A Lightweight and Practical Anonymous Authentication Protocol Based on Bit-Self-Test PUF

Yang An, Yuejiao Zhang, Wenjun Cao, Zhiyan Tong and Zhangqing He *

Abstract: Physical unclonable function (PUF), a cryptographic primitive, has recently been used in protocol design because it can ensure a tamper-evident feature. In many PUF-based protocol schemes, helper data algorithms (HDA) or fuzzy extractors (FE) are used to generate strong keys from unreliable PUF responses. However, these methods inevitably introduce complex error correction techniques, which not only increase the overhead of embedded devices but also pose some security risks. We propose a novel HDA technology, which does not use any high-overhead error correction mechanism, greatly reducing the implementation complexity and execution overhead. The novel HDA exploits the strategy of bit-self-test (BST) and the PUF can extract the robust responses by using the real-time generated reliable flags, and then an entropy extractor is used to generate the reliable and random key with high entropy. Based on this novel HDA, we design a lightweight anonymous authentication protocol. The protocol uses pseudo-random function (PRF) and XOR operation instead of the traditional hash function and symmetric encryption algorithm, which ensures security while reducing the overhead. Moreover, the proposed protocol does not require the server to store a large number of challenge–response pairs (CRPs), which reduces the storage overhead on the server while avoiding the risk of leakage of CRPs. Moreover, the device identity ID is updated during each round of the authentication process, which prevents the device from being tracked and protects the privacy of the device. The implementation and performance analysis of the protocol prototype on a Zynq-7000 SoC XC7Z010 FPGA shows that the proposed scheme solves the problems encountered with existing schemes and has additional security properties.

Keywords: BST-PUF; HDA; authentication protocol; lightweight

1. Introduction

Nowadays, with the popularity of IoT applications, the use of lightweight devices such as Radio Frequency Identification (RFID) and Near Field Communication (NFC) is becoming more widespread. When these devices communicate information over open and insecure communication channels, there is a high risk that data can be easily intercepted or tampered with by attackers, thus becoming an entry point for attackers to access IoT devices or personal data. This flexible access makes it possible for any insecure device endpoint to have an impact on the security of the entire IoT system [1–3]. Therefore, it is particularly important to design a lightweight, efficient, and reliable mutual authentication protocol. The unpredictable output of PUFs makes them a good choice for security primitives in lightweight systems such as RFID and NFC. Compared with traditional protocols, the PUF-based protocols have more security characteristics, such as preventing device-side tampering, probing, and cloning attacks [4–11].

PUF is a special physical entity that uses uncontrollable random process deviations in the manufacturing process of an integrated circuit to generate a unique mapping relationship, to which an unpredictable response \( R \) can be generated in real-time by inputting a random challenge \( C \) to the PUF circuit [12]. Due to this feature, the key information...
required for authentication does not need to be stored as plaintext or ciphertext in the non-volatile memory (NVM) inside the chip but is generated in real time by the PUF circuit inside the chip. Moreover, due to the unclonable property of the PUF, even if an attacker could probe the circuit structure of the chip, he would not be able to access the key information, thus ensuring the security of data storage. However, PUF responses are highly susceptible to environmental factors such as voltage, temperature, and pressure [13], so the original PUF response is not reliable enough and cannot be used directly in authentication, random number generation, and key generation. Existing schemes commonly use error correction codes (ECCs) [14–18] to recover the correct key from the original PUF response. Researchers have proposed techniques such as Pattern Matching [19], Index-Based Syndrome (IBS) [20], FE [17], or Multiple Evaluation (ME) [15] for removing noise from the PUF response or enhancing the reliability of the PUF response. However, these methods require the introduction of complex algorithms for generating helper data or involve complex mathematical operations, thus making the implementation process very complex with high implementation overheads, which is limited in the application fields of lightweight overhead. Moreover, there are some potential security issues with using the ECCs, such as the fact that the implementation is complicated and the public helper data may disclose the information of the keys [21–23].

In addition, the authentication protocol is vulnerable to an attacker stealing enough CRPs of the PUF from an insecure channel during the execution process to simulate the PUF mapping relationship and perform a modeling attack on the PUF [24] to decrypt the important information in the authentication, resulting in security vulnerability on both sides of the authentication. Researchers have increased the strength of PUF circuits against modeling attacks by adding non-linear units to the PUF [25] to increase the complexity of their mapping, or by encrypting the CRPs and helper data during the authentication process [26] before being transmitted. However, the introduction of various anti-modeling methods and encryption algorithms that alter the PUF structure undoubtedly increases the hardware overhead, which is not suitable for low-cost RFID with less than 2000 gates circuits. Therefore, the design of a suitable encryption algorithm and a lightweight authentication protocol is essential for RFID systems that require high security and low overhead.

Therefore, this paper starts the research from the lightweight PUF reliability enhancement method and the helper data algorithm based on PUF, proposing a lightweight HDA model based on BST-PUF [27]. In the key generation and recovery process, a reliable response is extracted from the original response $R$ by using the reliable flag $F$, and then an entropy extractor is used to generate a high entropy key that is reliable and uniform. The algorithm does not use any error correction codes, which greatly reduces the implementation complexity and execution overhead that can generate a large number of highly reliable keys with less hardware resource consumption, which significantly reduces the application cost of PUF. Moreover, the helper data and the PUF response are independent of each other, and will not reveal any information about the key. In addition, a lightweight practical anonymous authentication protocol is designed based on this HDA and enables mutual authentication between the terminal device containing a PUF entity and the server. The protocol uses the reliable flag to pre-process PUF responses, instead of the original error correction mechanism. While using PRF instead of hash functions and using XOR operation instead of symmetric encryption algorithms, it significantly reduces the overhead while ensuring security. During protocol execution, the server does not need to store a large number of CRPs, which reduces the storage overhead on the server side and avoids the risk of leakage of CRPs; during protocol authentication, the device does not transmit its ID explicitly, which prevents the device from being tracked and protects privacy; after security analysis, the proposed protocol proves to be resistant to various attacks against the protocol and PUF, such as modeling attacks, cloning attacks, and desynchronization attacks. The proposed protocol is also verified on the Zynq-7000 SoC XC7Z010 FPGA development board, and the experimental results show that the proposed protocol has significant advantages over similar protocols in terms of implementation overhead. The schematic diagram
process of the proposed lightweight anonymous authentication protocol based on the novel HDA is shown in Figure 1.

