An optimal probabilistic solution for information confinement, privacy, and security in RFID systems

Roberto Di Pietro
Dipartimento di Matematica, Università di Roma Tre, Roma, Italy
UNESCO Chair in Data Privacy, Universitat Rovira i Virgili, Tarragona - Spain
and
Refik Molva
Head of the Computer Communications Department, Institut Eurécom, Sophia-Antipolis, France

In this paper, we provide the following contributions to enhance the security of RFID based systems. First, we assume that among multiple servers storing the information related to the tags some of them can be compromised. For this new threat scenario, we devise a technique to make RFID identification server dependent, providing a different unique secret key shared by a tag and a server. The solution proposed requires the tag to store just a single key, thus fitting the constraints on tag's memory. Second, we provide a probabilistic tag identification scheme that requires the server to perform just bitwise operations and simple list manipulation primitives, thus speeding up the identification process. The tag identification protocol assures privacy, security and resilience to DoS attacks thanks to its stateless nature. Moreover, we extend the tag identification protocol to achieve mutual authentication and resilience to reply attacks. The proposed identification protocol, unlike other probabilistic protocols, never rejects a legitimate tag. Furthermore, the identification protocol requires the reader to access the local Data Base (DB) of tags' keys $O(n)$ times, while it has been shown in the literature that a privacy preserving identification protocol requires a reader to access $\Theta(n)$ times this DB. In this sense, our protocol is optimal. Finally, the three features suggested in this paper, namely, reader-dependent key management, tag identification, and mutual authentication, can be independently adopted to build alternative solutions.

Categories and Subject Descriptors: C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.2.4 [**Computer Systems Organization**]: computer communication networks—*Distributed Systems*; D.4.6 [**Operating systems**]: Security and Protection—*Cryptographic controls*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

General Terms: Algorithms, Security.

Additional Key Words and Phrases: RFID systems, Information confinement, security, privacy, probabilistic algorithm.

1. INTRODUCTION

Radio Frequency IDentification (RFID) is a technology for automated identification of objects and people. An RFID device, also known as tag, is a small microchip designed for wireless data transmission. It is generally attached to an antenna in a package that resembles an ordinary adhesive sticker. The applications of RFID ranges from cattle monitoring to e-passport [Juels 2006].

The other components of an RFID system are readers and servers. A reader is a device

querying tags for identification information, while all information about tags (ID, assigned keys, etc.) are maintained on servers. A server can be assigned multiple readers; in this case it only engages in communication with its constituent readers. It is generally assumed to have a single logical server that might resolve to multiple physically replicated servers. All communications between server and readers is assumed to be over private and authentic channels. Both readers and server do not suffer of constraints on power, processing, memory, and bandwidth.

Furthermore, based on a widely agreed assumption, servers, readers and the link between them are assumed to be trusted in that only the tags and the communication channel between the tag and the readers are assumed to be potentially vulnerable to malicious attacks [Juels 2006; Tsudik 2006]. In this paper, we relax this hypothesis by assuming a more general setting whereby tags, servers and readers can be subject to malicious attacks. In that context, we focus on the problem of tag identification by multiple servers that are either replicas of the same logical server or different servers governed by independent authorities. As a result of the relaxed security hypothesis, the new requirement in this setting is to cope with the compromise of servers. Apart from the obvious need to perform mutual authentication, as opposed to one-way authentication of the tag by the server, server compromise calls for new measures to prevent possible attacks originating from the leakage of secrets stored in the compromised server's authentication database. For instance, based on most existing tag authentication protocols, using the entries of a compromised server's authentication database, the attacker can fabricate duplicate tags.

The first contribution of this paper is an information confinement technique aiming at keeping the impact of server compromise limited. Thanks to this technique, the compromise of a server does not affect the authentication of any tag by other servers, be they replicas of the same logical server or different servers. A simple solution for confinement could consist of having each tag and server pair share a unique set of secrets. However, this solution would not be suitable with the memory constraints of RFID tags since with m servers, each RFID tag would have to store m pieces of information. The solution proposed in this paper requires the RFID tag to store a single secret key for all servers yet assuring the confinement property in case of server compromise.
Another challenging issue that affects the RFID systems is the responsiveness of the server during tag identification. It is usually the case that the server needs to search its DB of locally stored keys and to perform a cryptographic operation on each of these keys in order to identify the tag. In some scenarios the cost and the time required to identify a tag can be prohibitive due to the total number of tags that can potentially interact with the same server. Existing proposals for RFID identification try to reduce the complexity of the search operation performed by the server without requiring the tag to perform costly operations. Along the same lines, the second contribution of this paper is an efficient identification technique based on a probabilistic mechanism for the server to identify the tag that requires the tag to perform only bitwise operations, and the server to perform, other than bitwise operations, few simple list manipulation primitives. Note that our identification protocol requires the server to access just $O(n)$ its local DB of tags' keys, while it has been shown in [Damgård and Pedersen 2008] that, to preserve privacy, $\Theta(n)$ access to the DB are required. In this sense, our proposal is optimal. Further, note that our identification protocol, unlike other probabilistic solutions, by construction cannot reject a valid tag during the identification process. A thorough assessment of security and privacy achieved by the identification pro-

tocol is also provided.
Through a three-way handshake protocol this identification technique also achieves mutual authentication, as well as resilience against DoS and replay attacks.
Finally, the three features suggested in this paper, namely, reader-dependent key management, tag identification, and mutual authentication, can be independently adopted to build alternative solutions.

The sequel of the paper is structured as follows: next section introduces the related work; Section 3 outlines the system assumptions and Section 4 presents a mutual authentication protocol incorporating the confinement and probabilistic identification techniques, while Section 5 is devoted to assess the security and privacy of the proposed protocol. Section 6 focuses on further security properties and provides overhead analysis. Finally, in Section 7 we expose some concluding remarks.

2. RELATED WORK

A standard approach to provide security in RFID protocols [Molnar and Wagner 2004; Rhee et al. 2005] consists of using a unique key for each tag such that only the verifier (server) knows all the keys. This approach suffers from an expensive time complexity on the server side. Indeed, because only symmetric cryptographic functions can be used, the server needs to explore its entire database in order to retrieve the identity of the tag it is interacting with. If n is the number of tags managed by the server, $O(n)$ cryptographic operations are required to identify one tag. The advantage of the server over an adversary is that the server knows in which subset of identifiers it needs to search while the adversary has to explore the full range of identifiers.

In [Molnar and Wagner 2004] a proposal that requires just $\log_\delta n$ interactions between the server and a tag for the server to identify the tag is proposed. However, this approach requires $\log_\delta$ keys to be stored on each tag and in [Avoine et al. 2005] it has been proved that this technique weakens the privacy when an adversary is able to tamper with at least one tag. Further, the more tags an adversary tampers with, the more privacy is exposed.

A general solution, also adopted in [Tsudik 2006; Rhee et al. 2005] is to employ hash chains to allow tag identification and mutual authentication between the tag and the server. However, note that the hash chain length corresponds to the lifetime of the tag, which must be therefore stated in advance, leading to a waste of memory on the server side. Moreover, as the same author of [Tsudik 2006] recognizes, this solution is prone to DoS attack, in that an adversary can easily exhaust the hash chain via reading attempts.

In [Avoine et al. 2005; Avoine and Oechslin 2005] the authors optimizes a technique originally proposed in [Hellman 1980]. This technique allows to trade-off between time and the memory required on the reader. In particular, the time T required to invert any given value in a set of N outputs of a one-way function $h(\circ)$ with the help of M units of memory is $T = N^2\gamma/M^2$, where γ is a factor (usually a small one: < 10) to account for success probability. However, note that the technique is still prone to DoS attack and requires more computations on the server side. Leveraging this idea, in [Conti et al. 2007] the authors propose a new RFID identification protocol —RIPP-FS— that enforces privacy and forward secrecy, as well as resilience to a specific DoS attack, where the goal of the adversary is to force the tag to overuse the hash chain that has a finite length originally set to last for the tag's expected lifetime.

Aforementioned solutions assume that servers are trusted and cannot be compromised.

The first requirement raised by relaxing this hypothesis is for mutual authentication. An interesting solution to mutual authentication is exposed in [Juels and Weis 2005]: the authors are inspired by the work in [Hopper and Blum 2001] to introduce the HB+ protocol, a novel, symmetric authentication protocol with a simple, low-cost implementation. The security of the HB+ protocol against active adversaries is proved and based on the hardness of the Learning Parity with Noise (LPN) problem. The protocol is based on r rounds, where r is the security parameter, and each round requires: the tag and the server to send a message of $|\ell|$ bits to each other, where $|\ell|$ is the key length; to perform two inner product over terms of $|\ell|$ bits. A further work [Gilbert et al. 2005] showed the vulnerability of the HB+ protocol against a man in the middle (MIM) attack. A fix to the MIM attack HB+ was subject to was proposed in [Bringer et al. 2006], through the HB++ protocol. Furthermore, HB++ was proven in [Piramuthu 2006] to be subject to a particular attack in which the adversary could gain knowledge of the private key of the tag, hence jeopardizing the authentication mechanism. Finally, to complete (so far) the HB saga, in [Gilbert et al. 2008] the authors introduce a new protocol denoted random-HB$\sharp$. This proposal avoids many practical drawbacks of HB+, while remaining provably resistant to attacks in the model of Juels and Weis [Juels and Weis 2007]; at the same time it is provably resistant to a broader class of active attacks that include the attack of [Gilbert et al. 2005]. Further, the authors introduce an enhanced variant, called HB$\sharp$, that offers practical advantages over HB+.

