Information confinement, privacy, and security in RFID systems

Roberto Di Pietro[1] and Refik Molva[2]

[1] Dipartimento di Matematica
Università di Roma Tre
L.go S. Murialdo, 1 - 00149 Roma, Italy
dipietro@mat.uniroma3.it
[2] Institut Eurécom
2229, route des crêtes
Sophia-Antipolis, France
refik.molva@eurecom.fr

Abstract. This paper describes an identification and authentication protocol for RFID tags with two contributions aiming at enhancing the security and privacy of RFID based systems. First, we assume that some of the servers storing the information related to the tags can be compromised. In order to protect the tags from potentially malicious servers, we devise a technique that makes RFID identification server-dependent, providing a different unique secret key shared by each pair of tag and server. The proposed solution requires the tag to store only a single secret key, regardless of the number of servers, thus fitting the constraints on tag's memory. Second, we provide a probabilistic tag identification scheme that requires the server to perform simple bitwise operations, thus speeding up the identification process. The proposed tag identification protocol assures privacy, mutual authentication and resilience to both DoS and replay attacks. Finally, each of the two schemes described in this paper can be independently implemented to enhance the security of existing RFID protocols.

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 [1].

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 [1,2]. 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 like in the case of electronic passports. 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 (i.e. e-passports). 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 both the tag and the server to perform only bitwise operations. Through a three-way handshake protocol this identification technique also achieves mutual authentication, as well as resilience against DoS and replay attacks. Moreover, the proposed identification technique is shown to preserve the privacy of the tag. Finally, note that either of the two contributions can be independently incorporated into existing protocols.

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 the security evaluation and overhead analysis of this protocol. Finally, in Section 6 we expose some concluding remarks.

2 Related work

A standard approach to provide security in RFID protocols [3,4] 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 in order 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 [3] 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 [5] 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 [2,4] 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 [2] recognizes, this solution is prone to DoS attack, in that an adversary can easily exhaust the hash chain via reading attempts.

In [5,6] the authors optimizes a technique originally proposed in [7]. 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 [8] 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 [9]: the authors are inspired by the work in [10] 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 [11] 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 [12], through the HB++ protocol. Furthermore, HB++ was proven in [13] 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.

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 for every tag tag_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 tag_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) and a secure hash function $h(\circ)$, as assumed in literature [2,3,5]. The KDC assigns a unique key k_i to tag_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 entries and $KDB[g]$ to return the key $k_{g,j}$.

4 The protocol

This section presents first the protocol through which the confinement and probabilistic identification techniques are implemented. 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 threaten the integrity of the identification by 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 efficiency of 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 entry it is looking for. 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 half of the *active* entries in the KDB — where an active entry is an entry that has not been eliminated yet. By subsequently eliminating active entries at each step, 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 we provide is efficient in that it requires $O(n \log n)$ bit-wise operations (where n is the number of tags) and only uses simple comparison of memory cells.

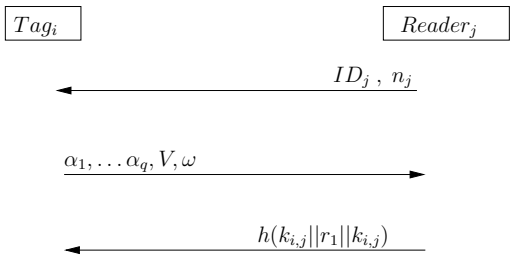

Fig. 1. The proposed protocol

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 and, for $p \in [1 \ldots q]$, it holds:

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

$$V[p] = DPM(r_p) \tag{2}$$

where the value r_p is the result of the invocation of the PRNG. Note that the bit length of r_p and $k_{i,j}$ is the same, that is $|k_{i,j}| = |r_p| = \ell$, and $r_p[i]$ denotes the i^{th} bit of the bit vector r_p. In the following we assume, without losing of generality, that ℓ is a multiple of 3. 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 its active entries. Let us focus on the g^{th} entry of the KDB and assume it is active; the following check will be performed:

1. compute $r' = KDB[g] \oplus \alpha_p$;
2. check if $DPM(r') = V[p]$.