**Figure 1.** The schematic diagram process of the proposed protocol.

### 2. Related Work

#### 2.1. Analysis and Comparison of Lightweight HDAs

We have analyzed and compared some existing lightweight HDA methods [6,19,24,28]. A pattern matching technique is proposed in [19] for processing responses, which can reliably provide and regenerate a finite and accurate sequence of bits for use by cryptographic applications by employing one or more PUF circuits. A 4-XOR Arbiter PUF circuit is used in [19] to prevent modeling attacks, but its error correction performance is degraded compared to traditional error correction mechanisms, which may lead to unsuccessful protocol authentication. In [6], a method for implementing lightweight authentication based on substring matching was designed using the idea of pattern matching in [19]. This method does not disclose the full PUF response on the communication channel, but displays a random subset of the response and sends it for authentication. After experimentation and analysis, it is concluded that the hardware overhead of the method satisfies the lightweight requirement, but the method indicates that if a close enough random subset is found to match during the authentication process, the authentication can be indicated as successful, which is a security hazard. Although the proposed PUF is resistant to most of the machine learning attacks, it is not effective against some advanced machine learning attacks, such as CMA-ES [29,30], and also the PUF-based protocol in [6] has weak resistance to physical probing attacks, man-in-the-middle attacks, and desynchronization attacks. While protocol designs based on this approach have some advantages in overhead, its shortcomings are obvious insecurity, and the approach [28] has similar advantages and disadvantages to what [6] has, in that it reduces overhead but is not fully secure against various attacks. A new self-correcting error correction mechanism is designed in [24] for traditional BCH error correction to correct output response errors generated by environmental transforma-
tions, in which each PUF response is mapped to a key during the design phase, and the actual key is determined from the PUF response only after the chip is fabricated. Since the chip foundry is unaware of the key, the proposed method can prevent forgery. It uses a two-stage finite state machine and helper data to make the actual response jump towards the correct response so that the final response is sufficiently reliable. However, the disadvantage is that the helper data increase with the number of bits in the response and need to be stored in the NVM on the device side, which consumes a lot of overhead, and if the HD between the actual response and the correct response is too large, it will lead to a further increase in overhead. This method is not suitable for applications requiring a large number of response bits and is more suited to local than remote authentication.

2.2. Analysis and Comparison of PUF-Based Protocols

PUF-based protocols fall into two main categories, those used between devices and servers and those used between two different devices. In the first category of protocols, Suh, G. Edward et al. [4] proposed a low-cost PUF authentication mechanism for RFID, where the tag containing the PUF produces a sufficient number of CRPs in a secure environment and stores them in the reader’s database, but the authentication mechanism transmits challenges and responses in plaintext during the information transfer process, which means the attacker can capture a large number of CRPs to model the PUF mapping and perform modeling attacks on PUFs. Meng-Day (Mandel) Yu et al. propose a lightweight authentication protocol with locking techniques for security issues such as machine learning on PUFs [4]. They limit the number of available CPRs; only trusted entities or servers can authorize access to new CPRs, and the method introduces random numbers on the device to prevent reliability-based attacks, increasing the security of the protocol. However, this method only supports a limited number of authentication cycles (approximately 10,000), which makes it unsuitable for IoT-based device deployments. Kong, J et al. proposed an embedded platform authentication protocol based on the arithmetic logic unit PUF [5], which has good resistance to bypass attacks and overclocking attacks and does not involve the storage of important information and CRPs. However, its use of PDL logic to improve the uniformity of PUF responses and the inclusion of a fuzzy extractor to enhance the reliability of PUF responses resulted in a large implementation overhead on the device.

Mehrdad Majzoobi et al. [6] proposed a lightweight PUF authentication protocol based on the substring matching technique, but this protocol does not implement mutual authentication between the two sides of the protocol while ensuring light weight, which has certain security risks. In [31], Aydin Aysu et al. proposed a provably secure PUF-based authentication protocol and gave an end-to-end design. The authors used FE based on ECCs to obtain PUF robust responses; moreover, they used a key update mechanism to achieve mutual authentication, satisfying most of the important required security requirements. However, there are some potential security and overhead issues with using the ECCs, such as that the implementation is complicated, the hardware execution overhead is large, and the public helper data may disclose the information of the keys [21–23]. In addition, when the FE generates the robust response \( r_1 \), the helper data \( hd \) will be generated at the same time, and the helper data need to be encrypted by the SKE module using the SIMON algorithm [32]. However, when the FE and SKE modules are implemented on the single-chip microcomputer MSP430, they occupy 49.26% of the total memory. Moreover, the execution time of the SIMON algorithm will increase exponentially with the number of bits of the data being encrypted, as in [31]; for 64-bit and 128-bit data, the clock cycle changes from 39,583 to 252,829. Therefore, there are some deficiencies in [31], both in terms of the overhead and execution time of the protocol, which is not suitable for resource-constrained RFID devices. Gope et al. [33] proposed a lightweight authentication protocol for RFID systems based on PUF that can guarantee forward and backward security; it also can resist DoS attacks and physical attacks, and meet most of the protocols’ required security performance. Two authentication protocols are proposed in the paper, each considering the presence or absence of noise in the PUF. Although the hardware overhead of the protocol is very friendly in the case of using an ideal PUF, it is clear that a PUF without noise does not
exist in practice. In the second noisy PUF scheme, the FE is still used in the process of extracting the robust response of the PUF, which means that the execution overhead is significantly increased and the scheme does not have the advantage of low overhead in practical applications.

Refs. [14,34] refer to the second category of protocols that are used between two different devices. Urbi Chatterjee et al. [14] combined the ideas of identity cryptography and hash functions to design a lightweight protocol based on PUFs. The protocol performs non-display processing of CRPs to prevent their leakage, which may affect the protocol security issues, thus making the protocol resistant to modeling attacks. However, they have designed a BCH encoder circuit to handle the noise to the PUF response, which makes the device overhead too high and limits the application of the protocol. The approach proposed in [34] is an improvement on the approach in [35], where the protocol proposed in [35] is vulnerable to man-in-the-middle, replay, and impersonation attacks. Ref. [34] proposes an alternative to [35] by designing an Elliptic Curve Qu-VanStone-based PUF protocol for solving the security issues faced by the protocol in [35]. Moreover, a more efficient Elliptic Curve signature secrecy scheme is proposed in the communication phase. The protocol proposed in [34] has a high level of security; the scheme is an improvement on [35] and resists the various security issues faced in the protocol. However, this scheme does not study how to reduce the high hardware overhead caused by PUF reliable response extraction.

3. Proposed Scheme

The proposed protocol is designed based on BST-PUF, which can generate the reliability flag [27], and the PUF model is shown in Figure 2. The PUF can generate a response $R$ and a reliable flag $F$ when the challenge $C$ is input, and $F$ can be used as the reliability of a response to pre-process the original PUF response. Figure 3 shows the process of extracting robust responses using the reliability flag $F$.

3.1. Lightweight Helper Data Algorithms

Based on the BST-APUF, we propose a lightweight HDA that can be used to extract stable cryptographic keys, or for security authentication. The proposed HDA schematic diagram process is shown in Figure 4. In the key generation process, the original challenge $c$ will be input into the LFSR to generate a challenge collection of the BST-PUF, and the BST-PUF will generate the original output $r_{m1}$ and the reliability flag $F$. The reliability flag $F$ is used as the helper data $h_1$ in this HDA to perform the key extraction and recovery operation. Moreover, the $r_{m1}$ and $h_1$ will be input to the robust response extractor to generate the highly reliable robust response $r_{c1}$. Although, after extracting, the PUF response $r_{c1}$ is reliable enough, the disclosure of the helper data in the response error correction process can lead to a large entropy loss, and the stronger the error correction capability, the relatively
larger the entropy loss [36]. It even causes the PUF response to be unavailable for key generation when the entropy value is too low. Therefore, the robust response of the PUF needs to be processed by the entropy extractor to generate the high-entropy key $K$.