As for recent attacks on a particular class of protocols, that is protocols that are based on linear transformations, there are two relevant papers to cite. In the first one [Ouafi and Phan 2008], the authors analyze the privacy of some recently proposed RFID authentication protocols and show how to compromise this property. It is worth noticing that the protocol in [Castelluccia and Soos 2007], inspired by [Di Pietro and Molva 2007], is subject to a linear algebra attack based on the fact that the protocol is based on linear transformation only. A general way to build attacks based on linear algebra for protocols that leverages linear transformations only and a new framework for the evaluation of privacy can be found in [van Deursen et al. 2008]. In particular, note that the same protocol in [Di Pietro and Molva 2007] —the one the solution presented in this paper is inspired by— is subject to a linear algebra attack, that will be discussed in detail in Section 5. However, in the same section we will prove that this proposal provides both key secrecy and privacy.

A recent paper with a fundamental result in the RFID area is [Damgård and Pedersen 2008]. In that paper the authors propose a model and definition for anonymous (group) identification that is well suited for RFID systems. Further, for the case where tags hold independent keys, they prove a conjecture by Juels and Weis, namely in a strongly private and sound RFID system using only symmetric cryptography, a reader must access virtually all keys in the system when reading a tag. This poses on a reader a lower bound of $\Theta(n)$ access to its local memory. Other interesting privacy models proposed in the literature are [Juels and Weis 2007] and [Vaudenay 2007]. In particular, the former will be adopted in this paper.

3. SYSTEM ASSUMPTIONS/MODEL

The components of the system are: tags, readers and key distribution centers (KDCs). KDCs represent the authorities ruling over a set of tags. Each KDC generates a unique key k_i of ℓ bits for every tag T_i that is under its jurisdiction and securely stores it in the tag.

The KDC also provides each reader $reader_j$ that is authorized to identify a tag T_i that is under its jurisdiction with a derived tag identification key $k_{i,j}$ along with the identifier ID_i of the tag. Each tag can thus be identified by one or several readers based on the derived tag identification keys distributed by the KDC. Each reader keeps in a secure key database (KDB) the set of derived tag identification keys and identifiers of the tags it is authorized to identify. It should be noted that in this model a reader can be associated with more than one KDC or be able to identify tags issued by several authorities.

Each tag has the capability to run a pseudo random number generator (PRNG) that generates at every invocation an output of $\ell + 1$ bits, and a secure hash function $h(\circ)$, with an output over ℓ bits. Note that these assumptions are coherent with the ones in the literature [Tsudik 2006; Molnar and Wagner 2004; Avoine et al. 2005]. The KDC assigns a unique key k_i to T_i. The derived tag identification key $k_{i,j}$ will be generated by the KDC during the initialization of $reader_j$'s KDB, based on the expression $k_{i,j} = h(k_i||reader_j||k_i)$, where $''||''$ denotes concatenation.
In the following we will assume the KDB to host n elements, where n is the number of tags in the system. These n elements are organized in a linked list data structure, with the usual operations associated to a list. In particular, if we are examining the g^{th} element of KDB, then KDB_g.*key* returns key $k_{g,j}$, while ***KDB.next*** returns the pointer to the next element in the list.

4. THE PROTOCOL

This section presents the protocol implementing information confinement and probabilistic identification. Further details are then provided on the mutual authentication and the lookup process that is the underpinning of the probabilistic identification technique.

4.1 Overview of the solution

Our proposal for tag identification and mutual authentication is based on a simple three-way handshake, as depicted in Figure 1. In the first flow, the reader sends a challenge and its identity to the tag. The tag replies with a response message computed based on its secret key, the identity of the reader, the challenge and a set of locally generated pseudo random numbers. The reader retrieves the identity of the tag through a lookup in its local database. If the lookup succeeds, the reader has authenticated the tag. The last flow of the protocol allows the tag to authenticate the reader. The main idea of our solution for information confinement is a reader-dependent identification mechanism that allows each reader (or the server to which the reader is connected to) to identify and authenticate a tag based on some long-term secret ($k_{i,j}$) that is different on each server whereas each tag keeps a unique secret identification key (k_i) for all readers. During the identification process each tag generates a temporary reader-dependent secret based on the identifier ID_j of the reader it is communicating with and its unique secret identification key k_i, computing $k_{i,j} = h(k_i||ID_j||k_i)$. The advantages of the reader-dependent mechanism are twofold:

—confinement of exposure: compromise of the long term secrets at a reader does not impact the identification of the same tag by other readers; in particular, the impersonation of these secrets does not allow an intruder to impersonate the tag with respect to other readers.

—selective reader access or non-transferable tag identification capability: the set of readers

authorized to perform tag identification can be controlled based on each reader's identity. Since the long-term identification secret for a tag is tightly bound with each reader's id, the identification capability cannot be transferred among readers with different identities and the set of tags each reader is authorized to identify can be determined based on the set-up of long-term identification keys.

Another innovative feature of our proposal is the the lookup process. Based on the response message transmitted by the tag, the reader searches the matching entry of its database (if any) by iterative elimination of the entries that cannot match with the message from the tag. The response message includes a series of verification values $(\alpha_1, \ldots, \alpha_q)$ computed under the key $k_{i,j}$ associated with the tag and the reader. Each verification value allows the reader to eliminate about one fourth of the elements in the KDB. By subsequently eliminating elements in the list, the reader achieves the identification of the tag. Unlike other solutions whereby each step of the lookup process requires encryption or hashing, the lookup process depicted in this paper is efficient in that it requires just $O(n)$ bit-wise operations (where n is the number of tags) and simple list manipulation primitives. By construction, the protocol never rejects legitimate tags. On the other hand, the probability for the protocol to accept an illegitimate tag (a tag for which there is no entry in the KDB) is a system design parameter that can be set to any small, non-zero value ($\epsilon > 0$).

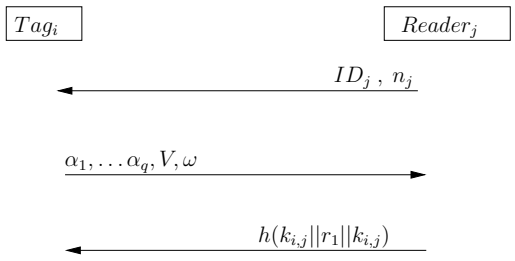

Fig. 1. The proposed protocol

4.2 Lookup Process

The lookup process allows the reader to identify the tag based on the following messages sent by the tag in the second flow of the protocol: $< \alpha_1, \ldots, \alpha_q, V, w >$, where $\omega = h(k_{i,j}||n_j||r_1||k_{i,j})$, V is a bit vector of length q. For $p \in [1 \ldots q]$, α_p and $V[p]$ are defined as follows:

$$\alpha_p = k_{i,j} \oplus r_p \tag{1}$$

$$V[p] = DPM(r_{p,0})z_p \oplus DPM(r_{p,1})(1 - z_p) \tag{2}$$

where the value r_p consists of the first ℓ bits generated by the invocation of the PRNG. To ease protocol exposition we will refer to the first $\ell/2$ bits of r_p with $r_{p,0}$ and to the remaining $\ell/2$ bits with $r_{p,1}$, that is $r_p = \{r_{p,0}||r_{p,1}\}$. The last bit returned by the invocation of the PRNG is assigned to the variable z_p. Note that the bit length of r_p and $k_{i,j}$ is the same, that is $|k_{i,j}| = |r_p| = \ell$. Also for $k_{i,j}$ we can introduce a different notation to refer

to the first $\ell/2$ bits ($k_{i,j,0}$) or the remaining $\ell/2$ bits ($k_{i,j,1}$) of the key. In the sequel of the paper, we assume, without loss of generality, that ℓ is a multiple of 6; further, for the sake of clarity and ease of notation, when it will be clear from the context to which key we are referring to, we will omit the indexes $\{i, j\}$ and will refer to $k_{i,j}$, $k_{i,j,0}$, and $k_{i,j,1}$ as k, k_0, and k_1 respectively.
The function $DPM : \{0,1\}^{\ell} \rightarrow \{0,1\}$ is defined as follows:

$$DPM(r_p) = P(M(S_1), \ldots, M(S_{\ell/3}))$$

where each S_i accounts for a triplet of bits of r_p as follows:

$$S_i =< r_p[3i-2], r_p[3i-1], r_p[3i] >, i = 1, \ldots, \ell/3$$

the function $M : \{0,1\}^3 \rightarrow \{0,1\}$ is the simple *majority* function, indicating whether its input has more 1s than 0s or viceversa:

$$M(b_1, b_2, b_3) = (b_1 \wedge b_2) \vee (b_1 \wedge b_3) \vee (b_2 \wedge b_3)$$

and $P : \{0,1\}^{\ell/3} \rightarrow \{0,1\}$ is the standard parity function; that is, given $T \in \{0,1\}^{\ell/3}$, it holds:

$$P(T) = \bigoplus_{i=1}^{\ell/3} T[i].$$

For each value α_p ($p \in [1, \ldots q]$) transmitted by the tag, the reader will perform a check for each of the elements in the list. Let us focus on the g^{th} element of the KDB; the following check will be performed:

(1) compute $r' = KDB_g.key \oplus \alpha_p$;
(2) check if $((DPM(r'_0) \neq V[p]) \wedge (DPM(r'_1) \neq V[p]))$.

