# Data Synchronization in Privacy-Preserving RFID Authentication Schemes*

Sébastien Canard and Iwen Coisel

Orange Labs R&D, 42 rue des Coutures, BP6243, F-14066 Caen Cedex, France
{sebastien.canard,iwen.coisel}@orange-ftgroup.com

**Abstract.** Massively deploying RFID systems, while preserving people's privacy and data integrity, is a major security challenge of the coming years. This is why research related to privacy-preserving authentication is growing, including design of schemes, cryptanalysis and security models. In nearly all such schemes secret key cryptography is used, since RFID tags are extremely constrained in time and space, and untraceability is achieved by updating some data at each authentication. Unfortunately, none of them entirely resists to denial of service attacks, those in which the enemy forces updating of the tag and/or the reader by sending fake messages. Moreover, literature lacks a clear and formal way of comparing schemes w.r.t. this kind of attack. In this paper, we introduce a new characterization, called synchronizability. This allows us to, first, establish a relevant model; second, evaluate existing schemes in this model and point out their deficiencies; third, present a new scheme with all desired features.

## 1 Introduction

Privacy-preserving authentication for RFID tags is a big issue in recent work. There exists a lot of schemes permitting a tag to authenticate itself to a given reader while being anonymous and untraceable for other possible actors. There are several approaches to construct such scheme, depending on the type of cryptographic key is used. Such scheme can for example be based on public key cryptography [3, 6, 14] or using a secret key centralized infrastructure [3].

Another possibility is to use a secret key infrastructure where each tag shares a personal secret key with the reader. The authentication protocol consists then for the tag in proving that it knows a valid secret key while protecting its privacy. One possibility is to use classical cipher-based challenge response protocols but they do not achieve the forward privacy property. Other solutions [2, 5, 9, 11, 16] may have a problem regarding the efficiency since the verification procedure mainly consists in searching the right tag. Obviously, the tag cannot send in clear any information about its identifier. As a consequence, most of existing

* This work has been financially supported by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT and by the French Agence Nationale de la Recherche under the RFID-AP project.

schemes [2, 5, 9, 11, 16] necessitate testing all the keys in the database until a match is found. These schemes should moreover include randomness to ensure the privacy property, which can be done either by a value sent by the reader [16] or by an update of the key [2, 5, 9, 11].

In this paper, we focus on schemes where the tag shares a personal secret key with the reader and where this key is updated regularly. The studied schemes consequently necessitate synchronization mechanisms to be sure that the same version of the key is used at both sides. And none of the state-of-the-art RFID authentication scheme [11, 2, 5] is at the same time correct (a valid tag is always accepted), sound (a fake tag is always rejected), privacy-preserving (a tag is anonymous, unlinkable and the scheme is forward secure) and resistant to a denial of service attack.

One other remaining problem is that it does not exist any formal mean to compare these schemes regarding first the capacity of an adversary to desynchronize a tag and the reader; second the way the scheme is able to resynchronize a tag and the reader; third the efficiency of the search procedure in the worst case. These comparisons are necessary to compare schemes and to know what are the advantages and the drawbacks of each of them.

In the following, we first describe in Section 2 the general context of our paper and the security model for privacy-preserving RFID authentication schemes. We next contribute to the current work on models for RFID protocols by giving in Section 3 relevant formal definitions and experiments to study and compare privacy-preserving RFID authentication scheme with key update while considering the three above points. We also apply in Section 4 our new method to existing schemes. Finally, we propose in Section 5 a new RFID authentication scheme with approximately the same features as the Dimitriou protocol but with the additional property that our scheme is privacy-preserving.

## 2 Context and Model

In this section, we set up our model for privacy-preserving authentication schemes.

### 2.1 Model for Privacy-Preserving RFID Authentication Schemes

A *tag* $\mathcal{T}$ is a transponder identified by a unique identifier $ID$ with limited abilities. A *reader* $\mathcal{R}$, composed of a transreceiver which communicates with tags and a back-end database $\mathcal{D}_{\mathcal{R}}$, is a more powerful device able to communicate with several tags up to limited distance. $\mathcal{D}_{\mathcal{R}}$ contains all identifiers $ID$ of valid tags and additional data such as keys[1]. In the following, we call *search procedure* the step that consists to find the right identifier of a tag in the database.

**Procedures.** A RFID scheme is composed of the following procedures, where $s$ is a security parameter.

[1] Note that in some other works, the database is outside the reader.