If the check fails, the g^{th} entry is discarded and the next entry of the KDB, if any, is examined. However, if the check succeeds, the current entry of KDB cannot be discarded. Indeed, if $KDB[g]$ is the actual entry associated with the tag, that is, if $KDB[g] = k_{i,j}$, the check will succeed by construction. On the other hand, if the check fails, the current entry definitely cannot be the one associated with the tag. Finally, note that for each α_p on the average one half of the active entries are eliminated. A thorough analysis of the lookup process can be found in Section 4.4.

4.3 Mutual authentication and session freshness

Assume that the look up process completes, returning a single entry of the KDB to the reader ($KDB[i]$). On one hand, as shown by the analysis of the lookup process in the next section, for an appropriate choice of the value q this will happen with high probability if the tag is a legitimate one, i.e. belonging to the set of tags recorded in the KDB. On the other hand, if the tag it is not a legitimate one, with high probability no entry will be returned. Hence, when the lookup procedure returns a single entry, we will assume in this subsection that the returned entry identifies the tag. Once the reader has identified the tag —let $k_{i,j}$ be the key in the KDB returned by the identification protocol —, the reader first recovers r_1 ($r_1 = \alpha_1 \oplus k_{i,j}$) and then proceeds to authenticate the tag and to verify the freshness of the session just computing $z = h(k_{i,j}||n_j||r_1||k_{i,j})$

```
Global variables: n ; q ; KDB
Input           : < α_1, ..., α_q, V, w >
Output          : The active entries of the KDB.
1.1  for i=1 to n do
1.2  |  Active[u] = True
1.3  end
1.4  count = 0; a = 0
1.5  while a < q do
1.6  |  u = 0
1.7  |  while u < n do
1.8  |  |  if Active[u] then
1.9  |  |  |  r' = α_a ⊕ KDB[u]
1.10 |  |  |  if DPM(r') ≠ V[a] then
1.11 |  |  |  |  Active[u] = False
1.12 |  |  |  |  count + +
1.13 |  |  |  end
1.14 |  |  end
1.15 |  |  u + +
1.16 |  end
1.17 |  a + +
1.18 end
1.19 if count = n then
1.20 |  fail
1.21 else
1.22 |  return KDB[j] s.t. Active[j] = True
1.23 end
```

Algorithm 1: Lookup

and verifying whether $z = \omega$. If the latter match succeeds, the reader has successfully authenticated the tag and verified the freshness of the session.

In the following we show how the tag authenticates the reader. We start by observing that 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: $r_p = \alpha_p \oplus KDB[i]$. Hence, the reader authenticates itself to the tag and assures the freshness of the session by sending to the tag the value $h(k_{i,j}||r_1||k_{i,j})$. 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.

4.4 Analysis

Server compromise: in case $reader_j$ is compromised the attacker can only access $k_{i,j}$, $i = 1..n$. Under the assumption that the hash function is one-way, it is impossible to derive k_i from $k_{i,j}$; 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 show that the lookup protocol completes and we prove its correctness.

Protocol termination: from Table 1 it can be verified that the protocol terminates after a finite number of iterations in the two inner loops; further, its completion takes at most $O(nq)$ steps, where each step consists of simple xor operations and a comparison. In the following it is shown that $q = O(\log n)$, yielding an overall complexity of $O(n \log n)$ bitwise comparisons.

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 entry of the KDB for a valid tag, with a probability ϵ, where ϵ can be decided at design phase.