If the test succeeds, the g^{th} element is removed from the list and the next element of the KDB, if any, is examined. However, if the test fails, the current element of KDB cannot be discarded. Indeed, if KDB_g is the pointer to the actual element associated with the tag (that is, if $KDB_g.key = k_{i,j}$), the test will fail by construction. On the other hand, if the test succeeds, the current element can be discarded from KDB since it definitely cannot be the one associated with the tag. Finally, for each α_p on the average one fourth of the elements of the list are eliminated. A thorough analysis of the lookup process can be found in Section 4.4.

4.3 Mutual authentication and session freshness

Once the reader has identified the tag —$k_{i,j}$ is the element pointed by KDB and returned by the identification protocol—, the reader recovers r_1 ($r_1 = \alpha_1 \oplus KDB.key$) and using that value it proceeds to the authentication of the tag. To that effect, the reader checks the freshness of the session by computing $z = h(k_{i,j}||n_j||r_1||k_{i,j})$ and verifying whether $z = \omega$. If the latter match succeeds, the reader has successfully authenticated the tag and verified the freshness of the session. Note that a simpler version of the protocol that provides only tag authentication based on the lookup process can be designed with a significant advantage of keeping the computational overhead of the tag at the lowest. Indeed, in that version of the protocol providing one-way authentication of the tag to the reader, the tag would not need to perform any hash computation for the purpose of authentication.

Global variables: $n\,;q\,;\ KDB$
Input : $<\alpha_1,\ldots,\alpha_q,V,w>$
Output : The elements left in the KDB.

1.1 /* Elements are organized in a list data structure */
1.2 /*$\hat{}KDB$ is a pointer to the first element of the list */
1.3 type node=record
1.4 $\quad key : tag\_key$
1.5 $\quad next :\hat{}node$
1.6 end
1.7 $aux = new(node)$
1.8 $aux.next =\hat{}KDB$
1.9 $a = 0$
1.10 **while** $a < q$ **do**
1.11 $\quad$ **while** $not(aux.next.next = null)$ **do**
1.12 $\quad\quad < r'_0, r'_1 >= \alpha_a \oplus aux.next.key$
1.13 $\quad\quad$ **if** $((\texttt{DPM}(r'_0) \neq V[a]) \wedge (\texttt{DPM}(r'_1) \neq V[a]))$ **then**
1.14 $\quad\quad\quad \texttt{remove}(aux.next)$
1.15 $\quad\quad$ **else**
1.16 $\quad\quad\quad aux.next = aux.next.next$
1.17 $\quad\quad$ **end**
1.18 $\quad$ **end**
1.19 $\quad a++;\ aux.next =\hat{}KDB$
1.20 **end**
1.21 **if** $\hat{}KDB.next = null$ **then**
1.22 $\quad \texttt{fail}$
1.23 **else**
1.24 $\quad$ **return** $\hat{}KDB$
1.25 **end**

Algorithm 1: Lookup

Indeed, for every $\epsilon > 0$, it is possible to select a value q such that the probability for the lookup process of accepting a bogus tag is below ϵ, as will be shown in Lemma 4.4.

We then turn to the authentication of the reader by the tag. Once the reader has successfully identified the tag, the reader can easily retrieve each of the q values r_p ($p \in [1,\ldots,q]$) generated by the tag. Indeed, from Equation 1, r_p can be computed by the reader as: $r_p = \alpha_p \oplus KDB.key$. Hence, the reader authenticates itself to the tag and assures the freshness of the session by sending the tag the value $h(KDB.key||r_1||KDB.key)$ that is, $h(k_{i,j}||r_1||k_{i,j})$. If this value matches with the one locally stored on the tag —computed by the tag when r_1 was generated—, then the tag authenticates the reader and it is also assured about the freshness of the session.

Note that in the event that the tag is not part of the reader's KDB, the identification will fail and the reader will not be able to reply with a well-formed authentication message. In this case the reader could reply with an error message but in order to prevent potential polling attacks through which an intruder would try to check if some tag is registered with a reader, the reader will generate instead a random string of $|h|$ bits and will send it back

to the tag.

4.4 Analysis

Server compromise: in case $reader_j$ is compromised the attacker can only access $k_{i,j}$, $i = 1 \ldots n$. Under the assumption that the hash function is one-way, it is impossible to derive k_i even having knowledge of $k_{i,j}$; hence the attacker cannot impersonate any of the n tags within any run of the protocol with any other reader. Further, note that the reader cannot impersonate any reader other than $reader_j$ either.

Identification protocol: in the sequel we analyse the termination and correctness of the protocol based on the properties of the lookup process.

Protocol termination: from Algorithm 1 it can be verified that the protocol terminates after a finite number of iterations in the two inner loops. As for its completion, it is straightforward to see that it requires, on the average, $4n$ steps —where each step has a cost proportional to the operations in lines 1.11-1.17, that is, bit-wise xor operations, a comparison, and a simple list manipulation operation. In what follows we sketch the proof that its completion takes, w.h.p., at most $O(n)$ steps.

LEMMA 4.1. *The protocol in Algorithm 1 terminates in $O(n)$ steps with high probability.*

PROOF SKETCH The proof consists of two steps. In the first step one can prove (using the Chernoff bounds) that w.h.p, for every value $\alpha_a = k_{i,j} \oplus r_a$ presented in input to the algorithm, this value allows to eliminate at least $1/8$ of the elements in the list —except the element corresponding to $k_{i,j}$ if any—, provided that there are $\sqrt{n}$ elements in the list. The elimination process fails ($S1$) with probability less than $1/n$.
The second step shows that with $\sqrt{n}(\log n)$ operations, the remaining $\sqrt{n}$ elements can be eliminated —except the element corresponding to $k_{i,j}$ if any. The elimination process ($S2$) fails with probability less than $1/n$ (this is a straightforward application of the occupancy problem). Hence, the algorithm does not terminate if step one or step two fail, that is:

$$Pr[S1 \vee S2] \leq Pr[S1] + Pr[S2] \leq 2Pr[S2] = \frac{2}{n}$$

□

Protocol correctness: the following lemma show that the proposed protocol will never reject a valid tag, while it could accept a bogus tag or return the wrong element of the KDB for a valid tag, with a probability ϵ, where ϵ can be decided at the design phase.

LEMMA 4.2. *For each valid input to the Lookup Process provided by a valid tag T_i, this tag will not be removed from the list KDB on all iterations of the Lookup Process.*

PROOF. By construction, a key $k_{i,j}$ corresponding to a valid input will never be discarded during any of the tests in the inner loop starting at line 1.10 of Algorithm 1; hence, T_i will not be removed from the list KDB. □

LEMMA 4.3. *Given $p, q \in_R \{0,1\}$, for $r' \in_R \{0,1\}^\ell$ it holds that:*

$$Pr[DPM(r'_0)p \neq q \wedge DPM(r'_1)(1-p) \neq q] = \frac{1}{4}$$

PROOF.

$$
\begin{aligned}
&Pr[DPM(r'_0)p \neq q \wedge DPM(r'_1)(1-p) \neq q] = \\
&1 - Pr[DPM(r'_0)p = q \vee DPM(r'_1)(1-p) = q] = \\
&1 - Pr[DPM(r'_0)p = q] + Pr[DPM(r'_1)(1-p) = q] - \\
&Pr[DPM(r'_0)p = q \wedge DPM(r'_1)(1-p) = q] = \\
&1 - Pr[DPM(r'_0)p = q] + Pr[DPM(r'_1)(1-p) = q] - \\
&Pr[DPM(r'_0)p = q] Pr[DPM(r'_1)(1-p) = q] = \\
&1 - \left(\frac{1}{2} + \frac{1}{2} - \frac{1}{2}\frac{1}{2}\right) = \frac{1}{4}
\end{aligned}
$$

□

LEMMA 4.4. *A randomly chosen input will be accepted by the Lookup Process with probability less than ϵ, where ϵ is chosen at the design phase.*

