Provably Secure Ubiquitous Systems: Universally Composable RFID Authentication Protocols

Mike Burmester* Tri van Le* Breno de Medeiros

Florida State University
Department of Computer Science
Tallahassee FL 32306

e-mail: {burmester, levan, breno}@cs.fsu.edu

Abstract

This paper examines two unlinkably anonymous, simple RFID identification protocols that require only the ability to evaluate hash functions and generate random values, and that are provably secure against Byzantine adversaries.

The main contribution is a universally composable security model tuned for RFID applications. By making specific setup, communication, and concurrency assumptions that are realistic in the RFID application setting, we arrive at a model that guarantees strong security and availability properties, while still permitting the design of practical RFID protocols. We show that the two previously proposed protocols are provably secure within the new security model. Our proofs do not employ random oracles—the protocols are shown to be secure in the standard model under the assumption of existence of pseudo-random function families.

1 Introduction

Radio Frequency Identification Devices (RFIDs) were initially developed as very small electronic hardware components having as their main function to broadcast a unique identifying number upon request. The simplest types of RFIDs are *passive tags*, that do not contain a power source, and are incapable of autonomous activity. These devices are powered by the reader's radio waves, and the antenna doubles as a source of inductive power. The low cost and high convenience value of RFIDs give them a potential for massive deployment, and it is expected that they will soon outnumber all other computing device types. Consequently, RFIDs are increasingly used in applications that interface with information security functions.

RFIDs are a challenging platform from an information assurance standpoint. Their extremely limited computational capabilities imply that traditional distributed multi-party computation techniques for securing communication protocols are not feasible, and instead that lightweight approaches must be considered. Yet the privacy and security requirements of RFID applications can be quite significant. Ultimately, these should be accomplished with as rigorous a view of security as other types of applications.

The goal of this paper is to consider unlinkably anonymous authentication protocols for secure RFID applications that:

1. are provably secure under a strong adversarial model, and that remain secure under universal composition with arbitrary applications.

2. are computationally lightweight, taking into consideration the hardware-imposed constraints of the platform.

3. are scalable to a large volume of devices

*This research is supported by the National Science Foundation under grants CCR-0209092 and ANI-0087641.

1.1 Why UC-style security?

In this section, we argue that UC-style security is needed in the context of RFID applications. We illustrate with examples of RFID authentication protocols that are provably secure in restricted models and that yet fail to provide adequate security under realistic attacks.

The first example is based on variants of the HB protocol, introduced in [21]. The HB protocol can be shown to be secure against passive adversaries by reduction to the so-called "Learning Parity with Noise" (LPN) problem. To achieve security against active adversaries, the HB protocol was adapted to include challenges from both readers and tags, leading to the HB+ protocol, which can be proven secure against active adversaries [22] in a simplified model where the adversary is a malicious reader attacking a single honest tag. The proof has been generalized to a parallel and concurrent setting in [23], by showing that rewinding techniques in the original security proof ([22]) are not necessary. All of these security results are established in a simple attack model, and cannot be held as providing evidence that the scheme is secure in practical applications, where the adversary may communicate with both readers and tags simultaneously. Indeed, man-in-the-middle attacks do exist [18] and result in a total protocol break.

The second example is based on a new scheme, named YA-TRAP [28], that uses timestamps. As pointed out by G. Tsudik, an adversarial reader may provide an expurious future time as the current timestamp, causing the tag to become fully or temporarily incapacitated. We describe herein a solution (first introduced in [15]) that addresses this issue without additional computational burden on the tag and/or extra bandwidth requirements, solving an open problem posed by G. Tsudik in that paper. Furthermore, this and related issues are captured directly in our security model via the availability requirement for RFID protocols, as described in Section §3.

The above examples provide evidence that comprehensive security models are relevant in the context of RFID application. In this paper we formulate security in terms of indistinguishability between real and ideal protocol simulations. This formalism is based on the premise that one should first define an *ideal functionality* for the protocol—i.e., how to achieve security in an ideal world where the honest parties have a secure communication channel to a trusted party. Then, one constructs a reduction that maps real protocol runs to protocol runs in the ideal world, and shows that honest parties cannot distinguish real and ideal protocol executions. The above formulation was outlined by Beaver [5, 4, 3], and extended by Canetti as the universal composability framework [8, 9, 10]. It has also been ported to a modular, formal models-type approach called *reactive systems*, which emphasizes independent analysis of cryptography and communication layers, by Pfitzmann and Waidner [26, 27].