- $\mathsf{SetupReader}(1^s)$ is a probabilistic algorithm which generates a key pair $(K_S, K_P)$ for the reader. We assume that $s$ is implicitly specified in $K_P$.
- $\mathsf{SetupTag}(ID, K_P)$ is a probabilistic algorithm which returns a tag-dependent secret key $K_{ID}$. $(ID, K_{ID})$ is added in the reader's database $\mathcal{D}_\mathcal{R}$ containing the whole set of legitimate tags.
- $\mathsf{Auth}$ is an interactive protocol $\pi$ between the reader $\mathcal{R}$ taking as inputs $K_S$, $K_P$ and $\mathcal{D}_\mathcal{R}$, and a tag $\mathcal{T}$ taking as inputs $K_{ID}$, $ID$ and $K_P$. At the end, the reader either accepts the tag and outputs $ID$ or outputs 0.

**Definition of the adversary.** In all experiments given below, a challenger $\mathcal{C}$ initializes the system and the adversary isgiven the public key $K_P$ of the reader. We distinguish in the following legitimate tags from corrupted one for which the adversary knows the secrets embeded in it. Moreover, the adversary plays any role in the $\mathsf{Auth}$ protocol by e.g. deleting or modifying some requests or responses. More precisely, in all below experiments, $\mathcal{A}$ has access at any time (except when stated) to the following oracles.

- $\mathcal{O}^{\mathsf{CreateTag}}()$**:** adds a tag to $\mathcal{D}_\mathcal{R}$ with unique identifier $ID$ and key $K_{ID}$.
- $\mathcal{O}^{\mathsf{Corrupt}}(ID)$**:** returns $K_{ID}$ and flags this tag as corrupted.
- $\mathcal{O}^{\mathsf{Launch}}()$**:** makes the reader launch the first request of a new $\mathsf{Auth}$ protocol instance $\pi$.
- $\mathcal{O}^{\mathsf{SendReader}}(m, \pi)$**:** sends a message $m$ to the reader for the protocol $\pi$ and outputs the response $r$.
- $\mathcal{O}^{\mathsf{SendTag}}(m, ID)$**:** sends a message $m$ to tag $ID$ and outputs its $r$.
- $\mathcal{O}^{\mathsf{Return}}(\pi)$**:** outputs the result of the protocol $\pi$, that is 0 if the output of the reader during $\pi$ is 0 and 1 otherwise.
- $\mathcal{O}^{\mathsf{Execute}}(ID)$**:** executes a complete $\mathsf{Auth}$ protocol between the reader and the tag $ID$. Its output is the one of the $\mathcal{O}^{\mathsf{Return}}$ oracle (1 if accepted and 0 is rejected) together with the transcript of the protocol.

### 2.2 Security Properties

**Correctness.** This property (also known as the completeness property) says that a legitimate tag is always accepted in the $\mathsf{Auth}$ protocol. The formal definition below is derived from [4].

**Definition 1 (Correctness).** *An RFID system is said to be* correct *if the probability that the reader outputs* 0 *during the* $\mathsf{Auth}$ *protocol* $\pi$ *with a legitimate tag* $ID$ *belonging to* $\mathcal{D}_\mathcal{R}$ *is negligible.*

In some cases, it is necessary to define a strong correctness, where the aim of the active adversary is to make rejected a legitimate tag.

- **Strong Correctness Experiment:**

1. At any time of the game, $\mathcal{A}$ chooses a legitimate tag $ID$.
2. $\mathcal{A}$ launches a request $\mathcal{O}^{\mathsf{Execute}}(ID)$ and obtains as output the bit $b$.

**Definition 2 (Strong Correctness).** *An RFID system is said to be* strong correct *if the probability that* $b = 0$ *at the end of the Strong Correctness Experiment is negligible.*

**Soundness.** This property states that a fake tag cannot be accepted by the system. It corresponds to the strong soundness in [4] where the adversary can corrupt tags.

– **Soundness Experiment:**
1. At any time of the game, $\mathcal{A}$ plays the role of a tag in the Auth protocol by using successful calls to the $\mathcal{O}^{\mathsf{SendReader}}$ oracle, and a final call to the $\mathcal{O}^{\mathsf{Return}}$ oracle.
2. The experiment's output is 1 if $\mathcal{A}$ is accepted during the Auth protocol and the outputted tag is not corrupted, and 0 otherwise.