PROOF. Let $I =< \alpha_1, \ldots, \alpha_q, V, w >$ be a randomly chosen input for the Lookup Process. Let $X_i[u]$ be the random variable that takes on the value 1 if the verification of value α_i does not cause the removal of an element from the KDB list as in Algorithm 1 — that is, if the test $(DPM(\{\alpha_i \oplus KDB_u.key\}_0) \neq V[i] \wedge DPM(\{\alpha_i \oplus KDB_u.key\}_1) \neq V[i])$ fails—, and 0 otherwise. In order for I to be considered a valid input with respect to a single element (KDB_u) of the KDB, all q tests have to fail. This happens with probability: $Pr[E_u] = Pr[X_1[u] = 1 \wedge X_2[u] = 1 \wedge \ldots \wedge X_q[u] = 1]$. Since the X_i are i.i.d, we have that $Pr[E_u] = Pr[X_1[u] = 1]^q$ where, as shown in Lemma 4.3:

$$Pr[X_1[u] = 1] = \frac{3}{4}.$$

Since there are n elements in the KDB, if E_i is the event that the i element survives, the probability that at least one of them survives after q steps is:

$$Pr[E_1 \vee \ldots \vee E_n] \leq nPr[E_1] < n(3/4)^q = \left(\frac{\sqrt{3}}{2}\right)^{2q - \log n}$$

Now let r be the highest value such that $\epsilon \leq \left(\frac{\sqrt{3}}{2}\right)^r$. If we set $q = \lceil \frac{r - \log n}{2} \rceil$, the lemma holds. □

In Figure 2 we plot the function $\left(\frac{\sqrt{3}}{2}\right)^{2q - \log n}$. The x-axis, that varies in the range $[2^{14} \ldots 2^{16}]$, refers to the number of tags in the system, while the y-axis, in the range $[28, \ldots, 64]$ represents the value of q. As it can be seen from this chart, the acceptance rate of the protocol is quite efficient, in the sense that small values of q are sufficient to greatly reduce the number of false negatives whereby bogus input would be accepted as legitimate ones —this number goes practically to zero, as q increases.

Figure 3 depicts an experiment that corroborates the previous result. Using a simulator that implements Algorithm 1, we generated a KDB of 32,768 entries, and tested the number of active entries that were left in the KDB for an increasing size of the value q, that is the number of random α_i values sent by the tag to the reader. In particular, q varies in the range $[\log n, \ldots, 4 \log n]$, that is in the range $[15, \ldots, 60]$, using an incremental step

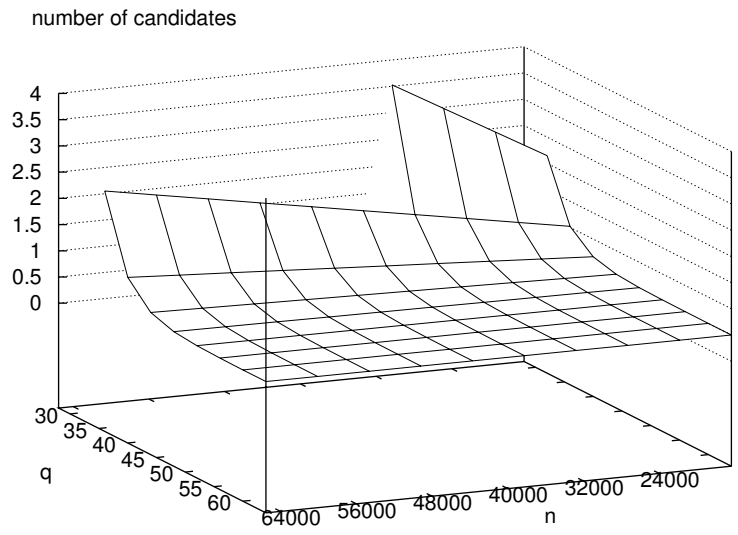

Fig. 2. Relationship between q, n, and the false acceptance rate.

of 1. The x-axis represents the value of q, while on the y-axis represents the number of active entries left in the KDB. To amortize statistical fluctuation, for each value of q, we performed 64 identification attempts, and we reported on the y-axis the number of active entries left in the KDB, averaged over these 64 protocol runs. As it can be derived from Figure 3, the number of active entries left in the KDB that result from the simulation is in accordance with the theoretical result of Lemma 4.4. In particular, the theoretical and the experimental curve tightly match for values of q below 30 (that is $2 \log n$), and the same qualitative behavior can be appreciated for increasing values of q, even if the experimental curve is more subject to the effect of the non null variance.

THEOREM 4.5. *On a valid input I generated by a legitimate tag (T_i) the Lookup Process will return only one element in the KDB list, corresponding to tag (T_i), with probability at least $1 - \epsilon$, where ϵ is chosen at the design phase.*

PROOF. This theorem can be reworded as: on a valid input, when the Lookup Process ends, the probability that the list KDB has just one element, and that this element is the one matching the input, is $1 - \epsilon$. The proof of this theorem follows from Lemma 4.2 and Lemma4.4. Based on Lemma 4.2 the probability that the list KDB has at least one entry after the last iteration of the Lookup Process is 1. The probability that the list KDB has more than one element is the same as the probability that a randomly chosen input is accepted, that is less than ϵ by Lemma 4.4. □

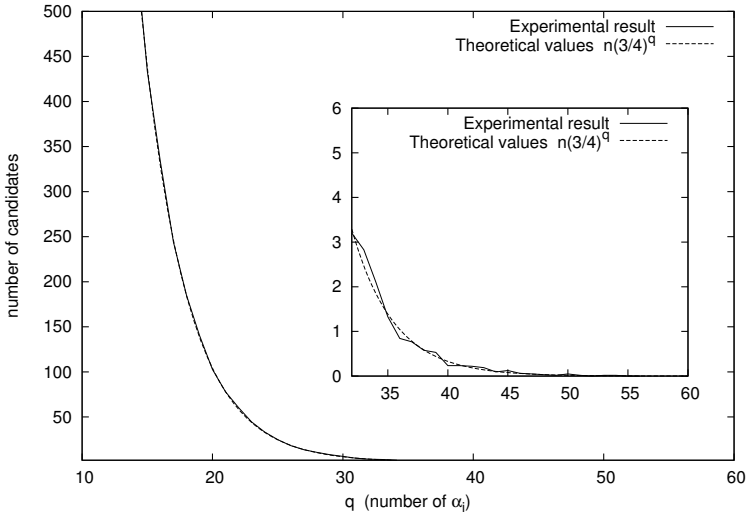

Fig. 3. Reader false acceptance rate: comparison of analytical and experimental results.

5. KEY SECRECY AND PRIVACY

5.1 Key secrecy

The proposed protocol cannot achieve perfect key secrecy, due to the fact that one bit of information on the key is leaked with every value α_i. Furthermore, linear algebra could be a powerful tool when used against a protocol that has only linear transformation. These observations have been used to mount an attack on the privacy of an earlier version of our protocol [Di Pietro and Molva 2007], as detailed in [van Deursen et al. 2008]. We first provide a brief summary of the attack method developed in this paper in order to further evaluate the current protocol with respect to this method.

In [van Deursen et al. 2008], the authors focus on the amount of information that is leaked through the DPM function and the resulting relation between α_i and $V[i]$. In particular, starting by a system composed of $\ell + 1$ tuples:

$$\begin{cases} k \oplus r_1, DPM(r_1) \\ \dots \\ k \oplus r_{\ell+1}, DPM(r_{\ell+1}) \end{cases} \tag{3}$$

the authors associate to each tuple the equation: $DPM(k \oplus \alpha_i) = DPM(r_i)$, where the unknown is just k; based on the previous equation, and with a few further linear algebra considerations, it is shown how to obtain a system of equations of the form:

$$Ax = v$$

where A is an $\ell \times \ell$ matrix that is non singular with probability $p > 0.2$. If the system has a solution, then x is the key k (with probability $(1/2)^{\ell/3}$). Apart from key secrecy, the privacy of the tag is also threatened based on this attack (refer to [van Deursen et al. 2008] for further details).

In the sequel we will consider the feasibility of this type of attack on the protocol sug-

gested in this paper. If $\mathcal{A}$ were able to obtain a system of tuples as in 3, then the security and privacy of the protocol should be analyzed with respect to the solution of the resulting system of equations. Initially we assume that the matrix A that would be obtained with the system of equations derived from our protocol is non singular, with a significant advantage for $\mathcal{A}$.

We first introduce an imaginary attack scenario in order to give the intuition behind the security evaluation that will be presented further.

Note that a tuple $\alpha_i, V[i]$ can be written as:

$$k \oplus r_i, DPM(r_{i,0})z_i \oplus DPM(r_{i,1})(1 - z_i)$$

The above tuple can be also written as the following system:

$$\begin{cases} k_0 \oplus r_{i,0}, DPM(r_{i,0})z_i \\ k_1 \oplus r_{i,1}, DPM(r_{i,1})(1 - z_i) \end{cases}$$

Note that each first element of the above tuples (that is, $k_0 \oplus r_{i,0}$ and $k_1 \oplus r_{i,1}$) now has a length of just $\ell/2$ bits. Assume that the value z_i is given (w.lo.g. $z_i = 1$): on one hand we can write a tuple in the form: $k_0 \oplus r_{i,0}, DPM(r_{i,0})$, on the other hand the other tuple will disclose no information at all.