Formal modeling of protocols and cryptographic primitives via real vs. ideal simulations is an increasingly respected paradigm for the analysis of multiparty protocols, including authentication and key-exchange [13, 20, 12], zero-knowledge proofs [11, 14], and the universe of cryptographic primitives [24]. More recently, an RFID privacy-oriented protocol has been proven secure in a strong real/ideal setting [1].

Our contribution. The novel contributions of this paper are:

- A universally composable security model for RFID applications, described in Section §3, that provides availability, authentication, and anonymity guarantees. The model accommodates the analysis of practical protocols, such as the previously proposed O-TRAP & YA-TRAP+, described in sections §2 and §5, respectively.
- A security proof for O-TRAP within the proposed UC-type security framework, in Section §4. The security proof for YA-TRAP+ is similarly constructed, but not included for the sake of brevity.
- Discussions on further security properties of the protocols, such as the extensibility of O-TRAP and YA-TRAP+ to accommodate kill-keys while avoiding vulnerability to side-channel attacks, on Section §5.

2 O-TRAP: an optimistic, secure 1-pass anonymous RFID authentication protocol

Chatmon, le Van, and Burmester introduced in [15] an RFID authentication protocol. The protocol is opti-

mistic, i.e., the security overhead is minimal when the parties are honest. The protocol is herein referred to as O-TRAP, which stands for Optimistic, Trivial (R)FID Authentication Protocol.

An RFID authentication system has three components: tags T_i, readers R_j, and a trusted server $\mathcal{TS}$. Tags are wireless transponders: they have no power of their own and respond only when they are in an electromagnetic field. Readers are transceivers and generate such fields, which they use to wirelessly transmit challenges to tags. Readers may issue two types of challenges: multicast and unicast. Multicast challenges are addressed to all tags in the range of a reader, whereas unicast challenges are addressed to specific tags. In our protocols below we consider both types of challenges. Note that our multicast challenges are just random strings, and *all* tags in the range of a reader R_j are challenged with the same random string; this kind of action is *not* usually counted as a communication pass in an authentication protocol.

We shall assume that all honest tags T_i adhere to the system specifications and the requirements of the authentication protocol. The same applies to the honest readers R_j, and to the trusted server $\mathcal{TS}$. Tags are issued with individual private keys K_i which they share only with the trusted server $\mathcal{TS}$. These keys are used by the tags for authentication. We denote by $\mathcal{K}$ the set of all authorised keys, i.e. the set of keys issued by $\mathcal{TS}$. Without loss of generality, we may assume that these keys are chosen at random from $\{0,1\}^\tau$, where τ is a security parameter.

In our RFID authentication protocols we shall assume that honest readers R_j and the server $\mathcal{TS}$ are linked by a secure communication channel (reliable and authenticated).

2.1 Protocol description

We now describe O-TRAP, a 1-pass[1] optimistic protocol that authenticates RFID tags anonymously. Figure 1 shows the protocol messages and how authentication checks are performed.

In this protocol, each reader R_j broadcasts a random string r^t_{sys} obtained from the server $\mathcal{TS}$, and

[1]Again, there is a step in which the reader broadcasts the same challenge to all tags. If one counts this as a communication round then our protocol is 2-pass.

Figure 1. O-TRAP. (Table L shown in Figure 2.)

$\mathcal{TS}$ sends to R_j, who broadcasts: r^t_{sys}

1. $T_i \rightarrow R_j \rightarrow \mathcal{TS} : r_i, \;\; h = H_{K_i}(r^t_{sys}, r_i)$

T_i updates $r_i = H_{K_i}(r_i)$

$\mathcal{TS}$ accepts T_i (as $T_{i'}$) if:

- $\exists\, K_{i'} : (r_i, K_{i'}) \in L \;\wedge h = H_{K_{i'}}(r^t_{sys}, r_i)$, or
- $\exists\, K_{i'} \in \mathcal{K}$ s.t. $h = H_{K_{i'}}(r^t_{sys}, r_i)$.

$\mathcal{TS}$ updates $r_{K_{i'}} = H_{K_{i'}}(r_i)$ in the lookup table L.

updated at regular intervals. Each tag T_i (containing key K_i) in the range of R_j uses the same r^t_{sys}, but will combine it with a locally generated string r_i, and sends (broadcasts) to the reader R_j the MAC: $h = H_{K_i}(r^t_{sys}, r_i)$. Here $H_{K_i}(\cdot)$ is a keyed hash. We require that $\{H_K(\cdot)\}_K$ be a pseudo-random function family [19]. T_i computes the value of local string r_i by taking the MAC of its previous value, stored locally. The server $\mathcal{TS}$ also updates the value r_i in a local key look-table –see Figure 2. From this table,

Figure 2. The key lookup table L.