**Definition 3 (Soundness).** *An RFID system is* sound *if the probability that the bit $b$ returned by the $\mathcal{O}^{\mathsf{Return}}$ oracle at the end of the Soundness Experiment is equal to 1 is negligible.*

**Privacy.** A tag should be anonymous and untraceable for everyone except the valid reader. Moreover, the scheme has to preserve the privacy of a tag in its previous authentications, even if an adversary compromises it and outputs its internal data: this is what is called forward-privacy. Many formal definitions concerning privacy in RFID systems have been proposed so far [1, 7, 8, 15, 12, 13] and we use here a definition which is closed to the ones in [7, 13].

All these features are described in the following experiment, where the goal of the adversary $\mathcal{A}$ is to recognize one tag among two.

– **Privacy Experiment:**
1. At any time of the game, $\mathcal{A}$ chooses two tags $ID_0$ and $ID_1$ in the set of legitimate tags and sends $(ID_0, ID_1)$ to $\mathcal{C}$.
2. $\mathcal{C}$ randomly chooses a bit $b$. The tag $ID_b$ is called the *challenge* tag. $ID_0$ and $ID_1$ are withdrawn from $\mathcal{D}_{\mathcal{R}}$ and thus cannot be manipulated by the adversary using oracles. $ID_b$ is added to $\mathcal{D}_{\mathcal{R}}$, as an exact copy of the tag $ID_0$ or $ID_1$.
3. Again, $\mathcal{A}$ interacts with the whole system through all oracles. Note that $\mathcal{A}$ can interact with the challenge tag without any restriction.
4. $\mathcal{A}$ finally outputs a bit $b'$.

**Definition 4 (Privacy).** *We say that an RFID scheme has the* privacy *property if the probability that $b = b'$ differs from $1/2$ by a fraction that is at most negligible.*

## 3 New Characterizations for RFID Schemes

### 3.1 On RFID Schemes with Synchronization

We now consider RFID authentication schemes where the reader shares a secret key with each tag and where this key is updated after each (not necessarily successful) authentication. In these schemes, it may be possible to desynchronize

several times a tag and the reader by e.g. forcing the tag to update its key whereas the reader does not, inducing a more important work by the reader during the search procedure.

In the RFID world, it exists in the literature several ways to fight against this possible desynchronization. One may design a scheme with a good resynchronization mechanism such that no matter being the number of times the tag and the reader are desynchronized, the system will always resynchronize. But this method can be subject to denial of service attack (see [11] and Section 4.1). A way to prevent this drawback is to limit the number of accepted resynchronization by the reader and reject a tag after that. But this may imply that a valid tag is rejected if it has been desynchronized a sufficient number of time (see [2] and Section 4.1). Another possibility for a scheme is to prevent a tag and the reader to be desynchronized more than a fixed number of times, so that the number of resynchronization is also limited (see [5] and Section 4.2).

As a consequence, since a scheme can use one of the above technique, it seems difficult to compare related work. In this paper, we construct a model to quantify the quality of a RFID authentication scheme when considering first the number of desynchronization the scheme is subject to; second the number of resynchronization the scheme can manage; third the efficiency of the search procedure in case of multiple desynchronizations.

### 3.2 Notation and Definition

Let us consider a valid RFID tag $ID$ with the corresponding key denoted $K_{ID}$. We need to introduce new notation that will be used in the rest of the paper.

First, to take into account the update of the key $K_{ID}$ during the authentication protocol, we will denote by $K_{ID}^{(0)}$ the key output by the $\mathsf{SetupTag}(ID)$ procedure, by $K_{ID}^{(j)}$ the $j$-th version of the key of the tag $ID$ after updating it $j$ times.

Second, as the key is stored both in the reader and the tag, as described in the SetupTag procedure, we need to distinguish the key that is embedded into the tag, denoted $TK_{ID}$, and the key stored in the database, denoted $RK_{ID}$, both initialized to $K_{ID}^{(0)}$. Usually $TK_{ID} = RK_{ID} = K_{ID}^{(*)}$ but, as the tag and the reader can be desynchronized by some adversaries, we may have $TK_{ID} = K_{ID}^{(i)}$ and $RK_{ID} = K_{ID}^{(j)}$ with $i \neq j$. We can now define a desynchronization as following.

**Definition 5.** *Considering an arbitrary tag $ID$ with $TK_{ID} = K_{ID}^{(i)}$ and reader with $RK_{ID} = K_{ID}^{(j)}$, a* desynchronization *is an operation inducing an incrementation of the value $|i - j|$ by 1. The value $|i - j| \in \mathbb{N}$ is called* number of desynchronizations.

