Privacy and Reader-first Authentication in Vaudenay's RFID Model with Temporary State Disclosure

Ferucio Laurenţiu Ţiplea and Cristian Hristea

Abstract

Privacy and mutual authentication under corruption with temporary state disclosure are two significant requirements for real-life applications of RFID schemes. No RFID scheme is known so far to meet these two requirements. In this paper we propose two practical RFID schemes that fill this gap. The first one achieves destructive privacy, while the second one narrow destructive privacy, in Vaudenay's model with temporary state disclosure. Both of them provide mutual (reader-first) authentication. In order to achieve these privacy levels we use Physically Unclonable Functions (PUFs) to assure that the internal secret of the tag remains hidden against an adversary with invasive capabilities. Our first RFID scheme cannot be desynchronized for more than one step, while the second one avoids the use of random generators on tags. Detailed security and privacy proofs are provided.

Index Terms

RFID scheme, PUF, authentication, privacy.

I. Introduction

RADIO Frequency Identification (RFID) refers to a technology whereby digital data encoded in *RFID tags* is transmitted to a *reader* via radio waves. A *back-end system*, which has an online database, is securely connected to the reader to collect, filter, process, and manage RFID data. It also stores complete information associated with the RFID tags in order to be able to authenticate them.

The RFID technology has been implemented in many significant areas such as toll collection systems, identification and tracking of various kinds of objects, consumer products, or access control. With the increasing usages to healthcare, electronic passports, and personal ID cards, the potential security threats and compliance risks have become enormous. In such a context, the need for secure and private communication protocols between reader and tags becomes crucial. Moreover, when developing such protocols, account must be taken of the adversary model to which they should resist. A widely accepted adversary model was proposed in [1], [2], now called *Vaudenay's RFID model*. According to it, an adversary is a probabilistic polynomial-time algorithm that can do a lot of things such as:

- create legitimate or illegitimate tags;
- draw one or more tags according to some chosen probability distribution, and release drawn tags;
- launch protocol instances with drawn tags;
- send messages to reader or drawn tags;
- corrupt drawn tags to retrieve their internal states;
- get the result of a completed protocol instance.

Strong adversaries have all the above capabilities. *Destructive adversaries* destroy tags after corrupting them. An adversary that can corrupt tags only at the end is a *forward adversary*; finally, an adversary that cannot corrupt tags is a *weak adversary*. Orthogonal to these classes of adversaries is the class of *narrow adversaries* that cannot see the result of completed protocol instances. Narrow adversaries with the capabilities discussed above give rise to *narrow strong*, *narrow destructive*, *narrow forward*,

The authors are with the Department of Computer Science, Alexandru Ioan Cuza University of Iaşi, Iaşi, Romania, e-mail: ferucio.tiplea@uaic.ro and cristi.hristea@gmail.com.

and *narrow weak* adversaries. Thus, Vaudenay's model comes with eight classes of adversaries and, consequently, eight levels of RFID privacy. Among these, destructive privacy together with reader-first authentication under corruption with temporary state disclosure plays an important role in practice. For instance, tag destruction under corruption is an important requirement when the tag is used for access control. Likewise, the disclosure of temporary state under tag corruption is a serious threat in practice. Reader-first authentication [3] assures that the tag will give its private data only when it authenticates the reader. Therefore, tag tracking and data theft are prevented when the reader is fake. All these together mean that we need RFID schemes that provide destructive privacy and reader-first authentication under corruption with temporary state disclosure.

Contribution: When Vaudenay's model was proposed, it was not very clear whether the tag corruption reveals the permanent state or the full (permanent and temporary) state of the tag. Later, this aspect was clarified and it was shown that the mutual authentication protocols proposed in [2] do not achieve the claimed privacy level under corruption with temporary state disclosure. Additionally, this does not even happen [4] with newer protocols like those in [5], [6]. We face thus the problem that no mutual authentication RFID scheme proposed so far achieves privacy under corruption with temporary state disclosure.

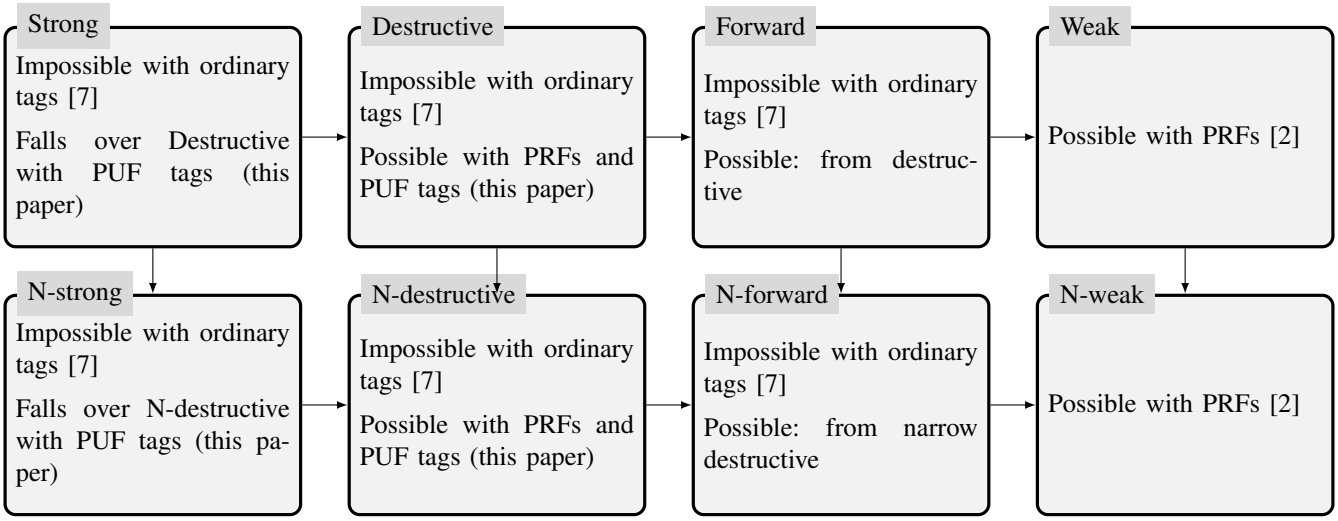

Fig. 1. Privacy and mutual authentication in Vaudenay's model with temporary state disclosure

In this paper we provide the first mutual authentication RFID schemes that achieve destructive and narrow destructive privacy in Vaudenay's model with temporary state disclosure. Moreover, in our schemes the tag authenticates first the reader (this is the reader-first approach [3]) that guarantees the information goes from tag to a trusted reader. The diagram in Figure 1 shows clearly our contribution in the general context of Vaudenay's model with temporary state disclosure.

It is known that no privacy level can be achieved with ordinary tags (that is, tags that only run cryptographic primitives) under mutual authentication and corruption with temporary state disclosure [7]. Therefore, the schemes we propose are based on *PUF tags*, that is tags endowed with *physically unclonable functions* (PUFs), a novel class of hardware security primitives that are in use for a while. The security proofs we provide to our schemes are very detailed. We also elaborate on the tag-reader desynchronization problem and its connection to narrow privacy.

Related work: The pseudo-random function (PRF) based RFID scheme in [2] achieves weak privacy and mutual authentication in Vaudenay's model. It is straightforward to see that the proof in [2] works even in the case of corruption with temporary state disclosure. The first PUF based RFID scheme that achieves destructive privacy and mutual authentication in Vaudenay's model (where corruption does not

disclose the temporary state of tags) was proposed in [4], as an extension of the scheme in [8], [9] (that only achieves unilateral authentication).

In [5], [6], two PUF based RFID schemes have been proposed and claimed that they achieve (narrow) destructive privacy and mutual authentication in Vaudenay's model with temporary state disclosure. Unfortunately, neither of them reaches even the narrow forward privacy level [4].

As far as we are concerned, these are the only RFID schemes related to our work.

Paper structure: The paper consists of seven sections, the first one being the introduction. The basic terminology and notation used throughout this paper is introduced in Sections II and III. A general discussion on privacy and mutual authentication in Vaudenay's model with temporary state disclosure is purported in Section IV. Our first RFID scheme, that achieves destructive privacy and mutual authentication in Vaudenay's model with temporary state disclosure, is presented in Section V. In the sixth section we introduce our second RFID scheme that achieves narrow destructive privacy and mutual authentication in the same model. The last section concludes the paper.

II. Basic definitions and notation

We fix in this section the basic terminology and notation used throughout this paper.

Probabilistic polynomial time algorithms and negligible functions: We use *probabilistic polynomial time* (PPT) algorithms $\mathcal{A}$ as defined in [10]. If $\mathcal{O}$ is an oracle, then $\mathcal{A}^{\mathcal{O}}$ denotes that $\mathcal{A}$ has oracle access to $\mathcal{O}$. When the oracle $\mathcal{O}$ implements some function f, we simply write $\mathcal{A}^f$ to denote that $\mathcal{A}$ has oracle access to f. This means that whenever $\mathcal{A}$ sends a value x to the oracle, it gets back $f(x)$.

If A is a set, then $a \leftarrow A$ means that a is uniformly at random chosen from A. If $\mathcal{A}$ is a probabilistic algorithm, then $a \leftarrow \mathcal{A}$ means that a is an output of $\mathcal{A}$ for some given input.

The asymptotic approach to security makes use of security parameters, denoted by λ in our paper. A positive function $f(\lambda)$ is called *negligible* if for any positive polynomial $poly(\lambda)$ there exists n_0 such that $f(\lambda) < 1/poly(\lambda)$, for any $\lambda \geq n_0$. $f(\lambda)$ is called *overwhelming* if $1 - f(\lambda)$ is negligible.

