of the arbiter paths during manufacturing processes. As a result, the mean and standard deviation values in the worst-case scenario were obtained $M = 52.3ps$ and $\sigma = 11ps$, respectively. In modeling with a gaussian distribution for each path, different values were assigned to simulate the state closest to reality. Totally, 100 DC-PUFs are simulated using Python. Different attack models including the LR and SVM-based attacks were conducted using the scikit-learn library to enable a fair comparison with the state-of-the-art PUF structures. In order to better justify the reliability of the proposed authentication method, we have also considered the ANN and CNN attack models where they are trained using the Keras library(Keras, 2022) with Tensor-Flow backend, Python, and Google celebratory. It should be noted that we used different attack models and scenarios similar to the models used by state-of-the-arts (Delvaux, 2019), (Konigsmark et al., 2016) and compare the prediction accuracy. Apart from the known attack models such as LR and ANN models, we also developed a new CNN-based attack model which is applied to the proposed DC-PUF and that of the other methods. The experimental results signify that the proposed DC-PUF surpasses the existing PUF structure in terms of resistance against ML attacks when the previous attack models (LR, and ANN) as well as the new CNN model are applied to these methods. We also applied the SVM attack model to the propose DC-PUF to better highlight its resiliency.

4.2. Resistance against ML attack models

Fig. 8 shows the effect of the number of DC-elements used in the proposed DC-PUF on resilience against the CNN-based attack model on the proposed DC-PUF with 5×10^5 CRPs. According to Fig. 8, the number of used DC-elements is defined as the percentage of challenge's bits which are modified by inserting DC-elements. Moreover, it is assumed that 90% of the inserted DC-elements are fed with response's bits, and the rest of them (10%) are directly connected to the challenge's bits. The input challenges and output responses are both 64 bits. The DC-elements are randomly mounted on the way of the challenge's bits which are intended to modify the incoming challenge to camouflage the hyperplane. It is apparent from Fig. 8 that inserting a few DC-elements can substantially augment the resilience against the ML attack as well as lowering the prediction accuracy, at the expense of extra hardware overhead. It is found that even by adding DC-elements to 10% of the challenge's bit (i.e., six DC-elements), the prediction accuracy significantly drops more than 20% compared with the direct connecting the challenge to the LFSR module. This reduction is about 40% once half of the challenge's bits are equipped with DC-element. It is worth pointing out that employing the DC-elements more than 50% of the challenge's bit does not notably drop the prediction accuracy as we are making the response more dependent on the previous response rather than the

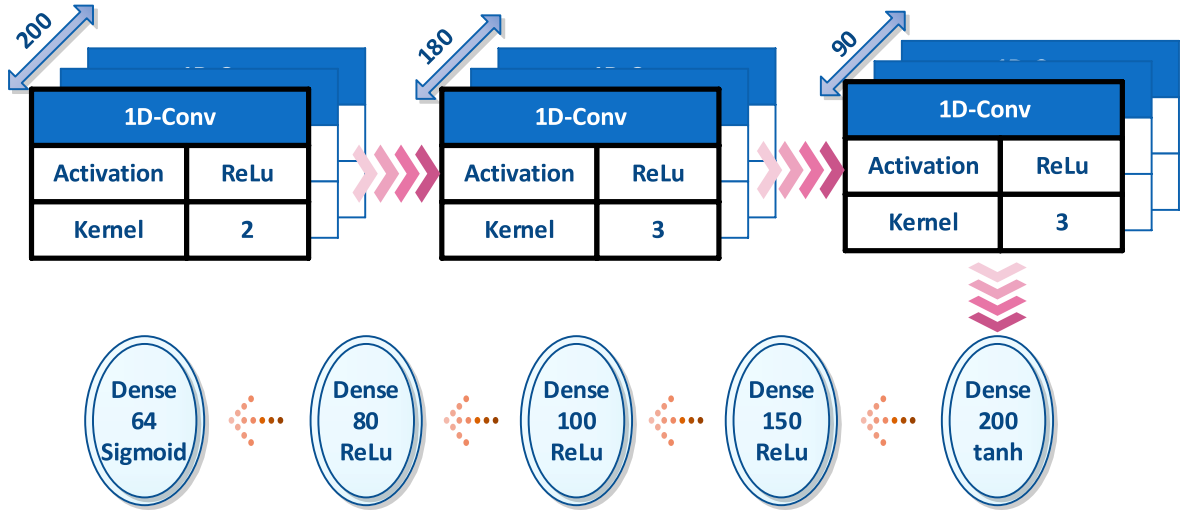


Fig. 7. Proposed CNN-Based attack scenario for cloning the PUF structure.

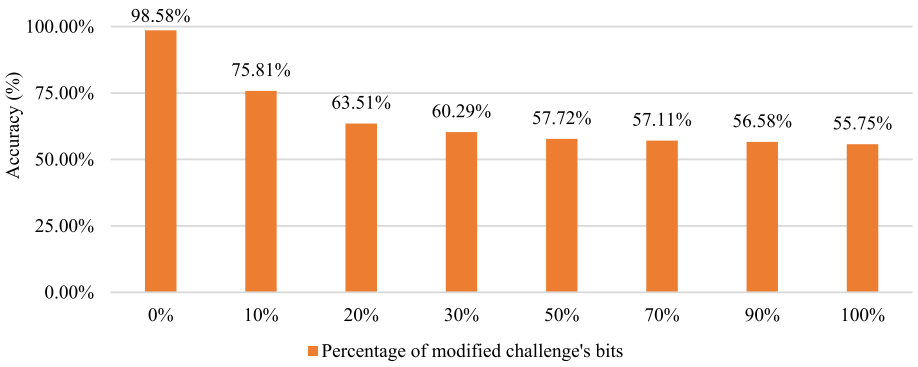


Fig. 8. The prediction accuracy versus the increase in percentage of challenge bits which are modified by DC-elements.

current challenge. For example, at 70%, 45 DC-elements are employed, where 40 elements are fed with the corresponding response's bit and 5 elements are connected to the input challenge. This means 40 bits of 64 bits (62%) of response are dependent on the previous response and the rest of them is dependent on input challenge. It is highly recommended to avoid employing DC-element by more than 50% of the challenge's bit to ensure higher resilience as well as lowering the area overhead. In the rest of this paper, we will consider 50% of challenge's bits candidate for DC-element insertion, 90% percent of DC-elements are supplied by corresponding response's bits, and all the challenges and responses are 64 bit long.

