Reverse Fuzzy Extractors: Enabling Lightweight Mutual Authentication for PUF-enabled RFIDs

Anthony van Herrewege[1], Stefan Katzenbeisser[2], Roel Maes[1], Roel Peeters[1], Ahmad-Reza Sadeghi[3], Ingrid Verbauwhede[1], and Christian Wachsmann[2]

[1] K.U. Leuven, ESAT/COSIC, Leuven, Belgium
{anthony.vanherrewege,roel.maes,roel.peeters,
ingrid.verbauwhede}@esat.kuleuven.be

[2] Technische Universität Darmstadt (CASED), Germany
katzenbeisser@seceng.informatik.tu-darmstadt.de
christian.wachsmann@trust.cased.de

[3] Technische Universität Darmstadt & Fraunhofer SIT Darmstadt, Germany
ahmad.sadeghi@trust.cased.de

Abstract. RFID-based tokens are increasingly used in electronic payment and ticketing systems for mutual authentication of tickets and terminals. These systems typically use cost-effective tokens without expensive hardware protection mechanisms and are exposed to hardware attacks that copy and maliciously modify tokens. Physically Unclonable Functions (PUFs) are a promising technology to protect against such attacks by binding security critical data to the physical characteristics of the underlying hardware. However, existing PUF-based authentication schemes for RFID do not support mutual authentication, are often vulnerable to emulation and denial-of service attacks, and allow only for a limited number of authentications.
In this paper, we present a new PUF-based authentication scheme that overcomes these drawbacks: it supports PUF-based mutual authentication between tokens and readers, is resistant to emulation attacks, and supports an unlimited number of authentications without requiring the reader to store a large number of PUF challenge/response pairs. In this context, we introduce reverse fuzzy extractors, a new approach to correct noise in PUF responses that allows for extremely lightweight implementations on the token. Our proof-of-concept implementation shows that our scheme is suitable for resource-constrained devices.

1 Introduction

Electronic payment and ticketing systems have been gradually introduced in many countries over the past few years. Typically, these systems use a large number of RFID-enabled tokens with constrained computing and memory capabilities (see, e.g., [35]). A fundamental security requirement in electronic payment and ticketing systems is *mutual authentication*: Only genuine tokens should be accepted by readers and only eligible readers should be able to modify the debit

of a user's token. The widespread use of these systems makes them attractive targets for different kinds of attacks. The most prominent example are attacks on widely used MiFare Classic tokens by NXP Semiconductors [35] that allow copying (cloning) and maliciously changing the debit of tokens [13]. Other MiFare products are claimed not to be affected. Existing solutions typically use cost-efficient tokens without expensive hardware protection mechanisms [35]. Hence, the authentication secrets of these tokens can often be recovered by basic side channel and invasive attacks, and used to emulate the token in software, which allows forging the information of the token (e.g., the debit of the ticket). To prevent such attacks, the secrets and information of the token should be cryptographically bound to the underlying RFID chip such that any attempt to extract or change them permanently deactivates the token.

In this context, Physically Unclonable Functions (PUFs) [31,27,2] promise to provide an effective and cost-efficient security mechanism. PUFs are physical systems embedded into a host device, that, when challenged with a stimulus, generate a noisy response. This means that, depending on environmental variations (e.g., temperature or voltage variations), a PUF will always return slightly different responses when challenged with the same stimulus. Further, due to manufacturing variations, responses to the same challenge vary across different PUFs and are typically hard to predict [27,2].

The common approach to authenticate a PUF-enabled token is querying its PUF with a challenge from a pre-recorded database of PUF challenges and responses. The token is accepted only if its response matches a PUF response in the database (such as in [32,6,10]). An alternative approach is using the PUF to generate the authentication secret of the token for use in a classical authentication protocol (such as in [38,34]). However, both approaches have serious drawbacks in practice: PUF-based key storage requires the token to reliably recover the (bit-exact) cryptographic secret from the noisy PUF response using some kind of error correction mechanism, which is expensive in terms of number of gates [12,7]. Further, existing PUF-based authentication schemes for RFID suffer from the following deficiencies: (1) there is no support for mutual authentication between token and reader; (2) most PUF types are vulnerable to emulation attacks [33] and would allow emulating the token in software; (3) some schemes are subject to denial-of-service attacks that permanently prevent tokens from authenticating to the reader [6]; and (4) all existing PUF-based authentication schemes are not scalable and allow only for a limited number of authentication protocol-runs since they rely on a database containing a large number of challenge/response pairs of the PUF of each token. It seems that emulation attacks could be mitigated by controlled PUFs [14]. However, controlled PUFs typically apply a cryptographic operation (such as a hash function) to the PUF response, which requires an expensive error correction mechanism on the token to maintain verifiability of the PUF.

Our Contribution. In this paper, we present the design and implementation of a new lightweight PUF-based authentication scheme for *mutual authentication* of RFID tokens and readers. Our scheme supports an unlimited number of authen-

tication protocol-runs, is resistant to emulation attacks, and does not require the reader to store a large number of PUF challenge/response pairs.

Furthermore, we introduce the concept of *reverse fuzzy extractors*, a novel approach to eliminate noise in PUF responses that moves the computationally expensive error correction process from the resource-constrained PUF-enabled token to the more powerful RFID reader. The resources required to implement our authentication scheme on the token are minimal since it is based on a reverse fuzzy extractor that requires significantly less hardware resources than the error correcting mechanisms used in existing PUF-based authentication schemes or PUF-based key storage.

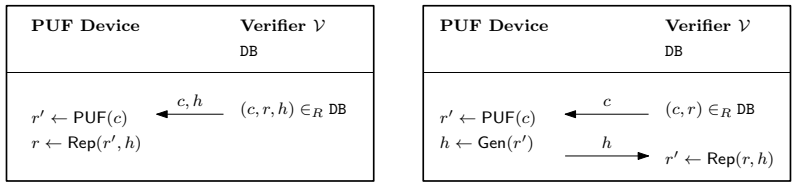
Outline. In Section 2, we provide background information on Physically Unclonable Functions (PUFs). We propose reverse fuzzy extractors in Section 3 and present our PUF-based mutual authentication scheme in Section 4. We describe the implementation and evaluate the performance of our scheme in Section 5, and analyze its security in Section 6. Finally, we survey on related work in Section 7 and conclude in Section 8.