Pseudo-random functions: A *pseudo-random function* (PRF) is a family of functions with the property that if we randomly choose a function from this family then its input-output behavior is computationally indistinguishable from that of a random function. To be more precise, consider and fix two polynomials ℓ_1 and ℓ_2 with positive values. Given a set $\mathcal{K}$ of *keys* and $\lambda \in \mathbb{N}$, define $\mathcal{K}_\lambda = \{K \in \mathcal{K} \mid |K| = \lambda\}$. A *family of functions* indexed by $\mathcal{K}$ is a construction $F = (F_K)_{K \in \mathcal{K}}$, where F_K is a function from $\{0,1\}^{\ell_1(|K|)}$ to $\{0,1\}^{\ell_2(|K|)}$. We also define $U_\lambda = \{f \mid f : \{0,1\}^{\ell_1(\lambda)} \to \{0,1\}^{\ell_2(\lambda)}\}$ and $U = (U_\lambda)_\lambda$.

We say that F is *computationally indistinguishable* from U if, for any PPT algorithm $\mathcal{A}$ with oracle access to functions, its *advantage*

$$Adv^{prf}_{\mathcal{A},F}(\lambda) = |P(1 \leftarrow \mathcal{A}^{F_K}(1^\lambda) \,:\, K \leftarrow \mathcal{K}_\lambda) - P(1 \leftarrow \mathcal{A}^{g}(1^\lambda) \,:\, g \leftarrow U_\lambda)|$$

is negligible (as a functions of λ).

$F = (F_K)_{K \in \mathcal{K}}$ is called a *pseudo-random function* if it is:

1) *Efficiently computable:* there exists a deterministic polynomial-time algorithm that on input λ, $K \in \mathcal{K}_\lambda$, and $x \in \{0,1\}^{\ell_1(\lambda)}$, returns $F_K(x)$;
2) *Pseudo-random:* F is computationally indistinguishable from U.

To prove that F is a PRF, we usually use a *bit guessing game* between a *challenger* $\mathcal{C}$ and an adversary $\mathcal{A}$ (the game is parameterized by a security parameter λ):

1) $\mathcal{C}$ randomly chooses $b \leftarrow \{0,1\}$;
2) if $b = 1$ then $\mathcal{C}$ randomly chooses $K \leftarrow \mathcal{K}_\lambda$ and sets $f = F_K$; otherwise, $\mathcal{C}$ randomly chooses $f \leftarrow U_\lambda$;
3) $\mathcal{C}$ provides oracle access to f for $\mathcal{A}$;
4) At some point, $\mathcal{A}$ outputs a bit b'.

The probability $\mathcal{A}$ wins the game is denoted $P(b' = b)$. Now, one can see that F is a PRF if it is efficiently computable and $Adv^{prf}_{\mathcal{A},F}(\lambda) = |P(b = b') - 1/2|$ is negligible.

Physically unclonable functions: A *physically unclonable function* (PUF) can be seen as a physical object that, when queried with a challenge x generates a response y that depends on both x and the specific physical properties of the object. PUFs are typically assumed to be *physically unclonable* (it is infeasible to produce two PUFs that cannot be distinguished based on their challenge/response behavior), *unpredictable* (it is infeasible to predict the response to an unknown challenge), and *tamper-evident* (any attempt to physically access the PUF irreversible changes the challenge/response behavior).

Unfortunately, PUFs are subject to noise induced by the operating conditions, such as supply voltage or ambient temperature. Therefore, PUFs return slightly different responses when queried with the same challenge multiple times. However, from a theoretical point of view it is assumed that PUFs return a similar response when queried with the same challenge multiple times (this is usually called *robustness*).

Based on these, we adopt here the concept of an *ideal PUF* slightly different than in [8]. Namely, an *ideal PUF* is a physical object with a challenge/response behavior that implements a function $P : \{0,1\}^p \to \{0,1\}^k$, where p and k are of polynomial size in λ, such that:

1) P is computationally indistinguishable from U;
2) Any attempt to physically tamper with the object implementing P results in destruction of P (P cannot be evaluated any more).

III. RFID SCHEMES AND SYSTEMS

From an informal point of view, an RFID system [11], [12] consists of a *reader*, a set of *tags*, and a *communication protocol* between reader and tags. The reader is a transceiver that has associated a database that stores information about tags. Its task is to identify *legitimate tags* (that is, tags with information stored in its database) and to reject all the other incoming communication. The reader and its database are trusted entities, and the communication between them is secure. A tag is a transponder device with much more limited computation capabilities than the reader. Depending on tag, it can perform simple logic operations, symmetric key, or even public key cryptography. Each tag has a *permanent* (or *internal*) *memory* that stores the state values, and a *temporary* (or *volatile*) *memory* that can be viewed as a set of *volatile variables* used to carry out the necessary computations.

RFID schemes: Let $\mathcal{R}$ be a *reader identifier* and $\mathcal{T}$ be a set of *tag identifiers* whose cardinal is polynomial in some security parameter λ. An ***RFID scheme over*** $(\mathcal{R}, \mathcal{T})$ [1], [2] is a triple $\mathcal{S} = (SetupR,\ SetupT,\ Ident)$ of PPT algorithms, where:

1) $SetupR(\lambda)$ inputs a security parameter λ and outputs a triple (pk, sk, DB) consisting of a key pair (pk, sk) and an empty database DB. pk is public, while sk is kept secret by reader;
2) $SetupT(pk, ID)$ initializes the tag identified by ID. It outputs an initial tag state S and a secret key K. A triple $(ID, f(S), K)$ is stored in the reader's database DB, where f is a public function that extracts some information from tag's initial state S;
3) $Ident(pk;\ \mathcal{R}(sk, DB);\ ID(S))$ is an interactive protocol between the reader identified by $\mathcal{R}$ (with its private key sk and database DB) and a tag identified by ID (with its state S) in which the reader ends with an output consisting of ID or $\perp$. The tag may end with no output (*unilateral authentication*), or it may end with an output consisting of OK or $\perp$ (*mutual authentication*).

The meaning of $SetupR(\lambda)$ is that it "creates" a reader identified by $\mathcal{R}$ and initializes it (and also establishes some public parameters of the system). We simply refer to the reader such created as being $\mathcal{R}$. The meaning of $SetupT(pk, ID)$ is that it "creates" a tag identified by ID, initializes it with an initial tag state, and also register this tag with the reader by storing some information about it in the reader's database. We denote this tag by $\mathcal{T}_{ID}$. The meaning of the reader's output ID ($\perp$) is that it authenticates (rejects) the tag. Similarly, the tag outputs OK ($\perp$) when it authenticates (rejects) the reader.

Mutual authentication where the tag (reader) authenticates first the reader (tag) is also called *reader-first* (*tag-first*) *authentication*.

The *correctness* of an RFID scheme means that, regardless of how the system is set up, after each complete execution of the interactive protocol between the reader and a legitimate tag, the reader outputs tag's identity with overwhelming probability. For mutual authentication RFID schemes, *correctness* means that the reader outputs tag's identity and the tag outputs OK with overwhelming probability.

RFID system: An *RFID system* is an instantiation of an RFID scheme. This is done by a trusted *operator* $\mathcal{I}$ who establishes the reader identifier $\mathcal{R}$, the set $\mathcal{T}$ of tag identifiers, and runs an RFID scheme over $(\mathcal{R}, \mathcal{T})$. In a given setting, the reader is initialized exactly once, while each tag at most once. Thus, the reader's database does not store different entries for the same tag. However, different settings with the same RFID scheme may initialize the reader and the tags in different ways.

Adversaries: The two most basic security requirements for RFID schemes are *authentication* and *untraceability*. To formalize them, the concept of an *adversary model* is needed. There have been several proposal for this, such as [1], [2], [13]–[18]. One of the most influential, which we follow in this paper, is *Vaudenay's model* [1], [2]. We recall below this model as in [4]. Thus, we assume first that some oracles the adversary may query share and manage a common list of tags $ListTags$, which is initially empty. This list includes exactly one entry for each tag created and active in the system. A tag entry consists of several fields with information about the tag, such as: the (permanent) identity of the tag (which is an element from $\mathcal{T}$), the temporary identity of the tag (this field may be empty saying that the tag is *free*), a bit value saying whether the tag is legitimate (the bit is one) or illegitimate (the bit is zero). When the temporary identity field is non-empty, its value uniquely identifies the tag, which is called *drawn* in this case. The adversary may only interact with drawn tags by means of their temporary identities.