Lemma 1. *For each valid input to the the Lookup Process provided by a valid tag tag_i, $Active[i]$ will take on value $True$ on all iterations of the Lookup Process.*

Proof. By construction, the key $k_{i,j}$ corresponding to a valid input will never fail any of the tests in the inner loop starting at line 1.7 of Algorithm 1; hence, $Active[i]$ will never be assigned with the value $False$. □

Lemma 2. *A randomly chosen input will be accepted by the Lookup Process with probability less than ϵ, where ϵ is chosen at 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 value α_i will not set an entry of the Active vector to $False$ in Algorithm 1 — that is if $V[i] = P(M(\alpha_i \oplus k_{u,j}))$— and 0 otherwise. In order for I to be considered a valid input with respect to a single entry ($KDB[u]$) of the KDB, all q tests have to succeed. 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 it will be shown in Lemma 3:

$$Pr[X_1[u] = 1] = \frac{1}{2}.$$

Since there are n entries in the KDB, the probability that at least one of them survives after q steps is $Pr[E_1 \vee \ldots \vee E_n] \leq nPr[E_1] < n(1/2)^q = (1/2)^{-q+\log n}$. Now let r be the highest integer such that $\epsilon \leq 2^{-r}$. If we set $q = r + \log n$, the lemma holds. □

In Figure 2 we report an experiment to support the previous result. We implemented a simulator for Algorithm 1. We generated a KDB of 65,536 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 α_i sent by the tag to the reader. In particular, we varied q in the range $[(\log n)/2, \ldots, 10 + \log n]$, that is in the range $[8, \ldots, 26]$, using an incremental step of 1. On the x-axis we report the value q, while on the y-axis the number of active entries left in the KDB. To amortize statistical fluctuation, for each value of q, we performed 256 identification attempts, and we reported on the y-axis the number of active entries left in the KDB, averaged over these 256 protocol runs. As it can be seen from Figure 2, the number of active entries left in the KDB that result from the simulation is in accordance with the theoretical result of Lemma 2.

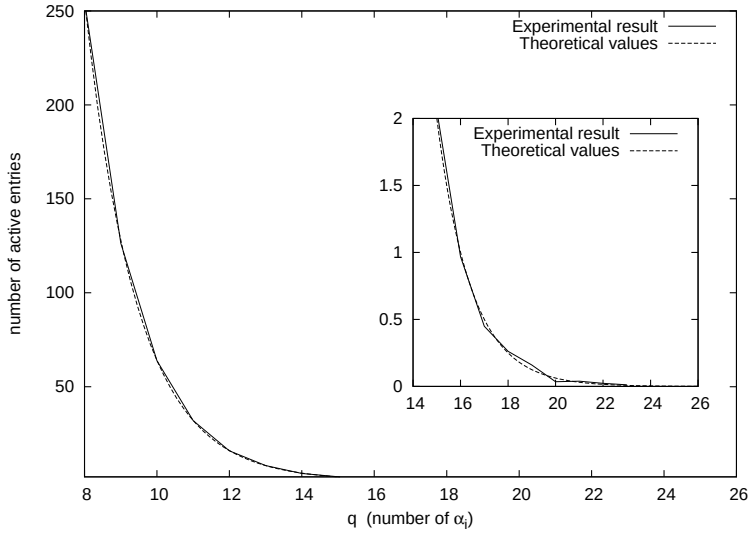

Fig. 2. Reader false acceptance rate: comparison of analytical and experimental results.

Theorem 1. *On a valid input I generated by a legitimate tag (tag_i) the Lookup Process will return only the entry $KDB[i]$ with probability at least $1-\epsilon$, where ϵ is chosen at design phase.*

Proof. This theorem can be reworded as: on a valid input, when the Lookup Process ends, the probability that only one entry of the $Active$ vector is still set to $True$ and that this entry is the one matching the input is $1-\epsilon$. The proof of this theorem follows from Lemma 1 and Lemma2. Based on Lemma 1 the probability that the entry of the $Active$ vector matching the input is set to $True$ after the last iteration of the Lookup Process is 1. The probability that at least one more entry has the value $True$ when the Lookup Process ends is the same as the probability that a randomly chosen input is accepted, that is less than ϵ by Lemma 2.

5 Security analysis and overhead

Due to space limitations, a formal proof of the security properties as well as a thorough comparison with [9] will appear in the extended version of this paper. Nevertheless, in the sequel of this paper we will provide an intuition on the soundness of the security properties.