2 Background on Physically Unclonable Functions (PUFs)

A PUF is a noisy function that is embedded into a physical object, e.g., an integrated circuit [31,27,2]. When queried with a *challenge* c, a PUF generates a *response* $r \leftarrow \mathsf{PUF}(c)$ that depends on both c and the unique device-specific intrinsic physical properties of the object containing the PUF. Since PUFs are subject to noise (e.g., environmental variations), they return slightly different responses when queried with the same challenge multiple times.

In literature, PUFs are typically assumed to be *robust*, *physically unclonable*, *unpredictable* and *tamper-evident*, and several approaches to heuristically quantify and formally define their properties have been proposed (see [2] for an overview). Robustness means that, when queried with the same challenge multiple times, the same PUF will return a similar response with high probability. Physical unclonability means that it is infeasible to produce two PUFs that cannot be distinguished based on their challenge/response behavior. Unpredictability requires that it is infeasible to predict the PUF response to an unknown challenge, even responses to other challenges can be obtained adaptively. Tamper-evidence means that any attempt to physically access the PUF irreversibly changes its challenge/response behavior.

There is a variety of PUF implementations (see [27] for an overview). The most appealing ones for integration into electronic circuits are electronic PUFs. The most prominent examples of this type are delay-based PUFs that exploit race conditions (arbiter PUFs [22,30]) and frequency variations (ring oscillator PUFs [15,37,28]) in integrated circuits; memory-based PUFs are based on the instability of volatile memory cells, like SRAM [16,18], flip-flops [26,23] and latches [36,21]; and coating PUFs [39] use capacitances of a dielectric coating applied to the chip housing the PUF.



(a) Typical use of fuzzy extractors (b) Reverse fuzzy extractor concept

Fig. 1. Concept of fuzzy extractors and reverse fuzzy extractors

Note that the number of responses of a memory-based PUF is limited by the number of its memory cells. Further, most delay-based PUFs are subject to model building attacks [22,30,25,33] that allow emulating the PUF in software. To counter this problem, additional primitives must be used: Controlled PUFs [14] use cryptography in hardware to hide the responses of the underlying PUF to mitigate model building attacks. This requires correcting the noise of PUF responses before they are processed by the cryptographic operation to maintain verifiability of the PUF. The cryptographic and error correcting components as well as the link between them and the PUF must be protected against invasive and side channel attacks.

Many PUF-based applications, including PUF-based identification and key storage, require PUF responses to be reliably reproducible while at the same time being unpredictable [3,27,2]. However, since PUFs are inherently noisy and their responses are not uniformly random, they are typically combined with *fuzzy extractors* [12] (Figure 1(a)). Fuzzy extractors consist of a *secure sketch*, which maps similar PUF responses to the same value, and a randomness extractor, which extracts full-entropy bit-strings from a partially random source.

Secure sketches generally work in two phases: in the *generation phase* some *helper data* $h = \mathsf{Gen}(r)$ is computed from PUF response r, which is used later in the *reproduction phase* to recover $r = \mathsf{Rep}(r', h)$ from a distorted PUF response $r' = r + e$, where e is the error caused by noise. An important property of secure sketches is that, after observing *one single* h, there is still some min-entropy left in r, which means that h can be stored and transferred publicly without disclosing the full PUF response [12].

3 Reverse Fuzzy Extractors

Fuzzy extractors and secure sketches [12] are commonly used to correct noisy PUF responses on the PUF-enabled device, which is required when the PUF response is used in a cryptographic algorithm or protocol (such as in [14,11,38,3]). However, the underlying error decoding algorithms are typically complex and require a large number of gates and/or long execution times when multiple bit errors must to be corrected [12,7]. Hence, implementing the decoding algorithm on the PUF-enabled device is a huge disadvantage in many applications.

To overcome this problem, we propose *reverse fuzzy extractors* that allow for very compact and fast implementations of secure sketches and fuzzy extractors. Instead of implementing the computationally intensive reproduction phase $\mathsf{Rep}()$, we implement the much more efficient helper data generation phase $\mathsf{Gen}()$ on the PUF-enabled device and move $\mathsf{Rep}()$ to the typically more powerful verifier (Figure 1(b)). As a consequence, new helper data h is generated each time the PUF is queried and the verifier corrects the reference value r of its database to the noisy PUF response r', which is different each time the PUF is evaluated.

There is one major pitfall that must be considered: Each execution of the helper data generator $\mathsf{Gen}()$ on a different noisy version of the same PUF response reveals new helper data. However, secure sketches give no guarantee about the min-entropy of the PUF response in case *multiple* helper data for different noisy variants of the same response is known [8]. Hence, reverse fuzzy extractors may leak the full PUF response, when $\mathsf{Gen}()$ and $\mathsf{Rep}()$ are based on a conventional fuzzy extractor. This is problematic in most PUF-based applications, such as controlled PUFs and PUF-based key storage (Section 2) that require at least some bits of the PUF response to be secret, and must be carefully considered when designing reverse fuzzy extractors.

We present an implementation of a reverse fuzzy extractor based on the syndrome construction [12], which is a secure sketch with a highly efficient helper data generation phase and that has been shown to ensure a certain amount of min-entropy in the PUF response even if multiple helper data for noisy variants of a response is known [8]. The syndrome construction implements the helper data generator $\mathsf{Gen}(r)$ as $h \leftarrow r \cdot H^T$, where H is the parity check matrix of a binary linear block code and h corresponds to the syndrome of r. The reproduction algorithm $\mathsf{Rep}(r', h)$ of the syndrome construction computes $r \leftarrow r' - e$, where e is determined by decoding the syndrome $s = h - r' \cdot H^T$ using the decoding algorithm of the underlying error correcting code. Note that $\mathsf{Gen}()$ corresponds to computing a matrix product of the PUF response with the parity-check matrix H of the underlying cyclic linear block code. Due to the special form of parity check matrices of these codes, this product can be computed very efficiently, as we show later when describing our prototype implementation in Section 5.

4 Our PUF-based Mutual Authentication Scheme

A naive approach to authenticate a PUF-enabled RFID token is the following: The verifier sends a random PUF challenge from a reference database to the token and accepts the token only when its response is similar to the one in the database. However, since the token always responds to the same PUF challenge with a similar PUF response, replay attacks are possible. Moreover, for most PUF implementations, sending the PUF response in clear allows cloning the token by model building attacks [33]. Further, it is not trivial to authenticate the reader to the token following this approach.

Our scheme solves these problems by merging the idea of controlled PUFs [14] and logically reconfigurable PUFs [20]: We amend a PUF with a control logic that

(1) hides the plain PUF response from the adversary and (2) allows dynamically changing the challenge/response behavior of the PUF in a random manner. Using reverse fuzzy extractors allows for a very compact implementation of our scheme that requires only minimal resources on the token.

