Privacy of Recent RFID Authentication Protocols

Khaled Ouafi[1] and Raphael C.-W. Phan *[2]

[1] Laboratoire de sécurité et de cryptographie (LASEC),
Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015, Switzerland
khaled.ouafi@epfl.ch
[2] Electronic & Electrical Engineering,
Loughborough University, LE11 3TU, Leics, United Kingdom
r.phan@lboro.ac.uk

Abstract. Privacy is a major concern in RFID systems, especially with widespread deployment of wireless-enabled interconnected personal devices e.g. PDAs and mobile phones, credit cards, e-passports, even clothing and tires. An RFID authentication protocol should not only allow a legitimate reader to authenticate a tag but it should also protect the privacy of the tag against unauthorized tracing: an adversary should not be able to get any useful information about the tag for tracking or discovering the tag's identity. In this paper, we analyze the privacy of some recently proposed RFID authentication protocols (2006 and 2007) and show attacks on them that compromise their privacy. Our attacks consider the simplest adversaries that do not corrupt nor open the tags. We describe our attacks against a general untraceability model; from experience we view this endeavour as a good practice to keep in mind when designing and analyzing security protocols.

Keywords: RFID, authentication protocols, privacy, untraceability, provably secure.

1 Introduction

RFIDs are widely used in inventory control and supply chain management [1, 7, 18, 19, 25], in e-passports [12, 6, 11, 15, 20] e.g. for US' visa waiver policies, in contactless credit cards [10]. Thus the daily dealings of the present day individual is in fact a wireless interconnected network involving interactions both within his connected personal area network (PAN) among the things carried in his bag or pocket, and between the PAN and the servers providing the services and connectivitiy to those things. Among the things that the individual is carrying on him would include those that are RFID enabled i.e. items he bought from a retail chain, the credit cards in his wallet that he uses to purchase the items, and his e-passport to identify himself to authorities.

Privacy, both in terms of tag *anonymity* and tag *untraceability* (or unlinkability), is a significant concern that needs to be addressed if RFIDs are to be as widely deployed as conceived by proponents. To date, a rigorous treatment of privacy for RFID models is still being developed, notably the work of Avoine [2], Juels and Weis [13], Le, Burmester and de Medeiros [16]; and Vaudenay [27,

* Work done while the author was with LASEC, EPFL.

28]. These models differ mainly in their treatment of the adversary's ability to corrupt tags. In fact, the recent privacy models [13, 27, 16, 28] define privacy in the untraceability sense. This is intuitive since untraceabile privacy (UPriv) is a strictly stronger notion (i.e. it implies) than anonymous privacy (APriv). To see this, note that if there exists an adversary breaking APriv then he can easily also break UPriv; while the converse is not necessarily true.

In this paper, we analyze the privacy issues of recently (in 2006 and 2007) proposed RFID protocols, namely [8, 14, 24, 4, 9]. Our attacks do not even need the strong requirement of corrupting tags [26, 13, 27, 17, 16, 28, 22]. To the best of our knowledge, attacks presented here are the first known analyses of ProbIP [8], MARP [14], Auth2 [24], YA-TRAP+ [4], O-TRAP [4] and RIPP-FS [9].

2 RFID Privacy Models

We describe for completeness, the general untraceable privacy (UPriv) model that will be the setting in which we use in later sections to demonstrate how to trace tags and thus show that the schemes do not achieve the notion of untraceable privacy. It is also good practice to design and analyze security protocols with reference to a clearly-defined model [23].

We do not claim to define a new model, for our emphasis in this paper is instead on the analysis of the privacy and security issues of recent RFID protocols. In fact, the model defined herein can be seen as an alternative definition of the Juels-Weis model [13] with some differences e.g. in constraints put on the adversary (see the discussion in section 6.1) in a style that is more in line with the Bellare et al. [3] models for authenticated key exchange (AKE) protocols, for which RFID protocols have close relationship with.

A protocol party is a $\mathcal{T} \in Tags$ or $\mathcal{R} \in Readers$ interacting in protocol sessions as per the protocol specifications until the end of the session upon which each party outputs Accept if it feels the protocol has been normally executed with the correct parties. Adversary $\mathcal{A}$ controls the communications between all protocol parties (tag and reader) by interacting with them as defined by the protocol, formally captured by $\mathcal{A}$'s ability to issue queries of the following form:

$\mathsf{Execute}(\mathcal{R}, \mathcal{T}, i)$ **query.** This models *passive* attacks, where adversary $\mathcal{A}$ gets access to an honest execution of the protocol session i between $\mathcal{R}$ and $\mathcal{T}$ by eavesdropping.

$\mathsf{Send}(U_1, U_2, i, m)$ **query.** This query models *active* attacks by allowing the adversary $\mathcal{A}$ to impersonate some reader $U_1 \in Readers$ (resp. tag $U_1 \in Tags$) in some protocol session i and send a message m of its choice to an instance of some tag $U_2 \in Tags$ (resp. reader $U_2 \in Readers$). This query subsumes the TagInit and ReaderInit queries as well as challenge and response messages in the Juels-Weis model.

$\mathsf{Corrupt}(\mathcal{T}, K)$ **query.** This query allows the adversary $\mathcal{A}$ to learn the stored secret K' of the tag $\mathcal{T} \in Tags$, and which further sets the stored secret to K. It captures the notion of *forward security* or *forward privacy* and the extent of the damage caused by the compromise of the tag's stored secret. This is the analog of the SetKey query of the Juels-Weis model.