### 3.3 The Desynchronization Capacity

We first study in depth the *desynchronization* characterization: we want to know for a given scheme the maximum number of desynchronizations between a tag

and the reader an adversary can create. We want to compute the corresponding scale, called *the desynchronization value*.

More precisely, to have a better characterization, we will distinguish on one side the maximum number of desynchronization $D_{\mathcal{R}}$ an adversary can create when only focusing on the reader, and on the other side this maximum number $D_{\mathcal{T}}$ when $\mathcal{A}$ only focuses on the tag. The desynchronization value consequently corresponds to the couple $(D_{\mathcal{R}}, D_{\mathcal{T}})$.

In fact, the Strong Correctness Experiment presented in Section 2.2 gives us a direct way to define more formally and compute these values. Let $\mathcal{A}$ being the adversary playing the experiment. At the end of step 1, $\mathcal{A}$ chooses a tag $ID$. We denote by $TK_{ID} = K_{ID}^{(i)}$ (resp. $RK_{ID} = K_{ID}^{(j)}$) the tag's version (resp. reader's version) of $K_{ID}$ at the end of this step. The maximum number of desynchronization obtained by the adversary $\mathcal{A}$, when $\mathcal{A}$ only focuses on the tag (resp. the reader), is $D_{\mathcal{T},\mathcal{A}} = j - i$ (resp. $D_{\mathcal{R},\mathcal{A}} = i - j$).

The desynchronization value can now be defined more formally as follows.

**Definition 6.** *For a given RFID authentication scheme, the* desynchronization value *of a scheme is the couple* $(D_{\mathcal{R}}, D_{\mathcal{T}})$ *with* $D_{\mathcal{R}} = \underset{\mathcal{A}}{Sup}(D_{\mathcal{R},\mathcal{A}})$ *and* $D_{\mathcal{T}} = \underset{\mathcal{A}}{Sup}(D_{\mathcal{T},\mathcal{A}})$*. The scheme is said* $(D_{\mathcal{R}}, D_{\mathcal{T}})$–desynchronizable*.*

### 3.4 The Resynchronization Capacity

Now, we study the *resynchronization* characterization: we study the capacity a scheme has to resynchronize a tag and the reader by computing the corresponding scale, called *the resynchronization value*, defined as follows.

**Definition 7.** *For a given RFID authentication scheme, when considering an arbitrary valid tag* $ID$ *with* $TK_{ID} = K_{ID}^{(i)}$ *and a reader with* $RK_{ID} = K_{ID}^{(j)}$*, we denote by* $R_{\mathcal{R}}$ *(resp.* $R_{\mathcal{T}}$*) the maximum number of desynchronizations the scheme can tolerate to accept the tag* $ID$ *during the* Auth *procedure, if only the tag (resp. the reader) has been updated, i.e. after the* Auth *procedure* $RK_{ID} = K_{ID}^{(j)}$ *(resp.* $TK_{ID} = K_{ID}^{(i)}$*). The* resynchronization value *of the scheme is the couple* $(R_{\mathcal{R}}, R_{\mathcal{T}})$ *and the scheme is said* $(R_{\mathcal{R}}, R_{\mathcal{T}})$–resynchronizable*.*

We now give methods to compute respectively $R_{\mathcal{R}}$ and $R_{\mathcal{T}}$. Let us consider a tag $\mathcal{T}$, the reader $\mathcal{R}$ (synchronized with $\mathcal{T}$) and a counter $C$ which will be incremented in each round of the two experiments. We first introduce two procedures: UpdateTag($ID$), which forces $TK_{ID}$ to be updated, and UpdateReader($ID$) which forces $RK_{ID}$ to be updated.

The computation of $R_{\mathcal{R}}$ (resp. $R_{\mathcal{T}}$) works as follows. During round $C$, we produce $C$ desynchronizations of the tag (resp. the reader) using UpdateTag($ID$) (resp. UpdateReader($ID$)) and then launch the Auth procedure between $\mathcal{R}$ and $\mathcal{T}$. If the reader accepts the tag, $TK_{ID}$ and $RK_{ID}$ are resynchronized (i.e. $TK_{ID} = RK_{ID}$), $C$ is incremented and a new round is started[2]. Else we stop the algorithm and output the value $C - 1$ which is exactly $R_{\mathcal{R}}$ (resp. $R_{\mathcal{T}}$).

[2] the tag (resp. the reader) will be updated once more.