4.1 System Model

The players in our scheme are (at least) a token issuer $\mathcal{I}$, a verifier $\mathcal{V}$ and a token $\mathcal{T}$. We denote the adversary with $\mathcal{A}$. Our scheme enables *mutual authentication* between $\mathcal{V}$ and $\mathcal{T}$. $\mathcal{V}$ has access to a database DB containing detailed information on all tokens $\mathcal{T}$ in the system. DB is initialized and maintained by $\mathcal{I}$.

4.2 Trust Model and Assumptions

Issuer $\mathcal{I}$ and verifier $\mathcal{V}$. We assume $\mathcal{I}$ and $\mathcal{V}$ to be trusted, which is a typical assumption in most RFID systems.[4] Further, $\mathcal{I}$ initializes $\mathcal{T}$ and $\mathcal{V}$ in a secure environment.

Token $\mathcal{T}$. We consider $\mathcal{T}$ to be a passive device that cannot initiate communication, has a narrow communication range (a few centimeters to meters) and erases its temporary state (all session-specific information and randomness) after it gets out of the electromagnetic field of $\mathcal{V}$. Further, we assume $\mathcal{T}$ to be equipped with a robust and unpredictable PUF (Section 2), a reverse fuzzy extractor (Section 3) and a lightweight hash function.

Adversary $\mathcal{A}$. As in most RFID security models, we assume $\mathcal{A}$ to control the wireless communication channel between $\mathcal{V}$ and $\mathcal{T}$. This means that $\mathcal{A}$ can eavesdrop, manipulate, delete and reroute all protocol messages sent by $\mathcal{V}$ and $\mathcal{T}$. Moreover, $\mathcal{A}$ can obtain useful information (e.g., by visual observation) on whether $\mathcal{V}$ accepted $\mathcal{T}$ as a legitimate token. Following the typical assumptions on PUF-based key storage (such as in [40,24,38]), we assume that $\mathcal{A}$ can read any information that is stored in the non-volatile memory of $\mathcal{T}$. However, $\mathcal{A}$ cannot access the responses of the PUF of $\mathcal{T}$ and cannot obtain temporary data stored in volatile memory (such as intermediate results of the computations) of $\mathcal{T}$ while it is participating in an authentication protocol. This can be achieved by using side-channel aware designs for the implementation of the underlying algorithms.

4.3 Protocol Specification

System Initialization. Token issuer $\mathcal{I}$ stores a random token identifier ID in the non-volatile memory of token $\mathcal{T}$. Moreover, $\mathcal{I}$ extracts $q > 0$ challenge/response pairs $(c_1, r'_1), \ldots, (c_q, r'_q)$ from the PUF of $\mathcal{T}$ and stores them together with ID in database DB, which is later used by verifier $\mathcal{V}$ in the authentication protocol.

[4] Note that there are papers considering revocation of malicious verifiers (such as in [4,29]). A simple approach to enable verifier revocation in our scheme is moving all computations of $\mathcal{V}$ to DB s.t. $\mathcal{V}$ has no access to the PUF challenge/response pairs.

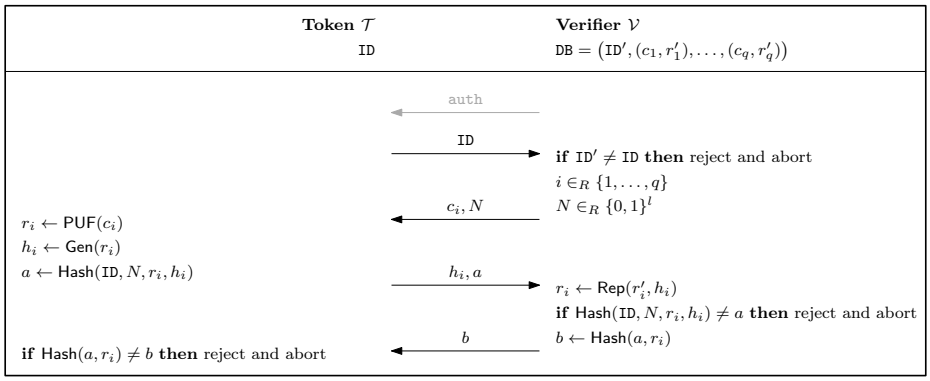


Fig. 2. Lightweight PUF-based mutual authentication protocol

Authentication Protocol. The authentication protocol (Figure 2) works as follows: Verifier $\mathcal{V}$ starts by sending an authentication request `auth` to token $\mathcal{T}$, which responds with its identifier `ID`. $\mathcal{V}$ chooses a random nonce N and a random challenge/response pair (c_i, r_i') from database `DB` and sends (c_i, N) to $\mathcal{T}$. Then, $\mathcal{T}$ evaluates $r_i \leftarrow \mathsf{PUF}(c_i)$, generates $h_i \leftarrow \mathsf{Gen}(r_i)$ using the reverse fuzzy extractor, computes $a \leftarrow \mathsf{Hash}(\mathtt{ID}, N, r_i, h_i)$ and sends (h_i, a) to $\mathcal{V}$. Next, $\mathcal{V}$ reproduces $r_i \leftarrow \mathsf{Rep}(r_i', h_i)$ using r_i' from `DB` and checks whether $\mathsf{Hash}(\mathtt{ID}, N, r_i, h_i) = a$. If this is not the case, $\mathcal{V}$ aborts and rejects. Otherwise, $\mathcal{V}$ sends $b \leftarrow \mathsf{Hash}(a, r_i)$ to $\mathcal{T}$ and accepts. Eventually, $\mathcal{T}$ accepts if $\mathsf{Hash}(a, r_i) = b$ and rejects otherwise.

Discussion. Note that the case $q = 1$ is equivalent to PUF-based key storage, where r_1 represents the authentication secret of $\mathcal{T}$. In this case, c_1 can be stored in the non-volatile memory of $\mathcal{T}$ and needs not to be sent from $\mathcal{V}$ to $\mathcal{T}$. Hence, two protocol messages can be saved: N can be sent with `auth` and `ID` can be sent with (h_i, a). Using multiple challenge/response pairs corresponds to storing multiple (session) keys in the PUF, which limits the impact of side channel attacks that may recover only a subset of these keys.