If $\mathcal{A}$ is able to produce:

—an assignment to $(\ell/2 + 1)$ z_i variables that can allow it to produce $\ell/2 + 1$ tuples of the form $k_0 \oplus r_{i,0}, DPM(r_{i,0})$;

—an assignment to further $(\ell/2+1)$ z_i variables that can allow it to produce $\ell/2+1$ tuples of the form $k_1 \oplus r_{i,1}, DPM(r_{i,1})$,

then it is able to build the systems:

$$\begin{cases} k_0 \oplus r_{1,0}, DPM(r_{1,0}) \\ \dots \\ k_0 \oplus r_{\ell+1,0}, DPM(r_{\ell+1,0}) \end{cases} \quad (4) \qquad \begin{cases} k_1 \oplus r_{1,1}, DPM(r_{1,1}) \\ \dots \\ k_1 \oplus r_{\ell+1,1}, DPM(r_{\ell+1,1}) \end{cases} \quad (5)$$

However, a correct assignment for these $(\ell + 2)$ z_i variables has probability: $(\frac{1}{2})^{\ell/2+1}(\frac{1}{2})^{\ell/2+1} = (\frac{1}{2})^{\ell+2}$ to occur.

In the above attack, the poor performance of $\mathcal{A}$ is due to the fact that it did not leverage the information disclosed by the tag (that is, $V[i]$). Leveraging this information, that is also the maximum amount of information that can be extracted from a tuple, paves the way to the following attack.

THEOREM 5.1. *Analyzing $\ell+2$ tuples of the form $< k \oplus r_i, V[i] >$, $\mathcal{A}$ has probability at most $(3/4)^{\ell+2}$ to disclose secret key k.*

PROOF. Let us focus on the two tuples:

$$\begin{cases} k_0 \oplus r_{i,0}, DPM(r_{i,0})z_i \\ k_1 \oplus r_{i,1}, DPM(r_{i,1})(1 - z_i) \end{cases}$$

Note that, by construction, the disclosed value $V[i]$ is either $DPM(r_{i,0})$, or $DPM(r_{i,1})$ —it depends on the variable z_i. The strategy played by $\mathcal{A}$ follows: it first randomly decides whether $z_i = 0$ or $z_i = 1$, and then assigns the value $V[i]$ consequently. Note that if $z_i = 1$ then $\mathcal{A}$ will add one tuple to the system in 4; if $z_i = 0$, then $\mathcal{A}$ will add one tuple to the

system in 5. Without losing of generality, let us assume that $\mathcal{A}$ chooses $z_i = 0$, that is it assigns $V[i]$ as the second element of the first tuple ($DPM(r_{i,0}) = V[i]$) and uses this tuple to populate the system in 4.

Note that the probability ($E1$) for $k_0 \oplus r_{i,0}$ to be assigned the exact result of $DPM(r_{i,0})$ is (3/4). Indeed, let A be the event that $\mathcal{A}$ correctly guessed the assignment to z_i, and denote with $\overline{A}$ the complementary event.

$$Pr[E1] = Pr[E1 \wedge (A \vee \overline{A})] = Pr[E1 \wedge A] + Pr[E1 \wedge \overline{A}] =$$
$$Pr[E1|A]Pr[A] + Pr[E1|\overline{A}]Pr[\overline{A}] = 1\frac{1}{2} + \frac{1}{2}\frac{1}{2} = \frac{3}{4}$$

That is, with probability $3/4$, $\mathcal{A}$ is able to add a tuple either to the system in 4 or to the system in 5. To complete the first system, $\ell/2 + 1$ tuples are needed, and the same holds for the second system. However, tuples are independent, hence the probability for $\mathcal{A}$ to find out a correct assignment for all of the $\ell + 2$ tuples is given by: $(3/4)^{\ell+2}$.

□

We can leverage the above theorem to find out an appropriate key bit length (ℓ). Indeed, if we set $\epsilon = 2^{-80}$, and we require the above probability to be below ϵ, we need to verify that: $(3/4)^{\ell+2} \leq \epsilon$; to satisfy the further requirement that ℓ has to be a multiple of 6, we just need to set $\ell = 192$.

5.2 Privacy

We first introduce a model that is widely adopted for privacy evaluation. We then proceed to the evaluation of our protocol based on this model.

5.2.1 *Privacy model.* Juels and Weis [Juels and Weis 2007] introduced a privacy model that provides one of the most comprehensive setting for privacy evaluation of RFID protocols.

In this model the system consists of a reader $\mathcal{R}$ and a set of n tags $T_1, \ldots, T_n$. Each party is a probabilistic interactive Turing machine with an independent source of randomness and unlimited internal storage. Tags and readers are modeled as "ideal functionalities", as in [Canetti 2001]. Functionalities may receive messages, and may respond with messages of their own through their interfaces.

Tag functionalities

Each tag functionality T_i stores an internal secret key and a session identifier sid. A tag can be assigned a new key via a SETKEY message. A tag responds to a SETKEY message by disposing its current key. The caller may then send an arbitrary new key to replace the prior key. A tag SID can be set to a new value sid via the message (TAGINIT,sid). TAGINIT messages delete information associated with an existing sid. In other words, a tag may be involved in only one protocol session at a time. A tag may respond to a protocol message or challenge, denoted c_j , with a response r_j.

Reader functionalities

The reader $\mathcal{R}$ is initialized with private key material. For the purposes of this model, this key material is immutable and internal to the reader. Tag data may be thought of as residing in a back-end database containing records of the tags "owned" by a particular reader. The reader functionality initializes a new session upon receipt of a message of the form READERINIT. When receiving a READERINIT message, $\mathcal{R}$ generates a fresh session identifier, sid, and the first challenge of an interactive challenge-response protocol,

c_0. For each READERINIT received, the reader creates a new internal entry of the form (sid, "open", c_0). Any responses containing sid are appended to that entry, as well as subsequent challenges, or any other auxiliary data. This entry is marked as "closed" and becomes read-only when the reader ultimately accepts or rejects a session.

For further details on *functionalities interaction* and *parametrization of the adversary*, the reader should refer to [Juels and Weis 2007].

Privacy experiment and definition A privacy experiment for an RFID system is denoted by $Exp^{priv}_{\mathcal{A},\mathcal{S}}[\ell, n, r, s, t]$. Here, $S = (GEN, R, \{T_i\})$, where $\{T_i\}$ contains n tags. Let ℓ be a security parameter. Adversary $\mathcal{A}$ with parameters r, s, and t is denoted by $\mathcal{A}[r, s, t]$, where r, s and t are respective parameters for reader initialization, computation steps, and tag initialization. Figure 4 provides a detailed description of the privacy experiment.

Experiment $\mathsf{Exp}^{\mathcal{A},S}_{priv}[\ell, n, r, s, t]$:

Setup:
1. $\mathsf{GEN}(1^\ell) \rightarrow (k_1, \ldots, k_n)$.
2. Initialize $\mathcal{R}$ with $(k_1, \ldots, k_n)$.
3. Set each the key of each T_i to k_i with a SETKEY call.

Phase 1 (Learning):
4. $\mathcal{A}$ may do the following in any interleaved order:
   (a) Make READERINIT calls, not exceeding r total calls.
   (b) Make TAGINIT calls, not exceeding t total calls.
   (c) Make arbitrary SETKEY calls to any $(n-2)$ tags.
   (d) Communicate and compute, not exceeding s total steps.

Phase 2 (Challenge):
5. A selects two tags T_i and T_j to which it did not send SETKEY messages.
6. Let $T^*_0 = T_i$ and $T^*_1 = Tj$ and remove both of these from the current tag set.
7. Let $b \in_R \{0, 1\}$ and provide $\mathcal{A}$ access to T^*_b .
8. A may do the following in any interleaved order:
   —Make READERINIT calls, not exceeding r total calls.
   —Make TAGINIT calls, not exceeding t total calls.
   —Make arbitrary SETKEY calls to any tag in the current tag set except T^*_b.
   —Communicate and compute, not exceeding s total steps.
9. $\mathcal{A}$ outputs a guess bit b'.

Exp succeeds if $b = b'$.

Fig. 4. Privacy experiment in [Juels and Weis 2007]

A protocol run within an RFID system $S = (\mathsf{GEN}, \mathcal{R}, \{T_i\})$ is defined to be private if no adversary $\mathcal{A}[r, s, t]$ has a non-negligible advantage in successfully guessing b in the

experiment in Figure 4. This intuition is captured in the following definition, where $poly(\ell)$ represents any polynomial function of ℓ.