In the key recovery process, the original challenge $c$ and the helper data $h_1$ are, respectively, input to the LFSR and the robust response extractor. Because the PUF output is susceptible to environmental factors, for the same challenge collection $c_1$ generated by the LFSR, the original output $r_{m2}$ of the PUF is not exactly the same as the PUF output $r_{m1}$. However, after processing by the $h_1$, the generated robust response $r_{c2}$ has no more than $10^{-9}$ bit error rate with $r_{c1}$, which ensures the stability and high efficiency of key recovery.

![Figure 4. The HDA based on BST-PUF.](image)

### 3.2. Proposed Authentication Protocol

In this paper, an authentication protocol is designed based on BST-PUF [27], and the security authentication can be performed between a cryptographic device with a PUF entity and the server. There is no need to use FE or any ECCs on the device during protocol execution, while pseudorandom functions are used instead of symmetric encryption algorithms, which further reduces the execution overhead of the device. In later sections, we will provide the formal security analysis of it. In order to describe the proposed protocol, the following formal definition is given.

- $y \leftarrow A$: $A$ is a set, and $y$ is a randomly selected element from $A$;
- $y = A(x)$: $A$ is a probabilistic algorithm and $y$ is the output produced by inputting $x$ to algorithm $A$;
- TRNG: Truly random number generator, which can derive a truly random number sequence;
- $PRF, PRF'$: Pseudorandom function $K' \times D' \rightarrow R$ uses the secret key $k \in K'$ and the message $m \in D'$ as input, an output $pr \in R'$;
- $KEY.Ext$: $P' \rightarrow K' \times L'$, uses PUF challenge $c' \in P'$ as input, generates a key $k \in K'$ and helper data $h \in L'$. The $KEY.Ext$ contains a BST-APUF circuit, which is described in Section 3.1;
- $KEY.Rec$: $P' \times L' \rightarrow K'$, takes as input the PUF noisy response $r' \in P'$ and helper data $h \in L'$ and recovers the key $k \in K'$.

The protocol execution process includes the setup phase and authentication phase. The setup phase is executed in a secure environment. The device is required to register with
the server, for which it generates a random challenge \( c_1 \) and a temporary ID and sends it to the device. The device produces the key and helper data \((k_1, h_1) := KEY.Ext(c_1)\) in the device using \(KEY.Ext\) and sends them to the server to set the initial shared key \( k_1 = k_{old} \) and initial temporary ID \( ID_1 = ID_{old} \). When the server and the device want to perform mutual authentication and message transmission, then the authentication phase begins. The specific execution process of mutual authentication is shown in Figure 5.

**Figure 5.** The proposed protocol.

In the authentication phase, the server generates and sends a random number \( \{m_1\} \) to the device. The device receives \( \{m_1\} \) and generates another random number \( \{m_2\} \), and
then reads the stored challenge $c_1$ and helper data $h_1$ from NVM and inputs them into the $KEY.\text{Rec}$ to generate the key $k_1$. Using $k_1$, $m_1 \parallel m_2$ as input, the PRF generates five pseudorandom numbers $s_1, \ldots, s_5$ for subsequent authentication and encryption operation. At the same time, the TRNG on the device generates a random challenge $c_2$, which is inputted into the $KEY.\text{Exc}$ to generate key $k_1$ and helper data $h_2$. Performing the XOR operation on $k_2$ and $s_2$ can generate the encrypted data $u$. Using $s_3$, $m_2 \parallel u$ as the input, the PRF can generate a pseudorandom number $v$.

Then, the message $\{s_3, m_2, u, v, ID_1\}$ will be sent to the server for authentication. Among them, the data $v$ are used as the Message Authentication Code (MAC). After the server receives the above information, it checks the temporary $ID_1$ to find the corresponding $k_1$ and $k_{old}$ in the database, and uses the key $k_1$ and $m_1 \parallel m_2$ as input to generate five pseudorandom numbers $s_1', \ldots, s_5'$ using the same PRF as on the device. If the device is legal, then $s_1', \ldots, s_5'$ will be the same as $s_1, \ldots, s_5$. Therefore, the server should first verify whether $s_1'$ and $s_1$ are equal; if they are equal and the MAC $v = \text{PRF}(s_2', m_2 \parallel u)$ are verified, the server will successfully authenticate the device. After this, it performs the later operation $k_2 := s_4' \oplus u$, $ID_2 := s_2' \oplus ID_1$, updates the stored data on the server: $\{ID_1, k_1\}, \{ID_{old}, k_{old}\} := \{ID_2, k_2\}, \{ID_1, k_1\}$, and sends the random number $s_4'$ to the device.

We need to consider a situation wherein $s_1'$ cannot be received by the device due to some attacks. This will make the device unable to update the authentication information and cause the information update with the server to not be synchronized. To solve this problem, if $s_1'$ is not equal to $s_1$, the server will use $k_{old}$ which replaces $k_1$ as the input to the PRF, and computes $s_1', \ldots, s_5'$ again, verifying whether $s_1'$ is equal to $s_1$. If they are still not equal, the authentication will fail and the server will return a random value to the device. In this way, the proposed authentication protocol can resist DoS and desynchronization attacks.

After the device receives the data $s_4'$, it will verify whether the $s_4'$ is same as the pseudorandom numbers $s_{4}$, and if they are equal, it performs the operation $ID_2 := s_{4} \oplus ID_1$ and updates the stored data $ID_1, (c_1, h_1) := ID_2, (c_2, h_2)$ on the device as the authentication information for the next authentication; otherwise, the authentication fails.

4. Formal Security Analysis

In this section, we list some possible security threats and attacks on the PUF-based protocols and provide a formal security analysis for the proposed protocols.

4.1. Probable Security Threats and Prospective Attacks

The insecure communication channels between the device and the server make the PUF-based authentication protocol that applies to RFID systems vulnerable to the following threats and attacks on security.

Tracking Attack: The attacker can trace the device by obtaining its ID through an insecure radio frequency channel and thus know the position and operation of the user. To secure against tracking attacks, the device must be anonymous throughout the authentication process.

Forward Security Threats: If an attacker is somehow able to obtain secret information about the device, then the attacker will be able to trace all communications between the device and the server. Therefore, the PUF used in the protocols should not only have good uniqueness, reliability, and randomness but should also have good resistance to physical attacks and machine learning, to ensure that the attacker cannot obtain the secret information generated in real time by the PUF in the device at the time of authentication, thus ensuring the backward untraceability of the protocol.