5 Implementation and Performance Evaluation

We demonstrate the feasibility of reverse fuzzy extractors by presenting a prototype implementation of the protocol depicted in Figure 2. The prototype comprises three main primitives (Figure 3): A challenge expander, a syndrome generator and a hash function. A controller orchestrates these primitives to execute the protocol in the correct order.

The prototype is designed to be used with an arbiter PUF but can be easily modified to work with most existing intrinsic PUFs. The used arbiter PUF implementation accepts 64-bit challenges and generates 1-bit responses. Since our protocol requires multiple response bits, we use a linear feedback shift register (LFSR) to expand a single challenge c into many consecutive 64-bit challenges

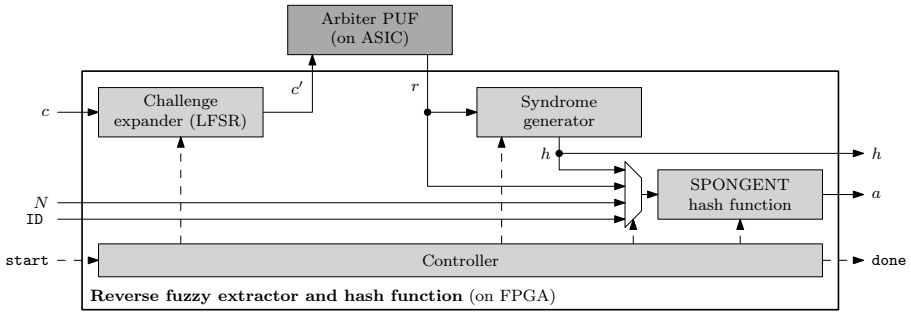


Fig. 3. Architecture of the reverse fuzzy extractor core

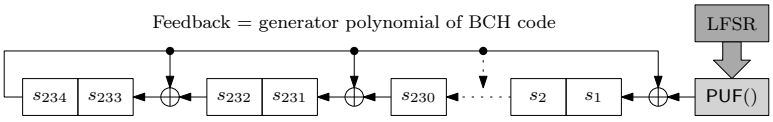


Fig. 4. Implementation of the syndrome generator

c', which are fed one after the other into the PUF. This allows generating PUF responses of arbitrary length for a single challenge. The expansion can be omitted for other PUF types that generate responses that have a sufficient length.

As described in Section 3, the syndrome generation consists of the matrix multiplication of an n-bit PUF response r with the $n \times (n-k)$ parity check matrix H^T of an error correcting linear block code. This $(n-k)$-bit result is called helper data h for r and can be used to correct a noisy version of r if the number of bit errors is small enough. For our implementation, we use the parity-check matrix of a $[n = 255, k = 21, t = 55]$ BCH block code that can correct up to $t = 55$ erroneous bits in a $n = 255$ bit PUF response, using a $(n-k) = 234$-bit helper data vector. In case the probability of a single bit error is 10%, then the probability of observing more than 55 errors in 255 response bits (resulting in a decoding failure) will only happen with probability $10^{-7.82}$. Due to the special structure of parity-check matrices of BCH codes, the matrix multiplication can be efficiently implemented as an LFSR where the feedback polynomial is determined by the BCH code and the feedback bit is added to the next bit of the PUF response (Figure 4). The helper data must be sent from the token to the verifier, which causes an entropy loss of the actual PUF response of up to $n-k$ bits. Assuming the PUF response has full entropy, there will be only $k = 21$ bits of uncertainty left after observing the helper data. In order to obtain a security level equivalent to a 128-bit key, we need at least $\lceil 128/21 \rceil = 7$ responses, each 255 bits in length, and 7 corresponding helper data vectors. This leads to an overall PUF response length of $7 \cdot 255 = 1,785$ bits and an overall helper data length of $7 \cdot 234 = 1,638$ bits. The probability of an authentication failure due to a decoding failure in one of the 7 blocks is $1 - (1 - 10^{-7.82})^7 = 10^{-6.97}$.

Table 1. Implementation size of the reverse fuzzy extractor core and its components when implemented on a Xilinx Virtex-5 FPGA (XC5VLX50).

Component	Registers	6-LUTs
SPONGENT hash	146	160
Syndrome generator	234	235
Challenge expander	65	67
Controller	51	196
Complete RFE core	496	658

This means that this prototype implementation of the protocol achieves a false rejection rate of only one in approximately 10 million authentications. The final building block of our prototype is the lightweight hash function SPONGENT [5], which seems to be perfectly suited for resource-constrained tokens.

Table 1 provides the implementation size results when synthesizing our prototype design for a field-programmable gate array (FPGA). The target device is a Xilinx Virtex-5. The complete core (Figure 3) can be implemented using 496 one-bit flip-flops and 658 6-input lookup-tables (LUTs). We note that these results can be further optimized by designing specifically for implementation on FPGA. The HDL description of our design is platform independent.

6 Security Analysis

We now prove the security properties of our reverse fuzzy extractor construction and mutual authentication scheme. In this context, we formalize all necessary aspects and set up formal security definitions.

6.1 Security of the Reverse Fuzzy Extractor

Secure sketch. Let $\mathtt{M}$ be a metric space with n elements and distance metric dist. Moreover, let $\mathtt{C} = \{w_1, \ldots, w_k\} \subseteq \mathtt{M}$ be an error correcting code with codewords w_i for $1 \leq i \leq k$. Let d be the minimum distance and t be the error correcting distance of $\mathtt{C}$, which means that $\mathtt{C}$ can detect up to d, and correct up to t errors. In this paper, we only consider linear binary block codes, where $\mathtt{M} = \mathbb{F}_2^n$ and dist corresponds to the Hamming distance. These codes are commonly denoted as $[n, k, d]$ codes and it holds $t = \lfloor (d-1)/2 \rfloor$. Following [12], we formally define a secure sketch as follows:

Definition 1. *A $(\mathtt{M}, m, m', t)$-secure sketch is a pair of probabilistic polynomial time algorithms $\mathsf{Gen}()$ and $\mathsf{Rep}()$ with the following properties: $\mathsf{Gen}()$ takes input $w \in \mathtt{M}$, which is chosen according to a distribution W on $\mathtt{M}$, and returns a bit-string $h \in \{0,1\}^*$. $\mathsf{Rep}()$ takes inputs $w' \in \mathtt{M}$ and h, and returns $w'' \in \mathtt{M}$. Correctness ensures that $w'' = w$ if $h = \mathsf{Gen}(w)$ and $\mathsf{dist}(w, w') \leq t$. The security property guarantees that for any distribution W over $\mathtt{M}$ with min-entropy m, w can be recovered from (a single) $h = \mathsf{Gen}(w)$ with at most probability $2^{-m'}$.*