Definition 5.2. **RFID (r, s, t)-Privacy.** A protocol initiated by $\mathcal{R}$ in an RFID system $S = (\mathsf{GEN}, \mathcal{R}, \{T_1, \ldots, T_n\})$ with security parameter ℓ is $(r, s, t) - private$ if:

$$\forall A[r,s,t] Pr\left[Exp^{priv}_{\mathcal{A},S}[\ell, n, r, s, t]\ guesses\ b\right] \leq \frac{1}{2} + \frac{1}{poly(\ell)}.$$

5.2.2 *Privacy analysis.* In this section we evaluate the privacy of our scheme using the privacy model introduced in the previous paragraph. To accommodate that framework to our scheme, note that the keys of the tag are completely independent of each other (under the assumption that hash functions cannot be inverted) thus the corruption of one tag does not affect the security of the rest of the system. Therefore, it is useless for adversary $\mathcal{A}$ to use the SETKEY procedure to change the key of tags. It is also useless for $\mathcal{A}$ to examine any other tags than the ones under its direct control, that is T_0^* and T_1^*. In our scheme, READERINIT is a simple triggering message, so there is no need at all for $\mathcal{A}$ to execute it. Therefore, an instance of the privacy experiment as suggested by the model can be set up using our protocol as in Figure 5.

In light of the randomization technique used in the protocol, in order to mount an attack on our scheme, $\mathcal{A}$ has to find a collision on X_b and one of the two sets: X_0, X_1. A collision is defined as follows:

Definition 5.3. *Collision.* Two tuples $< \alpha_i, V[i] >$ and $< \alpha_j, V[i] >$ collide if $\alpha_i = \alpha_j$.

A collision between the set of tuples obtained querying the two tags T_0^*, T_1^* and the set of tuples obtained querying T_b, can be leveraged to compromise privacy. For instance, assume that there is a collision between a tuple from X_0 and a tuple from X_b; further, assume that $T_0^* = T_b$. Then the probability that $V[i] = V[j]$ is equal to 3/4. However, if there is a collision and $T_0^* \neq T_b$, then the probability that $V[i] = V[j]$ is equal to 1/2.
The above observations could allow $\mathcal{A}$ to differentiate between the two tags, as proved in the sequel of this section.

LEMMA 5.4. *Given the two tags T_0^* and T_b^*, let $E1$ be the event that $T_0^* = T_b$ (that is, $b = 0$), $E2$ be the event that $T_0^* \neq T_b$, and let "coll" be the event that there is a tuple from T_0^* and one tuple from T_b that collide for some index i, j respectively. Further, assume that $V[i] = V[j]$. Then:*

$$Pr[E1|(V[i] = V[j]) \wedge coll] = \frac{3}{5} \tag{6}$$

PROOF.

$$Pr[E1|(V[i] = V[j]) \wedge coll] = \frac{Pr[E1 \wedge (V[i] = V[j]) \wedge cool]}{Pr[(V[i] = V[j]) \wedge coll]} =$$
$$\frac{Pr[(V[i] = V[j])|E1 \wedge coll] Pr[E1 \wedge coll]}{Pr[(V[i] = V[j]) \wedge coll]} =$$
$$\frac{Pr[(V[i] = V[j])|E1 \wedge coll] Pr[E1 \wedge] Pr[coll]}{Pr[(V[i] = V[j]) \wedge coll]} =$$
$$\frac{(3/4)(1/2) Pr[coll]}{Pr[(V[i] = V[j]) \wedge coll]} = \frac{(3/8) Pr[coll]}{Pr[(V[i] = V[j]) \wedge coll]}$$

Experiment $\mathsf{Exp}^{\mathcal{A},\mathcal{S}}_{priv}[\ell, n, x_0 + x_1 + x_b]$:

Setup

(1) Generate keys $(k_1, \ldots, k_n)$ uniquely and randomly with Gen;
(2) Initialize $\mathcal{R}$ with keys $(k_1, \ldots, k_n)$;
(3) Assign key k_i to T_i with a SETKEY call;

Learning

(4) Let $\mathcal{A}$ perform x_0 TAGINIT calls with T_0^* and let it record the received packets into the set X_0;
(5) Let $\mathcal{A}$ perform x_B TAGINIT calls with T_1^* and let it record the received packets into the set X_1;

Challenge

(6) Let $T_b \leftarrow_r \{T_0^*, T_1^*\}$
(7) Let $\mathcal{A}$ perform x_b TAGINIT calls with T_b and let it record the received packets into the set X_b:
(8) Let $\mathcal{A}$ perform calculations on the recorded packets in order to make an educated guess whether T_b is either T_0^* or T_1^*;

Exp succeeds if $\mathcal{A}$ can make and educated guess over T_b.

Fig. 5. Simplified version of the privacy experiment proposed in [Juels and Weis 2007], adapted to our protocol

We now need to assess $Pr[(V[i] = V[j]) \wedge coll]$:

$$
\begin{aligned}
&Pr[(V[i] = V[j]) \wedge coll] = \\
&Pr[(V[i] = V[j]) \wedge coll \wedge (E1 \vee E2)] = \\
&Pr[(V[i] = V[j]) \wedge coll \wedge E1] + Pr[(V[i] = V[j]) \wedge coll \wedge E2] = \\
&Pr[(V[i] = V[j])|coll \wedge E1]Pr[coll \wedge E1] + \\
&Pr[(V[i] = V[j])|coll \wedge E2]Pr[coll \wedge E2] = \\
&Pr[(V[i] = V[j])|coll \wedge E1]Pr[coll]Pr[E1] + \\
&Pr[(V[i] = V[j])|coll \wedge E2]Pr[coll]Pr[E2] = \\
&(3/4)Pr[coll](1/2) + (1/2)Pr[coll](1/2) = (5/8)Pr[coll]
\end{aligned}
$$

Hence, we have that:

$$
Pr[E1|(V[i] = V[j]) \wedge coll] = \frac{(3/8)Pr[coll]}{Pr[(V[i] = V[j]) \wedge coll]} = \frac{(3/8)Pr[coll]}{(5/8)Pr[coll]} = \frac{3}{5}
$$

□

However, once a collision is obtained, it is possible that the inequality $V[i] \neq V[j]$ holds as well. Based on the same reasoning as with Lemma 5.4, we get the following lemma:

LEMMA 5.5. *Let T_0 and T_b be two tags, $E2$ the event that $T_0 \neq T_b$, $E1$ the event that $T_0 = T_b$, and "coll" the event that there is a tuple from T_0 and one tuple from T_b that collide for some index i, j respectively. Further, assume that $V[i] \neq V[j]$. Then:*

$$Pr[E2|(V[i] \neq V[j]) \wedge coll] = \frac{2}{3} \tag{7}$$

PROOF.

$$Pr[E2|(V[i] \neq V[j]) \wedge coll] = \frac{Pr[E2 \wedge (V[i] \neq V[j]) \wedge cool]}{Pr[(V[i] \neq V[j]) \wedge coll]} =$$
$$\frac{Pr[(V[i] \neq V[j])|E2 \wedge coll]Pr[E2 \wedge coll]}{Pr[(V[i] \neq V[j]) \wedge coll]} =$$
$$\frac{Pr[(V[i] \neq V[j])|E2 \wedge coll]Pr[E2\wedge]Pr[coll]}{Pr[(V[i] \neq V[j]) \wedge coll]} =$$
$$\frac{(1/2)(1/2)Pr[coll]}{Pr[(V[i] \neq V[j]) \wedge coll]} = \frac{(1/4)Pr[coll]}{Pr[(V[i] \neq V[j]) \wedge coll]}$$

We have to compute $Pr[(V[i] \neq V[j]) \wedge coll]$, that is:

$$Pr[(V[i] \neq V[j]) \wedge coll] = Pr[(V[i] \neq V[j]) \wedge coll \wedge (E1 \vee E2)] =$$
$$Pr[(V[i] \neq V[j]) \wedge coll \wedge E1] + Pr[(V[i] \neq V[j]) \wedge coll \wedge E2] =$$
$$Pr[(V[i] \neq V[j])|coll \wedge E1]Pr[coll \wedge E1]+$$
$$Pr[(V[i] \neq V[j])|coll \wedge E2]Pr[coll \wedge E2] =$$
$$Pr[(V[i] \neq V[j])|coll \wedge E1]Pr[coll]Pr[E1]+$$
$$Pr[(V[i] \neq V[j])|coll \wedge E2]Pr[coll]Pr[E2] =$$
$$(1/4)Pr[coll](1/2) + (1/2)Pr[coll](1/2) = (3/8)Pr[coll]$$

Hence:

$$Pr[E2|(V[i] \neq V[j]) \wedge coll] = \frac{(1/4)Pr[coll]}{Pr[(V[i] \neq V[j]) \wedge coll]} = \frac{(1/4)Pr[coll]}{(3/8)Pr[coll]} = \frac{2}{3}$$

□

Lemma 5.4 and Lemma 5.5 show that, when a collision is found, $\mathcal{A}$ can decide with non negligible probability if either $T_b = T_0^*$ or $T_b = T_1^*$, as the following lemma shows.