### 3.5 Conclusions on the Synchronization

Given a RFID authentication scheme, we are now able to compare both desynchronization and resynchronization scales. Intuitively, if the desynchronization value is less or equal to the resynchronization value, an adversary will not be able to win the Strong Correctness Experiment by desynchronizing a tag or a reader: the scheme is considered secure.

**Definition 8.** *For a given RFID authentication scheme, if* $D_{\mathcal{R}} \leq R_{\mathcal{R}}$ *and* $D_{\mathcal{T}} \leq R_{\mathcal{T}}$*, the scheme is said* synchronizable*. Else, the scheme is said* desynchronizable*.*

Note that if an adversary $\mathcal{A}$ is able to desynchronize a tag (or a reader) more times than the scheme is able to resynchronize it, the scheme is obviously not strong correct.

### 3.6 Efficiency of the Search Procedure

During the protocol, the reader performs a search procedure in order to retrieve the identifier of the tag it is interacting with. The efficiency of a scheme is strongly connected to the one of this procedure. If some cases, an attacker can saturate a reader with bad requests inducing a denial of service attack.

Consequently, we need to compute, for a given RFID authentication scheme, the number of operations that are performed by the reader in the worst case to either retrieve an identifier or reject a tag. In the following, an operation corresponds to one of the following items[3].

- The call to a no-key mathematical or cryptographic function (e.g. a hash function, a modular reduction, a pseudo-random function).
- The call to a symmetric key cryptographic function (e.g. a HMAC).

## 4 Study of Existing Schemes

In the rest of the paper, we assume that all hash functions are cryptographically secure. More precisely, we need them to be one-way (i.e. given $H(m)$, it is infeasible to retrieve $m$) and collision-free (i.e. given $H(m)$, it is computationally infeasible to find $m' \neq m$ s.t. $H(m) = H(m')$) and we also use the random oracle. $H^k(m)$ denotes the $k$-composite of $H$ (i.e. $H \circ \ldots \circ H$ $k$ times).

In the following, we do not necessary detail the proofs of our theorems. This is due to the fact that most of these results are already known in the cryptographic community and also due to lack of space. Our aim in this section is to give the result of our analysis for existing schemes.

We here study existing and known secure schemes. For example, our study does not include the proposal of Lim and Kwon [10] and the one of Le et al. [8] since they have been recently shown as traceable [13], even if the second one [8] has interesting features concerning desynchronization and resynchronization.

[3] The search procedure also includes equality tests but, as they are negligible compared to the other kinds of operation, we neglect them.

### 4.1 The OSK Schemes

The OSK scheme has been introduced by Ohkubo, Suzuki and Kinoshita in 2003 [11]. Let $H_1, H_2$ be two secure hash functions. Each tag $ID$ of the system is set up with a secret key $TK_{ID} = K_{ID}^{(0)}$ and the database is set up with all the couples $(ID, RK_{ID} = K_{ID}^{(0)})$. In the life cycle of the tag $ID$, the key $K_{ID}^{(j)}$ is updated as $K_{ID}^{(j+1)} = H_2(K_{ID}^{(j)})$ after each (successful) authentication. During the protocol, the tag $ID$ simply response $\mathcal{H}_1(TK_{ID})$ to a request from the reader and then update its key.

The search procedure of the OSK protocol is very simple and works as follows. For each entry $i \in \mathcal{D}_{\mathcal{R}}$, the reader computes $H_1(RK_i)$ and compares it with $r$. If there is no match, the reader temporarily updates the entries of $\mathcal{D}_{\mathcal{R}}$ and starts again the search procedure. After a match is found, the key of the corresponding tag is updated.

**Theorem 1.** *The OSK scheme is* $(\infty, 0)$*–desynchronizable,* $(\infty, 0)$*–resynchronizable and the search procedure works in an infinite number of operations in the worst case.*

**The OSK$_m$ Scheme.** This protocol aims at increasing the efficiency of the search procedure by stopping it after after $m$ updates, where $m$ is fixed.

**Theorem 2.** *The OSK$_m$ scheme is* $(\infty, 0)$*–desynchronizable,* $(m, 0)$*–resynchronizable and the search procedure works in* $2m+1$ *operations per tag in the worst case. The scheme is consequently desynchronizable.*

**The OSK-AO Scheme.** Avoine and Oechslin have proposed in [2] another modification of the OSK protocol os as to reduce the complexity of the search procedure at the cost of a larger database. Let $n$ be the size of the database and $m$ be a security parameter. Due to space restriction, we refer the reader to [2] for details.