Next, we specify the syndrome construction that has been informally discussed in Section 3 and that has been shown to implement a secure sketch [12]:

Definition 2. *The syndrome construction is a* $(\mathbb{F}_2^n, n, k, t)$*-secure sketch (Definition1) that is based on a linear binary* $[n, k, d]$ *error correcting block code.* $\mathsf{Gen}(w)$ *computes* $h \leftarrow w \cdot H^T$*, where* H *is the parity check matrix of the underlying code.* $\mathsf{Rep}(w', h)$ *computes* $w \leftarrow w' - e$*, where* e *is determined by decoding the syndrome* $s = h - w' \cdot H^T$ *using the decoding algorithm of the underlying code.*

Note that helper data $h = w \cdot H^T$ corresponds to the syndrome of w. However, since the syndrome construction does not require w to be a codeword, decoding h may most likely fail. To overcome this problem, the reproduction algorithm $\mathsf{Rep}()$ of the syndrome construction decodes the syndrome $s = h - w' \cdot H^T = e \cdot H^T$.

Security definition of reverse fuzzy extractors. Similar to conventional fuzzy extractors, reverse fuzzy extractors should ensure that helper data does not leak the full PUF response. However, for reverse fuzzy extractors this must hold even when *multiple* different helper data for noisy variants of the *same* PUF response are known. This has been formalized by Boyen [8] as *outsider chosen perturbation security*, which is defined based on a security experiment between an unbounded adversary $\mathcal{A}$ and a challenger $\mathcal{C}_{\mathrm{PS}}$. In this experiment, $\mathcal{A}$ interacts with the helper data generator $\mathsf{Gen}()$ of a secure sketch (Definition 1) and obtains helper data for different $w_i = w + e_i$ for a fixed but secret w and different noise vectors (perturbations) e_i that can be adaptively chosen by $\mathcal{A}$. This allows $\mathcal{A}$ to influence the noise, which in case of PUFs can be done by changing the operating conditions such as ambient temperature or supply voltage. The outsider chosen perturbation security experiment is defined as follows: $\mathcal{A}$ sends a description of distribution W over $\mathtt{M}$ to $\mathcal{C}_{\mathrm{PS}}$, which then samples $w \in \mathtt{M}$ according to W. Next, $\mathcal{A}$ interacts with $\mathsf{Gen}()$ and obtains an arbitrary number of helper data $h_i = \mathsf{Gen}(w_i)$ for different $w_i = w + e_i$, where $e_i \in \mathtt{M}$ can be adaptively chosen by $\mathcal{A}$ with the only restriction that the Hamming weight of e_i is less or equal to t. Eventually, $\mathcal{A}$ returns a guess w^* for w. $\mathcal{A}$ wins if $w^* = w$. Based on this security experiment, Boyen [8] sets up the following security definition:

Definition 3. *A* $(\mathtt{M}, m, m', t)$*-secure sketch (Definition 1) is unconditionally secure against adaptive outsider chosen perturbation attacks, if no unbounded adversary* $\mathcal{A}$ *can win the outsider chosen perturbation security experiment with probability greater than* $2^{-m'}$ *for any distribution* W *over* $\mathtt{M}$ *with min-entropy* m*.*

Moreover, Boyen [8] shows that the at the syndrome construction achieves outsider chosen perturbation security:

Theorem 1. *The syndrome construction (Definition 2) is unconditionally secure against adaptive outsider chosen perturbation attacks (Definition 3).*

We now state the security of our reverse fuzzy extractor construction:

Theorem 2. *The reverse fuzzy extractor (Section 3) based on the syndrome construction (Definition 2) is a* $(\mathbb{F}_2^n, n, k, t)$*-secure sketch (Definition1) that achieves outsider perturbation security (Definition 3).*

Proof (Sketch, Theorem 2). Note that $\mathsf{Gen}()$ and $\mathsf{Rep}()$ of the syndrome construction and the the reverse fuzzy extractor based on the syndrome construction are identical. In fact, only the entities that execute $\mathsf{Gen}()$ and $\mathsf{Rep}()$ have been switched. Hence, it is easy to see that the syndrome construction and the reverse fuzzy extractor based on the syndrome construction are equivalent. Thus, since the syndrome construction is a $(\mathbb{F}_2^n, n, k, t)$-secure sketch, the reverse fuzzy extractor based on the syndrome construction is also a $(\mathbb{F}_2^n, n, k, t)$-secure sketch. Consequently, it follows from Theorem 1 that the reverse fuzzy extractor based on the syndrome construction achieves outsider perturbation security. □

6.2 Security of the Authentication Protocol

Correctness. Correctness means that, in case token $\mathcal{T}$ and verifier $\mathcal{V}$ are honest, mutual authentication should be successful.

Definition 4. *A mutual authentication scheme is correct, if an honest $\mathcal{T}$ always makes an honest $\mathcal{V}$ accept, and an honest $\mathcal{V}$ always makes an honest $\mathcal{T}$ accept.*

Theorem 3. *The authentication scheme in Section 4.3 is correct, when based on a PUF that generates responses of length n bits with at most t bit errors, and a $(\mathbb{F}_2^n, n, k, t)$-secure sketch (Definition 1).*

Proof (Sketch, Theorem 3). It is easy to see that the protocol in Section 4.3 is correct if $\mathsf{Rep}(r_i', \mathsf{Gen}(r_i)) = r_i$ for all (r_i, r_i'). The correctness property of the $(\mathbb{F}_2^n, n, k, t)$-secure sketch (Definition 1) ensures that $\mathsf{Rep}(r_i', \mathsf{Gen}(r_i)) = r_i$ if $\mathsf{dist}(r_i, r_i') \leq t$. If the PUF generates responses of length n bits with a bit error rate of at most ρ, then the probability of $\mathsf{dist}(r, r') \leq t$ can be expressed as the cumulative binomial distribution in t with parameters ρ and n. Note that t is chosen such that this probability, which is an upper bound for the false rejection rate of the authentication, becomes very small. Hence, a $(\mathbb{F}_2^n, n, k, t)$-secure sketch can then recover r from r' with overwhelming probability. □

Note that the implementation in Section 5 can handle PUFs with $\rho \leq 10\%$. When both verifier and token are trusted, it achieves an authentication failure rate of less than $10^{-6.97}$, which is acceptable for most commercial applications.