The oracles an adversary may query are:

1) $CreateTag^b(ID)$: Creates a free tag $\mathcal{T}_{ID}$ with the identifier ID by calling $SetupT(pk, ID)$ to generate a pair (K, S). If $b = 1$, $(ID, f(S), K)$ is added to DB and the tag is considered *legitimate*; otherwise ($b = 0$), the tag is considered *illegitimate*. Moreover, a corresponding entry is added to $ListTags$;
2) $DrawTag(\delta)$: This oracle chooses a number of free tags according to the distribution δ, let us say n, and draws them. That is, n temporary identities $vtag_1, \ldots, vtag_n$ are generated and the corresponding tag entries in $ListTags$ are filled with them. The oracle outputs $(vtag_1, b_1, \ldots, vtag_n, b_n)$, where b_i specifies whether the tag $vtag_i$ is legitimate ot not.
   As one can see, $DrawTag$ provides the adversary with access to some free tags by means of temporary identifiers, and gives information on whether the tags are legitimate or not (but no other information);
3) $Free(vtag)$: Removes the temporary identity $vtag$ in the corresponding entry in $ListTags$, and the tag becomes free. The identifier $vtag$ will no longer be used. We assume that when a tag is freed, its temporary state is erased. This is a natural assumption that corresponds to the fact that the tag is no longer powered by reader;
4) $Launch()$: Launches a new protocol instance and assigns a unique identifier to it. The oracle outputs the identifier;
5) $SendReader(m, \pi)$: Outputs the reader's answer when the message m is sent to it as part of the protocol instance π. When m is the empty message, abusively but suggestively denoted by $\emptyset$, this oracle outputs the first message of the protocol instance π, assuming that the reader does the first step in the protocol.
   We emphasize that the reader's answer is conceived as the message sent to the tag by the communication channel and not as the reader's decision output (tag identity or $\perp$). Therefore, if the reader does not send anything to the tag, the output of this oracle is empty;
6) $SendTag(m, vtag)$: outputs the tag's answer when the message m is sent to the tag referred to by $vtag$. When m is the empty message, this oracle outputs the first message of the protocol instance π, assuming that the tag does the first step in the protocol.

As in the case of the $SendReader$ oracle, we emphasize that the tag's answer is conceived as the message sent to the reader by the communication channel and not as the tag's decision output (OK or $\perp$). Therefore, if the tag does not send anything to the reader, the output of this oracle is empty;

7) $Result(\pi)$: Outputs $\perp$ if in session π the reader has not yet made a decision on tag authentication (this also includes the case when the session π does not exist), 1 if in session π the reader authenticated the tag, and 0 otherwise (this oracle is both for unilateral and mutual authentication);
8) $Corrupt(vtag)$: Outputs the current permanent (internal) state of the tag referred to by $vtag$, when the tag is not involved in any computation of any protocol step (that is, the permanent state before or after a protocol step). For an extended discussion on this oracle the reader is referred to [4].

We emphasize that $Corrupt$ does not return snapshots of the tag's memory during its computations. When the $Corrupt$ oracle returns the full state, we will refer to this model as being *Vaudenay's model with temporary state disclosure*.

Now, the adversaries are classified into the following classes, according to the access they get to these oracles:

- *Weak adversaries*: they do not have access to the $Corrupt$ oracle;
- *Forward adversaries*: once they access the $Corrupt$ oracle, they can only access the $Corrupt$ oracle;
- *Destructive adversaries*: after the adversary has queried $Corrupt(vtag)$ and obtained the corresponding information, the tag identified by $vtag$ is destroyed (marked as destroyed in $ListTags$) and the temporary identifier $vtag$ wil no longer be available. The database DB will still keep the record associated to this tag (the reader does not know the tag was destroyed). As a consequence, a new tag with the same identifier cannot be created (in this approach, the database cannot store multiple records for the same tag identifier);
- *Strong adversaries*: there are no restrictions on the use of oracles.

Orthogonal to these classes, there is the class of *narrow* adversaries that do not have access to the $Result$ oracle. We may now combine the narrow constraints with any of the previous constraints in order to get another four classes of adversaries, *narrow weak*, *narrow forward*, *narrow destructive*, and *narrow strong*.

Security: Now we are ready to introduce the *tag* and *reader authentication* properties as proposed in [1], [2], simply called the *security* of RFID schemes.

First of all, we say that a tag $\mathcal{T}_{ID}$ and a protocol session π *had a matching conversation* if they exchanged well interleaved and faithfully (but maybe with some time delay) messages according to the protocol, starting with the first protocol message but not necessarily completing the protocol session. If the matching conversation leads to tag authentication, then it will be called a *tag authentication matching conversation*; if it leads to reader authentication, it will be called a *reader authentication matching conversation*.

Now, the tag authentication property is defined by means of the following experiment that a challenger sets up for an adversary $\mathcal{A}$ (after the security parameter λ is fixed):

Experiment $RFID^{t\_auth}_{\mathcal{A},\mathcal{S}}(\lambda)$

1: Set up the reader;
2: $\mathcal{A}$ gets the public key pk;
3: $\mathcal{A}$ queries the oracles;
4: Return 1 if there is a protocol instance π s.t.:
   – π authenticates an uncorrupted legitimate tag $\mathcal{T}_{ID}$;
   – π had no tag authentication matching conversation with $\mathcal{T}_{ID}$.

Otherwise, return 0.

The advantage of $\mathcal{A}$ in the experiment $RFID^{t\_auth}_{\mathcal{A},\mathcal{S}}(\lambda)$ is defined as

$$Adv^{t\_auth}_{\mathcal{A},\mathcal{S}}(\lambda) = Pr(RFID^{t\_auth}_{\mathcal{A},\mathcal{S}}(\lambda) = 1)$$

An RFID scheme $\mathcal{S}$ achieves *tag authentication* if $Adv^{t\_auth}_{\mathcal{A},\mathcal{S}}$ is negligible, for any strong adversary $\mathcal{A}$.

The experiment for reader authentication, denoted $RFID^{r\_auth}_{\mathcal{A},\mathcal{S}}(\lambda)$, is quite similar to that above:

Experiment $RFID^{r\_auth}_{\mathcal{A},\mathcal{S}}(\lambda)$

Set up the reader;
$\mathcal{A}$ gets the public key pk;
$\mathcal{A}$ queries the oracles;
Return 1 if there is a protocol instance π with a tag $\mathcal{T}_{ID}$ s.t.:
- $\mathcal{T}_{ID}$ is an uncorrupted legitimate tag that authenticates the reader;
- π had no reader authentication matching conversation with $\mathcal{T}_{ID}$.

Otherwise, return 0.

The main difference compared to the previous experiment is that the adversary $\mathcal{A}$ tries to make some legitimate tag to authenticate the reader. As π and $\mathcal{T}_{ID}$ have no matching conversation, $\mathcal{A}$ computes at least one message that makes the tag to authenticate the reader.

An RFID scheme $\mathcal{S}$ achieves *reader authentication* if the advantage of $\mathcal{A}$, $Adv^{r\_auth}_{\mathcal{A},\mathcal{S}}$, is negligible, for any strong adversary $\mathcal{A}$ (the advantage of $\mathcal{A}$ is defined as above, by using $RFID^{r\_auth}_{\mathcal{A},\mathcal{S}}(\lambda)$ instead of $RFID^{t\_auth}_{\mathcal{A},\mathcal{S}}(\lambda)$).

Privacy: Privacy for RFID systems [2] captures anonymity and untraceability. It basically means that an adversary cannot learn anything new from intercepting the communication between a tag and the reader. To model this, the concept of a *blinder* was introduced in [2].

A *blinder for an adversary* $\mathcal{A}$ that belongs to some class V of adversaries is a PPT algorithm $\mathcal{B}$ that:

1) simulates the $Launch$, $SendReader$, $SendTag$, and $Result$ oracles for $\mathcal{A}$, without having access to the corresponding secrets;
2) passively looks at the communication between $\mathcal{A}$ and the other oracles allowed to it by the class V (that is, $\mathcal{B}$ gets exactly the same information as $\mathcal{A}$ when querying these oracles).

When the adversary $\mathcal{A}$ interacts with the RFID scheme by means of a blinder $\mathcal{B}$, we say that $\mathcal{A}$ is *blinded by* $\mathcal{B}$ and denote this by $\mathcal{A}^{\mathcal{B}}$. We emphasize that $\mathcal{A}^{\mathcal{B}}$ is allowed to query the oracles $Launch$, $SendReader$, $SendTag$, and $Result$ only by means of $\mathcal{B}$; all the other oracles are queried as a standard adversary.

Given an adversary $\mathcal{A}$ and a blinder $\mathcal{B}$ for it, define the following two experiments (privacy games):

Experiment $RFID^{prv-0}_{\mathcal{A},\mathcal{S}}(\lambda)$

Set up the reader;
$\mathcal{A}$ gets the public key pk;
$\mathcal{A}$ queries the oracles;
$\mathcal{A}$ gets the secret table of the $DrawTag$ oracle;
$\mathcal{A}$ outputs a bit b';
Return b'.

Experiment $RFID^{prv-1}_{\mathcal{A},\mathcal{S},\mathcal{B}}(\lambda)$

Set up the reader;
$\mathcal{A}^{\mathcal{B}}$ gets the public key pk;
$\mathcal{A}^{\mathcal{B}}$ queries the oracles;
$\mathcal{A}^{\mathcal{B}}$ gets the secret table of the $DrawTag$ oracle;
$\mathcal{A}^{\mathcal{B}}$ outputs a bit b';
Return b'.

Now, the *advantage* of $\mathcal{A}$ blinded by $\mathcal{B}$ is defined by

$$Adv^{prv}_{\mathcal{A},\mathcal{S},\mathcal{B}}(\lambda) =\mid P(RFID^{prv-0}_{\mathcal{A},\mathcal{S}}(\lambda)=1) - P(RFID^{prv-1}_{\mathcal{A},\mathcal{S},\mathcal{B}}(\lambda)=1) \mid$$

An RFID scheme achieves privacy for a class V of adversaries if for any adversary $\mathcal{A} \in V$ there exists a blinder $\mathcal{B}$ such that $Adv^{prv}_{\mathcal{A},\mathcal{S},\mathcal{B}}(\lambda)$ is negligible.

We thus obtain eight concepts of privacy: *strong privacy*, *narrow strong privacy*, *destructive privacy*, and so on.

We close this section by a remark on the blinded privacy game. In this game, the oracle $SendTag$ is simulated by the blinder. Therefore, the tags' states are not changed in this game.