$\mathsf{Test}_{\mathsf{UPriv}}(U, i)$ **query.** This query is the only query that does not correspond to any of $\mathcal{A}$'s abilities or any real-world event. This query allows to define the

indistinguishability-based notion of *untraceable privacy* (UPriv). If the party has accepted and is being asked a Test query, then depending on a randomly chosen bit $b \in \{0,1\}$, $\mathcal{A}$ is given $\mathcal{T}_b$ from the set $\{\mathcal{T}_0, \mathcal{T}_1\}$. Informally, $\mathcal{A}$ succeeds if it can guess the bit b. In order for the notion to be meaningful, a Test session must be *fresh* in the sense of Definition 2.

Definition 1 (Partnership & Session Completion) *A reader instance $\mathcal{R}_j$ and a tag instance $\mathcal{T}_i$ are partners if, and only if, both have output* Accept*$(\mathcal{T}_i)$ and* Accept*$(\mathcal{R}_j)$ respectively, signifying the completion of the protocol session.*

Definition 2 (Freshness) *A party instance is fresh at the end of execution if, and only if,*

1. *it has output* Accept *with or without a partner instance,*
2. *both the instance and its partner instance (if such a partner exists) have not been sent a* Corrupt *query.*

Definition 3 (Untraceable Privacy (UPriv)) *Untraceable privacy (*UPriv*) is defined using the game $\mathcal{G}$ played between a malicious adversary $\mathcal{A}$ and a collection of reader and tag instances. $\mathcal{A}$ runs the game $\mathcal{G}$ whose setting is as follows.*

Phase 1 (Learning): $\mathcal{A}$ is able to send any Execute, Send, and Corrupt queries at will.
Phase 2 (Challenge):

1. At some point during $\mathcal{G}$, $\mathcal{A}$ will choose a fresh session on which to be tested and send a Test query corresponding to the test session. Note that the test session chosen must be fresh in the sense of Definition 2. Depending on a randomly chosen bit $b \in \{0,1\}$, $\mathcal{A}$ is given a tag $\mathcal{T}_b$ from the set $\{\mathcal{T}_0, \mathcal{T}_1\}$.
2. $\mathcal{A}$ continues making any Execute, Send, and Corrupt queries at will, subjected to the restrictions that the definition of freshness described in Definition 2 is not violated.

Phase 3 (Guess): Eventually, $\mathcal{A}$ terminates the game simulation and outputs a bit b', which is its guess of the value of b.

The success of $\mathcal{A}$ in winning $\mathcal{G}$ and thus breaking the notion of UPriv is quantified in terms of $\mathcal{A}$'s advantage in distinguishing whether $\mathcal{A}$ received $\mathcal{T}_0$ or $\mathcal{T}_1$, i.e. it correctly guessing b. This is denoted by $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{UPriv}}(k)$ where k is the security parameter.

It remains to remark on the models other than Juels-Weis, namely the Le-Burmester-de Medeiros (LBdM) model and the Vaudenay model. The LBdM model similarly allows the corruption of tags. Nevertheless, proof of security is in the universal composability (UC) model [5].

The Vaudenay model [27, 28] is stronger than both the Juels-Weis and Le-Burmester-de Medeiros models in terms of the adversary's corruption ability. In more detail, it is stronger than the Juels-Weis model in the sense that it allows corruption even of the two tags used in the challenge phase. It is stronger than the Le-Burmester-de Medeiros model in the sense that it considers all its privacy notions even for corrupted tags, in contrast to the Le-Burmester-de Medeiros model that only considers corruption for its forward privacy notion.

Our choice to describe our tracing attacks in later sections with reference to a defined model is for more uniformity between similar attacks on different RFID protocols, and for better clarity to illustrate how an adversary can circumvent the protocols using precise types of interactions that he exploits, as captured by his oracle queries. This will facilitate the task of a designer when an attempt is made to redesign an attacked protocol.

3 ProbIP

At RFIDSec '07, Castellucia and Soos [8] proposed an RFID protocol (ProbIP) that allows tag identification by legitimate readers. Its security is based on the SAT problem, which is proven to be in the $\mathcal{NP}$ class of complexity. See Fig. 1, where the symbols in **bold** beneath each device denotes the stored state, and $K[a_i]$ represents the a_i-th bit of the ℓ-bit length secret K. The authors of ProbIP gave arguments [8] for its security in the Juels-Weis model.

For simplicity, we assume, and for the rest of this paper, that the reader and backend database server (if it exists) are one entity. This is sound since it is commonly assumed by RFID protocol designers that the channel between the reader and server are secure.

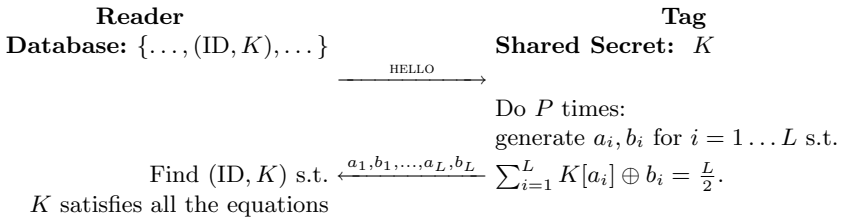

Fig. 1. The ProIP protocol

3.1 Violation of Anonymous Privacy

We first start by two remarks:

1. The tag does not update its secret key so at each authentication, some information is leaked from the *same* key.
2. The tag does not check the authenticity of the reader, i.e. an adversary can query the tag as many times as he likes.

From an information-theoretic point of view, a severe consequence of these two statements is that at one point an adversary will gather enough information to extract the key K from the responses of the tag.