$strings$	r_1	r_2	$\cdots$	r_n
$keys$	K_1	K_2	$\cdots$	K_n

and the value r_i sent by T_i, $\mathcal{TS}$ can find a corresponding key $K_{i'}$ and check that the value h is that same as $H_{K_{i'}}(r^t_{sys}, r_i)$. If the tag T_i has not been challenged by an unauthorised reader, the value h will be correct. In this case the cost for both the tag and the server is two MACs. However, if the tag has most recently interacted with a malicious reader, the stored values will be out-of-sync. Then $\mathcal{TS}$ will have to exhaustively search through all keys $K \in \mathcal{K}$ to find the correct value and resynchronize. Note that in the dishonest case the extra computational cost is borne out by the server and not by the tag. In what follows, we show that O-TRAP is a strong authentication protocol that provably hides the identity of tags from eavesdroppers and malicious readers without requiring the tag to ever perform expensive public-key operations. In all cases, the tag only needs to compute two MACs to authenticate it-

self. In the honest case, this is also the protocol cost for the central server.

Theorem 1 *O-TRAP guarantees availability, anonymity, and secure authentication in the security framework defined in Section §3, under the assumption that the keyed hash function is chosen from a pseudo-random function family* $\{H_K(\cdot)\}_K$.

To the best of our knowledge, this is the first anonymous, strong RFID authentication protocol to be shown secure within a comprehensive adversarial model.

3 Security model

In this section we formalize the security definitions for RFID protocols. The model largely follows existing paradigms for security of general-purpose network protocols, but becomes specific to the context of RFID applications in two aspects. First, we consider *availability* explicitly, capturing security against unauthorized disabling of tags directly within the model.

Secondly, we restrict concurrency by prohibiting tags from executing more than one session at a time. Note that this is a restriction only on individual, honest tags—many honest tags can be executing concurrently. In addition, readers (whether honest or corrupt), the central server, and dishonest tags can execute multiple sessions simultaneously. Yet, the requirement that a single honest tag can participate only in one session at a time facilitates the design of concurrently secure protocols. As the restriction is a mild one, and in accordance with the capabilities of RFID technology, it is beneficial in that it enables designers of security protocols to concentrate on the crucial security aspects and on how to balance competing interests, such as requirements of low computational cost and low memory utilization.

Proof structure. Our proof consists of two stages. First, we provide a mathematical description of real protocol executions. Protocol runs in this model are called real world simulations. Next, we describe an idealized protocol model, wherein honest parties have access to a trusted party (ideal functionality). Security in the ideal world can be readily seen to follow from the behavior of the ideal functionality in simulations. Finally, we show that the environment cannot distinguish between real and ideal world simulations. The adversary is allowed to schedule the actions of honest parties, eavesdrop in communications, and interact with the environment in an arbitrary manner (Byzantine adversary).

3.1 Simulations

Initialization of honest parties. Both real and ideal honest parties are initialized as follows. The trusted server—symbolized by oracle $\mathcal{O}_S$—creates a database of keys $K_i, i = 1, \ldots, n$—choosing keys at random from $\{0,1\}^\tau$, where τ is a security parameter (provided as input to the initialization step). For simplicity, we do not consider dynamic corruption of tags, though this can be easily accommodated within our model and proofs. Instead, the adversary is initialized with a subset of the valid keys $K_{\ell+1}, \ldots, K_n$, and so the first ℓ keys correspond to honest tags. During real-world simulations, the adversary interacts with honest tag T_i by accessing oracle $\mathcal{O}_i$, which emulates the behavior of the honest tag with corresponding key K_i.

The initialization also requires, for each ordered pair (i,j), $1 \leq i < j \leq \ell$, that one chooses two bits $b^1_{i,j}$ and $b^2_{i,j}$, independently at random. For each triple (i,j,c), with $c \in \{1,2\}$, an *ambivalent oracle* $\mathcal{O}^c_{i\vee j}$ will use key K_i or K_j in the simulation, respectively, if $b^c_{i,j} = 0$ or $b^c_{i,j} = 1$. The role of the ambivalent oracles will soon be made clear.