IV. Privacy and mutual authentication under temporary state disclosure

When Vaudenay's model was proposed [1], it was somewhat unclear whether the $Corrupt$ oracle returns the full (i.e., permanent and temporary) tag state or only the permanent one. This has also remained unclear in Paisie and Vaudenay's next year paper [2] on mutual authentication. While the distinction between full and permanent state did not have a negative impact on the results already obtained in the case of unilateral authentication, it highlighted several wrong results in the case of mutual authentication [7].

A. Understanding Theorem 1 in [7]

In the very interesting paper [7] a series of impossibility results were established, with respect to privacy and mutual authentication in RFID schemes. One of them, namely Theorem 1, says that there is no RFID scheme that achieves both reader authentication and narrow forward privacy in Vaudenay's model with temporary state disclosure. The argument is as follows. Given a blinder $\mathcal{B}$, one may construct an adversary $\mathcal{A}^{\mathcal{B}}_{sec}$ against reader authentication so that, if the scheme is narrow forward private then $\mathcal{A}^{\mathcal{B}}_{sec}$ has non-negligible advantage to authenticate itself as a valid reader. Going inside the proof, we remark that it is crucial the $Corrupt$ oracle returns the full state of a tag in order to allow an adversary to perform the test by which the tag authenticates the reader. By this test, the adversary distinguishes with non-negligible probability between the real privacy game and the blinded one.

As a conclusion, none of the random oracle (RO) or public-key cryptography (PKC) based RFID schemes in [2] achieves mutual authentication and the privacy level claimed in [2] if Vaudenay's model allows corruption with temporary state disclosure.

In our opinion, the RO based RFID scheme in [1], [2] needs a detailed discussion in order to understand why Theorem 1 applies to this case as well. To define this scheme, two public random oracles F and G running two random functions, one from $\{0,1\}^{k+\ell}$ to $\{0,1\}^k$ and the other one from $\{0,1\}^k$ to $\{0,1\}^k$, are needed. The $SetupT(pk, ID)$ algorithm creates a tag with the identity ID and a (permanent) state consisting of a key $K \leftarrow \{0,1\}^k$. The pair (ID, K) is stored in the reader's database DB. The reader and all tags are granted (secure) access to the oracles F and G. One may also think that copies of these oracles are distributed to the reader and all tags. The interactive protocol $Ident$ is pictorially represented in Figure 2.

Now, we have to clarify what corruption means in the case of this protocol. As F and G are public random oracles, the adversary is granted access to them as well. The $Corrupt$ oracle returns only the tag state but not the internal structure of the oracles F and G (which are thought as black boxes). Therefore, an adversary that corrupts a tag and gets a state $G(K)$ will not be able to "inverse" this value or to do any other computation derived from the internal structure of these oracles, except with negligible probability. This is somewhat opposite to pseudo-random functions whose internal structure is supposed to be known. For instance, if we consider the candidate pseudo-random function $DES = (DES_K)_{K\in\{0,1\}^{64}}$, a key K, and a cyphertext $c = DES_K(x)$, one may efficiently compute the plaintext x.

As a conclusion, an adversary that corrupts a tag and gets its key K may get $F(K, x)$ and $G(K)$ by querying the oracles F and G (but not by corrupting them). Therefore, the tag can perfectly be simulated by an adversary and Theorem 1 in [7] can be applied in this case (in fact, the adversary only needs to know w' in order to do the tag's test in the last step).

	Reader (DB, F, G)		**Tag** (F, G, K)
1	$x \leftarrow \{0,1\}^\ell$	$\xrightarrow{x}$	
2		$\xleftarrow{z}$	$z = F(0, K, x)$, $w' = F(1, K, x)$, $K = G(K)$
3	If $\exists (ID, K) \in DB$ and $0 \leq i < t$ s.t. $z = F(0, G^i(K), x)$ then output ID, $K = G^i(K)$, $K' = K$ else output $\perp$, $K' \leftarrow \{0,1\}^k$ $w = F(1, K', x)$	$\xrightarrow{w}$	
			If $w = w'$ then output OK else output $\perp$

Fig. 2. RO based RFID scheme in [2]

B. RFID design based on PUF tags

The newest technologies allow *PUF tags* that are tags with PUFs inside them. The tags that do not include PUFs will sometimes be referred to as *ordinary tags*. In order to adapt Vaudenay's model (with or without temporary state disclosure) to RFID schemes with PUF tags, we have to clarify what corruption means in this case. Taking into account that PUFs are temper-evident, at least two main scenarios are possible:

1) Any corruption on a PUF tag destroys the tag. By corruption, one gets the (full) state except for the values computed by the PUF (assuming that they were not saved in the tag's memory);
2) The PUF tag is destroyed by corrupting it, but some values returned by its PUFs are obtained (an example in this sense is the *cold boot attack* in [19] according to which the tag may be frozen at some time to obtain the PUF value).

The first scenario is the most used one and it is the one adopted in our paper. As corruption of PUF based tags does not reveal the full tag state, PUF tags cannot generally be simulated by adversaries. Working in this scenario, Theorem 1 in [7], at least in its present form, cannot be applied to RFID schemes with PUF tags. This leaves open the invitation to design RFID schemes that achieve mutual authentication and higher privacy levels than narrow forward in Vaudenay's model with temporary state disclosure. As we have already said, such schemes cannot be based on ordinary tags. A good choice is to use PUF tags, as it was done in [4]–[6], [8], [9]. However, the use of PUF tags does not mean that the schemes are immune to corrupting adversaries. This is because an adversary might not need the entire tag state to attack the scheme. An example in this sense is provided in [4] where it was shown that the RFID schemes proposed in [5], [6] do not achieve mutual authentication and (narrow) destructive privacy in Vaudenay's model with temporary state disclosure, as it was claimed by authors, although they use PUF tags. The proof exploits the fact that these schemes use volatile variables to carry values between protocol steps.

The second scenario was touched by several research papers such as [5], [6], [19]. We are not aware of any formal treatment of this scenario in Vaudenay's model. To implement this scenario in Vaudenay's model, the $Corrupt$ oracle should be changed to return snapshots of the tag's state during its computation (recall that the standard $Corrupt$ oracle returns the tag's state before or after a protocol step). A formal and complete treatment of such a corruption seems hard to reach; on the other side, such a corruption is very strong and probably no PUF based RFID scheme may achieve a privacy level higher than (narrow) weak under such a corruption. However, special cases may be relevant. One of them is the cold boot attack mentioned above. To defeat it, a PUF double evaluation technique was proposed in [19], which consists of two evaluations in a row of the same PUF. If the attack is applied immediately after the first

PUF evaluation, the second PUF evaluation is lost, and vice-versa. This technique was implemented in two RFID schemes [5], [6]. Unfortunately, the authors dis not pay much attention to the volatile variables, which made their schemes not to achieve even the narrow forward privacy level [4].

C. Strong and destructive privacy with PUF tags

Recall that (narrow) strong adversaries may corrupt a tag multiple times. However, working in the first corruption scenario mentioned above (with PUF tags), (narrow) strong adversaries become in fact (narrow) destructive. This is because corruption destroys the PUF tag and, therefore, it cannot be further used. Therefore, Vaudenay's model (with or without temporary state disclosure) for RFID schemes with PUF tags is limited to at most (narrow) destructive privacy.

D. Weak privacy and mutual authentication

The PRF based RFID scheme in [2] achieves mutual authentication and weak privacy in Vaudenay's model with temporary state disclosure. This simply follows from the proof in [2] together with the remark that weak adversaries are not allowed to corrupt tags.

V. Destructive privacy and reader-first authentication

An interesting question that arises when designing mutual authentication RFID schemes is whether the tag or the reader should be authenticated first. We have thus two approaches: *tag-first* and *reader-first authentication*, respectively [3]. The *tag-first authentication* has some advantage with respect to desynchronization: the tag computes its new state and sends information about it to the reader. However, the tag state is updated only when the reader authenticates the tag and confirms the new state to the tag. The disadvantage of this approach is that the tag should provide some information to the reader before it is confident of the reader's identity.

The *reader-first authentication* might enhance the tag privacy because the tag gives private information to the reader when it is confident of its identity. This also might help preventing adversaries from tracking tags. Another advantage is when the tag is designed only for a limited number of authentications. In such a case, the reader-first approach prevents a form of the denial of service attack that would "consume" all the tag's authentication answers.

In this section we address the problem to construct a destructive private and mutual authentication RFID scheme in Vaudenay's model with temporary state disclosure. For mutual authentication we follow the reader-first approach and, according to our discussion in Section IV, all tags are endowed with PUFs.

To describe our scheme, let us assume that λ is a security parameter, $\ell_1(\lambda)$ and $\ell_2(\lambda)$ are two polynomials, and $F = (F_K)_{K\in\mathcal{K}}$ is a pseudo-random function, where $F_K : \{0,1\}^{2\ell_1(\lambda)+2} \to \{0,1\}^{\ell_2(\lambda)}$ for all $K \in \mathcal{K}_\lambda$. Each tag is equipped with a (unique) PUF $P : \{0,1\}^{p(\lambda)} \to \mathcal{K}_\lambda$ and has the capacity to compute F, where $p(\lambda)$ is a polynomial. The internal state of the tag consists of a pair (s, x), where $s \in \{0,1\}^{p(\lambda)}$ is randomly chosen as a seed to evaluate P, and $x \in \{0,1\}^{\ell_1(\lambda)}$ is a random string used as a "dynamic" identifier of the tag. The reader maintains a database DB with entries for all legitimate tags. Each entry is a vector (ID, K, x), where ID is the tag's identity and $K = P(s)$, where P is the tag's PUF and (s, x) is its state.