DoS and Desynchronization Attacks: By blocking the information between the device and the server, the attacker could cause desynchronization problems. In many RFID-based authentication protocols, both the server and the device need to update their secret information to ensure forward secrecy. Therefore, when the response message from the server or device side is blocked, the other side cannot receive the message to update the
authentication information. This situation will result in one side updating the key and the other side failing to update the key, which leads to desynchronization.

**Impersonation Attacks:** The attacker intercepts information between the device and the server and reuses this information to impersonate a legitimate device or server to create separate connections with each end of the communication and exchange the data it receives. In this way, the attacker can make both ends of the communication believe they are talking directly to each other over a private connection, when in fact the entire session is fully controlled by the attacker.

**Physical Attack and Cloning Attacks:** Attackers compromise the chip and access its memory through a variety of attacks. For example, they can reverse-engineer the chip by opening the chip package and using optical images to reconstruct the layout, or they can probe the signals on the chip bus using micro-probes without destroying the chip, or using the side-channel attack to probe the chip contents directly from the outside. Moreover, since the chip in most devices is not tamper-proof, an attacker can build a cloned chip used for authentication by eavesdropping on communication channel information and physical attacks, which causes the device to be authenticated by the server as a legitimate device.

### 4.2. Security Analysis

First, we consider an adversary $A$ who has control over the radio communication channel between the device $D$ and the server $S$. $A$ is required to model the following query methods in polynomial time.

- **Transmit** $(S, M_1, A_2, M_3)$: After $A$ receives $M_1$ from $S$, $A$ sends message $A_2$ to $S$ and receives message $M_3$ from $S$;
- **Enquiry** $(D, A_1, M_2, A_3)$: $A$ sends $A_1$ to the device, receives $M_2$ from the device, and then sends $A_3$ to the device;
- **Execute** $(D, S)$: When executing a protocol between $D$ and $S$, $A$ needs to intercept this channel;
- **Blocking** $(A)$: $A$ is required to block a part of the protocol and interrupt the synchronization between $D$ and $S$;
- **Reveal** $(D)$: This query simulates the ability of the adversary to compromise the device and obtain confidential information stored in its memory.

$A$ can invoke the Transmit, Enquiry, Execute, and Blocking queries multiple times, but can only invoke the Reveal query once, since obtaining confidential information stored in the device memory can lead to physical destruction of the device chip.

**Lemma 1.** The proposed protocol can resist modeling attacks or machine learning.

The modeling attack on the PUF is predicated on obtaining a sufficient number of CRPs, whereas, in the proposed protocol, $c$ and $h$ are not transmitted in plaintext during protocol execution; the response $r$ is encrypted using a non-linear PRF when transmitted in the communication channel, and the attacker cannot have the opportunity to obtain enough CRPs on the communication channel, which makes it exponentially more difficult for an attacker to simulate the mapping relationship between $c$ and $r$, and modeling attacks on PUF become particularly difficult [25]. Moreover, the attacker who wants to obtain enough CRPs to perform machine learning and modeling attacks on the PUF can only obtain $c$ and $h$ stored in the NVM by probing the chip’s memory using the Reveal query, but this will simultaneously destroy the PUF circuit integrated into the chip and change its mapping relationships, so even if the attacker obtains $c$ and $h$, they will not be able to recover the response $r$. Thus, the proposed protocol based on BST-PUF has sufficient resistance to modeling attacks or machine learning.

**Lemma 2.** The secret data on the device will not be compromised in the proposed authentication protocol, even if the Reveal query is invoked.
The device does not need to store important information about the key, and the $k_1$ associated with response parameters is used as a shared key for both parties, which is generated with $c_1$ and $h_1$ as the input of the $KEY.Rec$ in real time. Thus, if adversary $A$ uses the Reveal query to obtain secret information on the device’s memory, it may only be able to steal helper data $h_1$ and challenge $c_1$. According to Lemma 1, we know that adversary $A$ cannot recover the response $r_1$ and thus cannot generate the key $k_1$. Therefore, the leakage of the challenge $c_1$ and helper data $h_1$ does not lead to the leakage of the response $r_1$. Moreover, any physical means of accessing the PUF will alter its CRP mapping, ultimately invalidating the device’s legal identity.

Lemma 3. The adversary $A$ invokes the Reveal query and also fails to associate the device’s identity.

Because the proposed protocol uses temporary ID, the adversary cannot associate the device by obtaining ID information. Moreover, because the data used for authentication at the device are updated after each successful authentication, and the challenge is randomly generated by the TRNG, the key information must be known if adversary $A$ wants to associate the identity of the device. However, even if the Reveal query is invoked, the adversary $A$ cannot obtain the key information (Lemma 1).

Theorem 1 (Untraceability). In the proposed authentication protocol, the device is untraceable.

Proof. If the adversary is unable to correlate two successful device authentication requests with two valid responses corresponding from the server, then the device is generally untraceable. This theory can be proved by modeling the game with adversary $A$ by setting $CH$ as the challenger of the RFID system. $CH$ chooses a valid server unit $S$ and two devices $D_1$ and $D_2$; $A$ invokes the following queries, Transmit, Enquiry, Execute, and Blocking, on $S$ and $D_1$ and $D_2$; After completing the invocation of the query, $A$ notifies $CH$; $CH$ randomly selects one of the devices $D$; $A$ invokes the following queries on $S$ and $D$: Transmit, Enquiry, Execute, Blocking. In addition, $A$ can invoke Reveal once; $A$ predicts her guess $D'$, and if $D' = D$, $A$ wins the game. Thus, the advantage of a successful guess is defined as $Adv_A = 2 \times (Sel[D' = D] - 1/2)$. If adversary $A$ does not have any advantage in random guessing, then $Sel[D' = D] = 1/2$, and therefore $Adv_A$ is zero, which means that it cannot be traced. In our proposed protocol, it is known from Lemma 2 that $A$ cannot infer the PUF response $r_1$ at round $i$ of authentication, and thus cannot generate the important information key $k_1$. From Lemma 3, we know that $A$ cannot correlate the identities of the device at round $i$ and $i + 1$. Therefore, adversary $A$ has only one choice, i.e., random guess $Sel[D' = D]$. Then, according to the above formula $Adv_A = 2 \times (Sel[D' = D] - 1/2)$, it is known that $Adv_{A=0}$ when $Sel[D' = D] = 1/2$. Therefore, the proposed authentication protocol can ensure untraceability. 

Theorem 2 (Forward secrecy). The proposed authentication protocol can ensure forward confidentiality while supporting backward untraceability.

Proof. To ensure that the proposed protocol has forward secrecy supported by backward untraceability, we model a game in Theorem 1. The difference is that adversary $A$ needs to call a Reveal query on the device to obtain the challenge $c$ and the helper data $h$, which are stored in the device memory, and finally, $A$ outputs the guessed $D$. By Lemma 2, $A$ cannot launch the key information $r$, and each CRP is randomly generated, so they are considered to be independent of each other. Since all the secret information used in the proposed protocol is one-time, if the attacker has access to challenge $c$, the helper data $h$, and a small quantity of CRPs, from Lemma 3, $A$ still cannot track the device by using the compromised information. Thus, adversary $A$ will not have any advantage over random guesses in predicting the device, and the proposed protocol can ensure forward secrecy with backward untraceability support. 

\[ \square \]
Lemma 4. The proposed scheme can ensure resilience against DoS attacks.