Fig. 9 shows the effect of the number of CRPs on the prediction accuracy of different PUFs when the CNN-based and LR-based attack models are conducted on different PUF models. It can be concluded from Fig. 9 that even with a large number of CRPs around 10^6, more than 58% prediction accuracy cannot be reached from the DC-PUF. Moreover, Fig. 9 compares the results of prediction accuracy of OB-PUF and XOR-PUF when they are under attack by CNN and LR-based attack models. In order to conduct an efficient attack on OB-PUF by using the CNN model, we develop a different attack scenario to eliminate the stage of training de-obfuscation tool using LR method while yielding the same prediction accuracy (i.e., more than 90%) as previously obtained by (Delvaux, 2019). We already discussed that the OB-PUF generate 3-bit response for each challenge and with this mechanism there are cases where the random pattern included in the OB-PUF does not influence on the generated response. As a matter of fact, the same 3-bit response is obtained for multiple applications of a fixed challenge and the randomization process will be neutralized. These challenges are referred to as nominal values that can be used against the OB-PUF. In the proposed attack model, we extract these nominal values and leverage the one-hot categorizing (PotdarT. and C., 2017) to extend all the 3-bit responses

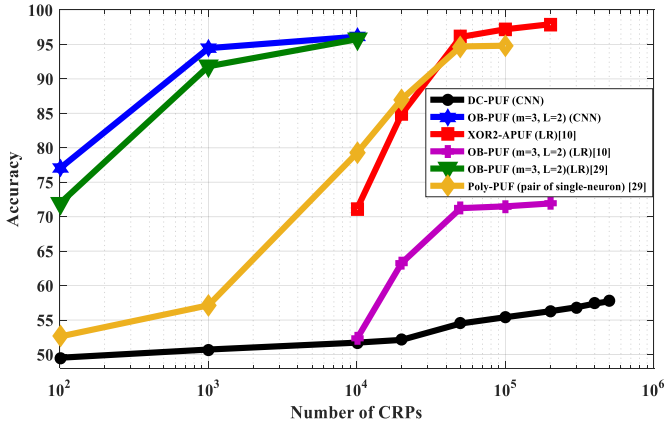


Fig. 9. The prediction accuracy versus number of CRPs for different PUF structures under different attack scenarios.

generated by nominal values into 8-bit response. Next, we used the CNN as the classification engine to find the correlation between CRPs. With this attack scenario, more than 95% of accuracy can be reached of the OB-PUF in the first round of attack with 5000 CRPs. This result is the same as previously obtained by (Delvaux, 2019)with the difference that they require an additional preprocess where a de-obfuscation tool is produced using LR method with 10^5 nominal values as shown by Fig. 9. For dataset larger than 10^4, the OB-PUF reveals around 95% of prediction accuracy when the proposed CNN-based attack model is applied. The same accuracy can also be reached by XOR-PUF when it is under LR attack with dataset larger than 10^5..

Table 1 reports the results of prediction accuracy for different PUF structures and the proposed DC-PUF when various ML-based attack models (e.g., LR, ANN, and CNN) are conducted. To highlight the higher resilience of the proposed DC-PUF in respect to the other PUFs, we used a much larger CRPs for DC-PUF when conducting attacks. In this experiment, the ANN model includes 10 and 30 neurons per hidden layers, respectively. In respect to the existing PUFs, it is found that proposed DC-PUF achieves higher resiliency (i.e., lower accuracy) when applying LR, ANN, and CNN attack models. The accuracy of the DC-PUF is 51.55%, 52.56%, and 56.27% for LR, ANN, and CNN respectively. This yields a significant drop of accuracy under LR attack around 47.9%, 43.4%, and 38.4% compared with the XOR-PUF, OB-PUF, and R–PUF, respectively. Moreover, we have further conducted SVM attack model for DC-PUF, which leads to 50.78% accuracy with 4×10^5 CRPs. In summary, we can claim that the proposed DC-PUF can achieve a considerable accuracy drop around 40%, on average, compared to the recent OB-PUF and R–PUF under LR and CNN-based attack models. We have also implemented the DC-PUF on FPGA, and a CNN attack model is conducted with 4096 CRPs. The experimental results signify that more than 50.38% prediction accuracy cannot be reached on DC-PUF, which indicates a promising resistance against ML attacks.

Table 1
Accuracy results for different PUFs against attack models.

Attack Model	CRPs	LR	ANN	CNN
Arbiter-PUF	10^3	99.9%	99.9%	99.9%
XOR-APUF (Majzoobi et al., 2008b)	78×10^3	99% (Ruhrmair et al., 2013)	–	–
Poly-PUF(Konigsmark et al., 2016)	10^5	–	≈95% (Delvaux, 2019) [a]	–
OB-PUF (Delvaux, 2019)	10^4	≈95% (Delvaux, 2019)		≈96%
OB-PUF (Gao et al., 2016)	2×10^5	71.92% (Gao et al., 2016)	–	–
R–PUF (L2) (Ye et al., 2016)	10^3	≈90% (Delvaux, 2019)	–	–
DC-PUF	4×10^5	51.55%	52.56%	56.27%

[a] A pair of single-neuron networks.

4.3. Uniformity & uniqueness

The uniformity is defined as the percentage of 1s among all the response's bits. Ideal uniformity is 50%, indicating that the same number of 1s and 0s should be available in the response. The proposed modification process on the challenge, including the randomization step and dependency chain should not affect too much the uniformity of the response. To evaluate the uniformity, we have randomly generated 2×10^4 challenges of each DC-PUF, and the experimental results of average uniformity are provided in Table 2. From Tables 2 and it is concluded that the uniformity of the proposed DC-PUF is around 51.57% which is higher than the value reported by Arbiter-PUF and R–PUF (Ye et al., 2016). Worthwhile to note that to inject the randomness into the responses, the R–PUF (Ye et al., 2016) used two-bit random number while we only used one-bit random number and still our uniformity result is better than R–PUF (Ye et al., 2016). The rationality is that our randomness does not only depend on the used random bit, but also on the dependency chain and the temporal correlation between the consecutive responses.

The uniqueness accounts the difference in responses generated by different PUFs when the same challenge is applied. This term is evaluated as the percentage of different response bits and its ideal value is 50%. To evaluate this term in DC-PUF, in addition to applying the same challenge, the identical sequence of challenge should also be applied to different PUFs due to the presence of dependency chain. There exist two possible responses for every challenge in DC-PUF owing to the 1-bit randomness which is injected by RNG unit. Therefore, the probability of finding the identical responses for every challenge is doubled after adding the randomization. This probability can be expressed as $2 \times p^m$, where m is the number of response bits, and p is the probability of finding the same single response bit when the same sub-challenge is applied to two different arbiter PUFs. The result of uniqueness reported in Table 2 implies that although the probability of finding identical responses is doubled after adding the randomization, the absolute value is very small and the uniqueness is close to the ideal value.