**Theorem 3.** *The OSK-AO scheme* $(\infty, 0)$*–desynchronizable,* $(m-1, 0)$*–resynchronizable and the search procedure works in* $2(t-1)^2/n$ *operations per tag in the worst case. The scheme is consequently desynchronizable.*

### 4.2 The Dimitriou Scheme

The Dimitriou protocol has been proposed in [5] and is presented in Figure 1. As the tag only updates its secret key if it has authenticate the reader, this scheme has not the drawback of the OSK like schemes, at the cost of more rounds. Note that contrary to [5], we need that the reader stores $ID$, the current and the last versions of the key, in order to prevent an attack against the strong correctness property.

In the following, we denote $r_1 = H_1(TK_{ID})$ and $r_2 = H_1(TK_{ID}||N_T||N_R)$. During our new search procedure, the reader compares the hash of each $RK_{ID}$

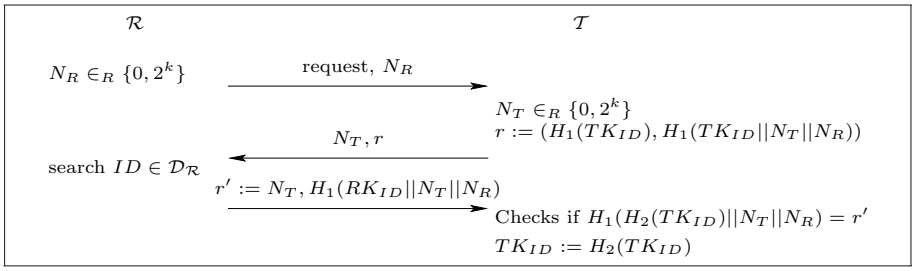


**Fig. 1.** Protocol of Dimitriou

with $r_1$. If a match is found and if $r_2$ is well constructed, it outputs $ID$ and updates $RK_{ID}$ and $RK'_{ID}$. If no match is found, the reader compares the received with all previous keys $RK'_{ID}$ and does the same as before, except updating the keys.

**Theorem 4.** *The Dimitriou scheme is* $(0, 1)$*–desynchronizable,* $(0, 1)$*–resynchronizable and the search procedure works in* 2 *operations per tag in the worst case. The scheme is consequently synchronizable.*

*Proof.*

- **Desynchronization:** tn adversary cannot update the tag since he cannot produce $H_1(RK_{ID}||N_T||N_R)$: $D_{\mathcal{R}} = 0$. Moreover, if the adversary block the last message of a valid protocol, the reader updates the key whereas the tag does not. As this can be done only once by the adversary, $D_{\mathcal{T}} = 1$: the scheme is $(0, 1)$–desynchronizable.
- **Resynchronization:** if the tag is updated, there is now way for the reader to make the link between the received value and the values stored in $\mathcal{D}_{\mathcal{R}}$. The tag is thus rejected and $R_{\mathcal{R}} = 0$. As described above, if the database is updated only once, the tag is accepted. On the other hand, if $RK_{ID}$ is updated twice, $TK_{ID}$ is no longer stored and $ID$ is rejected: $R_{\mathcal{T}} = 1$, the scheme is $(0, 1)$–resynchronizable and so synchronizable.
- **Search procedure efficiency:** in the worst case, the search procedure computes the hash of all keys in the actual and in the previous version: there are 2 operations per tag.

Note that the efficiency of the search procedure can be improved by directly storing the two versions of the key so that the reader has no operation to perform (only tests). The drawback is that the size of the memory is multiplied by two.

The main problem of the Dimitriou, whatever be the version, is that it is traceable w.r.t. the privacy experiment described in Section 2 since the adversary can call the $\mathcal{O}^{\mathsf{SendTag}}((request, N_R), ID_0)$ oracle for tag $ID_0$ (to obtain the value $H_1(TK_{ID_0})$) and choosing at random $ID_1$. If the response from the challenge contains $H_1(TK_{ID_0})$, $\mathcal{A}$ outputs $b' = 0$ and $b' = 1$ otherwise.