5.1 Key secrecy and Privacy

A run of the identification protocols sends q times over the communication channel between the tag and the reader the key of the tag, each time xored with a random value

(r_p). Hence, so far the security and privacy provided is that of the One Time Pad (OTP), that is perfect security and privacy. However, it should be noted that the protocol leaks one bit of information for each of the values r_i; this bit is conveyed in $V[i]$.

Key secrecy To provide an intuition of how the secrecy of the proposed scheme is affected by the leakage of $V[i] = DPM(r_i)$, let us compute what is the probability that a random values $r' \in \{0,1\}^\ell$ verifies $DPM(r') = DPM(r_i)$. Indeed, for an adversary to mount a successful attack, it is required to discriminate among the set of possible keys; the bigger the set of possible keys, the harder the task, and the bigger this set, the higher the above probability.

Lemma 3. *Given a value $p \in \{0,1\}$, for $r' \in_R \{0,1\}^\ell$ it holds that:*

$$Pr[DPM(r') = p] = \frac{1}{2}$$

Proof. Given a vector of d i.i.d binary random variable $X =< X_1, \ldots, X_d >$ and $p \in \{0,1\}$, there exist 2^{d-1} different assignment of values to $< X_1, \ldots, X_d >$ such that $\oplus_{j=1}^{d} X_j = p$. For instance, note that in our identification protocol, for a given S_i (e.g. $S_i =< 1,1,1 >$), it is possible to generate three other S_{i_j} ($S_{i_1} =< 1,1,0 >$, $S_{i_2} < 1,0,1 >$, $S_{i_2} =< 0,1,1 >$) such that: $M(S_i) = M(S_{i_j})$.

Let $X_i = M(S_i)$ and note that the S_i ($i = 1, \ldots, \ell/3$) are independent. It follows that it is possible to have $(4^{\ell/3})(2^{\ell/3-1}) = 2^{\ell-1}$ different r'_i such that $DPM(r_i) = DPM(r'_i)$. Hence,

$$Pr[DPM(r') = p] = \frac{2^{\ell-1}}{2^\ell} = \frac{1}{2}$$

. □

Privacy To study the impact of one bit leakage on privacy, we consider a hypothetical protocol based on the original identification protocol defined in Section 4.2 with a slight modification that consists of the substitution of the $DPM(\circ)$ function by the parity function. Thus in the hypothetical protocol, $V[i] = P(r_i) = D\hat{P}M(r_i)$. It can be shown that the modified version of the protocol is similar to the original one with respect to the number of messages required to identify the appropriate key in the KDB, that is $O(\log n)$ messages. Moreover, as for the confidentiality of the key, it easily follows from Lemma 3 that the leakage of the parity bit just halves the key space. However, when it comes to privacy, the hypothetical protocol is basically flawed.

Indeed, assume the attacker can observe two different runs of the identification protocol: $< \alpha_{1,1}, \ldots, \alpha_{1,q}, V_1, \omega_1 >$ and $< \alpha_{2,1}, \ldots, \alpha_{2,q}, V_2, \omega_2 >$. Its task is to distinguish, with a non negligible probability, if the two flows intercepted were originated by the same tag or not.

Note that for two random bit vectors r_1, r_2, it can be shown that $D\hat{P}M(r_1 \oplus r_2) = D\hat{P}M(r_1) \oplus D\hat{P}M(r_2)$ — that is, the parity computed over the xor of vectors r_1, r_2 is equal to xoring the result of the parity computed over each single vector; this introduces an imbalance in the probability distribution that could be leveraged by an adversary tampering with privacy. In particular, let δ be the event "$D\hat{P}M(\alpha_{1,i} \oplus \alpha_{2,i}) = V_1[i] \oplus$

$V_2[i]$". We can express $Pr[k_1 \neq k_2|\delta]$ as $Pr[k_1 \neq k_2|\delta] = 1/2 - |adv|$. In a perfect privacy preserving solution, we would have $|adv| = 0$. When trading off privacy with identification capabilities, as it is the case when leaking one bit of information, we would like to have $|adv|$ as small as possible, and possibly such that $\lim_{\ell \to \infty} |adv| = 0$; the faster the convergence to zero, the less is the advantage gained by the adversary. However, in this modified protocol, we have that $adv = 1/6$, as formalized with the following lemma:

Lemma 4. *Given the event* δ*="*$D\hat{P}M(\alpha_{1,i} \oplus \alpha_{2,i}) = V_1[i] \oplus V_2[i]$*". Then* $Pr[k_1 \neq k_2|\delta] = 1/2 - 1/6$.