4.4. Hardware complexity & overheads

Table 3 evaluates the hardware complexity and compares the number of primary components of different PUFs with 64-bit challenges and responses. The R–PUF uses two bits for randomization module in which 64 inverters and 64 multiplexers are connected to the 64-bit input challenges. The LFSR unit includes 64 DFFs to shift 64-bit sub-modules. In the arbiter-PUF, it uses 128 multiplexers and one DFF as arbiter unit, totally 256 multiplexers and 65 DFFs are included in R–PUF, as reported by Table 3.

In Poly-PUF, the CSD/RSD is input/output module which extends the random bits generated by TRNG unit while XORing the extended random bits with the challenge or responses. This module CSD in input and RSD in output totally requires 128 multiplexers and 128 XORs. Authors of Poly-PUF propose two different scenarios for implementing the PUF unit, where it can only use one PUF and one LFSR block to generate 64-bit responses in 65 clock cycles or multiple parallel PUF instances can be realized, and the LFSR unit is no longer required. The latter scenario incurs much higher power and design overheads while reducing the reliability due to the cross-talk problem between PUF instances. All the PUF instances are collaborated in generating the response and one PUF would issue the response to the server, so the time taken to generate response bits is shortened to only one clock cycles. To conduct a fair comparison, the first scenario is considered for the Poly-PUF in which 128 multiplexers and 65 DFFs are required to implement the arbiter PUF and LFSR unit, respectively. Totally, 256 multiplexers, 128 XORs, and 65 DFFs are included in Poly-PUF, as reported by Table 3. The next candidate is slender PUF, which uses four-stage XOR-PUFs with two 128-bit LFSR units. The two LFSR units are XORed and its 64-bit result is fed to the four-stage 64-bit PUF. Each PUF instance includes 128 multiplexers, and one DFF which totally produces four outputs for all stages. Finally, the four outputs of PUFs are XORed to generate a single bit response. The slender PUF embeds 128-bit TRNG unit which includes 128 multiplexers and 12 DFFs. In overall, the slender PUF includes 242 DFF, 67 XOR, and 640 MUX, as listed in Table 3.

The proposed DC-PUF only employs the DC-elements to 50% of the challenge's inputs while 90% of the DC-elements are input with the corresponding bits of the previous response. So, the number of DC-elements for 64-bit challenge is 32 and each of which includes one MUX and one XOR gate. Also, the 64-bit arbiter PUF and LFSR unit include 128 MUX+1 DFF and 64 DFF, respectively. In respect to the R–PUF and Poly-PUF, the number of MUX required for DC-PUF decreases by around 37%. This reduction is significantly higher in comparison to the Slender-PUF (i.e., 75%). In terms of the number of DFF, the DC-PUF shows a remarkable reduction around 73% compared with Slender-PUF, and the required DFF is the same as R–PUF and Poly-PUF. Moreover, the number of XOR required for DC-PUF is lower than Poly-PUF and no inverter gate is required. As a result, the hardware complexity of DC-PUF indicates a favorable decrease in comparison to the state-of-the-art PUFs. To better compare the results of hardware implementation, Table 4 reports the number of lookup tables (LUT) occupied in FPGA for each PUF instance. It is concluded from Table 4 that the DC-PUF yields a significant reduction in number of LUT around 73% compared to NB-PUF (Mandel Yu et al., 2014) and Slender PUF. Moreover, compared to the recent Poly-PUF, the number of LUTs included in the proposed DC-PUF decreases 20%.

Table 5 presents a qualitative comparison between different PUFs in terms of hardware complexity, prediction accuracy, obfuscation, and dependency types. According to Tables 5 and it is evident that the proposed DC-PUF incurs lower hardware complexity as well as yields much lower prediction accuracy under ML attacks. As reported by Table 5, the traditional arbiter PUF does not employ any technique to obfuscate the correlation between CRPs. Nevertheless, the DC-PUF employs 1-level ($L = 1$) randomization process accompanied by the dependency chain. That is, the input challenge is modified based on a

Table 2
Experimental results of uniformity and uniqueness.

Model	#CRP bits	Uniformity (%)	Uniqueness (%)
Arbiter- PUF	64	50.547%	49.197%
R–PUF(Ye et al., 2016)	64	51.10%	49.7%
DC-PUF	64	51.57%	49.23%

Table 3
Comparison of hardware complexity of different PUFs.

PUF Model	R–PUF (Ye et al., 2016)	Poly-PUF (Konigsmark et al., 2016)	Slender-PUF (Majzoobi et al., 2012)	DC-PUF
XOR	0	128	0	32
Inverter	64	0	67	0
MUX	256	256	640	160
DFF	65	65	242	65

Table 4
Number of LUTs required for different components of PUFs.

Component	NB-PUF (Mandel Yu et al., 2014)	Slender-PUF (Delvaux, 2019)	Poly-PUF (Konigsmark et al., 2016)	DC-PUF (Proposed)
PUF	4 × 128	4 × 128	128	128
LFSR	10	10	10	10
TRNG	128	128	N/A	N/A
CSD/RSD	N/A	N/A	61	N/A
DC-element	N/A	N/A	N/A	32
Total	650	650	213	170

Table 5
A qualitative comparison of different PUF models In terms of hardware complexity, accuracy, obfuscation, and dependency types.

Ref	Hardware Complexity	Prediction Accuracy	Obfuscation Type	Dependency Type
DC-PUF	*Low*	*Low*	$R(L1)+DC$	*Temporal + Spatial*
R–PUF (Ye et al., 2016)	*medium*	*High*	$R(L2)$	*Spatial*
OB-PUF (Gao et al., 2016)	*High*	*High*	$R(L1)$	*Spatial*
XOR-PUF (Majzoobi et al., 2008b)	*Very High*	*Very High*	*Redundancy*	*None*
Arbiter-PUF (Lim et al., 2005)	*High*	*Very High*	*None*	*None*

single random pattern and the previous responses, where the previous response itself is dependent on the previous challenge and prior responses. Therefore, it can be concluded that the DC-PUF, in addition to the spatial correlation, it also employs the temporal correlation between CRPs. The OB-PUF and R–PUF only use $L = 1$ and $L = 2$ randomization process, respectively, so they only employ the spatial correlation. The XOR-PUF uses the redundancy method by duplicating the PUF instances (i.e., XOR-mixed arbiter PUF) to an arbitrary extent to relieve the weak statistical property of the arbiter PUF. This weak statical property implies that one CRP may reveal information on arithmetically close other challenges.