The mutual authentication protocol is given in Figure 3. As we can see, the tag generates initially a random $u \leftarrow \{0,1\}^{\ell_1(\lambda)}$, computes $K = P(s)$ and $z = F_K(0, 0, u, x)$, and sends (u, z). The reader checks its database for a triple (ID, K, x) such that $z = F_K(0, 0, u, x)$ or $z = F_K(0, 0, u, x + 1)$. The reason is that at most one step desynchronization may occur between reader and tag; that is, when x is on tag, either x or $x - 1$ is on reader. When the reader finds out the right value, it re-synchronizes with the tag and prepares the answer (v, w). The tag evaluates the PUF, checks the value w received from reader, and takes a decision. It also updates x correspondingly and prepares the answer for the reader. On receiving the tag's answer, the reader checks it, takes a decision, and updates x correspondingly.

	Reader (DB, F)		**Tag** (P, s, F, x)
1			$u \leftarrow \{0,1\}^{\ell_1(\lambda)}$, $K = P(s)$
		$\xleftarrow{u, z}$	$z = F_K(0, 0, u, x)$
			erase K, u, z
2	If $\exists (ID, K, x) \in DB$ and $i \in \{0,1\}$		
	s.t. $z = F_K(0, 0, u, x + i)$		
	then $v \leftarrow \{0,1\}^{\ell_1(\lambda)}$, $x = x + i$		
	$w = F_K(0, 1, v, x)$		
	else $v \leftarrow \{0,1\}^{\ell_1(\lambda)}$, $w \leftarrow \{0,1\}^{\ell_2(\lambda)}$	$\xrightarrow{v, w}$	
3			$K = P(s)$, $w' = F_K(0, 1, v, x)$
			If $w = w'$
			then $x = x + 1$, $w' = F_K(1, 1, v, x)$
		$\xleftarrow{w'}$	else $w' \leftarrow \{0,1\}^{\ell_2(\lambda)}$
			erase K, v, w, w'
4	If $w' = F_K(1, 1, v, x + 1)$		
	then output ID, $x = x + 1$		
	else output $\perp$		

Fig. 3. Destructive private and reader-first authentication PUF based RFID scheme in Vaudenay's model with temporary state disclosure

Remark that if the reader does not update x (because it rejects the tag), then it will do so in step 2 of the next protocol session (with the same tag). Therefore, the desynchronization between reader and tag is at most one step.

Theorem 5.1: The RFID scheme in Figure 3 is correct.

Proof: Assuming that a tag $\mathcal{T}_{ID}$ is legitimate, the reader's database contains an entry (ID, K, x), where $K = P(s)$, (s, x) is the tag's state, and P is its PUF.

When the reader receives (u, z) from the tag $\mathcal{T}_{ID}$, exactly one of the two equalities $z = F_K(0, 0, u, x)$ and $z = F_K(0, 0, u, x + 1)$ holds with overwhelming probability (we use the notation in Figure 3).

If the reader has found the tag in its database (i.e., identified it), the equality $w = w'$ on tag's side holds with overwhelming probability. This means that the tag authenticates the reader. In such a case, the equality $w' = F_K(1, 1, v, x+1)$ holds with overwhelming probability, meaning that the reader authenticates the tag.

As a final remark, if the tag does not authenticate the reader, then the reader will authenticate the tag with negligible probability. ■

We will focus now on the security of our RFID scheme.

Theorem 5.2: The RFID scheme in Figure 3 achieves tag authentication in Vaudenay's model with temporary state disclosure, provided that F is a PRF and the tags are endowed with ideal PUFs.

Proof: Assume that the scheme does not achieve tag authentication, and let $\mathcal{A}$ be an adversary that has non-negligible advantage over the scheme, with respect to the tag authentication property. We will show that there exists a PPT algorithm $\mathcal{A}'$ that can break the pseudo-randomness property of the function F.

The main idea is the next one. Let $\mathcal{C}$ be a challenger for the pseudo-randomness security game of the function F. The adversary $\mathcal{A}'$ will play the role of challenger for $\mathcal{A}$. Thus, $\mathcal{A}'$ guesses the tag identity ID^* that $\mathcal{A}$ can authenticate with the reader with non-negligible probability (recall that there is a polynomial number $t(\lambda)$ of tags). Then, it creates the tag $\mathcal{T}_{ID^*}$ with the help of $\mathcal{C}$. Namely, the random key chosen

by $\mathcal{C}$ will be thought as the key generated by the tag's PUF. The adversary $\mathcal{A}'$ does not know this key but, in fact, it does not need to. As $\mathcal{A}'$ impersonates the reader, it can provide $\mathcal{A}$ with correct answers by querying $\mathcal{C}$. Therefore, $\mathcal{T}_{ID^*}$ will be regarded by $\mathcal{A}$ as a legitimate tag.

When $\mathcal{A}$ succeeds to authenticate $\mathcal{T}_{ID^*}$ to the reader with non-negligible probability, $\mathcal{A}'$ will use the information obtained from $\mathcal{A}$ to answer correctly, with overwhelming probability, some challenge of $\mathcal{C}$.

The details on $\mathcal{A}'$ are as follows (assuming a given security parameter λ):

1) The challenger $\mathcal{C}$ chooses uniformly at random a key for F and will answer all queries of $\mathcal{A}'$ with respect to this key;
2) $\mathcal{A}'$ plays the role of challenger for $\mathcal{A}$. It will run the reader and all tags created by $\mathcal{A}$, answering all $\mathcal{A}$'s oracle queries. Therefore, using $SetupR(\lambda)$ it generates a triple (pk, sk, DB), gives the public key pk to $\mathcal{A}$, and keeps the private key sk.
   $\mathcal{A}'$ will maintain a list of tag entries $\mathcal{A}'_{ListTags}$ similar to $ListTags$ (see Section III) but with the difference that each entry in this list also includes the current state of the tag as well as a special field designated to store the "key generated by the tag's internal PUF". The legitimate entries in this list define the reader's database DB. Initially, $\mathcal{A}'_{ListTags}$ is empty;
3) $\mathcal{A}'$ guesses the tag identity ID^* that $\mathcal{A}$ will authenticate to reader (recall that the number of tag identities is polynomial in the security parameter);
4) $\mathcal{A}'$ will simulate for $\mathcal{A}$ all the corresponding oracles in a straightforward manner, but with the following modifications:
   a) $CreateTag^b(ID)$: If $\mathcal{T}_{ID}$ was already created, then $\mathcal{A}'$ does nothing.
      If $\mathcal{T}_{ID}$ was not created and $ID \neq ID^*$, then $\mathcal{A}'$ randomly chooses $K \in \{0,1\}^\lambda$ and $x \in \{0,1\}^{\ell_1(\lambda)}$ and records a corresponding entry into $\mathcal{A}'_{ListTags}$ (K plays the role of the key generated by the tag's internal PUF). Thus, $\mathcal{T}_{ID}$ has just been created.
      If $\mathcal{T}_{ID}$ was not created and $ID = ID^*$, then $\mathcal{A}'$ records $(ID^*, ?, x)$ into $\mathcal{A}'_{ListTags}$, where $x \leftarrow \{0,1\}^{\ell_1(\lambda)}$. The meaning of "?" is that this field should have contained the key chosen by $\mathcal{C}$, which is unknown to $\mathcal{A}'$. However, $\mathcal{A}'$ does not need to know this key because it can answer all $\mathcal{A}$'s queries regarding ID^* with the help of $\mathcal{C}$.
      As the tags are endowed with ideal PUFs and the keys are uniformly at random chosen by $\mathcal{A}'$, including the key chosen by $\mathcal{C}$, $\mathcal{A}'$ implements correctly the functionality of all tags (including $\mathcal{T}_{ID^*}$);
   b) $DrawTag$ and $Free$: $\mathcal{A}'$ knows the list of all tags created by $\mathcal{A}$, and updates it correspondingly whenever $\mathcal{A}$ draws or frees some tag;
   c) $Launch()$: $\mathcal{A}'$ launches a new protocol instance whenever $\mathcal{A}$ asks for it;
   d) $SendTag(\emptyset, vtag)$: This is the first message $vtag$ sends in a protocol instance. If the tag referred by $vtag$ is ID^*, then $\mathcal{A}'$ will randomly generate $u \in \{0,1\}^{\ell_1(\lambda)}$ and query $\mathcal{C}$ for $(0, 0, u, x)$. If z is $\mathcal{C}$'s response, than $\mathcal{A}'$ answers with (u, z).
      If $vtag$ refers to some $ID \neq ID^*$, then $\mathcal{A}'$ can prepare by itself the answer because it knows the corresponding key for ID;
   e) $SendReader((u, z), \pi)$: Assume the reader (run by $\mathcal{A}'$) has received (u, z) in the protocol instance π from a tag identified by $vtag$ (in other words, $(u, z) \leftarrow SendTag(\emptyset, vtag)$).
      If $vtag$ refers to some tag ID such that $(ID, K, x) \in DB$ for some K, then the reader (run by $\mathcal{A}'$) can compute the answer according to the protocol.
      If $vtag$ refers to ID^*, then the reader (run by $\mathcal{A}'$) can compute the answer according to the protocol by queering $\mathcal{C}$ (recall that $\mathcal{T}_{ID^*}$ is regarded by $\mathcal{A}$ as a legitimate tag).
      If $vtag$ refers to some ID for which no entry can be found in DB, then the answer (v, w) is randomly chosen;
   f) $SendTag((v, w), vtag)$: If the tag referred by $vtag$ is ID^*, then $\mathcal{A}'$ queries $\mathcal{C}$ for $(0, 1, v, x)$ and then compares the answer with w. If they match, the tag outputs OK; otherwise, it outputs $\perp$. In the first case $\mathcal{A}'$ increments x and queries $\mathcal{C}$ for $(1, 1, v, x)$ to get w'; in the second case,

it chooses at random w'. If $vtag$ refers to some $ID \neq ID^*$ that has associated a pair (K, x), then $\mathcal{A}'$ can compute by itself w' (according to the protocol).
In all cases, the oracle returns w';

g) $Result(\pi)$: $\mathcal{A}'$ can infer the decision of the reader in the last step of π because it can obtain the value $F_K(1, 1, v, x + 1)$ for all tags (either it can compute it or query $\mathcal{C}$ for it). Therefore, $\mathcal{A}'$ can simulate $Result(\pi)$ according to its definition;

h) $Corupt(vtag)$: If the tag referred by $vtag$ is different from ID^*, then $\mathcal{A}'$ returns its current state; otherwise, it aborts;