Let us consider an adversary that will keep sending HELLO messages via Send queries to the tag until he gets ℓ equations. Since at each request tags generate P equations, an adversary would need to query the tag $\frac{n}{P}$ times. After that, she obtains the following system in which v_i^j denotes a boolean variable that is set to 1 if the $K[i]$-th bit of K is present in the j-th equation:

$$\begin{cases} \sum_{i=1}^{L} v_i^1 (K[i] \oplus b_i^1) & = \frac{L}{2} \\ \sum_{i=1}^{L} v_i^2 (K[i] \oplus b_i^2) & = \frac{L}{2} \\ \dots & \\ \sum_{i=1}^{L} v_i^\ell (K[i] \oplus b_i^\ell) & = \frac{L}{2} \end{cases} \tag{1}$$

As for any boolean v we can write $v + \bar{v} = 1$, we replace any $\bar{K[i]}$ by the value $1 - K[i]$. As a consequence we can deduce that there are as many as 3^n possible equations because every variable $K[i]$ can have three coefficients: $0, 1, -1$.

This way, the adversary gets a linear system of n equations and n variables that can be solved using standard methods such as the Gaussian elimination method. In the case where the n equations are not linearly independant, the adversary can still can obtain more equations from the tag by sending HELLO messages until she gets enough equations.

3.2 Countermeasure

The weakness of this authentication protocol comes from the fact that each round the advesary gets some information from the same key. So a quick way to counter our attack is to include a key-updating mechanism similar to OSK[21] at the end of the protocol using a one-way function.

In this case, adversaries do not get more than P equations for each key so that the security proof and reduction to the SAT problem become sound. The resulting protocol is even forward-private providing that adversaries do not get side-channel information from the reader [28].

4 MARP

MARP is proposed by Kim et al. [14] at CARDIS '06. They first describe a scheme consisting of separate phases, and then describe a more integrated one. For lack of better names, we denote these as MARP-1 and MARP-2, respectively. Here, a MARP is like a PDA to which several tags could be attached. The channels between reader and MARP, and between MARP and tag, are assumed by Kim et al. to be insecure.

We summarize these schemes in Figs. 2 and 3, which show only the bare minimal detail for understanding of our attacks. It suffices to note that $\langle K_d^g, K_e^g \rangle$ (resp. $\langle K_d^m, K_e^m \rangle$) is the private-public key pair of the reader (resp. MARP). Key_t and PIN_t are stored secrets of the tag. The reader is referred to [14] for more detailed descriptions.

4.1 Cryptanalysis of MARP-1

Tracing. Note that a_2 is fixed per tag, being a function of a particular tag T_t's unique identifier Uid_t and secret key Key_t. As the channel between the reader and the MARP is not confidential, an adversary via Execute queries (i.e. eavesdropping) can easily track the movement of T_t by checking for matches of a_2 with previously captured values, as long as the encryption is deterministic.

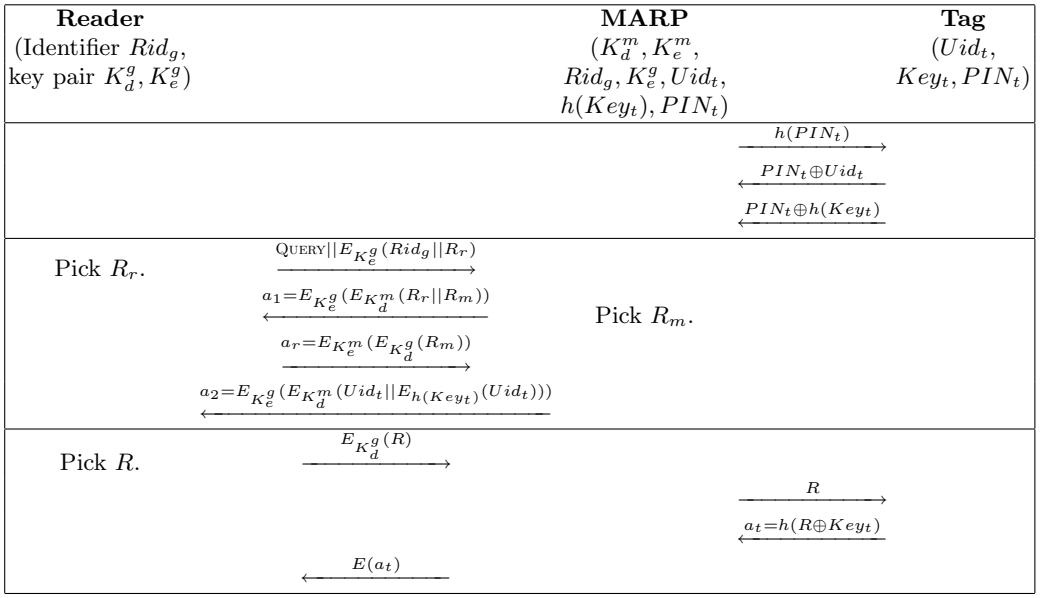

Fig. 2. The MARP-1 protocol, comprising 3 phases: setup, privacy protection, and authentication

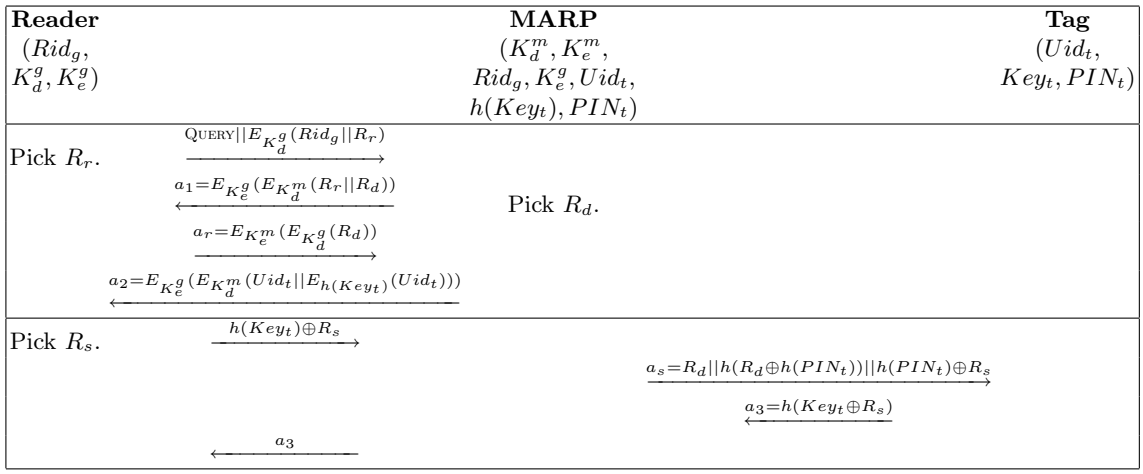

Fig. 3. The MARP-2 protocol, comprising 2 phases: MARP authentication and tag authentication