Proof.

$$\begin{aligned} Pr[k_1 \neq k_2|\delta] =& 1 - Pr[k_1 = k_2|\delta] = 1 - Pr[(k_1 = k_2) \wedge \delta]/Pr[\delta] = \\ & 1 - \frac{Pr[\delta|(k_1 = k_2)]Pr[k_1 = k_2]}{Pr[\delta]} = \\ & 1 - \frac{Pr[\delta|(k_1 = k_2)]Pr[k_1 = k_2]}{Pr[\delta|k_1 = k_2]Pr[k_1 = k_2] + Pr[\delta|k_1 \neq k_2]Pr[k_1 \neq k_2]}. \end{aligned}$$

Now, as noticed above, we have that: $Pr[\delta|(k_1 = k_2)] = 1$ and $Pr[\delta|(k_1 \neq k_2)] = 1/2$. Plugging these equalities in the above equation, we obtain:

$$Pr[k_1 \neq k_2|\delta] = 1 - \frac{1 \times 1/2}{(1 \times 1/2) + 1/2 \times 1/2} = 1 - \frac{2}{3} = \frac{1}{2} - \frac{1}{6}$$

□

We believe that capitalizing on the imbalance of the distribution probability a thorough attack targeting the privacy of the hypothetical protocol could be mounted. However, this is out the scope of the paper; the discussion so far was meant to give the reader the intuition behind the main rationale for the privacy evaluation of the original protocol. In particular, the main threat against privacy can be expressed as the *advantage —adv—* that was given to the adversary when computing: $Pr[\delta|k_1 = k_2] = Pr[D\hat{P}M(\alpha_{1,i} \oplus \alpha_{2,i}) = V_1[i] \oplus V_2[i]|k_1 = k_2] = \frac{1}{2} + |adv|$. Turning to the original protocol proposed in this paper, we notice that the following theorem holds.

Theorem 2. *Let* δ *be the event "*$DPM(\alpha_{1,i} \oplus \alpha_{2,i}) = V_1[i] \oplus V_2[i]$*". Then* $Pr[k_1 \neq k_2|\delta] = \frac{1}{2} - \frac{\epsilon}{2(2+\epsilon)}$*, where* $\epsilon = \left(\frac{1}{2}\right)^{\frac{2}{3}\ell}$.

Proof. Following the demonstration flow of Lemma 4 we can rewrite $Pr[k_1 \neq k_2|\delta]$ as:

$$Pr[k_1 \neq k_2|\delta] = 1 - \frac{Pr[\delta|(k_1 = k_2)]Pr[k_1 = k_2]}{Pr[\delta|k_1 = k_2]Pr[k_1 = k_2] + Pr[\delta|k_1 \neq k_2]Pr[k_1 \neq k_2]}.$$

It can be proved (see **conditioned** in Appendix) that $Pr[\delta|k_1 = k_2] = 1/2+(1/2)^{\frac{2}{3}\ell+1}$. Hence:

$$Pr[k_1 \neq k_2|\delta] = 1 - \frac{(1/2 + (1/2)^{\frac{2}{3}\ell+1})(1/2)}{(1/2 + (1/2)^{\ell+1})(1/2) + (1/2)(1/2)} =$$

$$1 - \frac{1/2 + (1/2)^{\frac{2}{3}\ell+1}}{1/2 + (1/2)^{\frac{2}{3}\ell+1} + 1/2} = \frac{1/2}{1 + (1/2)^{\frac{2}{3}\ell+1}} = \frac{1}{2 + (1/2)^{\frac{2}{3}\ell}} = \frac{1}{2} - \frac{\epsilon}{2(2+\epsilon)},$$