LEMMA 5.6. *If there is a collision among the tuples in X_b and either the tuples in X_0 or the tuples in X_1, then $\mathcal{A}$ can guess the value of b with probability at least $(1/2 + 1/6)$.*

PROOF. Let us assume w.lo.g that there is a collision between T_b and T_0^* (for index i, j respectively) and, on one hand, we have that $V[i] = V[j]$. Then by Lemma 5.4: $Pr[T_0 = T_b] = 3/4 = 1/2 + 1/4$. On the other hand, if $V[i] \neq V[j]$, then by Lemma 5.5: $Pr[T_0 \neq T_b] = 2/3$. Since b is either 0 or 1, it follows that $Pr[T_1 = T_b] = 2/3 = 1/2 + 1/6$. □

According to the privacy model, $\mathcal{A}$ can perform x_0 queries to T_0^* and x_1 queries to T_1^*. When finally querying x_b times tag T_b, as a direct consequence of the above lemma, $\mathcal{A}$ needs just to find out a collision between the tuples in X_b and the tuples collected in X_0, X_1 to succeed in the Exp described in Figure 5, that is violating the privacy of our protocol.

In the remainder of the analysis, we are interested in finding an assignment to the security parameter ℓ such that the probability of finding a collision in the described privacy model is negligible. In particular, let us define X'_d the set built from set X_d as follows: if $< \alpha_i, V[i] > \in X_d$, then $\alpha_i \in X'_d$ ($d \in \{0, 1, b\}$). To capture the probability that a collision occurs, we have to compute:

$$Pr[c_{x_b}] = Pr[(X'_0 \vee X'_1) \wedge X'_b]$$

LEMMA 5.7. *Let $(Pr\,[c_{x_b}])$ be the probability of having a collision between the x_0+x_1 tuples provided by the tags T_0^* and T_1^*, where x_b represents tuples provided by the tag T_b (obtained by $\mathcal{A}$ querying tag T_0^*, T_0^*, and T_b x_0, x_1, and x_b times respectively). Let $g = x_0 + x_1 + x_b$, the following inequality holds: $Pr[c_{x_b}] \leq 1 - exp\left(-\frac{g^2}{2^{\ell+1}}\right)$.*

PROOF. Assume that x_0, x_1 tuples have been collected querying tag T_0^*, T_1^* respectively. Collecting the tuples obtained querying tag T_b exactly x_b times, $\mathcal{A}$ will *not* obtain a collision with probability:

$$Pr[\overline{c}_{x_b}] = \left(1 - \frac{(x_0 + x_1)}{2^\ell}\right)^{x_b} \geq exp\left(-\frac{2(x_0 + x_1)x_b}{2^\ell}\right) \tag{8}$$

Note that the above probability decreases if we assume that there are no collisions among the two set X'_0 and X'_1; it further decreases if we assume there are no collisions among the elements in X'_b. Then we give $\mathcal{A}$ the advantage of considering: $X'_0 \cap X'_1 = \emptyset$ and $\alpha_e \neq \alpha_d, \forall \alpha_e, \alpha_d \in X'_b, e \neq d$.

One goal of $\mathcal{A}$ is to minimize the overall number of queries ($g = x_0 + x_1 + x_b$), while minimizing the probability in Equation 8. Hence, for a given budget of queries (g), simple math can confirm that the above equation is minimized for $(x_0 + x_1) = x_b = g/2$.

Hence we have: $Pr[\overline{c}_{x_b}] \geq exp\left(-\frac{g^2}{2^{\ell+1}}\right)$, therefore:

$$Pr[c_{x_b}] \leq 1 - exp\left(-\frac{g^2}{2^{\ell+1}}\right) \tag{9}$$

□

THEOREM 5.8. *Given $f > 0$, for $\mathcal{A}$ to win the privacy experiment $\mathsf{Exp}_{priv}^{\mathcal{A},\mathcal{S}}[\ell, n, x_0 + x_1 + x_b]$ with probability greater than 2^{-f}, it is required to issue at least $g \geq \sqrt{2^{\ell+1-f}}$ queries.*

PROOF. We have shown that, as a consequence of Lemma 5.6, if there is a collision in the privacy experiment $\mathsf{Exp}_{priv}^{\mathcal{A},\mathcal{S}}[\ell, n, x_0 + x_1 + x_b]$, $\mathcal{A}$ wins. The upper-bound ($\epsilon = 2^{-f}$) for the probability of having a collision can be computed as follows:

$$Pr[\overline{c}_{x_b}] \leq 1 - exp\left(-\frac{g^2}{2^{\ell+1}}\right) < \epsilon$$

that is: $exp\left(-\frac{g^2}{2^{\ell+1}}\right) \geq (1-\epsilon)$, hence:

$$g \leq \sqrt{2^{\ell+1}(-\ln(1-\epsilon))}$$

We can simplify the above inequality by noticing that, for $0 < \epsilon < 1$, we have $-\ln(1-\epsilon) > \epsilon$. Note that this gives an additional advantage to the adversary. Rewriting the above inequality, we obtain: $g < \sqrt{2^{\ell+1}\epsilon}$, that is:

$$g < \sqrt{2^{\ell+1-f}} \tag{10}$$

Therefore, if $\mathcal{A}$ does not perform at least $g = \sqrt{2^{\ell+1-f}}$ queries to the tags T_0^*, T_1^*, and T_b, its probability of obtaining a collision is below $\epsilon = 2^{-f}$. □

Finally, note that we can leverage the above theorem to find out an appropriate key bit length (ℓ). Indeed, if we set $\epsilon = 2^{-80}$, and we require the above probability to be below ϵ, we need to verify that: $\sqrt{2^{\ell+1-f}} \leq \epsilon$. The assignment $\ell = 240$ is the minimum value of ℓ that satisfies the above constraints, as well as the initial condition that ℓ has to be a multiple of 6.

6. FURTHER SECURITY PROPERTIES AND OVERHEAD

6.1 Mutual authentication

By Lemma 4.4 a bogus reply message generated by an attacker can be accepted with probability less than ϵ only. Further, such a scenario can be made practically impossible by setting appropriate values for q in order to keep ϵ below a negligible value. Besides, even a successful attempt that achieves acceptance of the random input by the Lookup Process cannot compromise authentication, since the attacker would not be able to complete the remainder of the protocol flows without the knowledge of the legitimate tag's secret key. The choice of the particular expression $h(k_{i,j}||n_j||r_1||k_{i,j})$ combining the key and the nonces as part of the authentication scheme is justified in [Menezes et al. 1996].

As for reply attacks, the freshness of a session is granted by binding the messages exchanged to the random values generated by both the tag (r_1), and the reader (n_j), as in Figure 1.

6.2 DoS resilience

Opposed to other approaches [Tsudik 2006; Avoine et al. 2005; Avoine and Oechslin 2005], our protocol is stateless in that there is no need to store any state information such as timestamps or counter values beyond the execution of each protocol instance. The only piece of information that the tag has to persistently keep in memory is the key k_i. Hence, even if a tag is triggered t consecutive times by an attacker attempting to impersonate a legitimate reader, if the next reading is performed by a legitimate reader, the tag will be correctly identified since the state has not been modified. Statelessness thus bestows our protocol with an inherent countermeasure against denial of service attacks.

Furthermore, as a side advantage of statelessness, our protocol allows a tag to be read a practically unbounded number of times by a legitimate reader.

6.3 Overhead

The main computational overhead on the tag is due to the generation of the q values r_p. These values could be computed via a PRNG. Similarly to what proposed in [Tsudik 2006],

in practice it can be resolved as an iterated keyed hash (e.g., HMAC) computed on some cheap, weak pseudo random source (for instance circuitry noise) and keyed on $k_{i,j}$. The solutions in [Feldhofer et al. 2005; Pramstaller et al. 2006], matching the tight hardware constraints of RFID, could be adopted to serve as hash function. Further, the tag requires one hash functions to generate $k_{i,j}$ and $q\ell$ more "xor" ($\oplus$) due to the invocation of the function $P(\circ)$. Note that the cost of all these "xor", operations can be considered negligible.

As for the communications overhead, the tag is required to send q messages of ℓ bits (α_p), plus q bits (the bit vector V), and the result of the hash function, that can be considered of 160 bits. We focus on the main source of overhead, that is the q messages. From Lemma 4.4, a practical value for q could be $3 \log n$; in this way the reader lookup protocol will return, when triggered by a legitimate query, more than one element with probability $\left(\frac{\sqrt{3}}{2}\right)^{5 \log n}$ only. As discussed before note that, in case the lookup protocol returns a bogus element, the authentication protocol will reject that element. Note that a new round of the protocol could be invoked in case of such a failure. What is more important, in case of a protocol re-run due to the fact that in the KDB there are too many elements left, is that the new values α_i can be matched against the elements left in the KDB. In other words, the computations performed by the reader in the previous run will be leveraged to pursue identification.