Alternatively, the adversary can replay an old R from MARP to the tag via Send queries, and check if the response a_t matches the old value of a_t corresponding to the replayed R.

We remark that these attacks have less requirement than the ones performed by Juels and Weis [13] on some other older RFID protocols that require Corrupt queries.

Violating the anonymous privacy. Note that the initial setup messages allow to compute

$$\begin{aligned} z &= [PIN_t \oplus Uid_t] \oplus [PIN_t \oplus h(Key_t)] \\ &= Uid_t \oplus h(Key_t). \end{aligned}$$

Then the adversary simply issues Execute queries to be able to compute z, and then issues a Send query to replace the message R from MARP to the tag with $R' = 0$, and so the tag responds with $a_t = h(Key_t)$. This allows to compute:

$$\begin{aligned} z \oplus a_t &= [Uid_t \oplus h(Key_t)] \oplus h(Key_t) \\ &= Uid_t, \end{aligned}$$

and so reveals a potential unique identifier of the tag, which can be cross-checked against the possible list of identifiers for a match.

4.2 Tracing MARP-2

MARP-2 also allows tracing. By eavesdropping both messages via Execute queries between the reader and the MARP and between the MARP and the tag, an adversary gets $h(key_t) \oplus R_s$ and $h(PIN_t) \oplus R_s$. By XOR-ing these two values, the adversary gets $h(PIN_t) \oplus h(key_t)$ which does not depend on the session parameters and can be used to trace a tag.

This scheme is also vulnerable to a replay attack since the response of the tag does only depend on the parameters sent by the MARP. So if an adversary sends twice the same message a_s via Send queries, she will get the same response a_3 which can also be used for tracing.

5 Auth2

At PerCom '07, Tan et al. [24] proposed two RFID protocols. We are interested here in the second protocol, and more exactly to the first variant described therein. For lack of better names we simply call it Auth2, see Fig. 4 for a complete description of the protocol, where r_j is a unique identifier of the reader, f and h are two collisions-resistant hash functions and $h(.)_m$ denotes the function that truncates the output of h to its m first bits.

5.1 Cryptanalysis of Auth2

Definite tracing. It was noted by Auth2 designers that indefinite tracing is possible but not a concern since many tags could result in the same $h(f(r_j||t_i))_m$ value. We show how this tracing can be made definite, i.e. it can precisely track a unique tag, not just a group of them that have the same $h(f(r_j||t_i))_m$.

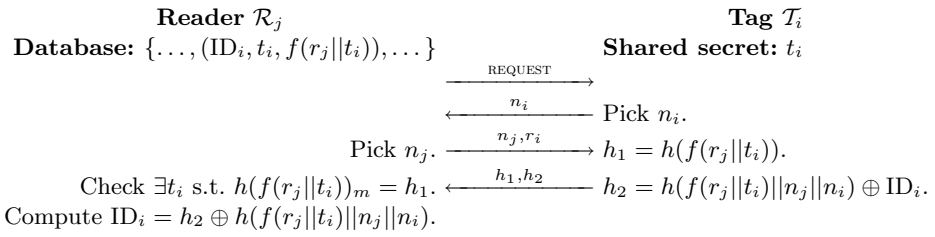

Fig. 4. The Auth2 protocol

1. **Learning**: The adversary eavesdrops via Execute queries for a short period during the protocol sessions involving tag $\mathcal{T}_0$ and two readers $\mathcal{R}_1, \mathcal{R}_2$ to obtain $\langle r_1, h(f(r_1||t_0))_m\rangle$ and $\langle r_2, h(f(r_2||t_0))_m\rangle$.
2. **Challenge**: Some time later, when the adversary wishes to track the tag $\mathcal{T}_0$, he starts a session with the challenge tag $\mathcal{T}_b \in \{\mathcal{T}_0, \mathcal{T}_1\}$ replaying r_1 via a Send query and checks the response from the tag for a match on the first message component with $h(f(r_1||t_0))_m$. He starts another session replaying r_2 via a Send query and checks the response from the tag for a match on the first message component with $h(f(r_2||t_0))_m$. With both matches, it is highly likely that this is the same tag whose session he had initially eavesdropped on, i.e. $\mathcal{T}_b = \mathcal{T}_0$. Else $\mathcal{T}_b = \mathcal{T}_1$.

Violating the anonymous privacy. When analyzing the anonymous privacy of their Auth2 scheme, the authors [24] assume that the adversary has access to the reader's list L of data corresponding to targeted tags, i.e. each entry in L is of the form $\langle ID_i, f(r_j||t_i)\rangle$.

The adversary only needs two entries in L corresponding to the targeted tag T_i, i.e.$\langle ID_i, f(r_1||t_i)\rangle$ and $\langle ID_i, f(r_2||t_i)\rangle$.

Issue a Send query with r_1 in a session, and then another Send query with r_2 in another session to T_i. Check if both responses match $f(r_1||t_i)$ and $f(r_2||t_i)$ respectively.