Token Authentication. Token authentication means that adversary $\mathcal{A}$ should not be able to make a legitimate verifier $\mathcal{V}$ to accept $\mathcal{A}$ as a legitimate token $\mathcal{T}$. Following [34,1], we formalize token authentication based on a security experiment, where $\mathcal{A}$ must make an honest $\mathcal{V}$ to authenticate $\mathcal{A}$ as $\mathcal{T}$. Hereby, $\mathcal{A}$ can arbitrarily interact with $\mathcal{T}$ and $\mathcal{V}$, which both are simulated by a challenger $\mathcal{C}_{\mathrm{TA}}$. However, since in general it is not possible to prevent simple relay attacks, $\mathcal{A}$ is not allowed to just forward all messages from $\mathcal{T}$ to $\mathcal{V}$.[5] This means that at least

[5] Note that simple relay attacks can be mitigated by distance bounding techniques. However, for simplicity we excluded relay attacks because the main focus of the protocol is demonstrating the use of reverse fuzzy extractors.

some of the protocol messages that made $\mathcal{V}$ accept must have been computed by $\mathcal{A}$. More specifically, the token authentication experiment is as follows: $\mathcal{C}_{\mathrm{TA}}$ initializes $\mathcal{T}$ and $\mathcal{V}$. Then, $\mathcal{C}_{\mathrm{TA}}$ initializes $\mathcal{A}$ with the public system parameters. Next, $\mathcal{A}$ can arbitrarily interact with $\mathcal{T}$ and $\mathcal{V}$ that are simulated by $\mathcal{C}_{\mathrm{TA}}$. Hereby, $\mathcal{A}$ can eavesdrop on authentication protocol-runs between an honest $\mathcal{T}$ and an honest $\mathcal{V}$, and manipulate protocol messages exchanged between $\mathcal{V}$ and $\mathcal{T}$. Further, $\mathcal{A}$ can start authentication protocol-runs as $\mathcal{V}$ or $\mathcal{T}$ with $\mathcal{C}_{\mathrm{TA}}$. $\mathcal{A}$ wins, if it makes $\mathcal{V}$ accept after a polynomial (in l) number of queries to $\mathcal{C}_{\mathrm{TA}}$.

Definition 5. *An authentication scheme achieves μ-token authentication, if no probabilistic polynomial time adversary $\mathcal{A}$ wins the token authentication experiment with probability greater than $2^{-\mu}$.*

Theorem 4. *The authentication scheme in Section 4.3 achieves k-token authentication (Definition 5) in the random oracle model, when using the reverse fuzzy extractor (Section 3) based on the syndrome construction (Definition 2).*

In the following, we focus on the variant of our authentication scheme that uses only one single challenge/response pair, i.e., where $q = 1$ (Section 4.3). The proof can be easily extended for $q > 1$. Due to space restrictions we give only proof sketches and provide detailed proofs in the full version of the paper [17].

Proof (Sketch, Theorem 4). We show that, if there is an adversary $\mathcal{A}$ that violates token authentication (Definition 5) with probability greater than 2^{-k}, then $\mathcal{A}$ can be transformed into an adversary $\mathcal{B}$ that violates outsider chosen perturbation security of the reverse fuzzy extractor (Theorem 2). Note that, in the chosen perturbation security experiment (Definition 3), $\mathcal{B}$ interacts with a helper data generator oracle $\mathsf{Gen}()$ that, when queried with e_j, returns $h_j = \mathsf{Gen}(r + e_j)$ for a fixed but unknown $r \in \mathbb{F}_2^n$. Based on this $\mathsf{Gen}()$-oracle, $\mathcal{B}$ simulates challenger $\mathcal{C}_{\mathrm{TA}}$ of the token authentication security experiment (Definition 5) such that $\mathcal{A}$ cannot distinguish between $\mathcal{B}$ and $\mathcal{C}_{\mathrm{TA}}$. Hereby, $\mathcal{A}$ and $\mathcal{B}$ have access to the same random oracle $\mathsf{Hash}()$, and $\mathcal{B}$ records all queries x made by $\mathcal{A}$ to $\mathsf{Hash}()$ and the corresponding responses $\mathsf{Hash}(x)$ in a list L. Since, $\mathcal{A}$ cannot distinguish $\mathcal{B}$ from $\mathcal{C}_{\mathrm{TA}}$, by assumption $\mathcal{A}$ violates token authentication (Definition 5) with probability greater than 2^{-k}. $\mathcal{B}$ uses L to extract $r^* = r$ from the protocol message (h, a) generated by $\mathcal{A}$ that finally makes $\mathcal{V}$ accept. Note that, the random oracle ensures that $(x, a) \in L$. Hence, $\mathcal{B}$ can extract r with probability greater than 2^{-k}, which contradicts outsider chosen perturbation security of the reverse fuzzy extractor (Theorem 2). □

Note that in practice, the success probability $2^{-\mu}$ (Definition 5) of $\mathcal{A}$ may depend on the output length t of the hash function implementing the random oracle: In case $t < k$ $\mathcal{A}$ could simply guess the correct hash digest a with probability 2^{-t}. For the implementation of the syndrome construction based reverse fuzzy extractor (Section 5), we have $t = 128 < k = 147$, and thus $\mu = 128$.

Verifier Authentication. Verifier authentication means that adversary $\mathcal{A}$ should not be able to make an honest token $\mathcal{T}$ to accept $\mathcal{A}$ as a legitimate verifier $\mathcal{V}$.

This is formalized by a verifier authentication security experiment between $\mathcal{A}$ and a challenger $\mathcal{C}_{\text{VA}}$ that is identical to the token authentication experiment with the only difference that $\mathcal{A}$ wins, if $\mathcal{A}$ makes $\mathcal{T}$ accept after a polynomial (in t and the bit length of PUF responses) number of queries.