where $\epsilon = \left(\frac{1}{2}\right)^{\frac{2}{3}\ell}$. □

Therefore, using the attack that originally targeted the hypothetical protocol, the advantage of the attacker on our protocol decreases exponentially fast with the length of the key. In particular, this also provides a method for determining the appropriate key length. Indeed, by setting the maximum advantage for the adversary to $2^{-\tau}$, a key length that provides to the adversary an advantage less than $2^{-\tau}$ is given by $\ell = \lceil (3/2)(\tau-2) \rceil$. The previous equation was based on the following consideration: $\frac{\tau}{2(2+\tau)} \leq \frac{\tau}{4}$. As an illustration, to provide the adversary with an advantage less than 2^{-80}, the key length could be set to $\ell = \lceil (3/2)(\tau - 2) \rceil = \lceil (3/2)(80 - 2) \rceil = 117$.

5.2 Mutual authentication

By Lemma 2 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 [14].

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.

5.3 DoS resilience

Opposed to other approaches [2,5,6], 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.

5.4 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 [2], 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 [15,16], matching the tight hardware constraints of RFID, could be adopted to serve as hash function. Further, the tag requires to compute $q\ell$ "and" ($\wedge$), $q\ell$ "or" ($\vee$) —due to the recurrent invocation of the function $M(\circ)$— and $q\ell/3$ more "xor" ($\oplus$) due to the invocation of the function $P(\circ)$. Note that the sum of the cost of all these "xor", "or" and "and" operations can be considered negligible.

As for the communications overhead, the tag is required to send q messages of $|\ell|$ 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 2, a practical value for q could be $2 \log n$; in this way the reader lookup protocol will return, when triggered by a legitimate query, more than one entry only with probability $1/n$ on the average. As discussed before note that, in case the lookup protocol returns a bogus entry, the authentication protocol will reject that entry. 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 active entries left, is that the new values α_i can be matched against the active entries 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 $O(n \log n)$ bitwise operations and $O(n \log n)$ bit comparison. 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 2, where n is the total number of tags the system is composed of. Hence, 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.

5.5 Protocol comparison

A concise comparison of the properties provided by our protocol with regard to a few reference protocols is given in Table 1. Note that our protocol is the only one that fulfils all the properties. Due to page limitation, a detailed discussion enriched with few more properties will be provided in the extended version of this paper.

Table 1. Comparison of our proposal with some protocols in Section 2.

Protocol	Properties			
	Privacy	Mutual auth.	DoS resilience	reply attack res.
Our [this paper]	Yes	Yes	Yes	Yes
OSK/OA [5]	Yes	Yes	No	Yes
CR/MW [3]	weak	Yes	Yes	Yes
Ya-Trap[2]	Yes	No	No	Yes

6 Concluding remarks

As a first contribution of this paper we have relaxed the assumption that servers cannot be compromised and 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. Second, we have proposed a probabilistic mechanism that preserves privacy and allows mutual authentication between server and tag. This mechanism is also resilient to DoS and replay attacks. Further, it only requires $O(n \log n)$ bitwise operations and comparisons on the data base of keys stored in a server, hence speeding up the search process. Moreover, the tag just requires to store a single key and the capability to run a PRNG and a hash function. Finally, the information confinement technique and the tag identification protocol could be independently incorporated into existing solutions.

Current work is aimed at devising a possibly general formal framework to evaluate the security and privacy of the proposed solution.