6 YA-TRAP, YA-TRAP+ and O-TRAP

At SecureComm '06, Burmester et al. [4] proposed two RFID protocols with formal proofs of security in the universal composability model [5], namely YA-TRAP+ and O-TRAP. These were inspired by YA-TRAP proposed at PerCom '06 by Tsudik [26].

6.1 YA-TRAP

The steps of YA-TRAP [26] are given in Fig. 5, where HMAC is a message authentication code and PRNG is a pseudo-random number generator. It works as follows: a tag is initialized with an initial timestamp t_0 and the top timestamp value t_{max}, as well as with a unique secret value K_i. Tags are also assumed to be able to compute a PRNG, where PRNG_i^j denotes the jth invocation by the tag $\mathcal{T}_i$ of its own PRNG.

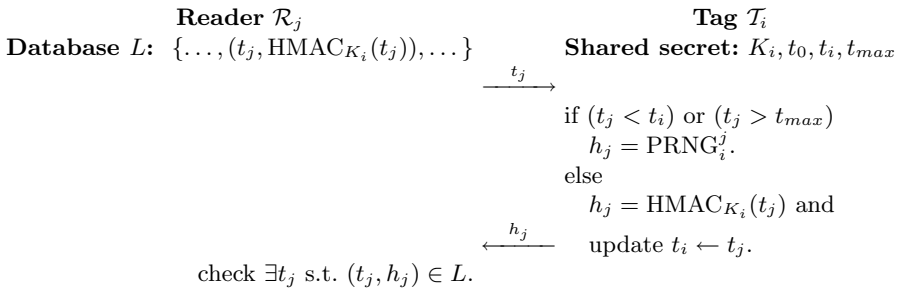

Fig. 5. The YA-TRAP protocol

The main goal of YA-TRAP's design was to achieve untraceable privacy (UPriv) with adversaries assumed to be able to corrupt tags.

Two operating modes were proposed [26] for YA-TRAP, namely *real-time* and *batch*. The difference is that for batch mode, responses from tags are collected by the reader in batches for later communication to the server for offline processing and identification. This latter mode is suited for settings e.g. inventory control where tags are assumed honest, since they will only be authenticated later in batches rather than online. Thus, this mode is not suitable for applications where feedback is required on the spot, e.g. library check-outs, or retail outlets for both tags in purchased items as well as tags in credit cards.

Tsudik observed that it was possible for denial of service (DoS) attacks to be launched towards YA-TRAP, and remarks that DoS resistance is not among the key goals of YA-TRAP.

What is more subtile, however, is the fact that a denial-of-service kind of attack could lead to an adversary being able to track a tag in the YA-TRAP protocol.

Tracing tags in real time. In the YA-TRAP specification, it was suggested [26] that the top value t_{max} of a tag's timestamp need not be unique but could instead be shared by a batch of tags.

Consider a scenario where tags have different t_{max}, operating in real-time mode. Indeed, acknowledging the fact that tags are produced by different manufacturers for diverse applications, it seems inevitable that some tags will have differing t_{max}. An adversary can trace a tag, i.e. distinguish between two tags (corresponding to a break of the privacy notion in the Juels-Weis model [13] and the UPriv notion we described in Section 2), as follows. For simplicity, assume two tags $\mathcal{T}_0$ and $\mathcal{T}_1$ with respective t_{max0} and t_{max1}, where $t_{max0} < t_{max1}$.

1. **Learning**: Issue a Send query with $t_j = t_{max0}$ to a tag $\mathcal{T} \in \{\mathcal{T}_0, \mathcal{T}_1\}$. Since t_{max0} is much into the future than current t_i value, a response $h_j = \mathrm{HMAC}_{K_i}(t_j)$ is expected, irrespective of which tag it is. Furthermore, the tag will update its local time counter as $t_i = t_{max0}$. This action serves to send the tag into the future by marking it for future tracing.
2. **Challenge**: Some time later, when it is desired to trace the tag, issue a Send query with t_j for $t_{max0} < t_j < t_{max1}$. If $\mathcal{T} = \mathcal{T}_0$, it will respond $h_j = \mathrm{PRNG}_i^j$ and will not successfully pass the validation check by the reader. If $\mathcal{T} = \mathcal{T}_1$, it will respond $h_j = \mathrm{HMAC}_{K_i}(t_j)$ and will successfully pass the validation

check. Thus by observing the reader-tag interaction via Execute queries, an adversary can distinguish between $\mathcal{T}_0$ and $\mathcal{T}_1$ and win the privacy game.

Juels and Weis [13] gave two tracing attacks on YA-TRAP that are valid in their privacy model, thus showing YA-TRAP does not meet their definition of strong privacy. Nevertheless, their tracing attacks would no longer apply in a weaker privacy model, and in fact one which better models the practical setting, where the adversary is further restricted by limiting its access to the TagInit message [13] as follows: when the TagInit message is issued to its two selected tags $\mathcal{T}_0$ and $\mathcal{T}_1$ used during the challenge phase, the adversary does not know which one of them was issued the message. This better models the practical privacy setting as the adversary is unaware during the learning phase which tag it has queried.

In contrast, our attack still applies in this weakened-adversary setting, and thus our result shows that setting a common t_{max} for tags offers more advantage over having individual t_{max} for each tag.

YA-TRAP was designed to specifically output a random response even if the tag does not want to be validated by the reader, such that an adversary is unable to distinguish between that random response and a proper response. Yet, by observing the output of the reader-tag interaction, i.e. seeing if the tag passes the validation or not, still allows the distinguishing. In this sense, using the YA-TRAP approach of generating random responses by itself is not sufficient to prevent tracing.