As the simulation starts, each tag oracle or ambivalent oracle is marked as `available`. Each tag oracle or ambivalent oracle independently initializes values r_i, $r^c_{i\vee j}$ at random. The server $\mathcal{O}_S$ generates a random value r^0_{sys} which will be broadcast by readers as challenge to tags during the first *server period*, or simply *period*. Subsequently, the adversary may cause new periods to commence by telling $\mathcal{O}_S$ to refresh the value r^t_{sys} with a new random value, where t counts how many periods have completed before the current one.

3.1.1 Real simulation model

Let $\mathcal{A}$ be the adversary. $\mathcal{A}$ can internally represent many adversarial tags T' (with compromised valid keys or invalid keys) and dishonest readers R', but we represent it as a single party $\mathcal{A}$.

At the beginning of the simulation, the total number of tags n is provided to the adversary. The adversary interacts in an arbitrary manner with the simulation environment $\mathcal{Z}$. Consider a single communication session between $\mathcal{A}$ and some honest party. $\mathcal{Z}$ maintains a notion of time—we do not require synchronized clocks, $\mathcal{Z}$ only needs to discern which adversarial actions precede other adversarial actions. We now describe what types of messages can be understood by honest parties in the real protocol simulation. We note that, since individual tags execute sequentially, they are not always available to initiate new communication sessions with $\mathcal{A}$, for instance if already communicating with $\mathcal{A}$.

REFRESH(): Called by $\mathcal{A}$ to cause the beginning of a new server period. $\mathcal{O}_S$ increments the period counter ($t \leftarrow t+1$) and generates a new random value r^t_{sys}. This value will be broadcast by honest readers as challenge to tags, until the beginning of the next server period—i.e., until another call to REFRESH() occurs.

START(i): If $\mathcal{O}_i$ is not `available`, this call is ignored. Otherwise $\mathcal{O}_i$ changes status to `communicating` with $\mathcal{A}$, and all oracles of the type $\mathcal{O}^c_{i\vee?}$ are marked as `unavailable`.

START($i \vee j, c$): If $\mathcal{O}^c_{i,j}$ is not `available`, this call is ignored. Otherwise $\mathcal{O}^c_{i,j}$ changes status to `communicating` with $\mathcal{A}$, and $\mathcal{O}_i$, $\mathcal{O}_j$ become `unavailable`, as well as all *other* oracles of the type $\mathcal{O}^{c'}_{i\vee?}$, $\mathcal{O}^{c'}_{j\vee?}$, with $c' \in \{1,2\}$.

SEND(i, m): If $\mathcal{O}_i$ is not `communicating` with $\mathcal{A}$ this call is ignored. Otherwise, $\mathcal{O}_i$ responds with the pair $r_i, h = H_{K_i}(m, r_i)$, and updates $r_i \leftarrow H_{K_i}(r_i)$.

SEND($i \vee j, c, m$): If $\mathcal{O}^c_{i\vee j}$ is not `communicating` with $\mathcal{A}$ this call is ignored. Otherwise, let ι be either i or j, corresponding to whether $\mathcal{O}^c_{i\vee j}$ was initialized with key K_i or K_j, respectively. Then $\mathcal{O}^c_{i\vee j}$ responds with the pair $r^c_{i\vee j}, h = H_{K_\iota}(m, r^c_{i\vee j})$, and updates $r^c_{i\vee j} \leftarrow H_{K_\iota}(r^c_{i\vee j})$.

SEND($\mathcal{TS}, m$): $\mathcal{O}_S$ parses m as a string $r||h$. It then consults its lookup table for an entry of the type (r, K_i). If such an entry is found, $\mathcal{O}_S$ further checks if $h = H_{K_i}(r^t_{sys}||r)$, replying to $\mathcal{A}$ with 1 (indicating authentication success) if the equality holds. If either a match is not found or the check fails, $\mathcal{O}_S$ searches its key database $\mathcal{K}$ for any K_i such that $h = H_{K_i}(r^t_{sys}||r)$. If such K_i is found, it replies to $\mathcal{A}$ with 1, or 0 otherwise. $\mathcal{O}_S$ outputs the identity i if the authentication is successful with key K_i, else it outputs nothing. This output is not observed by the environment $\mathcal{Z}$.

END(i): If $\mathcal{O}_i$ is not `communicating`, the call is ignored. Otherwise, $\mathcal{O}_i$ becomes `available`, as well as any $\mathcal{O}^c_{i\vee j}$ such that $\mathcal{O}_j$ is also `available`.