4.4.1. DC-PUF reliability

The stability of the PUF is a crucial factor in the reliability of the authentication methodology. A PUF that is unstable can lead to the inconsistent responses, which can compromise the authentication process, and also increase the false-positive and false-negative rates. There are several factors that can adversely affect the stability of a PUF, including power supply fluctuations and temperature changes. Thus, it is important to ensure that the PUF is designed to be stable under different operating conditions. First of all, it is worthwhile to note that the base architecture of the proposed DC-PUF is an Arbiter-PUF (Ruhrmair and Solter, 2014) (Gassend et al., 2004). Experimental results in (Hemavathy and Bhaaskaran, 2023) already reported the reliability of the Arbiter-PUF for 32-bit and 64-bit challenges which are 99.52%, and 97.96%, respectively, validating a reliable random output under different operating conditions. In the proposed DC-PUF, we aim to augment the resilience against the ML-based modeling attacks as well as increasing the randomness into the responses. To achieve this, we introduce the dependency chain providing a temporal correlation between consecutive responses which in turn, increases the randomness and provides a remarkable resilience against the cloning and modeling attacks. To ensure the reliability of the PUF instance, the delay difference between the upper and lower paths should be significant enough to generate a reliable bit. In fact, the chance of bit flip in response increases only when the delay between two paths intersects over the operating temperature and the path delay difference is small (Hemavathy and Bhaaskaran, 2023). Thus, adding the DC elements to the original Arbiter-PUF does not affect the path delay difference; thereby, no impact on reliability is expected. However, in the proposed DC-PUF, we are essentially creating a feedback loop in the Arbiter-PUF to form the dependency chain. That is, some challenge bits are substituted by some response bits by employing the DC elements. Therefore, it is expected that the reliability of the DC-PUF is lower than the original Arbiter-PUF due to the bias amplification caused by the feedback loop. In fact, there is a chance that the flipped bit is among the selected response bits for forming the dependency chain and the reliability degrades as the authentication round continues and dependency chain grows. The reliability of PUF can be measured by calculating the Intra-distance which is the distance between two responses when a specific challenge applied twice to one device (El-Hajj et al., 2021), and can be expressed as (1):

$$HD(Intra) = \frac{1}{n(d-1)} \sum_{j=2}^{d} \sum_{k=1}^{n} HD\left(R_{(i,k)}, R_{(j,k)}\right) \times 100 \quad (1)$$

In (1), the average number of unreliable response bits is calculated, where d is the total number of devices under evaluation, n is the number of response bits, and $R_{(i,k)}, R_{(j,k)}$ are the k th bits of i th and j th response for the same device under different operating conditions. Accordingly, the reliability can be defined as (2):

$$Reliability = 100 - HD(Intra) \quad (2)$$

Ideal PUF instance with reliability of 100% means the device generates the same responses for a given challenges despite hostile environmental conditions. We performed the reliability analysis on the proposed DC-PUF based on the simulation in MATLAB, similar to the approach adopted in (Rührmair et al., 2013) (Becker, 2015) (Sahoo et al., 2018). In our simulation, we assumed that the delay components of the proposed DC-PUF are independent and identically distributed in a normal distribution with $M = 10$ and $\sigma = 0.05$ (Sahoo et al., 2018). To simulate the effect of experimental noise on PUF performance owing to the temperature and supply voltage fluctuations (i.e., noisy system), we used an additive noise with a normal distribution around $\mathcal{N}(0, \sigma^2_{noise})$. Thus, each delay component of the PUF behaves a normal distribution around $\mathcal{N}(10, \sigma^2 + \sigma^2_{noise})$ in the presence of noisy system, where $\sigma_{noise} = \sigma/2$, and the σ is the standard deviation of delay distribution of delay component. The simulation is performed on both 32-bit and 64-bit models of the DC-PUF and 10 instances is considered for reliability analysis, where each instance was evaluated 30 times with 2×10^4 CRPs. Also, to control the reliability of DC-PUF, we performed the same simulation for different authentication rounds $\mathscr{R} = \{1, 2, .., 6\}$ and different percentage of response bits $\mathscr{S} = \{20, 30, 40, 50\}$ are considered for forming the dependency chain. Fig. 10 shows the reliability of 32-bit and 64-bit DC-PUF for different $\mathscr{S}$ values during different authentication rounds $\mathscr{R}$.

In order to better control the reliability of the DC-PUF and lower the chance of propagation of flipped bit, one simple solution is to reduce the authentication rounds and also decrease the number of response bits that are selected for the dependency chain, albeit with trade-offs in terms of prediction accuracy and randomness. In practice, there is no need to continue the dependency chain forever and after some successful sessions the chain can be reset and recreated in both sides for preventing instability propagation to the next authentication rounds. It is concluded from Fig. 10 that the authentication reliability for 32-bit and 64-bit DC-PUF is about 97.52% and 90.64%, respectively, during four authentication rounds when 30% of the response bits are selected for

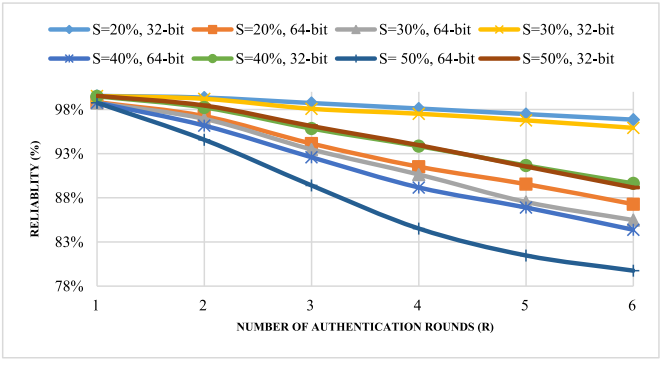


Fig. 10. Reliability of 32-bit and 64-bit DC-PUF during different authentication rounds ($\mathscr{R}$).

dependency chain. In this case, the prediction accuracy of the PUF instance increases to 60.29% for 64-bit DC-PUF according to the results of Fig. 8. In case of 20% selection of response bits, the reliability for 32-bit and 64-bit in four authentication rounds are 98.11% and 91.52%, respectively. Moreover, the prediction accuracy in this configuration is about 63.51% for 64-bit DC-PUF. It is evident from Fig. 10, as the authentication rounds and the number of response bits selected for the chain formation decreases, the reliability significantly improves, albeit at the expense of increasing the prediction accuracy during modeling attacks. The hardware complexity is also reduced when the number of selected response bits are decreased due to the reduction in the number of DC-elements.