To reiterate, our attack can be prevented if the adversary is unable to observe the output of the reader-tag interaction, i.e. it does not know if the tag successfully passes the reader's validation check. This inability in fact corresponds to the *narrow* adversary model defined in Vaudenay's privacy model [28]. One example setting that fits this narrow model is the batch mode suggested by Tsudik [26] for YA-TRAP. Nevertheless, it is worth recalling here that batch mode is not relevant for applications where immediate feedback is required e.g. retail and library check-outs, and furthermore is only meaningful in the setting where tags are assumed to be honest (not usually the case) since they are not authenticated on the spot but later.

Cloning. An adversary can issue Send queries to the tag with arbitrarily many values of t_j and obtain the corresponding responses h_j. These values allow the tag to be cloned so that when the cloned tag is queried a particular t_j value, it will reply with the captured response h_j. The problem here stems from the fact that tag responses h_j are pre-computable only with the presence of the tag and not the reader since the supposed reader-supplied challenge is a predictable monotonically increasing timestamp t_j.

6.2 Tracing YA-TRAP+ with Second Pass

The steps of YA-TRAP+ are shown in Fig. 6, where $H_K(\cdot)$ denotes a keyed hash function and the steps preceeded by $[*]$ are optional, and only meant to be used by the reader if it is felt that DoS attacks are rampant. Legitimate readers share with the tags their secret keys K_i.

It turns out that the tracing attack of subsection 6.1 is simpler when applied to YA-TRAP+ if its optional second pass (preceeded in Fig. 6 by $[*]$) is made compulsory.

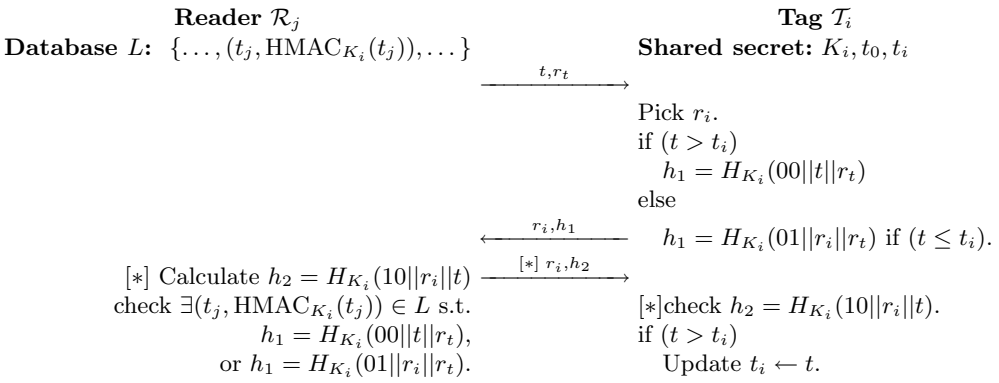

Fig. 6. The YA-TRAP+ protocol

1. **Learning**: An adversary first issues Send queries to the tag $\mathcal{T}_0$ with some r_t and a value t that is predictably much larger than the tag's t_i, obtaining the response $r_i, h_1 = H_K(00||t||r_t)$. It then intentionally modifies via a Send query the message h_2 from reader to tag such that the tag does not successfully authenticate the reader and thus the tag does not update its internal time counter t_i to t.
2. **Challenge**: Issue a Send query to the tag in future (i.e. let the challenge tag $\mathcal{T}_b \in \{\mathcal{T}_0, \mathcal{T}_1\}$ during the challenge phase) with the same r_t and t. Since $t > t_i$, it will return the response $r_i', h_1 = H_K(00||t||r_t)$ for which h_1 is the same if the challenge tag $\mathcal{T}_b = \mathcal{T}_0$. Otherwise, the adversary knows $\mathcal{T}_b = \mathcal{T}_1$. This allows to track the tag and win the privacy game.

Note that YA-TRAP+ was specifically designed to resist the kind of tracing attack on its predecessor YA-TRAP that we mounted in subsection 6.1, and yet this result shows that the optional second pass of YA-TRAP+ that requires to check h_2 before updating the stored secret, although meant to provide additional security to resist denial of service attacks, will in fact cause the protocol to fall to tracing.

6.3 Tracing O-TRAP

The steps of O-TRAP are shown in Fig. 7. The reader contains a hash table indexed by r_i with entries $\langle r_i, K_i \rangle$ where r_i and K_i correspond to secrets of tags to which it has legitimate access.

1. **Learning**: An adversary can issue a Send query to the tag $\mathcal{T}_0$ with random values r_t repeatedly, causing the tag to update its r_i each time such that it is way into the future compared to its synchronization with the reader.
2. **Challenge**: The adversary observes the future interaction between a tag $\mathcal{T}_b \in \{\mathcal{T}_0, \mathcal{T}_1\}$ and a reader via Execute queries to see if the reader accepts the tag as valid. If not, then the adversary knows this was the tag that it marked during the learning phase, i.e. $\mathcal{T}_b = \mathcal{T}_0$. Else, $\mathcal{T}_b = \mathcal{T}_1$.

Note that this kind of attack has been independently applied by Juels and Weis [13] to a couple of other older RFID protocols. Yet what is interesting as has been

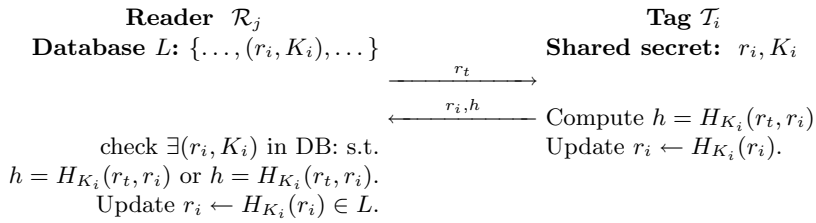

Fig. 7. The O-TRAP protocol

demonstrated here, is that recent provably secure protocols like YA-TRAP+ and O-TRAP in some sense still allow for tracing.