5) If $\mathcal{A}$ is able to make the reader to authenticate the tag ID^*, then this means that $\mathcal{A}$ can compute $w' = F_{K^*}(1, 1, v, x + 1)$ without knowing K^*, provided that K^* is the key chosen by $\mathcal{C}$ and x is the current third component of the entry $(ID^*, ?, x)$. Then, $\mathcal{A}'$ can prepare the challenge phase for $\mathcal{C}$ as follows:
   a) $\mathcal{A}'$ sends $(1, 1, v, x + 1)$ to $\mathcal{C}$;
   b) $\mathcal{C}$ randomly chooses $b \in \{0, 1\}$; if $b = 1$, then $\mathcal{C}$ returns $w'' = F_{K^*}(1, 1, v, x+1)$, else $\mathcal{C}$ returns a random w'';
   c) $\mathcal{A}'$ prepares its guess b' as follows: if $w' = w''$, then $b' = 1$, else $b' = 0$.

The probability that $\mathcal{A}'$ guesses the bit chosen by $\mathcal{C}$ can be computed as the product between the probability that $\mathcal{A}'$ guesses ID^* and the probability that $\mathcal{A}$ makes the reader to authenticate the tag ID^*.

The probability that $\mathcal{A}'$ guesses ID^* is $1/t(\lambda)$, where $t(\lambda)$ is the polynomial number of tag identities. If we assume now that $\mathcal{A}$ has non-negligible probability to make the reader authenticate the tag ID^*, then $\mathcal{A}'$ can successfully answer $\mathcal{C}$'s challenge with non-negligible probability; this contradicts the fact that F is a pseudo-random function. ■

As with respect to the reader authentication property, we have the following result.

Theorem 5.3: The RFID scheme in Figure 3 achieves reader authentication in Vaudenay's model with temporary state disclosure, provided that F is a PRF and the tags are endowed with ideal PUFs.

Proof: Assume that our scheme does not achieve reader authentication, and let $\mathcal{A}$ be an adversary that has non-negligible advantage over the scheme, with respect to the reader authentication property. We will show that there exists a PPT algorithm $\mathcal{A}'$ that can break the pseudo-randomness property of the function F.

The main idea is somewhat similar to the one in the Theorem 5.2. Let $\mathcal{C}$ be a challenger for the pseudo-randomness property of the function F. The adversary $\mathcal{A}'$ will play the role of a challenger for $\mathcal{A}$. First, $\mathcal{A}'$ guesses the tag identity ID^* that authenticates $\mathcal{A}$ as a valid reader, with non-negligible probability (recall that there is a polynomial number $t(\lambda)$ of tags). Then, it creates the tag $\mathcal{T}_{ID^*}$ with the help of $\mathcal{C}$, exactly as in the proof of Theorem 5.2. This tag will be regarded by $\mathcal{A}$ as a legitimate one. When $\mathcal{A}$ succeeds in making $\mathcal{T}_{ID^*}$ to authenticate it as a valid reader, $\mathcal{A}'$ will use the message sent by $\mathcal{A}$ to answer some challenge of $\mathcal{C}$.

The description of $\mathcal{A}'$ is very similar to the one in the proof of Theorem 5.2, so we will focus on the differences between them (λ denotes a security parameter):

1) The challenger $\mathcal{C}$ chooses uniformly at random a key for F and will answer all queries of $\mathcal{A}'$ with respect to this key;
2) $\mathcal{A}'$ plays the role of challenger for $\mathcal{A}$. It will run the reader and all tags created by $\mathcal{A}$, answering all $\mathcal{A}$'s oracle queries. Therefore, using $SetupR(\lambda)$ it generates a triple (pk, sk, DB), gives the public key pk to $\mathcal{A}$, and keeps the private key sk.
   $\mathcal{A}'$ will maintain a list of tag entries $\mathcal{A}'_{ListTags}$ exactly as in the proof of Theorem 5.2;
3) $\mathcal{A}'$ guesses the tag identity ID^* that authenticates $\mathcal{A}$ as a valid reader;
4) $\mathcal{A}'$ will simulate for $\mathcal{A}$ all the corresponding oracles exactly as in the proof of Theorem 5.2;
5) If $\mathcal{A}$ is able to make $\mathcal{T}_{ID^*}$ to authenticate it as a valid reader, then this means that $\mathcal{A}$ can compute $w = F_{K^*}(0, 1, v, x)$ without knowing K^* (provided that K^* is the key chosen by $\mathcal{C}$), where x is the current identifier of $\mathcal{T}_{ID^*}$. Then, $\mathcal{A}'$ can prepare the challenge phase for $\mathcal{C}$ as follows:

a) $\mathcal{A}'$ sends $(0, 1, v, x)$ to $\mathcal{C}$;
b) $\mathcal{C}$ randomly chooses $b \in \{0, 1\}$; if $b = 1$, then $\mathcal{C}$ returns $w' = F_{K^*}(0, 1, v, x)$, else $\mathcal{C}$ returns a random w';
c) $\mathcal{A}'$ prepares its guess b' as follows: if $w = w'$, then $b' = 1$, else $b' = 0$.

The probability that $\mathcal{A}'$ guesses the bit chosen by $\mathcal{C}$ is non-negligible if $\mathcal{A}$ has a non-negligible probability to make $\mathcal{T}_{ID^*}$ to authenticate it as a valid reader (this is similar to the proof of Theorem 5.2). Therefore, the assumption that $\mathcal{A}$ has a non-negligible probability to make $\mathcal{T}_{ID^*}$ to authenticate it as a valid reader contradicts the pseudo-randomness of the function F. ■

By using the *sequence-of-games* approach [20] we will prove that our protocol reaches destructive privacy. With this approach, a sequence of games (probabilistic experiments) is defined. The initial game is the original privacy game with respect to a given adversary. The transition from one game G_i to another one G_{i+1} is done by indistinguishability in our case. This means that a probability distribution in G_i is replaced by another one that is indistinguishable from the previous one. In this way, the difference between the probabilities the adversary wins G_i and G_{i+1}, is negligible.

Theorem 5.4: The RFID scheme in Figure 3 achieves destructive privacy in Vaudenay's model with temporary state disclosure, provided that F is a PRF and the tags are endowed with ideal PUFs.

Proof: Let $\mathcal{A}$ be a destructive adversary against our RFID scheme denoted $\mathcal{S}$. We will show that there is a blinder $\mathcal{B}$ such that $Adv^{prv}_{\mathcal{A},\mathcal{S},\mathcal{B}}(\lambda)$ is negligible. The blinder $\mathcal{B}$ that we construct, which has to answer to the oracles *Launch*, *SendReader*, *SendTag*, and *Result* without knowing any secret information, works as follows:

- $Launch()$: returns a unique identifier π for a new protocol instance;
- $SendTag(\emptyset, vtag)$: returns (u, z), where $u \leftarrow \{0,1\}^{\ell_1(\lambda)}$ and $z \leftarrow \{0,1\}^{\ell_2(\lambda)}$;
- $SendReader((u, z), \pi)$: returns (v, w), where $v \leftarrow \{0,1\}^{\ell_1(\lambda)}$ and $w \leftarrow \{0,1\}^{\ell_2(\lambda)}$;
- $SendTag((v, w), vtag)$: returns $w' \leftarrow \{0,1\}^{\ell_2(\lambda)}$;
- $SendReader(w', \pi)$: the blinder does not do anything because, in this case, the reader does not answer;
- $Result(\pi)$: if the session π does not exist or exists but is not completed, the blinder outputs $\perp$. If π has been issued by the $Launch()$ oracle and a protocol transcript $tr_\pi = ((u, z), (v, w), w')$ has been generated by
  - $(u, z) \leftarrow SendTag(\emptyset, vtag)$,
  - $(v, w) \leftarrow SendReader((u, z), \pi)$,
  - $w' \leftarrow SendTag((v, w), vtag)$, and
  - $SendReader(w', \pi)$,

  where $vtag$ refers to some legitimate tag, the blinder outputs 1; otherwise, outputs 0 (remark that the blinder sees what $\mathcal{A}$ sees and, therefore, it knows whether $vtag$ refers to some legitimate tag or not).

We further prove that $Adv^{prv}_{\mathcal{A},\mathcal{S},\mathcal{B}}(\lambda)$ is negligible. To this we define a sequence of games $G_0, \ldots, G_7$, where G_0 is the experiment $RFID^{prv-0}_{\mathcal{A},\mathcal{S}}$ and G_{i+1} is obtained from G_i as described below, for all $0 \leq i < 7$. By $P(G_i)$ we denote the probability the adversary $\mathcal{A}$ wins the game G_i.