Another solution that can remarkably augment the reliability owing to the PUF instability is to change the way the dependency chain is formed in the server side. One option is to form the dependency chain at the server side based on the response generated by the server to cancel the effect of channel noise on the identification accuracy. However, to cancel the effect of PUF instability itself and also avoid propagating inconsistent response bits, the chain alternatively can be formed at the server side based on the receiving response generated at the client side. It should be noted that this choice mainly depends on the channel parameters and the environment conditions where the PUF-enabled device is operating. Also, the change in the way of forming the dependency chain at the server side can be configurable in software, thus no change is required in the device at field.

To elaborate the reliability in terms of channel interferences, we need to explain the authentication flow again. In this case, it is assumed that the inherent PUF stability is guaranteed and the server contains an accurate trained model or reference lookup table including the challenge-response pairs for each device installed in the field. In fact, before the installation, the server needs to create this lookup table for the authentication flow. However, due to the one-bit random used in the client-side, the server must generate two possible responses for each challenge and compare them with the receiving response. The receiving response may differ from the response generated in each side due to the presence of channel interference and noise. Information required for authentication in every session including the device ID (the first previous response) and the other previous response from the successful requests is available on the server as well as the client side. For each session, the server keeps track of the previous authenticated response to form a dependency chain. Note that to ensure the reliability, we need to properly keep track of the dependency chain in both server and client sides. That is, the generated response for each challenge can be the same at both sides if the client is an authorized node. The first previous response is initialized as the device ID in both sides, which is sent by each client at the beginning of the session. In the next successful authorized session, the previous response is updated with the current response generated by that side or remains the same when the authentication is rejected. In fact, the dependency chain continues to grow in each successful session and stops growing when the session is rejected. That is, the breach into the proposed authentication mechanism gets harder as time goes by, since the attacker are not aware of the history of the previous successful sessions.

In each authentication session, the device sends the request including the device ID to the server. The server looks up the last authenticated response for this device and sends back a random or specified challenge to the client. Next, the client generates a response based on the incoming challenge, the random bit, and the previous authenticated response. At the server side, the server generates two possible responses based on the transmitted challenge and the last authenticated response, and compared with the receiving response. If the similarity between these two responses and receiving response is less than a pre-defined margin, the authentication is passed, the current generated response by the server with less similarity to the receiving response is set as the last successful response and the authentication result sent back to notify the client. At the client side, the generated response by the client is set to the previous response when the authentication is passed as well. So, the last successful response in both authorized sides is equally updated, so does the dependency chain.

A key point in the reliability analysis in terms of channel interference for the proposed authentication methodology is that the dependency chain remains the same at both sides of the server and authorized nodes and updated accordingly in each successful session, since the receiving response can be used to form the dependency chain at the server side. In fact, the bit-error rate in each authentication session is not propagated to the next sessions as the previous response in each side is updated based on the same PUF model, input challenge, and current authenticated response generated in that side. It is expected that in the first session, the response generated by the device and one of the responses generated by server is equal when the PUF model, ID, and the incoming challenge are similar. So, the chain starts with the same result in both sides and this equality continues to propagate to the next sessions. As a result, the proposed authentication flow does not let the bit-error be accumulated in the dependency chain and affects the next authentication sessions. Worthwhile to note that if an unauthorized access in somehow is considered true to access by our method, it does not mean it can continue to access forever and the breach is permanent. It is because, our results show that the attacker cannot predict the response more than 58% even using a sophisticated SNN model and the false access is mainly due to the channel disturbance and the Bit-Error-Rate (BER). In other words, the BER may help the attacker to somehow resemble the response something similar to the original response by the server. Even so, in this case the dependency chain is not similar in both sides as the response in server is updated with the legitimate response generated by the server rather than the receiving response. Therefore, the next requests after a successful request of a malicious node will likely fail due to inconsistency between the dependency chains. Important to note that our results show that the percentage of the successful sessions coming from the unauthorized node that are wrongly considered authorized by the server is zero for less than 20 number of mismatches in bits of the receiving response.

Besides the channel interference, the other terms affecting the reliability includes the PUF strength, the PUF randomness, and the accuracy of the model used in the server side. The PUF strength can be evaluated as the uniformity, uniqueness, and its resistance against machine learning (ML) attacks. The uniformity and the uniqueness metrics are already reported in the experimental section, where these metrics are close to the ideal values for the proposed PUF. Moreover, the resistance against different ML attacks is also assessed with only 58% of prediction accuracy for the proposed PUF, which is far better than the previous attacks on already-existing PUF structures as reported by Table I and the chart shown in Fig. 9. In terms of PUF randomness, unlike the way proposed in (Konigsmark et al., 2016), (Ye et al., 2016), we only use a single bit of randomness to simplify the authentication process at the server side, while hardening the cloning process for attacker. Unlike (Konigsmark et al., 2016), (Ye et al., 2016), this single bit is sufficient in our method as we not only rely on the PUF randomness, but also on the dependency chain (i.e., temporal correlation between consecutive responses) to harden the cloning process while preventing to predict the PUF model using ML attacks. In terms of the accuracy of the model used in the server, the server can use a lookup table to search for the correct responses (two possible response for each challenge) corresponding to each challenge and compared with the receiving response.

To evaluate the efficiency of the proposed authentication method, the IEEE 802.15.4 standard is taken into account as the physical layer (Rührmair et al., 2013). This standard defines the data communication using low-rate, low-power, and low-complexity short-range radio frequency transmission in a wireless personal area network (WPAN) which is highly suitable for resource-constrained IoT devices. We implemented the physical layer of this standard using the Communication Toolbox in MATLAB software with the O-QPSK modulation at 2450 MHz band. Also, the effect of channel disturbance on the authentication reliability is

measured to obtain the False-Positive and False-Negative rates. In our experiments, 1000 IoT nodes with five consecutive authorized sessions is assumed to model the dependency chain. To compare the False-Positive and False-Negative rates in the presence of channel variations during data transmission, the E_b/N_0 parameter is measured, where E_b is the signal energy associated with each data bit and N_0 is the noise spectral density. It is proved that as the noise spectral density increases, so does the BER, since the receiving response affects more by the channel variations. The False-Negative rate is an important factor in authentication reliability which defines the percentage of the authorized sessions for different nodes that are wrongly denied by the server. Fig. 11 shows the False-Negative rate in the presence of channel variations for IEEE 802.15.4 standard with O-QPSK modulation at 2450 MHz frequency. To include the channel noise and improving the authentication reliability, we consider a pre-defined range of mismatch in bits of the receiving response in terms of the Hamming-Distance as depicted in Fig. 11. It is concluded from Fig. 11 that the average False-Negative rate in the proposed authentication is around 1% when 3-bit mismatches (i.e., HD = 3) is specified at the receiver side. Nevertheless, the HD = 2 can also be recommended for authentication in less noisy channels. Note that the negative values of E_b/N_0 represents the cases where the noise spectral density gets larger than the signal energy associated with each data bit. Therefore, the largest negative value of E_b/N_0 highlights the worst-case scenario in our experimental. In case of HD = 2, the average of False-Negative rate is the same as 1% by excluding the $E_b/N_0 = -2\,dB$. It is expected that the increase in HD also increases the False-Positive rate, as the attacker can hide the prediction inaccuracy in the mismatch bits of the receiving response which is caused by the channel variations.