Definition 6. *An authentication scheme achieves μ-verifier authentication, if no probabilistic polynomial time adversary $\mathcal{A}$ wins the verifier authentication experiment with probability greater than $2^{-\mu}$.*

Theorem 5. *The authentication scheme in Section 4.3 achieves k-verifier authentication (Definition 6) in the random oracle model, when using the reverse fuzzy extractor (Section 3) based on the syndrome construction (Definition 2), when the underlying PUF generates at least ρ bit errors each time it is evaluated.*

Proof (Sketch, Theorem 5). We show that, if there is an adversary $\mathcal{A}$ that violates verifier authentication (Definition 6) with probability greater than 2^{-k}, then $\mathcal{A}$ can be transformed into an adversary $\mathcal{B}$ that violates outsider chosen perturbation security of the reverse fuzzy extractor (Theorem 2). $\mathcal{B}$ simulates challenger $\mathcal{C}_{\text{VA}}$ of the verifier authentication security experiment (Definition 5) based on the $\mathsf{Gen}()$-oracle such that $\mathcal{A}$ cannot distinguish between $\mathcal{B}$ and $\mathcal{C}_{\text{VA}}$ in a similar way as in the proof of Theorem 4. Hereby, $\mathcal{A}$ and $\mathcal{B}$ have access to the same random oracle, and $\mathcal{B}$ records all queries x made by $\mathcal{A}$ to $\mathsf{Hash}()$ and the corresponding responses $\mathsf{Hash}(x)$ in a list L. Since, $\mathcal{A}$ cannot distinguish between $\mathcal{B}$ and $\mathcal{C}_{\text{VA}}$, by assumption $\mathcal{A}$ violates verifier authentication (Definition 6) with probability greater than 2^{-k}. $\mathcal{B}$ uses L to extract $r^* = r$ from the protocol message b generated by $\mathcal{A}$ that finally makes $\mathcal{T}$ accept. Note that, the random oracle assumption ensures that $(x, b) \in L$, while the bit errors in the PUF responses ensure that $\mathcal{A}$ cannot just replay an old b. Hence, $\mathcal{B}$ can extract r with probability greater than 2^{-k}, which contradicts outsider chosen perturbation security of the reverse fuzzy extractor (Theorem 2). □

7 Related Work

One of the first proposals of using PUFs in RFID systems is by Ranasinghe et al. [32], who propose storing a set of PUF challenge/response pairs (CRPs) in a database that can later be used by RFID readers to identify a token. The idea is that the reader queries the PUF of the token with a random challenge from the database and verifies whether the response of the token is similar to the database entry. One problem of this approach is that CRPs cannot be re-used since this enables replay attacks. Hence, the number of token authentications is limited by the number of CRPs in the database. This scheme has been implemented and analyzed by Devadas et al. [10]. Holcomb et al. [18] present a similar scheme based on an SRAM-PUF on RFID chips. Another approach to PUF-based authentication by Bolotnyy and Robins [6] aims to prevent unauthorized tracking of tokens. A major drawback of their scheme is that tokens can only be authenticated a limited number of times without being re-initialized, which enables denial-of-service attacks.

Tuyls and Batina [38] propose using a PUF to reconstruct the authentication secret of a token whenever it is needed instead of storing it in secure non-volatile memory. Since the key is inherently hidden in the PUF, obtaining the key by hardware-related attacks is supposed to be intractable. However, the scheme proposed by Tuyls and Batina [38] relies on public-key cryptography, which is still much too expensive for low-cost RFID tokens. Several other authentication schemes for RFID exist that use PUF-based key storage to protect against unauthorized tracking of tokens [9,34] and relay attacks [19]. However, these schemes require the expensive decoding operation of a fuzzy extractor to be implemented on the token, which is too expensive for low-cost RFIDs.

8 Conclusion

We presented a new lightweight PUF-based authentication scheme providing mutual authentication of RFID tokens and readers. Our scheme is resistant to emulation attacks, supports an unlimited number of token authentications, and does not require the reader to store a large number of PUF challenge/response pairs. Furthermore, we introduce the concept of *reverse fuzzy extractors*, a novel approach to correct noise in PUF responses moving the computationally expensive error correction process from the resource-constrained PUF-enabled token to the more powerful RFID reader. Reverse fuzzy extractors are applicable to device authentication and PUF-based key storage (where the key is used to communicate with an external entity) and can significantly reduce the area costs of secure PUF implementations. Future work includes a highly optimized implementation of our scheme and developing lightweight privacy-preserving authentication protocols based on PUFs and reverse fuzzy extractors.

Acknowledgement. This work has been supported in part by the European Commission under grant agreement ICT-2007-238811 UNIQUE.

References

1. Armknecht, F., Chen, L., Sadeghi, A.R., Wachsmann, C.: Anonymous authentication for RFID systems. In: Radio Frequency Identification: Security and Privacy Issues (RFIDSec), LNCS, vol. 6370, pp. 158–175. Springer (2010)
2. Armknecht, F., Maes, R., Sadeghi, A.R., Standaert, F.X., Wachsmann, C.: A formal foundation for the security features of physical functions. In: IEEE Symposium on Security and Privacy. pp. 397–412. IEEE (May 2011)
3. Armknecht, F., Maes, R., Sadeghi, A.R., Sunar, B., Tuyls, P.: Memory leakage-resilient encryption based on physically unclonable functions. In: Advances in Cryptology (ASIACRYPT). LNCS, vol. 5912, pp. 685–702 (2009)
4. Avoine, G., Lauradoux, C., Martin, T.: When compromised readers meet RFID. The 5th Workshop on RFID Security (RFIDSec) (2009)
5. Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K., Verbauwhede, I.: SPONGENT: A Lightweight Hash Function. In: Cryptographic Hardware and Embedded Systems (CHES). LNCS, vol. 6917, pp. 312–327. Springer (2011)