In a DOS attack, the adversary $A$ tends to invoke the Blocking query to stop the device from receiving the message $s'_4$ in the third transmission; hence, the information on the device remains the $ID_1, (c_1, h_1)$ and cannot be updated. To deal with this attack, we store the auxiliary key $k_{old}$ and temporary ID $ID_{old}$ at the server. If, in the next authentication, the server cannot find the correct $ID$, it will use the $(ID_{old}, k_{old})$ instead of $(ID_1, k_1)$ to perform the authentication. This will serve as a defense against DoS attacks.

Lemma 5. The proposed protocol can ensure security against a cloning attack.

Most existing authentication protocols store the secret in the tag’s NVM. However, this approach is prone to key leakage. In our proposed scheme, we do not store any keys in the tag, and the device’s PUF is considered to be inseparable from the microprocessor. Therefore, even if an adversary has access to the device, it cannot compromise the security of the proposed protocol (Lemma 1). Furthermore, the PUFs cannot be recreated as they are immune to cloning [23], and each device is equipped with a PUF; the proposed authentication protocol can be considered secure against cloning attacks.

Theorem 3. The proposed protocol accomplishes mutual authentication.

Proof. Now, to prove the legitimacy of $A$, $A$ must respond to the query from the server $S$. Alternatively, $A$ needs to generate a valid response message $s_1, m_2, u, v, ID_1$ after receiving a message $m_1$ from the server $S$. In this case, $A$ must know the PUF response (i.e., $r_1$) used for this authentication. However, according to Lemma 2, $A$ cannot obtain the secret response $r_1$. Since $A$ cannot infer $r_1$, it cannot generate valid parameters $s_1, m_2$, and $v$ associated with $r_1$. We know from Lemma 3 that $A$ cannot associate devices, and it is known from Lemma 1 and Lemma 5 that the proposed protocol is resistant to DOS attacks, cloning attacks, or modeling attacks. Therefore, $A$ cannot impersonate a legitimate server. In our proposed scheme, only legitimate devices and servers can authenticate each other. □

5. Implementation and Analysis

We use the Zynq-7000 SoC XC7Z010 FPGA board (Xilinx, San Jose, CA, USA) and a 6-core 3.4 GHz AMD 2600 (AMD, Santa Clara, CA, USA) desktop computer as the device and server, respectively, to implement the proposed protocol. First, we use the programmable logic (PL) part of the Zynq board to implement a finite state machine (FSM) to serialize the authentication phase of the device side of the protocol. The FSM is used to call the main modules of the authentication phase on the device. We use the PC to implement and execute each component of the server and use the processing system (PS) part of the Zynq board to communicate with the desktop running MATLAB R2020a through a universal asynchronous receiver-transmitter (UART). The hardware overhead of the Zynq board should be as low as possible to emulate lightweight devices. We implement the protocol using the method described in the previous sections and measure the device execution overhead. We also analyze the fault tolerance and scalability of the proposed scheme.

5.1. Implementation and Performance Evaluation of the BST-PUF

Since arbiter PUFs (APUFs) have lower hardware requirements than RO PUFs and provide a larger number of CRPs than memory-based PUF, we have chosen the scheme in [27], which applies the BST-PUF strategy to the arbiter PUF circuit as the implementation of the PUF section in the proposed protocol. It is known that the BST strategy can be applied to a variety of PUFs and is not limited to a particular PUF. The BST-APUF was chosen as the implementation in this paper because of its ability to output a large number of CRPs and its lower overhead. Compared to the traditional Arbiter PUF, a Threshold Generator (TG) is added to the BST-APUF to generate additional delays. The TG feeds into the up-layer and down-layer delay paths separately via the multiplexer module, generating
the PUF response $R_i$ and the corresponding reliability flag $F_i$. The specific circuit design is shown in Figure 6. It consists of four modules: an Arbiter PUF with a 64-stage delay switch module, a BST module, a reliability flag generator, and a control logic unit. The principles of BST-APUF are described in Sections II and III of [27].

We tested three performance metrics (uniqueness, reliability, and bias) of the selected robust responses at 1.0 V and $-25^\circ\text{C}$, $0^\circ\text{C}$, $25^\circ\text{C}$, and $80^\circ\text{C}$. The reliability can be expressed in terms of the bit error rate (BER), which is the percentage of false response bits acquired on average over different operating environments, and the results are shown in Figure 7. We found that when the selection rate of the selected robust response is 66.85%, the uniqueness of the PUF response is 49.1%, the bias is 50.3%, and the BER is less than $10^{-9}$. The reliability is high enough to be directly used for security authentication.

---

**Figure 6.** Structure of the BST Arbiter PUF.

**Figure 7.** Performance test results of BST-APUF with different numbers of delay modules.

### 5.2. Implementation and Analysis of the Protocol

It is important to note that this paper focuses on the PUF-related part of the authentication protocol and the secure communication between the server and the device. During the interaction phase between the user and the device, we connect the interaction interface to the backend database, and the user uses the interaction interface to perform the corresponding authentication while the backend database makes the corresponding query. To use the system, users need to register first, storing their ID and key in the database. When we need to authenticate the user with the device, the key entered by the user is checked in the
database for the match, and only if the key entered is identical to the key corresponding to the ID in the database can the system be used. For establishing a secure channel in the setup phase of the proposed protocol, we first use Cookie+Session for the front-end login. HTTP is a stateless protocol; each time the client sends a request, it should first establish a connection with the server, and after the request is completed, the connection will be disconnected again. This method can save the connection resources occupied during the transmission. The Cookie is a special piece of information (the password set by the user) that the server sends to the client, which is stored in text form on the client side. Each time the client sends a request to the server, it will send this special information together. With the Cookie, the server can access the information transmitted by the client, but if it needs to authenticate the information, the Session is required. The client requests the server, which opens up memory space for the request, which is the Session object. Once we have the Cookie and Session, we will be able to log in and perform authentication. In the secure communication phase, we can use covert timing channels to transmit secret information from one process to another. Moreover, we refer to the hierarchical entropy algorithm proposed by Omar Darwish et al. [37,38] for the efficient and accurate detection of covert timing channels.