# 5 Our Proposal: the $C^2$ Scheme

## 5.1 Description of the $C^2$ Scheme

Our scheme, presented on Figure 2, aims at proposing a secure construction and being good regarding synchronisation and efficiency as defined in Section 3. As

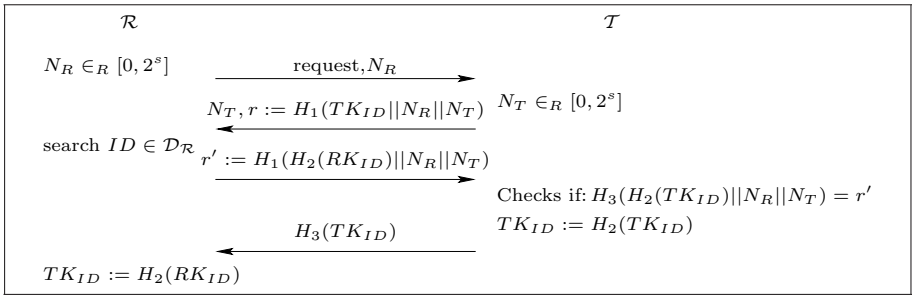


**Fig. 2.** The $C^2$ Protocol

in the Dimitriou scheme, the random values $N_R$ and $N_T$ are used to protect the scheme against replay attack. We next protect the scheme on privacy at the cost of a less efficient search procedure. We next design our scheme in such a way that the tag updates its key only after having authenticated the reader. Finally, the last message from the tag permits to convince the reader that the tag has updated its internal key.

In the search procedure, for each $ID \in \mathcal{D}_{\mathcal{R}}$, the reader computes the value $H_1(RK_{ID}||N_R||N_T)$ and compares it with $r$. In case of a match, the procedure outputs the corresponding identifier. Else, for each $ID \in \mathcal{D}_{\mathcal{R}}$, the reader computes the ephemeral identifier $EID = H_2(RK_{ID})$, computes $H_1(EID||N_R||N_T)$ and compares it with $r$. In case of a match, the reader outputs the corresponding identifier and updates $RK_{ID}$ (so $RK_{ID} = TK_{ID}$). Otherwise, the tag is rejected.

We thus obtain a $(1, 0)$-synchronizable scheme instead of $(0, 1)$-synchronizable for the Dimitriou scheme. It seems better in practice as we do not need to store the previous identifier of the tag and so use less memory for the database.

## 5.2 Security Arguments

We first prove that the $C^2$ scheme is secure.

**Theorem 5.** *The $C^2$ scheme is sound under the security of the hash function and in the random oracle.*

*Proof.* To win the Soundness Experiment, an adversary can first guess which random value $N_R$ will be sent. If $\mathcal{A}$ has beforehand asked the tag the corresponding

answer: this is not possible if the security parameter $s$ is well chosen. Another possibility for the adversary is to produce a valid message $H_1(TK_{ID}||N_R||N_T)$ without knowing a valid (and uncorrupted) value $TK_{ID}$: this is not possible under the security of the hash function. So the scheme is sound.

**Theorem 6.** *The $C^2$ scheme is private under the security of the hash function and in the random oracle.*

*Proof.* The scheme clearly provides the anonymity of the tag under the assumption that the hash function is one-way. Moreover, the scheme is unlinkable since $\mathcal{A}$ has no control on the value $N_T$. Again, if the used hash functions are secure, the three first messages are useless for $\mathcal{A}$. One possibility for $\mathcal{A}$ is to use $H_3(TK_{ID})$ but this message is only sent when the key is updated: $\mathcal{A}$ cannot learn any information of it. Finally, the scheme provide the forward-privacy property using the same argument. So the scheme is private.

**Theorem 7.** *The $C^2$ scheme is $(1,0)$–desynchronizable, $(1,0)$–resynchronizable and the search procedure works in 3 operations per tag in the worst case. The scheme is consequently synchronizable.*