Game G_1: This is identical to G_0 except that the game challenger will not use the PRF keys generated by PUFs to answer the adversary's oracle queries, but randomly generated keys, one for each tag created by adversary. Of course, the game challenger must maintain a secret table with the association between each tag and this new secret key. From the adversary's point of view, this means that the probability distribution given by each tag's PUF (in G_0) is replaced by the uniform probability distribution (in G_1). As the PUFs are ideal, the two distributions are indistinguishable. Taking into account that there are a polynomial number of tags, it must be the case that $|P(G_0) - P(G_1)|$ is negligible. We will provide below a proof sketch of this.

Assume $\mathcal{A}$ is an adversary that can distinguish between G_0 and G_1 with non-negligible probability. Define then a new adversary $\mathcal{A}'$ that can break the PUF security with non-negligible probability. In order

to interact with the RFID system, the adversary $\mathcal{A}$ must create some tags. As the tags' PUFs, as well as their seeds, are independently at random chosen, we may assume, without loss of generality, that $\mathcal{A}$ creates exactly one tag with some identity ID, interacts with it, and draws the final conclusion based on this interaction.

Now, the proof goes in a way somewhat similar to the proof of Theorem 5.2. Assume that $\mathcal{C}$ is a challenger for some PUF P. $\mathcal{A}'$ will play the role of challenger for $\mathcal{A}$. When $\mathcal{A}$ queries $CreateTag$ to create the tag $\mathcal{T}_{ID}$, legitimate or not, $\mathcal{A}'$ chooses at random a state (s, x) for this tag and sends s to the challenger $\mathcal{C}$. The challenger chooses at random a bit $b \leftarrow \{0,1\}$ and answers with $K = P(s)$, if $b = 0$, or $K \leftarrow \{0,1\}^\lambda$, if $b = 1$. The adversary $\mathcal{A}'$ will then use K to create the tag $\mathcal{T}_{ID}$. It will also answer $\mathcal{A}$'s all other oracle queries (similar to the proof of Theorem 5.2).

Remark that $\mathcal{A}$ will play the game G_b, without knowing b. After some time, $\mathcal{A}$ will output a guess $b' \in \{0,1\}$ about the game it thinks it is playing. Then, $\mathcal{A}'$ can make a decision about the key K: it was computed as $P(s)$, if $b' = 0$, or it is randomly chosen, if $b' = 1$. Clearly, the probability the adversary $\mathcal{A}'$ wins the PUF security game is the probability that $\mathcal{A}$ distinguishes between the two worlds, G_0 and G_1. If this is non-negligible, then $\mathcal{A}'$ has non-negligible probability to break the PUF.

Game G_2: We replace in G_1 the oracle $Result$ by the oracle $Result_{\mathcal{B}}$ which is the simulation of $Result$ by the blinder $\mathcal{B}$ (please see above the definition $\mathcal{B}$). Denote by G_2 the game such obtained. We prove that $P(G_1) = P(G_2)$.

Recall first that in game G_1 the tags are still endowed with PUFs, but their secret PRF keys are not computed by PUFs. They are randomly generated by the game challenger that maintains a secret table with the key associated to each tag. In this way, the $Corrupt$ oracle will never reveal the secret key, but it destroys the tag when queried.

If $\mathcal{A}$ queries the oracle $Result$ or $Result_{\mathcal{B}}$ for a protocol session that does not exist or is incomplete, both oracles return $\perp$. Therefore, let us assume that these oracles are queried on a complete protocol session π. In this case we will show that $Result(\pi) = 1$ if and only if $Result_{\mathcal{B}}(\pi) = 1$.

Assume $Result(\pi) = 1$. Then, there is a transcript $tr_\pi = ((u, z), (v, w), w')$ defined by a sequence of oracle queries $(u, z) \leftarrow SendTag(\emptyset, vtag)$, $(v, w) \leftarrow SendReader((u, z), \pi)$, $w' \leftarrow SendTag((v, w), vtag)$, and $SendReader(w', \pi)$ such that $vtag$ refers to some tag $\mathcal{T}_{ID}$ whose state is (s, x) and secret key is K, $u \leftarrow \{0,1\}^{\ell_1(\lambda)}$, $z = F_K(0, 0, u, x)$, and (ID, K, x) is in the reader's database (that is, $\mathcal{T}_{ID}$ is legitimate). All these facts show that $Result_{\mathcal{B}}(\pi) = 1$ (recall that the blinder $\mathcal{B}$ sees what $\mathcal{A}$ sees and, therefore, it knows whether $vtag$ refers to some legitimate tag or not).

The inverse implication is a bit more elaborate. Assume that $Result_{\mathcal{B}}(\pi) = 1$. This means that there is a transcript $tr_\pi = ((u, z), (v, w), w')$ defined by a sequence of oracle queries as those above and the tag $\mathcal{T}_{ID}$ referred by $vtag$ is legitimate. Assume that the tag's key is K and its state is (s, x), and in DB there is a record (ID, K, x'). According to the description of the protocol, x' is either $x - 1$ or x. Because the oracles $SendReader$ and $SendTag$ are the real ones (and not simulated by blinder), the reader finds $i \in \{0,1\}$ such that $z = F_K(0, 0, u, x-i)$. Therefore, w must be of the form $F_K(0, 1, v, x-i)$, and this value will match $F_K(0, 1, v, x)$ computed by tag. Therefore, the tag authenticates the reader and replies by $w' = F_K(1, 1, v, x + 1)$. But then, the reader will successfully check the equality between w and $F_K(1, 1, v, x - i + 1)$ (computed by itself) and, therefore, authenticates the tag. As a conclusion, $Result(\pi) = 1$.

This shows that $P(G_1) = P(G_2)$.

Game G_3: This game is identical to G_2 except that the $Launch()$ oracle is simulated according to the blinder description. No difference is encountered between the two games and, therefore, $P(G_2) = P(G_3)$.

Game G_4: This game is identical to G_3 except that the $SendTag(\emptyset, vtag)$ oracle is simulated according to the blinder description. By doing this, the probability distribution

$$\{(u, z) \mid u \leftarrow \{0,1\}^{\ell_1(\lambda)},\ z = F_K(0, 0, u, x)\}$$

is replaced by

$$\{(u, z) \mid u \leftarrow \{0,1\}^{\ell_1(\lambda)},\ z \leftarrow \{0,1\}^{\ell_2(\lambda)}\}.$$

As F is a PRF, $|P(G_3) - P(G_4)|$ is negligible. The proof is quite straightforward. The main idea is as follows. Assume that an adversary $\mathcal{A}$ can distinguish with non-negligible probability between G_3 and G_4. Define an adversary $\mathcal{A}'$ for PRF that uses $\mathcal{A}$ as a subroutine and sends $(0, 0, u, x)$ as a challenge. When the PRF challenger returns, with equal probability, either $z = F_K(0, 0, u, x)$ or $z \leftarrow \{0,1\}^{\ell_2(\lambda)}$, $\mathcal{A}'$ sends this value to $\mathcal{A}$. The probability $\mathcal{A}'$ guesses between the two possibilities for z is exactly the probability $\mathcal{A}$ distinguishes between the two games.

Game G_5: This game is identical to G_4 except that the $SendReader((u, z), \pi)$ oracle is simulated according to the blinder description. That is, for each tag $\mathcal{T}_{ID}$ whose secret key is K and current state is (s, x), one of the probability distributions

$$\{(u, z, v, w) \mid u, v \leftarrow \{0,1\}^{\ell_1(\lambda)},\ z \leftarrow \{0,1\}^{\ell_2(\lambda)},\ w = F_K(0, 0, v, x + i)\},$$

where $i \in \{0, 1\}$, or

$$\{(u, z, v, w) \mid u, v \leftarrow \{0,1\}^{\ell_1(\lambda)},\ z, w \leftarrow \{0,1\}^{\ell_2(\lambda)}\}$$

is replaced by

$$\{(u, z, v, w) \mid u, v \leftarrow \{0,1\}^{\ell_1(\lambda)},\ z, w \leftarrow \{0,1\}^{\ell_2(\lambda)}\}.$$

As F is a PRF and the key K was chosen at random, it must be the case that $|P(G_4) - P(G_5)|$ is negligible. The proof is by contradiction and it is quite similar to the proof that establishes the transition from G_3 to G_4.

Game G_6: This game is identical to G_5 except that the $SendTag((v, w), vtag)$ oracle is simulated by blinder. That is, for each tag $\mathcal{T}_{ID}$, the probability distribution

$$\{(u, z, v, w, w') \mid u, v \leftarrow \{0,1\}^{\ell_1(\lambda)},\ z, w \leftarrow \{0,1\}^{\ell_2(\lambda)},\ w' = F_K(1, 1, v, x)\},$$

is replaced by

$$\{(u, z, v, w, w') \mid u, v \leftarrow \{0,1\}^{\ell_1(\lambda)},\ z, w, w' \leftarrow \{0,1\}^{\ell_2(\lambda)}\}.$$

As F is a PRF and the key K was chosen at random, it must be the case that $|P(G_5) - P(G_6)|$ is negligible. The proof is by contradiction and it is quite similar to the proof in Game G_5. Therefore, it is omitted.

Game G_7: This game is identical to G_6 except that the $SendReader(w', \pi)$ oracle is simulated by blinder. However, this does not change the probability distribution from G_6. Therefore, $P(G_6) = P(G_7)$.