The False-Positive rate is defined as the percentage of the unauthorized sessions coming from the malicious nodes that are wrongly passed by the server. To measure the False-Positive rate, it is assumed the attacker has full access to the network between the client and the authorization servers and therefore is aware of the incoming challenges, the previous outgoing responses, and the ID of an authorized device. Moreover, the attacker can use the challenge-response pairs to model our DC-PUF and to conduct a ML attack scenario. The experimental results signify that the accuracy of this model cannot be reached more than 58%. By taking this assumption into account Fig. 12 shows the False-Positive rate versus the HD in the presence of channel variations. It is apparent from Fig. 12 that False-Positive rate is around zero percentage for HD less than 20 and after that it increases exponentially.

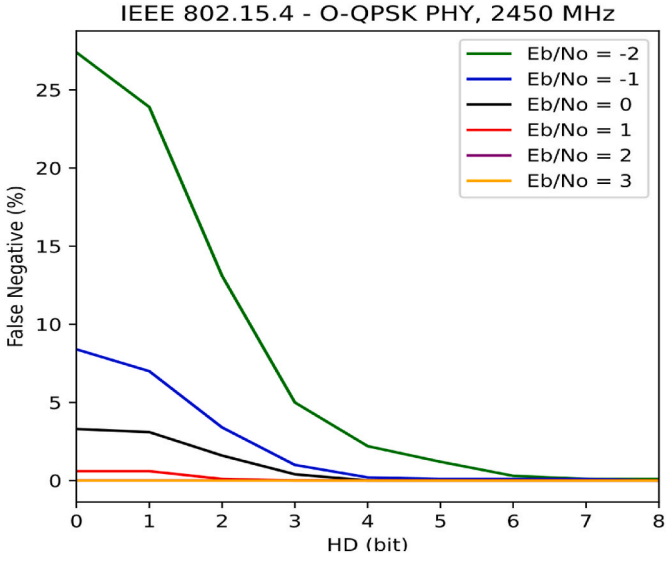


Fig. 11. The false-negative rate versus different HDs in the presence of channel variations.

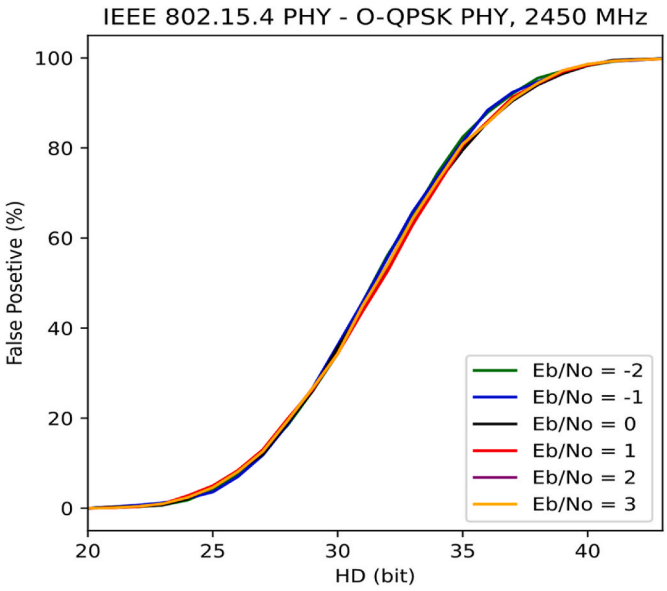


Fig. 12. The false-positive rate versus different HDs in the presence of channel variations.

Also, the False-Positive rate is less affected by the channel variations as the Bit-Error-Rate (i.e., the number of bit errors in the receiving message divided by the total number of transferred bits) in communication channel can help the attacker to correct the false prediction in some bits as well as to subvert the correct prediction in that of the other bits. According to Figs. 11 and 12, for HD more than 6 and less than 20, both the False-Negative and False-Positive rates are zero. As a result, we can recommend the higher value of HD to decrease the False-Negative rate, while the False-Positive rate remains in zero percentage. value of HD to decrease the False-Negative rate, while the False-Positive rate remains in zero percentage.

5. Conclusion

In this paper, a new PUF-based authentication mechanism called DC-PUF is proposed. The main goal was to weaken the correlation between the challenge-response pairs (CRPs) to harden or significantly postpone the replication or cloning process, which can be exploited by the attacker. Accordingly, we take advantage of the dependency chain mechanism which obfuscates the relation between CRPs so that the attacker cannot predict any relevant dependency. The experimental results imply that even a sophisticated CNN network with a large number of data-set cannot successfully clone the PUF model with more than 58% of accuracy. This accuracy is considerably larger for that of the other PUFs which resulted in more than 95% accuracy when the CNN-based attack is conducted. Moreover, the DC-PUF structure incurs a less hardware overhead compared to the existing designs, and the uniformity and uniqueness remain almost the same as the ideal value. The reliability of the proposed DC-PUF in terms of PUF instability is also evaluated and can be controlled by adjusting the number of authentication rounds and the percentage of response bits that are selected for the dependency chain. Moreover, the results showed that the average of the False-Negative rate of 1% is obtained for HD = 3 in the presence of channel variations, while the False-Positive rate remains in zero percentage. This method provides a lightweight mechanism to securely authenticate the resource constraint IoT devices while resisting against sophisticated machine-learning attacks.

CRediT authorship contribution statement

Abolfazl Sajadi: Conceptualization, Software, Validation,

Resources, Data curation. **Ahmad Shabani:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing. **Bijan Alizadeh:** Supervision, Investigation, Validation, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Amsaad, F., et al., 2021. Enhancing the performance of lightweight configurable PUF for robust IoT hardware-assisted security. IEEE Access 9, 136792–136810. https://doi.org/10.1109/ACCESS.2021.3117240.

Ashtari, A., Shabani, A., Alizadeh, B., 2019. A new RF-PUF based authentication of internet of things using random forest classification. In: Proceedings of 16th International ISC Conference on Information Security and Cryptology. ISCISC, pp. 21–26. https://doi.org/10.1109/ISCISC48546.2019.8985161. Aug. 2019.