7 RIPP-FS

RIPP-FS was proposed by Conti et al. [9] at PerCom '07. The steps of RIPP-FS are given in Fig. 8.

Each tag $\mathcal{T}_i$ is initialized with a tag key $K^0_{\mathcal{T}_i}$ that it shares with the reader, as well as the initial value-pair (K_0, t_0) generated by the reader, where K_0 is the last value in a hash chain

$$K_\ell = w$$
$$K_i = H(K_{i+1}) = H^{\ell-1}(w), i = 0, \dots, \ell - 1$$

for w a seed, and t_j $(j = 0, \dots, \ell)$ is a time interval counter.

A tag is also assumed to be able to compute a pseudo-random number generator (PRNG), where PRNG_i^j denotes the jth invocation by the tag $\mathcal{T}_i$ of its own PRNG.

One of the goals of RIPP-FS's design was to achieve untraceable privacy (UPriv) against adersaries able to corrupt tgs, and it is claimed to offer more security properties than YA-TRAP, YA-TRAP+ and O-TRAP.

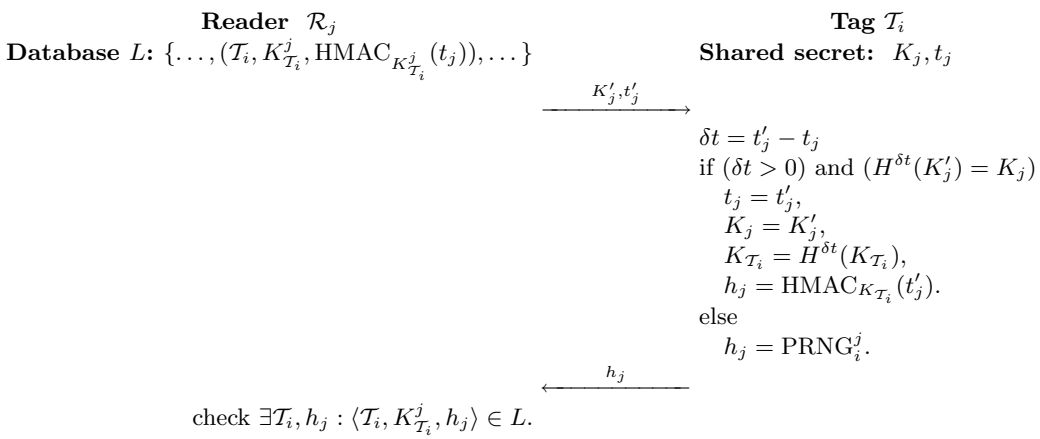

Fig. 8. The RIPP-FS protocol

7.1 Tracing Tags

We show how to trace tags in the RIPP-FS protocol.

1. **Learning**:
   (a) Query Send to the reader to initiate two protocol sessions, obtaining (K'_j, t'_j) and (K'_{j+1}, t'_{j+1}), where $t'_{j+1} > t'_j$, and $K'_j = H(K'_{j+1})$.
   (b) Make a Send query to a tag $\mathcal{T}_0$ with the value (K'_{j+1}, t'_{j+1}). Since this is a valid message generated from the reader, a response $h_j = \text{HMAC}_{K_{\mathcal{T}_i}}(t'_{j+1})$ is expected. More importantly, the tag will update its time interval counter as $t_j = t'_{j+1}$, as well as the other secrets $K_j = K'_{j+1}$ and $K_{\mathcal{T}_i} = H^{t'_{j+1}-t_j}(K_{\mathcal{T}_i})$.
2. **Challenge**: Some time later, when it is desired to trace the tag, issue a Send query with (K'_j, t'_j) to the challenge tag $\mathcal{T}_b$, and pass the response h_{j+1} to the reader. If $\mathcal{T}_b = \mathcal{T}_0$, it will respond $h_{j+1} = \text{PRNG}_i^j$ and will not successfully pass the validation check by the reader. If $\mathcal{T}_b = \mathcal{T}_1$, it will respond $h_{j+1} = \text{HMAC}_{K_i}(t'_j)$ and will successfully pass the validation check. Thus by observing the reader-tag interaction via Execute queries, an adversary can distinguish between $\mathcal{T}_0$ and $\mathcal{T}_1$ and win the privacy game.

8 Concluding Remarks

We first provided an alternative description of privacy models that captures the notion of untraceable privacy (UPriv) and discussed its relation to existing models. This was to pave the way for our analysis results in later sections. We showed how the notion of UPriv cannot be achieved by some recent RFID protocols.

Our emphasis in this paper was to analyze the level of untraceable privacy offered by the protocols. We only discussed reasons why our attacks worked and intentionally did not propose any tweaks nor fixes on the protocols; mainly because there are already many available in literature, and so we feel this was not necessary unless there is a serious void of well designed provably secure RFID protocols.

Final remarks: while a uniformly accepted privacy model for RFID protocols is still being developed by the community, the results here serve to strengthen the need for such a standard model to facilitate better design of RFID protocols that offer both privacy and security. This has to be fulfilled if RFIDs are ever to be widely used by each individual within his network space of interconnected things.

References

1. "Albertsons Announces Mandate," RFID Journal, 5 March, 2004. Available online at `http://www.rfidjournal.com/article/articleview/819/1/1/`.
2. G. Avoine, "Adversarial Model for Radio Frequency Identification," Cryptology ePrint Archive, report 2005/049, 20 February, 2005. Available at IACR ePrint Archive, `http://eprint.iacr.org/2005/049`.
3. M. Bellare, D. Pointcheval and P. Rogaway, "Authenticated Key Exchange Secure against Dictionary Attacks," *Advances in Cryptology - EUROCRYPT '00*, LNCS 1807, pp. 139–155, 2000.