Now, we show that G_7 is in fact $RFID^{prv-1}_{\mathcal{A},\mathcal{S},\mathcal{B}}$. The blinded adversary $\mathcal{A}^{\mathcal{B}}$ sees each tag as a standard PUF tag, although random secret keys are used instead of the keys generated by PUFs. The oracles $CreateTag$, $Draw$, $Free$, and $Corrupt$ that can be queried directly by $\mathcal{A}$ do not use the keys generated by PUFs in order to answer the adversary's queries (in fact, they do not use any secret key). The answer to the other oracles is simulated by blinder which does not use the secret keys either. Therefore, G_7 is indeed $RFID^{prv-1}_{\mathcal{A},\mathcal{S},\mathcal{B}}$.

Now, to derive the final conclusion of the proof we remark that $P_{\mathcal{A}}(G_0) = P(RFID^{prv-0}_{\mathcal{A},\mathcal{S}}(\lambda) = 1)$ and $P_{\mathcal{A}}(G_7) = P(RFID^{prv-1}_{\mathcal{A},\mathcal{S},\mathcal{B}}(\lambda) = 1)$. Combining all the probabilities $P(G_i)$ together, we obtain that $Adv^{prv}_{\mathcal{A},\mathcal{S},\mathcal{B}}(\lambda)$ is negligible and, therefore, our protocol achieves destructive privacy. ■

VI. Narrow destructive privacy and reader-first authentication

With little effort, the RFID scheme in Figure 3 can be simplified to a narrow destructive private and reader-first authentication RFID scheme in Vaudenay's model with temporary state disclosure. The mutual authentication protocol of this new RFID scheme is presented in Figure 4; all the other elements are as in Section V, except that F_K is a function from $\{0,1\}^{\ell_1(\lambda)+2}$ to $\{0,1\}^{\ell_2(\lambda)}$ and t is polynomial in the security parameter. As one can see, there is no random generator on tag. Because of this, the synchronization between tag and reader can be lost. The only thing we can do is to check (on the reader side) for a polynomial bounded desynchronization. Due to this, the scheme can be at most narrow destructive

private: if an adversary desynchronizes the tag and reader sufficiently enough (for more than t steps), then it will be able to distinguish the real privacy game from the blinded one by means of the $Result$ oracle. Roughly speaking, this is because in the real privacy game the $Resut$ oracle returns 0 (when the tag and reader are desynchronized for more than t steps), while in the blinded privacy game it returns 1.

	Reader (DB, F)		**Tag** (P, s, F, x)
1			$K = P(s)$
		$\xleftarrow{z}$	$z = F_K(0, 0, x)$
			erase K, z
			$x = x + 1$
2	If $\exists (ID, K, x) \in DB$ and $0 \leq i < t$		
	s.t. $z = F_K(0, 0, x + i)$		
	then $x = x + i$, $w = F_K(0, 1, x + 1)$		
	else $w \leftarrow \{0, 1\}^{\ell_2(\lambda)}$	$\xrightarrow{w}$	
3			$K = P(s)$
			If $w \neq F_K(0, 1, x)$
			then $w' = F_K(1, 1, x)$
		$\xleftarrow{w'}$	else $w' \leftarrow \{0, 1\}^{\ell_2(\lambda)}$
			erase K, w, w'
	If $w' = F_K(1, 1, x + 1)$		
	then output ID, $x = x + 1$		
	else output $\perp$		

Fig. 4. Narrow destructive private and reader-first authentication PUF based RFID scheme in Vaudenay's model with temporary state disclosure

We therefore have the following result.

Theorem 6.1: The RFID scheme in Figure 4 achieves mutual authentication and narrow destructive privacy in Vaudenay's model with temporary state disclosure, provided that F is a PRF and the tags are endowed with ideal PUFs.

Proof: It is straightforward to see that the proof follows a similar line to the proofs of Theorems 5.2 and 5.3 for mutual authentication, and Theorem 5.4 for narrow destructive privacy. Remark that for privacy, the $Result$ oracle is not used. ■

It is good to remark that our RFID scheme in Figure 4 also provides an appropriate practical solution to the narrow destructive privacy in the plain Vaudenay's model, where the existing solution is based on random oracles (please see Section IV-A).

A few more words on desynchronization are in order. If we look to the protocol in Figure 4 we remark that the desynchronization is a result of the fact that the tag and reader share a common variable x that is updated by tag before authenticating the reader. This allows an adversary to query a tag for more than t times and, therefore, to desynchronize the tag and the reader.

To prevent desynchronization between reader and tag in reader-first authentication RFID schemes, the tag should update the shared permanent variables after authenticating the reader, and not before.

VII. Conclusions

Modern applications of RFID systems ask for advanced security and privacy properties. For instance, tag destruction under corruption is an important requirement when the tag is used for access control.

Likewise, the disclosure of temporary state under tag corruption is a serious threat in practice. Reader-first authentication [3] assures that the tag will give its private data only when it authenticates the reader. Therefore, tag tracking and data theft is prevented when the reader is fake. All these together mean that we need RFID schemes that provide destructive privacy and reader-first authentication under corruption with temporary state disclosure.

As far as we are concerned, no RFID scheme developed so far meets these requirements in Vaudenay's model with temporary state disclosure. The aim of this paper is to propose two RFID schemes that fill this gap. The first one is destructive private and the second one is narrow destructive private. Both of them assure reader-first authentication, are practical, and efficient. The first one may be desynchronized for at most one step. The second scheme avoids random generators on tags. As (narrow) destructive privacy cannot be achieved with ordinary tags, we have used PUFs as secure hardware containers for the secret key of tags. Detailed security and privacy proofs are provided for our schemes.

REFERENCES

[1] S. Vaudenay, "On privacy models for RFID," in *Proceedings of the Advances in Crypotology 13th International Conference on Theory and Application of Cryptology and Information Security*, ser. ASIACRYPT'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 68–87.

[2] R.-I. Paise and S. Vaudenay, "Mutual authentication in RFID: Security and privacy," in *Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS '08. New York, NY, USA: ACM, 2008, pp. 292–299.

[3] R. Peeters, J. Hermans, and J. Fan, "IBIHOP: Proper privacy preserving mutual RFID authentication," in *The 2013 Workshop on Radio Frequency Identification/Internet of Things Security (RFIDsec'13 Asia)*, ser. Cryptology and Information Security Series, C. Ma and J. Weng, Eds., vol. 11. IOS Press, 2013.

[4] C. Hristea and F. L. Ţiplea, "Destructive privacy and mutual authentication in Vaudenay's RFID model," Cryptology ePrint Archive, Report 2019/073, 2019, https://eprint.iacr.org/2019/073.

[5] S. Kardaş, S. Çelik, M. Yildiz, and A. Levi, "PUF-enhanced offline RFID security and privacy," *J. Netw. Comput. Appl.*, vol. 35, no. 6, pp. 2059–2067, Nov. 2012.

[6] M. Akgün and M. U. Çaglayan, "Providing destructive privacy and scalability in RFID systems using PUFs," *Ad Hoc Netw.*, vol. 32, no. C, pp. 32–42, Sep. 2015.

[7] F. Armknecht, A.-R. Sadeghi, A. Scafuro, I. Visconti, and C. Wachsmann, in *Transactions on Computational Science XI*, M. L. Gavrilova, C. J. K. Tan, and E. D. Moreno, Eds. Berlin, Heidelberg: Springer-Verlag, 2010, ch. Impossibility Results for RFID Privacy Notions, pp. 39–63.

[8] A.-R. Sadeghi, I. Visconti, and C. Wachsmann, "PUF-enhanced RFID security and privacy," in *Workshop on secure component and system identification (SECSI)*, vol. 110, 2010.

[9] ——, *Enhancing RFID Security and Privacy by Physically Unclonable Functions*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 281–305.

[10] M. Sipser, *Introduction to the Theory of Computation*. Cengage Learning, 2012.

[11] K. Finkenzeller, *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification*, 3rd ed. Wiley Publishing, 2010.

[12] Y. Li, H. R. Deng, and E. Bertino, *RFID Security and Privacy*, ser. Synthesis Lectures on Information Security, Privacy, and Trust. Morgan & Claypool Publishers, 2013.

[13] A. Juels and S. A. Weis, "Defining strong privacy for RFID," *ACM Trans. Inf. Syst. Secur.*, vol. 13, no. 1, pp. 7:1–7:23, Nov. 2009.

[14] S. Canard, I. Coisel, J. Etrog, and M. Girault, "Privacy-preserving RFID systems: Model and constructions," https://eprint.iacr.org/-2010/405.pdf, 2010.

[15] R. H. Deng, Y. Li, M. Yung, and Y. Zhao, "A new framework for RFID privacy," in *Proceedings of the 15th European Conference on Research in Computer Security*, ser. ESORICS'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 1–18.

[16] J.-M. Bohli and A. Pashalidis, "Relations among privacy notions," *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 1, pp. 4:1–4:24, Jun. 2011.

[17] J. Hermans, F. Pashalidis, Andreasand Vercauteren, and B. Preneel, "A new RFID privacy model," in *Computer Security – ESORICS 2011*, V. Atluri and C. Diaz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 568–587.

[18] J. Hermans, R. Peeters, and B. Preneel, "Proper RFID privacy: Model and protocols," *IEEE Transactions on Mobile Computing*, vol. 13, no. 12, pp. 2888–2902, Dec 2014.

[19] S. Kardaş, M. S. Kiraz, M. A. Bingöl, and H. Demirci, "A novel RFID distance bounding protocol based on physically unclonable functions," in *RFID. Security and Privacy*, A. Juels and C. Paar, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 78–93.

[20] V. Shoup, "Sequences of games: A tool for taming complexity in security proofs," 2004.