END($i \vee j, c$): If $\mathcal{O}^c_{i\vee j}$ is not `communicating`, the call is ignored. Otherwise, $\mathcal{O}^c_{i\vee j}$ becomes `available`, as well as $\mathcal{O}_i$, $\mathcal{O}_j$, and any $\mathcal{O}^{c'}_{i\vee?}$, $\mathcal{O}^{c'}_{j\vee?}$, for $c' \in \{1,2\}$.

The role of the identity-ambivalent oracles. The ambivalent oracles $\mathcal{O}^c_{i\vee j}$ enable $\mathcal{A}$ to interact with parties whose identity is one of two possible choices. This enables attacks against anonymity, where $\mathcal{A}$'s objective is to determine if $\mathcal{O}^1_{i\vee j}$ and $\mathcal{O}^2_{i\vee j}$ represent the same or different identities. Note that the concurrency-prevention rules (enforced via the tags maintaining a status among `available`, `communicating`, and `unavailable`) are designed to prevent that $\mathcal{A}$ may disambiguate the ambivalent oracles simply on the basis of availability conflicts, while at the same time preventing that a single tag executes two sessions concurrently.

3.2 Security definitions

We now formally define the security goals of anonymous authentication protocols. We define a session ses with honest tag T_i as a time-interval between the first call to START(i) after either the beginning of the simulation or the most recent call to END(i), and the first subsequent call to END(i).

Availability holds when there is no efficient adversary $\mathcal{A}$ that during the course of the simulation, has non-negligible probability in preventing a tag T_i from authenticating itself to a reader R_j during a session

ses, without changing T_i's interaction with R_j in session ses. This should remain true even if $\mathcal{A}$ has interacted with T_i or $\mathcal{TS}$ arbitrarily in the past, perhaps attempting to force either or both into an inconsistent state. Note that $\mathcal{A}$ is still allowed to interact with all other honesty parties, including reader R_j, during ses. The advantage $adv^{\mathcal{A},T_i}_{AVLB}$ of $\mathcal{A}$ in this game against T_i is the maximum probability that $\mathcal{TS}$ rejects T_i in any session.

$$adv^{\mathcal{A},T_i}_{AVLB} := Prob[\mathcal{TS} \text{ rejects } T_i \text{ in } ses \mid \mathcal{A} \text{ only relays between } \mathcal{O}_i \text{ and } R_j \text{ during } ses],$$

and $adv^{\mathcal{A}}_{AVLB}$ is defined as the maximum of the $adv^{\mathcal{A},T_i}_{AVLB}$, over all honest tags T_i in any session.

An important concern in regard to the management of RFIDs is to have a kill process, in which a reader can instruct an RFID tag to disable its functionality permanently. Current methods for disabling EPC tags have been recently shown ([25]) to allow an attacker to perform a power-analysis based recovery of the kill-key. Such attacks violate the above definition of availability. Our protocols can be adapted to support a kill-key while still guaranteeing availability, as discussed in Section §5.

Authentication holds when there is no efficient adversary that, during the simulation, succeeds with non-negligible probability in authenticating itself to an honest reader R_j during some session ses, and moreover: (a) The server $\mathcal{TS}$ believes $\mathcal{A}$ to have authenticated itself as tag T_i in ses; and (b) the duration interval [start-time, end-time] for session ses is disjoint from the duration intervals of all of $\mathcal{A}$'s sessions with oracle $\mathcal{O}_i$ as well as with any ambivalent oracle $\mathcal{O}^c_{i,j}$ that was initialized as $\mathcal{O}_i$. We note that in this definition, $\mathcal{A}$ is not required to know under which identity T_i it has succeeded in authenticating itself. Furthermore, it accommodates man-in-the-middle attacks, as long as the attack leads to $\mathcal{A}$'s acquiring knowledge (such as keys) that can be used for subsequent authentication attempts, while ruling out scenarios in which the adversary simply relays messages between honest parties as successful attacks. The advantage $adv^{\mathcal{A},T_i}_{\mathsf{AUTH}}$ of the adversary against authentication is simply the probability that it succeeds.

$$adv^{\mathcal{A},T_i}_{\mathsf{AUTH}} := Prob[\ \mathcal{A} \text{ authenticates as } T_i \text{ in } ses; ses \cap \text{Sessions}(\mathcal{A}, \mathcal{O}_i) = \emptyset],$$

where i is the index of an honest user. The advantage $adv^{\mathcal{A}}_{\mathsf{AUTH}}$ it the maximum of the $adv^{\mathcal{A},T_i}_{\mathsf{AUTH}}$ over all tags T_i.