References

1. Juels, A.: Rfid security and privacy: A research survey. IEEE Journal on Selected Areas in Communications **24**(2) (2006) 381–394
2. Tsudik, G.: Ya-trap: Yet another trivial rfid authentication protocol. In: IEEE PerCom Workshops. (2006) 640–643
3. Molnar, D., Wagner, D.: Privacy and security in library rfid: issues, practices, and architectures. In: CCS '04: Proceedings of the 11th ACM conference on Computer and communications security, New York, NY, USA, ACM Press (2004) 210–219
4. Rhee, K., Kwak, J., Kim, S., Won, D.: Challenge-response based RFID authentication protocol for distributed database environment. In Hutter, D., Ullmann, M., eds.: International Conference on Security in Pervasive Computing – SPC 2005. Volume 3450 of LNCS., Boppard, Germany, Springer-Verlag (April 2005) 70–84
5. Avoine, G., Dysli, E., Oechslin, P.: Reducing time complexity in RFID systems. In Preneel, B., Tavares, S., eds.: Selected Areas in Cryptography – SAC 2005. Volume 3897 of LNCS., Kingston, Canada, Springer-Verlag (August 2005) 291–306
6. Avoine, G., Oechslin, P.: A scalable and provably secure hash based RFID protocol. In: International Workshop on Pervasive Computing and Communication Security – PerSec 2005, Kauai Island, Hawaii, USA, IEEE, IEEE Computer Society Press (March 2005) 110–114

7. Hellman, M.: A cryptanalytic time-memory tradeoff. IEEE Transactions on Information Theory **26** (1980) 401–406
8. Conti, M., Di Pietro, R., Mancini, L.V., Spognardi, A.: RIPP-FS: an rfid identification, privacy preserving protocol with forward secrecy. In: Proceedings of the 3^{rd} IEEE International Workshop on Pervasive Computing and Communication Security, IEEE Press, to appear (2007)
9. Juels, A., Weis, S.: Authenticating pervasive devices with human protocols. In Shoup, V., ed.: Advances in Cryptology – CRYPTO'05. Volume 3126 of LNCS., Santa Barbara, California, USA, IACR, Springer-Verlag (August 2005) 293–308
10. Hopper, N.J., Blum, M.: Secure human identification protocols. In: ASIACRYPT. (2001) 52–66
11. Gilbert, H., Robshaw, M., Sibert, H.: An active attack against HB+ - a provably secure lightweight authentication protocol. Cryptology ePrint Archive, Report 2005/237 (2005)
12. Bringer, J., Chabanne, H., Emmanuelle, D.: 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, Lyon, France, IEEE, IEEE Computer Society Press (June 2006)
13. Piramuthu, S.: HB and related lightweight authentication protocols for secure RFID tag/reader authentication. In: Collaborative Electronic Commerce Technology and Research – CollECTeR 2006, Basel, Switzerland (June 2006)
14. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Chapter 9 - Hash Functions and Data Integrity. In: Handbook of applied cryptography. CRC Press (1996)
15. Feldhofer, M., Wolkerstorfer, J., Rijmen, V.: Aes implementation on a grain of sand. IEE Proceedings - Information Security **152**(1) (October 2005) 13–20
16. Pramstaller, N., Rechberger, C., Rijmen, V.: 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, New York, NY, USA, ACM Press (2006) 159–166
17. Matsui, M.: Linear cryptanalysis method for des cipher. In Springer, ed.: Advances in Cryptology-Eurocrypt '93, Lecture Notes in Computer Science n. 765 (1993) 386–397

Appendix

conditioned In the sequel, we want to evaluate the *advantage —adv—* that is given to the adversary in Equation 3:

$$Pr[DPM(\alpha_{1,i} \oplus \alpha_{2,i}) = V_1[i] \oplus V_2[i] | k_1 = k_2] = \frac{1}{2} + adv. \tag{3}$$

Note that the lower $|adv|$, the harder would be for the adversary to perform an educated guess in distinguishing between the two tags.

$$\begin{aligned}
&Pr[DPM(\alpha_{1,i} \oplus \alpha_{2,i}) = V_1[i] \oplus V_2[i] | k_1 = k_2] = \\
&Pr[DPM(r_{1,i} \oplus r_{2,i}) = V_1[i] \oplus V_2[i]] = \\
&Pr[M(S_{1,1} \oplus S_{2,1}) \oplus \ldots \oplus M(S_{1,\ell/3} \oplus S_{2,\ell/3}) = \\
&M(S_{1,1}) \oplus \ldots \oplus M(S_{1,\ell/3}) \oplus M(S_{2,1}) \oplus \ldots \oplus M(S_{2,\ell/3})] = \\
&Pr[(M(S_{1,1} \oplus S_{2,1}) \oplus M(S_{1,1}) \oplus M(S_{2,1})) \oplus \ldots \oplus \\
&(M(S_{1,\ell/3} \oplus S_{2,\ell/3}) \oplus M(S_{1,\ell/3}) \oplus M(S_{2,\ell/3})) = 0]
\end{aligned}$$