In Section 3.1, we introduced a novel HDA based on BST-PUF. In this paper, the KEY.Ext / KEY.Rec model is implemented according to the principles described in Figure 4, and the KEY.Ext / KEY.Rec execution process of this paper is shown in Figure 8. In the KEY.Ext phase, a string of random numbers generated by the TRNG is input to the LFSR as the original challenge of the PUF to generate the 64-group (because we use a 64-level cascade switch in the BST-APUF) N-bit challenge collection \( c_i \). Inputting \( c_i \) to the BST-PUF can generate the N-bit original response \( r_{m1} \) and the N-bit helper data \( h_i \). Because the device uses \( c_i \) and \( h_i \) for key recovery during authentication, \( c_i \) and \( h_i \) are stored in the device’s NVM. The \( r_{m1} \) and \( h_i \) are input to the robust response extractor to output the highly reliable response \( r_c \). To increase the entropy value of the key while ensuring low overhead, we choose the SPONGENT [39] lightweight hash as the entropy extractor to generate the high-entropy key \( k \). In the KEY.Rec phase, the \( c_i \) and \( h_i \), which are stored in the NVM in the KEY.Rec phase, is input into the BST-PUF and the robust response extractor, respectively. Although the collection of challenges used for key recovery is the same as that of the key generation phase, the output of the PUF may be biased due to environmental factors, so the original response \( r_{m2} \) of the key recovery phase is extracted using \( h_i \) to generate the robust response \( r_{c2} \). Since the BER of the response is tested in Section 5.1 to be below \( 10^{-9} \) when the PUF has three delay gates, the key \( k \) can be recovered from the response \( r_{c2} \) after entropy extraction.

![Figure 8. KEY.Ext / KEY.Rec model.](image-url)

We use SPONGENT-128/256/128 to hash the robust response \( r_c \) to generate a 128-bit key \( k \). The generated keys are tested by NIST and the results show that after 15 sets of tests, the \( p \)-value is greater than 0.01, which indicates that the keys have passed all experiments in the NIST Statistical Test Suite. We control the operation of TRNG by generating true
random numbers in the PL part of the Zynq-7000 SoC XC7Z010 FPGA using the method described in the reference [40], which enforces a sub-stable state on the flip-flop through closed-loop feedback control. While having good randomness, the TRNG has a very low hardware overhead, making it ideal for generating random numbers in a lightweight authentication protocol.

The PRF and PRF′ are based on the pseudorandom number function designed by the SIMON block cipher in [32]. According to the requirements of the proposed protocol authentication process, we selected SIMON128/128 for the PRF and encryption operations.

Table 1 shows the execution overhead required for each module of the protocol execution once and the total execution overhead on the device for the whole authentication process. Tables 2 and 3, respectively, show the comparison between the proposed protocol and past schemes in terms of various security properties and hardware overhead. Although the overhead of [6] has a small advantage over the proposed protocol, it lacks the very important feature of mutual authentication. Furthermore, the protocol in [6] is resistant to modeling attacks, but the protocol has weak forward security and the protocol is also less resistant to DoS attacks. As analyzed in Section 4, our proposed scheme satisfies all the important security features required, achieves mutual authentication, and performs the protocol process as it uses the temporary IDs in every round, which can be used as antitracking, while having good forward secrecy and backward security as well; furthermore, the proposed protocol has the ability to handle the problem of desynchronization, providing the ability to resist DoS attacks while ensuring the session security. Moreover, the resource overhead on the device of this protocol is 870 LUTs and 547 registers, which is a great advantage in terms of security and overhead compared to the recent PUF-based lightweight authentication protocols.

Table 1. The proposed protocol’s execution costs.

| Module      | LUTs | Registers |
|-------------|------|-----------|
| PUF         | 150  | 47        |
| KEY.E/R     | 241  | 165       |
| TRNG        | 128  | 12        |
| PRF         | 267  | 154       |
| Controller  | 84   | 169       |
| Total       | 870  | 547       |

Table 2. Comparison with other protocols’ execution costs.

| Protocols  | LUTs | Registers |
|------------|------|-----------|
| Majzoobi [6] | 656  | 537       |
| Chatterjee [14] | 1591 | 1933      |
| Che [41] | 6038 | 1724      |
| Proposed work | 870  | 547       |

Table 3. Performance comparison based on required security.

| Protocols        | SP1 | SP2 | SP3 | SP4 | SP5 | SP6 | SP7 |
|------------------|-----|-----|-----|-----|-----|-----|-----|
| Majzoobi [6]     | ✗   | ✓   | ✗   | ✓   | ✓   | ✗   | ✓   |
| Chatterjee [14]  | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ✗   |
| Che [41]         | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   |
| Proposed work    | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   | ✓   |

SP1: Mutual Authentication; SP2: Untraceability; SP3: Forward Secrecy; SP4: Modeling Robust; SP5: Unclonability; SP6: Resilience to DoS Attacks; SP7: Security Against Physical Attacks.

As shown in Table 4, when we implement the protocol using the Zynq-7000 SoC XC7Z010 FPGA (Xilinx, San Jose, CA, USA) board with a 6-core 3.4 GHz AMD 2600 (AMD,
Santa Clara, CA, USA) desktop computer, the total execution time measured at a clock frequency of 2.5 MHz averaged 509,913 clock cycles, and the actual execution time on the device side averaged 16.65 ms. The execution time in [41] containing the instruction transfer between token and verifier is around 1.25 s, and the protocol has a high level of security. However, compared to [41], our proposed scheme has significant advantages in terms of execution time and hardware overhead. The protocol in [14] is implemented on the Edison platform and its end-to-end execution time is measured to be around 480.11 ms, but [14] uses BCH codes on the device side to correct the noisy response, which leads to a higher hardware overhead. The hardware overhead of our proposed scheme is much lower than that of the protocol in [14], which is more suitable for RFID, a resource-constrained Internet of Things device.

| Step | Implementation Module | Clock Cycles |
|------|------------------------|--------------|
| Read $c_1$, $h_1$, $ID_1$ | Read ROM | 33,394 |
| $m_1 \leftarrow$ TRNG; $c_1 \leftarrow$ TRNG | Programmable logic (PL) | 8426 |
| $k_1 :=$ KEY.Rec($c_1$, $h_1$); $(k_2, h_2) :=$ KEY.Ext($c_2$) | Key. Ext/Rec Model and BST-APUF | 16,439 |
| $(s_1, \ldots, s_3) :=$ PRF($k_1$, $m_1 \parallel m_2$); $v :=$ PRF'($s_3$, $m_2 \parallel u$) | SIMON PRF/PRF’(128-bit) | 375,866 |
| Update $c_2$, $h_2$, $ID_2$ | Write ROM | 75,788 |
| Total | - | 509,913 |

Nevertheless, [6] does not use any error correction logic, which makes its overhead lower than our scheme. However, it provides an open interface for the adversary to obtain information about CRPs as long as access to the device’s interface is maintained. Compared to [6], our proposed scheme has higher security. Reference [14] generates a new security association information between prover and verifier that hides the correlation between the challenge and response of the PUF. However, the server will send the original challenge and the corresponding helper data to the device during the authentication phase. Since the device’s interface is open to random queries, the exposure of these helper data and the challenge may lead to a side-channel attack against the device PUF. Moreover, the strategy uses a BCH to obtain PUF robust responses with high hardware overhead; thus, it is not suitable for resource-constrained devices. In contrast, although the protocol design in [41] satisfies all security criteria, its cost is too high and it does not have the advantage of lightweight overhead.