4. M. Burmester, T.V. Le and B. de Medeiros, "Provably Secure Ubiquitous Systems: Universally Composable RFID Authentication Protocols," *Proceedings of Securecomm '06*, pp. 1–9, 2006. Full version available at IACR ePrint Archive, `http://eprint.iacr.org/2006/448`, last revised 5 December 2006.
5. R. Canetti, "Universally Composable Security: A New Paradigm for Cryptographic Protocols," *Proc. IEEE FOCS '01*, pp. 136–145, 2001. Full version available at IACR ePrint Archive, `http://eprint.iacr.org/2000/067`, last revised 13 December 2005.
6. D. Carluccio, K. Lemke and C. Paar, "E-Passport: The Global Traceability or How to Feel Like a UPS Package," *Proceedings of WISA '06*, LNCS 4298, pp. 391–404, 2007.
7. CASPIAN, "Boycott Benetton," accessed 19 September 2007. Available online at `http://www.boycottbenetton.com`.
8. C. Castelluccia and M. Soos, "Secret Shuffling: A Novel Approach to RFID Private Identification," *Proceedings of RFIDSec '07*, pp. 169–180, 2007.
9. M. Conti, R. Di Petro, L.V. Mancini and A. Spognardi, "RIPP-FS: An RFID Identification, Privacy Preserving Protocol with Forward Secrecy," *Proceedings of PerCom '07*, pp. 229–234, 2007.
10. T.S. Heydt-Benjamin, D.V. Bailey, K. Fu, A. Juels and T. O'Hare, "Vulnerabilities in First-Generation RFID-enabled Credit Cards," *Proceedings of Financial Cryptography '07*, LNCS, to appear.
11. J.-H. Hoepman, E. Hubbers, B. Jacobs, M. Oostdijk and R.W. Schreur, "Crossing Borders: Security and Privacy Issues of the European e-Passport," *Proceedings of IWSEC '06*, LNCS 4266, pp. 152–167, 2006.
12. A. Juels, D. Molnar and D. Wagner, "Security and Privacy Issues in E-Passports," *Proceedings of SecureComm '05*, pp. 74–88, 2005. Full version available at IACR ePrint Archive, `http://eprint.iacr.org/2005/095`, last revised 18 September 2007.
13. A. Juels and S.A. Weis, "Defining Strong Privacy for RFID," *Proceedings of PerCom '07*, pp. 342–347, 2007. Full version available at IACR ePrint Archive, `http://eprint.iacr.org/2006/137`, 7 April 2006.
14. S.-C. Kim, S.-S. Yeo and S.K. Kim, "MARP: Mobile Agent for RFID Privacy Protection," *Proceedings of CARDIS '06*, LNCS 3928, pp. 300–312, 2006.
15. E. Kosta, M. Meints, M. Hensen and M. Gasson, "An Analysis of Security and Privacy Issues Relating to RFID Enabled ePassports," *Proceedings of IFIP-SEC '07*, LNCS, to appear.
16. T.V. Le, M. Burmester and B. de Medeiros, "Universally Composable and Forward-Secure RFID Authentication and Authenticated Key Exchange," *Proceedings of ASIACCS '07*, pp. 242–252, 2007. Full version available at IACR ePrint Archive, `http://eprint.iacr.org/2007/051`, 14 February 2007.
17. C.H. Lim and T. Kwon, "Strong and Robust RFID Authentication Enabling Perfect Ownership Transfer," *Proceedings of ICICS '06*, LNCS 4307, pp. 1–20, 2006.
18. "Michelin Embeds RFID Tags in Tires," RFID Journal, 17 January, 2003. Available online at `http://www.rfidjournal.com/article/articleview/269/1/1/`.
19. "Mitsubishi Electric Asia Switches on RFID," RFID Journal, 11 September, 2006. Available online at `http://www.rfidjournal.com/article/articleview/2644/`.
20. J. Monnerat, S. Vaudenay and M. Vuagnoux, "About Machine-Readable Travel Documents: Privacy Enhancement using (Weakly) Non-Transferable Data Authentication," *Proceedings of RFIDSec '07*, pp. 15–28, 2007.
21. M. Ohkubo, K. Suzuki and S. Kinoshita, "RFID Privacy Issues and Technical Challenges," *Communications of the ACM*, Vol. 48, No. 9, pp. 66–71, 2005.
22. R.-I. Paise and S. Vaudenay, "Mutual Authentication in RFID: Security and Privacy," *Proceedings of AsiaCCS '08*, to appear.
23. P. Rogaway, "On the Role Definitions in and Beyond Cryptography," *Proceedings of Asian Computing Science Conference (Asian 2004)*, LNCS 3321, pp. 13-32, 2004.

24. C.C. Tan, B. Sheng and Q. Li, "Serverless Search and Authentication Protocols for RFID," *Proceedings of PerCom '07*, pp. 3–12, 2007.
25. "Target, Wal-Mart Share EPC Data," RFID Journal, 17 October, 2005. Available online at `http://www.rfidjournal.com/article/articleview/642/1/1/`.
26. G. Tsudik, "YA-TRAP: Yet Another Trivial RFID Authentication Protocol," *Proceedings of PerCom '06*, pp. 640–643, 2006.
27. S. Vaudenay, "RFID Privacy based on Public-Key Cryptography," *Proceedings of ICISC '06*, LNCS 4296, pp. 1–6, 2006.
28. S. Vaudenay, "On Privacy Models for RFID," *Advances in Cryptology - Asiacrypt '07*, LNCS 4833, pp. 68-87, 2007.