The main computational overhead sustained by the reader is the tag identification; this operation requires in the worst case no more than just $O(n)$ steps, where a step consists of bitwise operations, comparisons, and simple list manipulation operations. As for the number of messages, the reader just sends three values for a total of $(h + m + n_o)$ bits where h is the size in bit of the output of the hash function, m is the number of bits required to identify a reader, and n_o is the size in bit of the nonce.

Last, one should note one caveat: the proposed protocol is particularly sensitive to the value n, as shown in Lemma 4.4, where n is the total number of tags the system is composed of. Indeed, the protocol requires to devise at design time an upper bound n' on the number of tags. We believe this is not a critical limitation, since this upper bound will impact on the protocol requiring just $c \log n'$ messages, where c is a small constant as seen before and computing the logarithm over n' will attenuate the overhead of considering an upper bound. Furthermore, the value n' does not affect the storage requirements of the reader since the reader is only required to store the keys of the n tags that are actually deployed.

6.4 Protocol comparison

A concise comparison of the properties provided by our protocol with regard to a few reference protocols is given in Table I. Note that our protocol is the only one that fulfils all the security properties, while providing no false positive.

7. CONCLUDING REMARKS

A first contribution of this paper is to have relaxed the assumption that servers cannot be compromised and to have provided a solution that limits the impact of server compromise. In particular, thanks to the confinement technique we provide, the compromise of a server has no impact on other servers, such as rekeying or update of critical data, or on the privacy of tags since the secret database of each server is made server-dependent. We further pro-

Table I. Comparison of our proposal with some protocols in Section 2.

Protocol	Properties					
	Privacy	Mutual auth.	DoS resilience	reply attack res.	false positive	false negative
Our [this paper]	Yes	Yes	Yes	Yes	No	Yes
OSK/OA [Avoine et al. 2005]	Yes	Yes	No	Yes	Yes	No
CR/MW [Molnar and Wagner 2004]	No	Yes	Yes	Yes	No	Yes
Ya-Trap [Tsudik 2006]	Yes	No	No	Yes	No	No
HB+ [Juels and Weis 2005]	Yes	No	No	Yes	Yes	Yes

pose a probabilistic mechanism that preserves key secrecy and tag privacy, while allowing mutual authentication between server and tag. This mechanism is also resilient to DoS and replay attacks. Furthermore, this mechanism only requires $O(n)$ bitwise operations and comparisons on the data base of keys stored in a server, hence speeding up the search process and reaching a theoretical lower bound for the number of access to the data base of keys to have privacy preserved. In this sense, our identification protocol is optimal. Another property that our identification protocol enjoys is that, unlike other probabilistic protocols, a legitimate tag cannot be rejected by the reader. Moreover, the tag just requires to store a single key and the capability to run a PRNG and a hash function. Note that on one extreme, the computational requirements on the tag can be kept minimal by eliminating the need for the hash function, resulting in a lightweight protocol that achieves only the authentication of the tag by the reader. Finally, the information confinement technique and the tag identification protocol could be independently incorporated into existing solutions.

We believe that the new identification protocol provided could foster further research in the area and that the techniques presented in this paper could be adopted in various different settings as well.

REFERENCES

AVOINE, G., DYSLI, E., AND OECHSLIN, P. 2005. Reducing time complexity in RFID systems. In *Selected Areas in Cryptography – SAC 2005*, B. Preneel and S. Tavares, Eds. Lecture Notes in Computer Science, vol. 3897. Springer-Verlag, Kingston, Canada, 291–306.

AVOINE, G. AND OECHSLIN, P. 2005. A scalable and provably secure hash based RFID protocol. In *International Workshop on Pervasive Computing and Communication Security – PerSec 2005*. IEEE, IEEE Computer Society Press, Kauai Island, Hawaii, USA, 110–114.

BRINGER, J., CHABANNE, H., AND EMMANUELLE, D. 2006. HB^{++}: a lightweight authentication protocol secure against some attacks. In *IEEE International Conference on Pervasive Services, Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing – SecPerU 2006*. IEEE, IEEE Computer Society Press, Lyon, France.

CANETTI, R. 2001. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS '01: Proceedings of the 42nd IEEE symposium on Foundations of Computer Science*. IEEE Computer Society, Washington, DC, USA, 136–145.

CASTELLUCCIA, C. AND SOOS, M. 2007. Secret shuffling: A novel approach to RFID private identification. In *Conference on RFID Security (RFIDSec07)*.

CONTI, M., DI PIETRO, R., MANCINI, L. V., AND SPOGNARDI, A. 2007. RIPP-FS: an RFID identification, privacy preserving protocol with forward secrecy. In *Proceedings of the 3rd IEEE International Workshop on Pervasive Computing and Communication Security*. IEEE Press, 229–234.

DAMGÅRD, I. AND PEDERSEN, M. Ø. 2008. RFID security: Tradeoffs between security and efficiency. In *CT-RSA*. Lecture Notes in Computer Science, vol. 4964. Springer, 318–332.

DI PIETRO, R. AND MOLVA, R. 2007. Information confinement, privacy, and security in RFID systems. In

12th European Symposium On Research In Computer Security (ESORICS 2007). Lecture Notes in Computer Science, vol. 4734. Springer, 187–202.

FELDHOFER, M., WOLKERSTORFER, J., AND RIJMEN, V. October 2005. AES implementation on a grain of sand. *IEE Proceedings - Information Security 152,* 1, 13–20.

GILBERT, H., ROBSHAW, M., AND SIBERT, H. 2005. An active attack against HB+ - a provably secure lightweight authentication protocol. Cryptology ePrint Archive, Report 2005/237.

GILBERT, H., ROBSHAW, M. J. B., AND SEURIN, Y. 2008. $HB\sharp$: Increasing the security and efficiency of $HB+$. In *Advances in Cryptology - EUROCRYPT 2008*. Lecture Notes in Computer Science, vol. 4965. Springer, Instanbul, Turkey, 361–378.

HELLMAN, M. 1980. A cryptanalytic time-memory tradeoff. *IEEE Transactions on Information Theory 26*, 401–406.

HOPPER, N. J. AND BLUM, M. 2001. Secure human identification protocols. In *Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT '01)*. Lecture Notes in Computer Science, vol. 2248. Springer-Verlag, London, UK, 52–66.

JUELS, A. 2006. RFID security and privacy: A research survey. *IEEE Journal on Selected Areas in Communications 24,* 2, 381–394.

JUELS, A. AND WEIS, S. 2005. Authenticating pervasive devices with human protocols. In *Advances in Cryptology – CRYPTO'05*. Lecture Notes in Computer Science, vol. 3126. Springer-Verlag, Santa Barbara, California, USA, 293–308.

JUELS, A. AND WEIS, S. A. 2007. Defining strong privacy for RFID. In *PERCOMW '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops*. IEEE Computer Society, Washington, DC, USA, 342–347.

MENEZES, A. J., VAN OORSCHOT, P. C., AND VANSTONE, S. A. 1996. *Handbook of applied cryptography*. CRC Press, Chapter 9 - Hash Functions and Data Integrity.

MOLNAR, D. AND WAGNER, D. 2004. Privacy and security in library RFID: issues, practices, and architectures. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*. ACM Press, New York, NY, USA, 210–219.

OUAFI, K. AND PHAN, R. C.-W. 2008. Privacy of recent RFID authentication protocols. In *Information Security Practice and Experience, 4th International Conference (ISPEC 2008)*. Lecture Notes in Computer Science, vol. 4991. Springer, 263–277.

PIRAMUTHU, S. 2006. HB and related lightweight authentication protocols for secure RFID tag/reader authentication. In *Collaborative Electronic Commerce Technology and Research – CollECTeR 2006*. Basel, Switzerland.

PRAMSTALLER, N., RECHBERGER, C., AND RIJMEN, V. 2006. A compact fpga implementation of the hash function whirlpool. In *FPGA '06: Proceedings of the 2006 ACM/SIGDA 14th international symposium on Field programmable gate arrays*. ACM Press, New York, NY, USA, 159–166.

RHEE, K., KWAK, J., KIM, S., AND WON, D. 2005. Challenge-response based RFID authentication protocol for distributed database environment. In *International Conference on Security in Pervasive Computing – SPC 2005*, D. Hutter and M. Ullmann, Eds. LNCS, vol. 3450. Springer-Verlag, Boppard, Germany, 70–84.

TSUDIK, G. 2006. Ya-trap: Yet another trivial RFID authentication protocol. In *PERCOMW '06: Proceedings of the 4th annual IEEE international conference on Pervasive Computing and Communications Workshops*. IEEE Computer Society, Washington, DC, USA, 640–643.

VAN DEURSEN, T., MAUW, S., AND RADOMIROVIĆ, S. 2008. Untraceability of RFID protocols. In *Information Security Theory and Practices. Smart Devices, Convergence and Next Generation Networks*. Lecture Notes in Computer Science, vol. 5019. Springer, Sevilla, Spain, 1–15.

VAUDENAY, S. 2007. On Privacy Models for RFID. In *Advances in Cryptology - Asiacrypt 2007*. Lecture Notes in Computer Science, vol. 4833. Springer, 68–87.