Now, we analyze the scheme proposed in this paper in terms of fault tolerance and scalability. The fault tolerance of a system can be subdivided into three indicators: reliability, availability, and testability. In terms of reliability, we divide the software and hardware of the system into various modules, and when there is a problem with the system, we will use a virtual machine to test whether the operation of each module can transmit data, i.e., to increase the size of MTTR. In terms of availability, the system has been in development for less than a year at present, but, based on the results of the initial operation, the system has a high rate of non-failure. For the system, we calculated the MTBF of the system during the development process and found that when there is a problem with the system, the system can run through a round of data normally before reporting errors. The system uses callback function constructs and a highly functional logical code structure in terms of scalability to handle a larger number of users.

6. Conclusions

We propose a lightweight and practical anonymous authentication protocol based on BST-PUF and implement it on an FPGA. Analyses show that our protocol still remains safe
even if an adversary has physical access to the device. Specifically, our protocol effectively ensures the low overhead and required security properties by exploiting the lightweight and inherent security features of the BST-PUF. Therefore, the proposed scheme ensures security while effectively reducing the implementation overhead and solving the problems of existing PUF-based authentication protocols. Because of its low overhead and high security properties, our proposed scheme can be used in resource-constrained IoT tags such as RFID in various fields, such as healthcare, the automotive industry, ubiquitous electric IoT, and security intelligent control systems, and it provides a sufficiently reliable and inexpensive solution for the development of IoT. However, there are also some limitations to our work. Firstly, we have implemented the device side of the proposed protocol on an FPGA development board, but have not evaluated the design to measure its precise power consumption. The second is that the scheme was developed less than a year ago, although we have proven the security of the protocol and the system runs with high reliability at present. However, because the proposed scheme can be used in a wide range of scenarios and the application environment of the device is complex, other security threats may arise in practical applications and more practical engineering applications are lacking to test the reliability of the scheme. In our coming work, we will continue to verify the security of the protocol and improve the reliability of the system. The proposed methods in this paper can be applied to other field such as system identification \[42-44]\ and signal and information processing.

**Author Contributions:** Conceptualization, Y.A.; Funding acquisition, Z.H.; Investigation, Y.A. and Y.Z.; Methodology, Y.A.; Supervision, Y.Z., W.C. and Z.T.; Validation, Y.A. and Y.Z.; Original draft writing, Y.A.; Review and editing, Y.A., Y.Z., Z.H. and W.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the National Natural Science Foundation of China (No. 62174050) and Hubei Provincial Natural Science Foundation of China (No. 2020CFB814).

**Acknowledgments:** We would like to acknowledge Guisheng Liao, for his contributions to this work.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Aghili, S.F.; Ashouri-Talouki, M.; Mala, H. DoS, impersonation and de-synchronization attacks against an ultra-lightweight RFID mutual authentication protocol for IoT. *J. Supercomput.* 2018, 74, 509. [CrossRef]
2. Juels, A. RFID security and privacy: A research survey. *IEEE J. Sel. Areas Commun.* 2006, 24, 381. 5.861395. [CrossRef]
3. Madlmayr, G.; Langer, J.; Kantner, C.; Scharinger, J. NFC devices: Security and privacy. In Proceedings of the 2008 Third IEEE International Conference on Availability, Reliability and Security, Barcelona, Spain, 4–7 March 2008; p. 642. [CrossRef]
4. Suh, G.E.; Devadas, S. Physical unclonable functions for device authentication and secret key generation. In Proceedings of the 2007 44th ACM/IEEE Design Automation Conference (DAC), San Diego, CA, USA, 4–8 June 2007; p. 9.
5. Kong, J.; Koushanfar, F.; Pendyala, P.K.; Sadeghi, A.R.; Wachsmann, C. PUFatt: Embedded platform attestation based on novel processor-based PUFs. In Proceedings of the 2014 51st ACM/EDAC/IEEE (DAC), San Francisco, CA, USA, 1–5 June 2014; p. 1. [CrossRef]
6. Majzoobi, M.; Rostami, M.; Koushanfar, F.; Wallach, D.S.; Devadas, S. Slender PUF protocol: A lightweight, robust, and secure authentication by substring matching. In Proceedings of the 2012 IEEE Symposium on Security and Privacy Workshops, San Francisco, CA, USA, 24–25 May 2012; p. 33. [CrossRef]
7. Yu, M.D.; Hiller, M.; Delvaux, J.; Sowell, R.; Devadas, S.; Verbauwhede, I. A lockdown technique to prevent machine learning on PUFs for lightweight authentication. *IEEE Trans. Multi-Scale Comput. Syst.* 2016, 2, 146–159. 2553027. [CrossRef]
8. Pang, Z.; Zhang, J.; Zhou, Q.; Gong, S.; Qian, X.; Tang, B. Crossover ring oscillator PUF. In Proceedings of the 2017 18th International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, USA, 14–15 March 2017; p. 237. [CrossRef]
9. Gope, P.; Hwang, T. Untraceable Sensor Movement in Distributed IoT Infrastructure. *IEEE Sens. J.* 2015, 15, 5340. [CrossRef]
10. Sahoo, D.P.; Nguyen, P.H.; Mukhopadhyay, D.; Chakraborty, R.S. A case of lightweight PUF constructions: Cryptanalysis and machine learning attacks. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* 2015, 34, 1334. [CrossRef]
11. Vijayakumar, A.; Kundu, S. A novel modeling attack resistant PUF design based on non-linear voltage transfer characteristics. In Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 9–13 March 2015; p. 653. [CrossRef]
12. Maes, R. Physically Unclonable Functions. In *Properties*; Springer: Berlin/Heidelberg, Germany, 2013.
13. Lim, D.; Lee, J.W.; Gassend, B.; Suh, G.E.; Van Dijk, M.; Devadas, S. Extracting secret keys from integrated circuits. IEEE Trans. Very Large Scale Integr. (Vlsi) Syst. 2005, 13, 1200–1205.

14. Chatterjee, U.; Govindan, V.; SadhuKhun, R.; Mukhopadhyay, D.; Chakraborty, R.S.; Mahata, D.; Prabhu, M.M. Building PUF based Authentication and Key Exchange Protocol for IoT without Explicit CRPs in Verifier Database. IEEE Trans. Dependable Secur. Comput. 2019, 16, 424. [CrossRef]

15. Bhargava, M.; Cakir, C.; Mai, K. Reliability enhancement of bi-stable PUFs in 65nm bulk CMOS. In Proceedings of the 2012 IEEE International Symposium on Hardware-Oriented Security and Trust, San Francisco, CA, USA, 3–4 June 2012; p. 25. [CrossRef]

16. Beckmann, K.; Manem, H.; Cady, N.C. Performance Enhancement of a Time-Delay PUF Design by Utilizing Integrated Nanoscale ReRAM Devices. IEEE Trans. Emerg. Top. Comput. 2017, 5, 304. [CrossRef]

17. Dodis, Y.; Reyzin, L.; Smith, A. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. In International Conference on the Theory and Applications of Cryptographic Techniques; Springer: Berlin/Heidelberg, Germany, 2004; p. 323.