Anonymity holds when no efficient adversaries have non-negligibly better-than-even chances of, at any time in the simulation, outputting a triple (i, j, b), where $1 \le i < j \le n$, and either (1) $b = 0$ and $\mathcal{O}^1_{i\vee j} \neq \mathcal{O}^2_{i\vee j}$, or (2) $b = 1$ and $\mathcal{O}^1_{i\vee j} = \mathcal{O}^2_{i\vee j}$. The advantage of the adversary in distinguishing T_i and T_j, $Adv^{\mathcal{A},i\vee j}_{\mathsf{ANON}}$, is defined as the difference between winning and losing probabilities when the adversarial guess bit equals 1:

$$adv^{\mathcal{A},i\vee j}_{\mathsf{ANON}} := Prob[(i,j,1) \leftarrow \mathcal{A} \mid \mathcal{O}^1_{i\vee j} = \mathcal{O}^2_{i\vee j}] - Prob[(i,j,1) \leftarrow \mathcal{A} \mid \mathcal{O}^1_{i\vee j} \neq \mathcal{O}^2_{i\vee j}],$$

and the adversarial advantage against anonymity, $adv^{\mathcal{A}}_{\mathsf{ANON}}$ is the maximum of the $adv^{\mathcal{A},i\vee j}_{\mathsf{ANON}}$ over all pairs (i, j), with $i < j$.

This is a unified framework because the adversary does not need to identify, at any particular point in the simulation, which security property it seeks to defeat. Instead, it may weigh its knowledge and adjust its strategy during the simulation to maximize its success in violating any of the security requirements.

3.2.1 Ideal simulation

Recall that we number server periods with counter t. Within each period, the ideal functionality $\mathcal{F}$ maintains a database of the messages generated by all honest (non-adversarial) tags.

In the ideal simulation, honest parties are represented by the ideal functionality $\mathcal{F}$. We now describe $\mathcal{F}$'s behavior in interactions between the oracles and the ideal adversary $S_{\mathcal{A}}$—in the UC framework, the ideal adversary $S_{\mathcal{A}}$ reproduces actions of $\mathcal{A}$ in the real world. We have several cases to consider, depending on which messages $S_{\mathcal{A}}$ sends to $\mathcal{F}$. Initialization is identical to real world simulations.

REFRESH(): $S_{\mathcal{A}}$ causes $\mathcal{F}$ to start a new period t, generating a fresh random value r^t_{sys}, and sending r^t_{sys} to $S_{\mathcal{A}}$. It resets its database, so that it is empty at the start of the period.

START(i), START($i \vee j, c$), END(i), END($i \vee j, c$): $\mathcal{F}$ annotates some oracles as `available`,

`communicating`, or `unavailable` exactly as in the real world simulation.

SEND(i, m): If $\mathcal{O}_i$ is not `communicating` with $S_\mathcal{A}$, $\mathcal{F}$ ignores the message m. Otherwise, $\mathcal{F}$ generates a new random value r, and returns r to $S_\mathcal{A}$. If $m = r^t_{sys}$, then $\mathcal{F}$ also stores the pair (i, r) in its database.

SEND$(i \vee j, c, m)$: If $\mathcal{O}^c_{i\vee j}$ is not `communicating` with $S_\mathcal{A}$, the message m is ignored. Otherwise, $\mathcal{F}$ generates random value r (returned to $S_\mathcal{A}$) and if $m = r^t_{sys}$, it stores the triple $(i \vee j, c, r)$ in its database.

SEND$(\mathcal{TS}, m)$: First, $\mathcal{F}$ checks if there exists a tuple (i, m) or triple $(i \vee j, c, m)$ in its database. If so, it replies to $S_\mathcal{A}$ with the bit value 1. Otherwise, $\mathcal{F}$ parses m as a string $r||h$. For each key K_i, $i = \ell + 1, \ldots, n$, $\mathcal{F}$ checks if $h = H(K_i, s^t||r)$. If any match is found, it replies to $S_\mathcal{A}$ with 1, or 0 otherwise. Finally, if authentication was successful (1 returned) for key K_i, $\mathcal{F}$ computes $\mathcal{O}_S$'s private output as i.

3.3 Security in the ideal simulation

Availability: If the ideal adversary makes a call to SEND$(\mathcal{TS}, m)$ in period t, where m is a value returned by $\mathcal{F}$ as a result of a call to SEND(i, r^t_{sys}), then $\mathcal{F}$ returns 1.