*Proof.*

- **Desynchronization:** Here we first highlight the fact that $\mathcal{A}$ is not able to produce valid messages for this protocol. Indeed, the only way to do this is to know the secret key used either by the tag or the reader. As the tag is uncorrupted and the hash function is one-way, $\mathcal{A}$ cannot learn anything about this key. By blocking the last message of a protocol, $\mathcal{A}$ desynchronize the tag as it updates its secret key contrary to the reader. $\mathcal{A}$ cannot use this technique twice as the reader resynchronize its key on reception of the second message (during the search procedure). As a consequence, $\mathcal{A}$ must produce the third messages without interacting with the reader. As said before it is impossible and so $D_{\mathcal{R}} = 1$. The only way for $\mathcal{A}$ to force the update of $RK_{ID}$ without updating $TK_{ID}$ is to produce the last message. As $\mathcal{A}$ has no information about $TK_{ID}$, he is not able to produce such a message. So, $D_{\mathcal{T}} = 0$. Finally, the scheme is $(1,0)$–desynchronizable.
- **Resynchronization:** By definition of the scheme, the tag is still accepted if $TK_{ID}$ is updated once. This is not the case if it is updated twice. So, $R_{\mathcal{R}} = 1$. If the reader updates the stored keys once, $TK_{ID}$ is no longer stored in the database and the reader does not find a match, even if it updates all the stored keys once. As a consequence the tag is rejected, $R_{\mathcal{T}} = 0$. The scheme is $(1,0)$–resynchronizable and so synchronizable.
- **Search procedure efficiency:** On reception of a random value, the search procedure computes, for all $ID \in \mathcal{D}_{\mathcal{R}}$, $H_1(RK_{ID}||N_R||N_T)$, with $RK_{ID} = K_{ID}^{(*)}$ and its update $K_{ID}^{(*+1)}$. Next, it compares these values with the received one. This implies per tag 3 calls to a hash function (two for the computation of messages and one for the update).

Note that the search procedure can be fastened by storing in the database the updated key and obtain 2 operations in the search procedure.

## 6 Conclusion

Inspired by the state of the art, we have suggested a new model for RFID authentication most adapted for schemes using a secret key infrastructure with an update key mechanism and we have designed a protocol with good properties while being secure in the privacy model.

**Acknowledgments.** We are grateful to Marc Girault for his suggestions of improvement, and to anonymous referees for their valuable comments.

## References

1. G. Avoine. Adversarial model for radio frequency identification. Cryptology ePrint Archive, Report 2005/049, 2005.
2. G. Avoine and P. Oechslin. A scalable and provably secure hash based RFID protocol. In *PerSec 2005*. IEEE Computer Society Press, 2005.
3. B. Calmels, S. Canard, M. Girault, and H. Sibert. Low-cost cryptography for privacy in rfid systems. In *CARDIS 2006*, volume 3928 of *LNCS*, pages 237–251. Springer, 2006.
4. I. Damgård and M. Ø. Pedersen. Rfid security: Tradeoffs between security and efficiency. In *CT-RSA 2008*, volume 4964 of *LNCS*, pages 318–332. Springer, 2008.
5. T. Dimitriou. A lightweight RFID protocol to protect against traceability and cloning attacks. In *SecureComm 2005*. IEEE Computer Society Press, 2005.
6. M. Girault and D. Lefranc. Public key authentication with one (online) single addition. In *CHES 2004*, volume 3156 of *LNCS*, pages 413–427. Springer, 2004.
7. A. Juels and S. A. Weis. Defining strong privacy for rfid. In *PERCOMW '07*, pages 342–347, Washington, DC, USA, 2007. IEEE Computer Society.
8. T. V. Le, M. Burmester, and B. de Medeiros. Universally composable and forward-secure rfid authentication and authenticated key exchange. In *ASIACCS 2007*, pages 242–252. ACM, 2007.
9. S. Lee, T. Asano, and K. Kim. RFID mutual authentication scheme based on synchronized secret information. In *Symposium on Cryptography and Information Security*, Hiroshima, Japan, January 2006.
10. Chae Hoon Lim and Taekyoung Kwon. Strong and robust rfid authentication enabling perfect ownership transfer. In *ICICS*, volume 4307 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2006.
11. M. Ohkubo, K. Suzuki, and S. Kinoshita. Cryptographic approach to "privacy-friendly" tags. In *RFID Privacy Workshop 2003*, 2003.
12. K. Ouafi and R. C.-W. Phan. Traceable privacy of recent provably-secure rfid protocols. In *ACNS 2008*, volume 5037 of *LNCS*, pages 479–489, 2008.
13. R.-I. Paise and S. Vaudenay. Mutual authentication in rfid: security and privacy. In *ASIACCS 2008*, pages 292–299. ACM, 2008.
14. S. Vaudenay. RFID privacy based on public-key cryptography (abstract). In *ICISC 2006*, volume 4296 of *LNCS*, pages 1–6, Busan, Korea, November-December 2006. Springer-Verlag.
15. S. Vaudenay. On privacy models for rfid. In *ASIACRYPT 2007*, pages 68–87, 2007.
16. S. Weis, S. Sarma, R. Rivest, and D. Engels. Security and privacy aspects of low-cost radio frequency identification systems. In *SPC 2003*, volume 2802 of *LNCS*, pages 454–469. Springer-Verlag, 2003.