Ashtari, A., Shabani, A., Alizadeh, B., 2022. A comparative study of machine learning classifiers for secure RF-PUF-based authentication in internet of things. Microprocess. Microsyst. 93, 104600. https://doi.org/10.1016/j.micpro.2022.104600.

Becker, G.T., 2015. The gap between promise and reality: on the insecurity of XOR arbiter PUFs. In: Güneysu Tim, H., Handschuh (Eds.), Cryptographic Hardware and Embedded Systems – CHES 2015. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 535–555.

Becker, G.T., Kumar, R., 2014. Active and Passive Side-Channel Attacks on Delay Based PUF Designs. IACR Cryptology ePrint Archive, p. 287, 2014, [Online]. Available: http://eprint.iacr.org/2014/287.pdf.

Chatterjee, B., Das, D., Sen, S., 2018. "RF-PUF: IoT security enhancement through authentication of wireless nodes using in-situ machine learning,". In: Proceedings of the 2018 IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2018. Institute of Electrical and Electronics Engineers Inc., Jun, pp. 205–208. https://doi.org/10.1109/HST.2018.8383916.

Delvaux, J., 2019. Machine-Learning Attacks on PolyPUFs, OB-PUFs, RPUFs, LHS-PUFs, and PUF-FSMs. IEEE Trans. Inf. Forensics Secur. 14 (8), 2043–2058. https://doi.org/10.1109/TIFS.2019.2891223.

Delvaux, J., Verbauwhede, I., Gu, D., 2017. Security Analysis of PUF-Based Key Generation and Entity Authentication [Online]. Available: https://www.esat.kuleuven.be/cosic/publications/thesis-290.pdf.

El-Hajj, M., Fadlallah, A., Chamoun, M., Serhrouchni, A., 2021. A taxonomy of PUF Schemes with a novel Arbiter-based PUF resisting machine learning attacks. Comput. Network. 194, 108133 https://doi.org/10.1016/J.COMNET.2021.108133. Jul.

Feldhofer, M., Dominikus, S., Wolkerstorfer, J., 2004. Strong authentication for RFID systems using the AES algorithm. Lect. Notes Comput. Sci. 3156, 357–370. https://doi.org/10.1007/978-3-540-28632-5_26.

Gao, Y., et al., 2016. Obfuscated challenge-response: a secure lightweight authentication mechanism for PUF-based pervasive devices. In: 2016 IEEE International Conference on Pervasive Computing and Communication Workshops, PerCom Workshops. Institute of Electrical and Electronics Engineers Inc., Apr https://doi.org/10.1109/PERCOMW.2016.7457162, 2016.

Gassend, B., Lim, D., Clarke, D., van Dijk, M., Devadas, S., 2004. Identification and authentication of integrated circuits. Concurr Comput. Pract. Ex. 16 (11), 1077–1098. https://doi.org/10.1002/cpe.805.

Hemavathy, S., Bhaaskaran, V.S.K., 2023. Arbiter PUF—a review of design, composition, and security aspects. IEEE Access 11, 33979–34004. https://doi.org/10.1109/ACCESS.2023.3264016.

Holcomb, D.E., Burleson, W., Fu, K., 2007. Initial SRAM State as a Fingerprint and Source of True Random Numbers for RFID Tags.

Keras, 2022. The Python Deep Learning API accessed Feb. 03.

Khormizi, F., Shabani, A., Alizadeh, B., 2022. Hardware patching methodology for neutralizing timing hardware Trojans using vulnerability analysis and time borrowing scheme. IEEE Transactions on Circuits and Systems II: Express Briefs 69 (6), 2937–2941. https://doi.org/10.1109/TCSII.2022.3160489.

Konigsmark, S.T.C., Chen, D., Wong, M.D.F., 2016. PolyPUF: physically secure self-divergence. IEEE Trans. Comput. Aided Des. Integrated Circ. Syst. 35 (7), 1053–1066. https://doi.org/10.1109/TCAD.2015.2488493. Jul.

Lee, J.W., Lim, D., Gassend, B., Suh, G.E., van Dijk, M., Devadas, S., 2004. A technique to build a secret key in integrated circuits for identification and authentication applications. In: IEEE Symposium on VLSI Circuits. Digest of Technical Papers, pp. 176–179. https://doi.org/10.1109/vlsic.2004.1346548.

Lim, D., Lee, J.W., Gassend, B., Suh, G.E., van Dijk, M., Devadas, S., 2005. Extracting secret keys from integrated circuits. IEEE Trans Very Large Scale Integr VLSI Syst 13 (10), 1200–1205. https://doi.org/10.1109/TVLSI.2005.859470.

Maes, R., Maes, R., 2013. Physically unclonable functions: concept and constructions. In: Physically Unclonable Functions. Springer Berlin Heidelberg, pp. 11–48. https://doi.org/10.1007/978-3-642-41395-7_2.

Majzoobi, M., Koushanfar, F., Potkonjak, M., 2008a. Lightweight Secure PUFs," *IEEE/ACM International Conference on Computer-Aided Design*. Digest of Technical Papers, ICCAD, p. 670. https://doi.org/10.1109/ICCAD.2008.4681648. –673.

Majzoobi, M., Koushanfar, F., Potkonjak, M., 2008b. Testing techniques for hardware security. Proceedings - Int Test Conf. https://doi.org/10.1109/TEST.2008.4700636.

Majzoobi, M., Koushanfar, F., Potkonjak, M., 2009. Techniques for design and implementation of secure reconfigurable PUFs. ACM Trans. Reconfigurable Technol. Syst. (TRETS) 2 (1). https://doi.org/10.1145/1502781.1502786. Mar.

Majzoobi, M., Rostami, M., Koushanfar, F., Wallach, D.S., Devadas, S., 2012. Slender PUF Protocol: A Lightweight, Robust, and Secure Authentication by Substring Matching. Proceedings - IEEE CS Security and Privacy Workshops, pp. 33–44. https://doi.org/10.1109/SPW.2012.30. SPW 2012.

Mandel Yu, M.D., M'Raihi, D., Verbauwhede, I., Devadas, S., 2014. A noise bifurcation architecture for linear additive physical functions. In: Proceedings of the 2014 IEEE International Symposium on Hardware-Oriented Security and Trust. HOST, pp. 124–129. https://doi.org/10.1109/HST.2014.6855582, 2014.