Authentication: $\mathcal{F}$ only returns 1 if the ideal adversary $S_\mathcal{A}$ calls SEND$(\mathcal{TS}, m)$ in period t, where m was either computed as $m \leftarrow r||h$, where $h = H(K_i, r^t_{sys}||r)$, and K_i is an adversary-controlled key ($i > \ell$); or if m was itself a value produced by $\mathcal{F}$ as an honest-tag response to a call SEND(i, r^t_{sys}) by $S_\mathcal{A}$.

Anonymity: It is obvious, as the values returned by honest parties are generated independently at random by $\mathcal{F}$. Therefore, the ideal adversary is only able to distinguish between parties by checking for simultaneous unavailability/availability. However, the rules preventing concurrent execution of ambivalent oracles and tag oracles are designed to prevent this.

4 Security reduction for O-TRAP

We prove this theorem by constructing, for each real adversary $\mathcal{A}$, an ideal adversary $S_\mathcal{A}$, such that no probabilistic polynomial time environment $\mathcal{Z}$ can distinguish an execution of $S_\mathcal{A}$ in the ideal world from an execution of $\mathcal{A}$ in the real world.

We accomplish this by replacing all oracle calls of $\mathcal{A}$ to honest parties in the real world simulation into identically-named calls of $S_\mathcal{A}$ to $\mathcal{F}$ in the ideal world one. We then show that $\mathcal{Z}$ cannot distinguish between values returned by $\mathcal{F}$ in the ideal simulation form values returned by honest parties' oracles in the real world simulation.

First, note that calls to START$(\cdot)$, END$(\cdot)$, and REFRESH$()$ have identical observable effects in the real and ideal worlds.

A SEND(i, m) call by $\mathcal{A}$ in the real world results in $\mathcal{O}_i$ returning the pair $r, H(K_i, m||r)$, where r is a new pseudo-random value generated by $\mathcal{O}_i$. A SEND(i, m) call by $S_\mathcal{A}$ in the ideal world results in a true random pair of same length being returned. These pairs are indistinguishable by $\mathcal{Z}$—who does not have the key K_i—due to the pseudo-randomness of the function family $\{H(K, \cdot)\}_K$.

A SEND$(\mathcal{TS}, m)$ call by $\mathcal{A}$ in the real world results in the value 1 being returned if m can be parsed as $r||h$, where $h = H(K_i, r^t_{sys}||r)$, for some $K_i, i = 1, \ldots, n$. We distinguish two subcases:

(a) $i \leq \ell$ (honest tags), and

(b) $i > \ell$ (adversarial tags).

A SEND$(\mathcal{TS}, m)$ call by $S_\mathcal{A}$ in the ideal world results in 1 if either:

(c) There is an entry of the type (i, m) or $(i \vee j, c, m)$ in $\mathcal{F}$'s database, where $1 \leq i \leq \ell$; or

(d) m can be parsed as $r||h$, where $h = H(K_i, r^t_{sys}||r)$, for some K_i, where $\ell < i \leq n$.

We now consider the cases where the outcomes for $\mathcal{A}$ and $S_\mathcal{A}$ differ, and argue that these only happen with negligible probability.

First, note that cases (b) and (d) correspond exactly. Now, suppose that case (a) occurs in the real world, but (c) does not occur in the ideal world. In this

case, $\mathcal{A}$ was able to compute an authentication value $r, H(K_i, r^t_{sys}||r)$, with $i \leq \ell$, without obtaining this from an oracle call to $\mathcal{O}_i$ or some $\mathcal{O}^c_{i \vee j}$—otherwise (c) would occur in the ideal world. This may only happen with negligible probability since $\mathcal{A}$ does not have the key K_i and the function family $\{H(K, \cdot)\}_K$ is assumed pseudo-random.

Secondly, if (c) happens in the ideal world without (a) happening in the real world, then it follows that some real world computation by $\mathcal{A}$ that does not result in a valid authentication value (in the real world) happens to serendipitously reproduce a correct value when executed by $S_{\mathcal{A}}$ in the ideal world. Since the ideal world values are chosen truly at random, this can only happen with negligible probability. It follows that (a) and (c) correspond to each other with overwhelming probability.

Finally, the authentication outcomes could be identical for $\mathcal{A}$ and $S_{\mathcal{A}}$ and yet the authenticated identities output by $\mathcal{O}_S$ in the real and ideal worlds might differ. Clearly, if this happens with one of the identities corresponding to an honest party, it would imply either a collision between outputs of two independent pseudo-random functions (causing identity mismatch in the real world) or that a pseudo-random function matched a random value chosen by $\mathcal{F}$ (causing identity mismatch in the ideal world). Both cases can only happen with negligible probability.