Now, let $Z_i = (M(S_{1,i} \oplus S_{2,i}) \oplus M(S_{1,i}) \oplus M(S_{2,i})$. We can rewrite Equation 3 as:

$$Pr[DPM(\alpha_1 \oplus \alpha_2) = V[1] \oplus V[2] | k_1 = k_2] = Pr[Z_1 \oplus \ldots \oplus Z_{\ell/3} = 0].$$

Note that the Z_i are independent, hence we can apply the piling-up-lemma [17], obtaining:

$$Pr[DPM(\alpha_1 \oplus \alpha_2) = V[1] \oplus V[2] | k_1 = k_2] =$$
$$Pr[Z_1 \oplus \ldots \oplus Z_{\ell/3} = 0] = \frac{1}{2} + 2^{\frac{\ell}{3}-1} \prod_{i=1}^{\ell/3} \left(Pr[Z_i = 0] - \frac{1}{2} \right)$$

It can be shown (see **Prob.** here below) that $Pr[Z_i = 0] = 10/16$, hence we have that:

$$Pr[DPM(\alpha_1 \oplus \alpha_2) = V[1] \oplus V[2] | k_1 = k_2] = \frac{1}{2} + \left(\frac{1}{2}\right)^{\frac{\ell}{3}-1} \prod_{i=1}^{\ell/3} \frac{1}{8}$$
$$= \frac{1}{2} + \left(\frac{1}{2}\right)^{\frac{2}{3}\ell+1}$$

Prob. Let $Z_i = (M(S_{1,i} \oplus S_{2,i}) \oplus M(S_{1,i}) \oplus M(S_{2,i}))$. We have that:

$$Pr[Z_i = 0] = Pr[(M(S_{1,i} \oplus S_{2,i}) \oplus M(S_{1,i}) \oplus M(S_{2,i}) = 0] =$$
$$Pr[M(S_{1,i} \oplus S_{2,i}) = 0 \wedge M(S_{1,i}) = 0 \wedge M(S_{2,i}) = 0] +$$
$$Pr[M(S_{1,i} \oplus S_{2,i}) = 0 \wedge M(S_{1,i}) = 1 \wedge M(S_{2,i}) = 1] +$$
$$Pr[M(S_{1,i} \oplus S_{2,i}) = 1 \wedge M(S_{1,i}) = 0 \wedge M(S_{2,i}) = 1] +$$
$$Pr[M(S_{1,i} \oplus S_{2,i}) = 1 \wedge M(S_{1,i}) = 1 \wedge M(S_{2,i}) = 0] =$$
$$\frac{10}{64} + \frac{10}{64} + \frac{10}{64} + \frac{10}{64} = \frac{5}{8}.$$

Indeed, let:
$[M(S_{1,i} \oplus S_{2,i}) = 0 \wedge M(S_{1,i}) = 0 \wedge M(S_{2,i}) = 0]$ = C1;
$[M(S_{1,i} \oplus S_{2,i}) = 0 \wedge M(S_{1,i}) = 1 \wedge M(S_{2,i}) = 1]$ = C2;
$[M(S_{1,i} \oplus S_{2,i}) = 1 \wedge M(S_{1,i}) = 0 \wedge M(S_{2,i}) = 1]$ = C3;
$[M(S_{1,i} \oplus S_{2,i}) = 1 \wedge M(S_{1,i}) = 1 \wedge M(S_{2,i}) = 0]$ = C4,
it is possible to build the truth table for the above variables that confirm our numerical results (the truth table is omitted due to space limitation).