Mispan, M.S., Su, H., Zwolinski, M., Halak, B., 2018. Cost-efficient design for modeling attacks resistant PUFs. In: Proceedings of the 2018 Design, Automation and Test in Europe Conference and Exhibition, DATE 2018. Institute of Electrical and Electronics Engineers Inc., Apr, pp. 467–472. https://doi.org/10.23919/DATE.2018.8342054.

Potdar, K., T, S., C, D., 2017. A comparative study of categorical variable encoding techniques for neural network classifiers. Int. J. Comput. Appl. 175 (4), 7–9. https://doi.org/10.5120/IJCA2017915495. Oct.

Ruhrmair, U., Solter, J., 2014. PUF Modeling Attacks: an Introduction and Overview. EDAA, pp. 1–6. https://doi.org/10.7873/date.2014.361. Apr.

Rührmair, U., Sölter, J., 2014. PUF modeling attacks: an introduction and overview. In: Proceedings -Design, Automation and Test in Europe, DATE. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.7873/DATE2014.361

Ruhrmair, U., et al., 2013. PUF modeling attacks on simulated and silicon data. IEEE Trans. Inf. Forensics Secur. 8 (11), 1876–1891. https://doi.org/10.1109/TIFS.2013.2279798.

Rührmair, U., et al., 2013. PUF modeling attacks on simulated and silicon data. IEEE Trans. Inf. Forensics Secur. 8 (11), 1876–1891. https://doi.org/10.1109/TIFS.2013.2279798.

Sabri, M., Shabani, A., Alizadeh, B., 2021. SAT-based integrated hardware trojan detection and localization approach through path-delay analysis. IEEE Transactions on Circuits and Systems II: Express Briefs 68 (8), 2850–2854. https://doi.org/10.1109/TCSII.2021.3074549.

Sahoo, D.P., Saha, S., Mukhopadhyay, D., Chakraborty, R.S., Kapoor, H., 2014. Composite PUF: a new design paradigm for physically unclonable functions on FPGA. In: Proceedings of the 2014 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2014. IEEE Computer Society, pp. 50–55. https://doi.org/10.1109/HST.2014.6855567.

Sahoo, D.P., Mukhopadhyay, D., Chakraborty, R.S., Nguyen, P.H., 2018. A multiplexer-based arbiter PUF composition with enhanced reliability and security. IEEE Trans. Comput. 67 (3), 403–417. https://doi.org/10.1109/TC.2017.2749226.

Shabani, A., Alizadeh, B., 2020. PODEM: a low-cost property-based design modification for detecting hardware trojans in resource-constraint IoT devices. J. Netw. Comput. Appl. 167, 102713 https://doi.org/10.1016/J.JNCA.2020.102713.

Shabani, A., Alizadeh, B., 2021. Enhancing hardware trojan detection sensitivity using partition-based shuffling scheme. IEEE Transactions on Circuits and Systems II: Express Briefs 68 (1), 266–270. https://doi.org/10.1109/TCSII.2020.3001263.

Suh, G.E., Devadas, S., 2007. "Physical Unclonable Functions for Device Authentication and Secret Key Generation,", p. 9. https://doi.org/10.1145/1278480.1278484.

Trout, J.E., Betts, C.P., 1988. Security engineering. Mil. Eng. 80 (522), 368–371. https://doi.org/10.4018/978-1-59140-911-3.ch011.

Tuyls, P., Schrijen, G.J., Škorić, B., van Geloven, J., Verhaegh, N., Wolters, R., 2006. Read-proof hardware from protective coatings. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer Verlag, pp. 369–383. https://doi.org/10.1007/11894063_29.

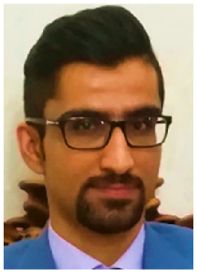
Vijayakumar, A., Kundu, S., 2015. A novel modeling attack resistant PUF design based on non-linear voltage transfer characteristics. In: Proceedings -Design, Automation and Test in Europe, DATE. Institute of Electrical and Electronics Engineers Inc., pp. 653–658. https://doi.org/10.7873/date.2015.0522

Wen, Y., Lao, Y., 2018. PUF modeling attack using active learning. In: Proceedings - IEEE International Symposium on Circuits and Systems. Institute of Electrical and Electronics Engineers Inc., Apr https://doi.org/10.1109/ISCAS.2018.8351302.

Ye, J., Hu, Y., Li, X., 2016. RPUF: physical unclonable function with randomized challenge to resist modeling attack. In: Proceedings of the 2016 IEEE Asian Hardware Oriented Security and Trust Symposium. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/AsianHOST.2016.7835567, 2017.

Ye, J., Guo, Q., Hu, Y., Li, H., Li, X., 2018. Modeling attacks on strong physical unclonable functions strengthened by random number and weak PUF. In: Proceedings of the IEEE VLSI Test Symposium. IEEE Computer Society, pp. 1–6. https://doi.org/10.1109/VTS.2018.8368627.

Abolfazl Sajadi received his M.S. degree in digital electronic systems from University of Tehran in 2021. As an active member of the Design, Verification, and Debugging of Embedded Systems (DVDES) Laboratory at Tehran University, he contributed to various research projects. Presently, he is pursuing his Ph.D. degree at Leiden University. His primary research interests include cryptographic primitives and digital hardware design. Additionally, he delves into the fields of hardware security, system-level hardware/software co-design, hardware acceleration, and hardware solutions for deep learning applications.



Ahmad Shabani received the B.S. degree in electrical engineering from Shiraz University, Shiraz, Iran, in 2014, the M.S. degree in digital electronic systems from Shahid Beheshti University, Tehran, Iran, in 2016, and the Ph.D. degree in digital electronic systems from University of Tehran, in 2021. He is currently an Adjunct Lecturer at University of Tehran, and also has a joint collaboration with the Design, Verification and Debugging of Embedded Systems (DVDES) laboratory of Tehran University. His research interests include image processing and video coding standards, medical image processing, hardware security, Hardware Trojans, and low-power & high-speed digital VLSI circuits.

Bijan Alizadeh (SM'13) received his Ph.D. degree in electrical and computer engineering from the University of Tehran, Iran, in 2004. He was with the School of Electrical Engineering, Sharif University of Technology, Iran, as an Assistant Professor from 2005 to 2007 and VDEC, The University of Tokyo, Japan, as a Research Associate from 2007 to 2010. In 2011, he joined the School of Electrical and Computer Engineering at the University of Tehran as an assistant professor, where he has been an Associate Professor since August 2017. He has authored or co-authored over 130 publications in international scientific journals and conferences. He has been engaged in the research and development of VLSI systems, reconfigurable computing, formal verification and debug, hardware security, and high-level synthesis.