18. Yu, M.D.; Devadas, S. Secure and Robust Error Correction for Physical Unclonable Functions. IEEE Des. Test Comput. 2010, 27, 48. [CrossRef]

19. Paral, Z.; Devadas, S. Reliable and efficient PUF-based key generation using pattern matching. In Proceedings of the 2011 IEEE International Symposium on Hardware-Oriented Security and Trust, San Diego, CA, USA, 5–6 June 2011; pp. 128–133.

20. Becker, G.T.; Wild, A.; Güneyis; T. Security analysis of index-based syndrome coding for PUF-based key generation. In Proceedings of the 2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), Washington, DC, USA, 5–7 May 2015; pp. 20–25.

21. Suzuki, M.; Ueno, R.; Homma, N.; Aoki, T. Efficient fuzzy extractors based on ternary debiasing method for biased physically unclonable functions. IEEE Trans. Circuits Syst. 2018, 66, 616. [CrossRef]

22. Maes, R.; Herrewege, A.V.; Verbauwhede, I. PUFKY: A fully functional PUF-based cryptographic key generator. In International Workshop on Cryptographic Hardware and Embedded Systems; Springer: Berlin/Heidelberg, Germany, 2012; p. 302. [CrossRef]

23. Ueno, R.; Suzuki, M.; Homma, N. Tackling Biased PUFs Through Biased Masking: A Debiasing Method for Efficient Fuzzy Extractor. IEEE Trans. Comput. 2019, 68, 1091–1104. [CrossRef]

24. Lao, Y.; Yuan, B.; Kim, C.H.; Parhi, K.K. Reliable PUF-Based Local Authentication With SelfCorrection. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 2017, 36, 201. [CrossRef]

25. Zhang, J.; Wan, L.; Wu, Q.; Dmos-Puf, G. Dynamic multi-key-selection obfuscation for strong PUFs against machine learning attacks. arXiv 2018, arXiv:1806.02011.

26. Gope, P.; Sikdar, B. Privacy-Aware Authenticated Key Agreement Scheme for Secure Smart Grid Communication. IEEE Trans. Smart Grid 2018, 10, 3953–3962. [CrossRef]

27. He, Z.; Chen, W.; Zhang, L.; Chi, G.; Gao, Q.; Harn, L. A Highly Reliable Arbiter PUF With Improved Uniqueness in FPGA Implementation Using Bit-Self-Test. IEEE Access 2020, 8, 181751. [CrossRef]

28. Qureshi, M.A.; Munir, A. PUF-RAKE: A PUF-based Robust and Lightweight Authentication and Key Establishment Protocol. IEEE Trans. Dependable Secur. Comput. 2021. [CrossRef]

29. Becker, G.T. On the Pitfalls of Using Arbiter-PUFs as Building Blocks. IEEE Trans. Comput. Dependable Integr. Circuits Syst. 2015, 34, 1295–1307. [CrossRef]

30. Delvaux, J. Machine Learning Attacks on PolyPUF, OB-PUF, RPUE, and PUF-PSM. IEEE Trans. Inf. Forensics Secur. 2017, 14, 2043–2058. [CrossRef]

31. Aysu, A.; Gulcan, E.; Moriyama, D.; Schaumont, P.; Yung, M. End-to-end design of a PUF-based privacy preserving authentication protocol. In International Workshop on Cryptographic Hardware and Embedded Systems; Springer: Berlin/Heidelberg, Germany, 2015; p. 556. [CrossRef]

32. Beaulieu, R.; Treatman-Clark, S.; Shors, D.; Weeks, B.; Smith J.; Wingers, L. The SIMON and SPECK lightweight block ciphers. In Proceedings of the 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, 8–12 June 2015.

33. Gope, P.; Lee, J.; Quek, T.Q. Lightweight and Practical Anonymous Authentication Protocol for RFID Systems Using Physically Unclonable Functions. IEEE Trans. Inf. Forensics Secur. 2018, 13, 2831–2843. [CrossRef]

34. Braeken, A. PUF based authentication protocol for IoT. Symmetry 2018, 10, 352. [CrossRef]

35. Urbi Chatterjee, Rajat Subhra Chakraborty, and Debdeep Mukhopadhyay. A PUF-Based Secure Communication Protocol for IoT. ACM Trans. Embed. Comput. Syst. 2017, 16, 67. [CrossRef]

36. Koeberl, P.; Li, J.; Rajan, A.; Wu, W. Entropy loss in PUF-based key generation schemes: The repetition code pitfall. In Proceedings of the IEEE International Conference on Hardware-Oriented Security & Trust, Arlington, VA, USA, 6–7 May 2014.

37. Darwish, O.; Al-Fuqaha, A.; Anan, M.; Nasser, N. The role of hierarchical entropy analysis in the detection and time-scale determination of covert timing channels. In Proceedings of the 2015 International Wireless Communications and Mobile Computing Conference (IWCMC), Dubrovnik, Croatia, 24–28 August 2015; pp. 153–159. [CrossRef]

38. Darwish, O.; Al-Fuqaha, A.; Brahim, G.B.; Javed, M.A. Using MapReduce and hierarchical entropy analysis to speed-up the detection of covert timing channels. In Proceedings of the 2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC), Valencia, Spain, 26–30 June 2017; pp. 1102–1107. [CrossRef]

39. Bogdanov, A.; Knèzević, M; Leander, G.; Toz, D.; Varici, K.; Verbauwhede, I. SPONGENT: A lightweight hash function. In International Workshop on Cryptographic Hardware and Embedded Systems; Springer: Berlin/Heidelberg, Germany, 2011; p. 312.
40. Majzoobi, M.; Koushanfar, F.; Devadas, S. FPGA-based true random number generation using circuit metastability with adaptive feedback control. In *International Workshop on Cryptographic Hardware and Embedded Systems*; Springer: Berlin/Heidelberg, Germany, 2011; p. 17. [CrossRef]

41. Che, W.; Martin, M.; Pocklassery, G.; Kajuluri, V.K.; Saqib, F.; Plusquellic, J. A privacy-preserving, mutual PUF-based authentication protocol. *Cryptography* 2017, 1, 3. [CrossRef]

42. Zhang, X.; Ding F. Optimal adaptive filtering algorithm by using the fractional-order derivative. *IEEE Signal Process Lett.* 2022, 29, 399–403. [CrossRef]

43. Zhou, Y.H.; Ding, F. Modeling nonlinear processes using the radial basis function-based state-dependent autoregressive models. *IEEE Signal Process Lett.* 2020, 27, 1600–1604. [CrossRef]

44. Zhou, Y.H.; Zhang, X.; Ding, F. Hierarchical estimation approach for RBF-AR models with regression weights based on the increasing data length. *IEEE Trans. Circuits Syst. II Express Briefs* 2021, 68, 3597–3601. [CrossRef]