6. Bolotnyy, L., Robins, G.: Physically unclonable Function-Based security and privacy in RFID systems. In: IEEE International Conference on Pervasive Computing and Communications (PerCom). pp. 211–220. IEEE (2007)
7. Bösch, C., Guajardo, J., Sadeghi, A.R., Shokrollahi, J., Tuyls, P.: Efficient helper data key extractor on FPGAs. In: Cryptographic Hardware and Embedded Systems (CHES). LNCS, vol. 5154, pp. 181–197. Springer (2008)
8. Boyen, X.: Reusable cryptographic fuzzy extractors. In: ACM Conference on Computer and Communications Security (ACM CCS). pp. 82–91. ACM (2004)
9. Bringer, J., Chabanne, H., Icart, T.: Improved privacy of the Tree-Based hash protocols using physically unclonable function. In: Security and Cryptography for Networks (SCN), LNCS, vol. 5229, pp. 77–91. Springer (2008)
10. Devadas, S., Suh, E., Paral, S., Sowell, R., Ziola, T., Khandelwal, V.: Design and implementation of PUF-based unclonable RFID ICs for Anti-Counterfeiting and security applications. In: International Conference on RFID. pp. 58–64. IEEE (2008)
11. Dodis, Y., Katz, J., Reyzin, L., Smith, A.: Robust fuzzy extractors and authenticated key agreement from close secrets. In: Advances in Cryptology (CRYPTO), LNCS, vol. 4117, pp. 232–250. Springer (2006)
12. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In: Advances in Cryptology (EUROCRYPT). LNCS, vol. 3027, pp. 523–540 (2004)
13. Garcia, F.D., de Koning Gans, G., Muijrers, R., van Rossum, P., Verdult, R., Schreur, R.W., Jacobs, B.: Dismantling MiFare classic. In: Jajodia, S., Lopez, J. (eds.) 13th European Symposium on Research in Computer Security (ESORICS). LNCS, vol. 5283, pp. 97–114. Springer (2008)
14. Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Controlled physical random functions. In: Computer Security Applications Conference. pp. 149–160. IEEE (2002)
15. Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Silicon physical random functions. In: ACM Conference on Computer and Communications Security (ACM CCS). pp. 148–160 (2002)
16. Guajardo, J., Kumar, S.S., Schrijen, G.J., Tuyls, P.: FPGA intrinsic PUFs and their use for IP protection. In: Workshop on Cryptographic Hardware and Embedded Systems (CHES). LNCS, vol. 4727, pp. 63–80 (2007)
17. van Herrewege, A., Katzenbeisser, S., Maes, R., Peeters, R., Sadeghi, A.R., Verbauwhede, I., Wachsmann, C.: Reverse fuzzy extractors: Enabling lightweight mutual authentication for PUF-enabled RFIDs. Cryptology ePrint Archive (to appear)
18. Holcomb, D.E., Burleson, W.P., Fu, K.: Initial SRAM state as a fingerprint and source of true random numbers for RFID tags. In: Conference on RFID Security (RFIDSec) (2007)
19. Kardas, S., Kiraz, M.S., Bingol, M.A., Demirci, H.: A novel RFID distance bounding protocol based on physically unclonable functions. Cryptology ePrint Archive, Report 2011/075 (2011)
20. Katzenbeisser, S., Kocabaş, U., van der Leest, V., Sadeghi, A.R., Schrijen, G.J., Schröder, H., Wachsmann, C.: Recyclable PUFs: Logically reconfigurable PUFs. In: Workshop on Cryptographic Hardware and Embedded Systems (CHES). LNCS, vol. 6917, pp. 374–389. Springer (2011)
21. Kumar, S., Guajardo, J., Maes, R., Schrijen, G.J., Tuyls, P.: Extended abstract: The butterfly PUF protecting IP on every FPGA. In: IEEE Workshop on Hardware-Oriented Security and Trust (HOST). pp. 67–70 (2008)

22. Lee, J.W., Lim, D., Gassend, B., Suh, G.E., van Dijk, M., Devadas, S.: A technique to build a secret key in integrated circuits for identification and authentication application. In: Symposium on VLSI Circuits. pp. 176–159 (2004)
23. van der Leest, V., Schrijen, G.J., Handschuh, H., Tuyls, P.: Hardware intrinsic security from D flip-flops. In: ACM Workshop on Scalable Trusted Computing (ACM STC). pp. 53–62 (2010)
24. Lim, D., Lee, J.W., Gassend, B., Suh, G.E., van Dijk, M., Devadas, S.: Extracting secret keys from integrated circuits. IEEE Transactions on VLSI Systems 13(10), pp. 1200–1205 (2005)
25. Lin, L., Holcomb, D., Krishnappa, D.K., Shabadi, P., Burleson, W.: Low-power sub-threshold design of secure physical unclonable functions. In: International Symposium on Low Power Electronics and Design (ISLPED). pp. 43–48 (2010)
26. Maes, R., Tuyls, P., Verbauwhede, I.: Intrinsic PUFs from flip-flops on reconfigurable devices. In: Workshop on Information and System Security (WISSec). p. 17 (2008)
27. Maes, R., Verbauwhede, I.: Physically unclonable functions: A study on the state of the art and future research directions. In: Towards Hardware-Intrinsic Security, pp. 3–37. Springer (2010)
28. Maiti, A., Casarona, J., McHale, L., Schaumont, P.: A large scale characterization of RO-PUF. In: IEEE Symposium on Hardware-Oriented Security and Trust (HOST). pp. 94–99 (2010)
29. Nithyanand, R., Tsudik, G., Uzun, E.: Readers behaving badly: Reader revocation in PKI-based RFID systems. Cryptology ePrint Archive, Report 2009/465 (2009)
30. Öztürk, E., Hammouri, G., Sunar, B.: Towards robust low cost authentication for pervasive devices. In: International Conference on Pervasive Computing and Communications (PerCom). pp. 170–178. IEEE (2008)
31. Pappu, R.S., Recht, B., Taylor, J., Gershenfeld, N.: Physical one-way functions. Science 297, pp. 2026–2030 (2002)
32. Ranasinghe, D.C., Engels, D.W., Cole, P.H.: Security and privacy: Modest proposals for low-cost RFID systems. Auto-ID Labs Research Workshop (2004)
33. Rührmair, U., Sehnke, F., Sölter, J., Dror, G., Devadas, S., Schmidhuber, J.: Modeling attacks on physical unclonable functions. In: ACM conference on Computer and communications security (ACM CCS). pp. 237–249 (2010)
34. Sadeghi, A.R., Visconti, I., Wachsmann, C.: Enhancing RFID Security and Privacy by Physically Unclonable Functions. In: Towards Hardware-Intrinsic Security, pp. 281–305. Springer (2010)
35. NXP Semiconductors: Web site of MiFare. `http://mifare.net/` (December 2011)
36. Su, Y., Holleman, J., Otis, B.: A 1.6pJ/bit 96% stable chip-ID generating circuit using process variations. In: IEEE International Solid-State Circuits Conference (ISSCC). pp. 406–611 (2007)
37. Suh, G.E., Devadas, S.: Physical unclonable functions for device authentication and secret key generation. In: Design Automation Conference. pp. 9–14 (2007)
38. Tuyls, P., Batina, L.: RFID-tags for anti-counterfeiting. In: The Cryptographers' Track at the RSA Conference. LNCS, vol. 3860, pp. 115–131. Springer (2006)
39. Tuyls, P., Schrijen, G.J., Škorić, B., van Geloven, J., Verhaegh, N., Wolters, R.: Read-proof hardware from protective coatings. In: Workshop on Cryptographic Hardware and Embedded Systems (CHES). LNCS, vol. 4249, pp. 369–383 (2006)
40. Škorić, B., Tuyls, P., Ophey, W.: Robust key extraction from physical uncloneable functions. In: Applied Cryptography and Network Security (ACNS). LNCS, vol. 3531, pp. 407–422 (2005)