5 Extensions

In this section we describe a 2-pass optimistic RFID authentication protocol, introduced in [15], that addresses most of the drawbacks of the authentication protocols in [2, 16, 28, 17, 29], and that also thwarts power analysis attacks. The protocol is an extension of YA-TRAP—Yet Another Trivial Authentication Protocol, proposed by G. Tsudik [28].

We also discuss how to accommodate kill-keys in both O-TRAP and YA-TRAP+ without introducing side-channel vulnerabilities (such as power analysis attacks).

5.1 YA-TRAP+

We describe the extension YA-TRAP+ of YA-TRAP. This extension includes one extra optional pass, to deal with large scale DoS attacks. In the first step the tag is authenticated, whereas in the second optional step the server authenticates the timestamp. The protocol is given in Figure 3 ([15]).

Figure 3. YA-TRAP+

$\mathcal{TS}$ sends to R_j, who broadcasts (t, r^t_{sys}).

1. $T_i \rightarrow R_j \rightarrow \mathcal{TS}$: $r_i,\ h_1 = H_K(00||t||r^t_{sys})$, if $t > t_i$.
   $r_i,\ h_1 = H_K(01||r_i||r^t_{sys})$, if $t \leq t_i$.
2. $\mathcal{TS} \rightarrow R_j \rightarrow T_i$: $r_i,\ h_2 = H_K(10||r_i||t)$, (optional).

– $\mathcal{TS}$ accepts T_i as authentic only if $\exists\ K \in \mathcal{K}$:
$(h_1 = H(K, 00||t||r^t_{sys}))$ or
$(h_1 = H(K, 01||r_i||r^t_{sys}))$.

– T_i verifies that $h_2 = H(K, 10||r_i||t)$ (optional)

– T_i sets $t_i = t$ if $t > t_i$.

Pass 2 is optional, used by the server during a time period when the number of attacks that occur is beyond a certain threshold and the server would like to resynchronize the correct timestamp t for all the tags. The optional pass is used with all tags during such a time period so that no identity information is revealed. When this period is over, the server may return to normal 1-pass authentication. This makes the scheme resistant to DoS while being almost as efficient as the YA-TRAP protocol.

We obtain the following new result for YA-TRAP+, whose proof will be provided in a full version of this paper:

Theorem 2 *YA-TRAP+ guarantees availability, anonymity, and secure authentication in the security framework defined in Section §3, under the assumption that the keyed hash function is chosen from a pseudo-random function family* $\{H_K(\cdot)\}_K$.

6 Further Research

6.1 Kill-keys

Both O-TRAP and YA-TRAP+ may be extended to accommodate kill-keys.

In the case of YA-TRAP+, the disabling mechanism is very simple. The server executes the authentication protocol with $t > t^i_{max}$—known to $\mathcal{TS}$—and executes the extra optional step. The tag accepts t as valid, and becomes disabled.

To accomplish the same with O-TRAP, the protocol is modified so that each tag T_i is initialized with two keys, the authentication tag K_i and the kill-key $\hat{K}_i$, which is computed as $\hat{K}_i = H_{K_i}(f_i)$, for some value f_i stored by $\mathcal{TS}$.

To disable T_i, $\mathcal{TS}$ sends a special command $\mathsf{KILL}(r)$. The tag computes $H_{K_i}(r)$, and if that matches its stored $\hat{K}_i$, the tag becomes disabled. Otherwise, it does nothing.

6.2 Corruption of tags

Forward secrecy can be addressed by having the tags and the trusted server update their "long term" private keys. In this case both the tag and the trusted server must be mutually authenticated, e.g. by having an extra pass.

6.3 Timing attacks

In our security model the tags and the trusted server take exactly one computation step between sending and receiving authentication data. A secure implementation should reflect this semantic. In particular the time taken for each pass must be constant. This can be done by inserting an artificial delay on the trusted server. This does not effect the throughput and workload of the server, which is the objective of our scalable optimistic protocols.

7 Conclusion

We present a new, universally composable framework to study the security of RFID authentication protocols. Two optimistic, anonymous RFID authentication protocols, O-TRAP and YA-TRAP+ (both introduced in [15]) are proven secure in the new framework. In addition, we propose extensions of both protocols to provide for access-controlled tag disabling (kill-keys) in a way that tolerates side-channel attacks.

As future work, we plan to formally examine the possibility of extending this security model to examine DoS resilience and/or to incorporate specific descriptions of side-channel information leakage.

References

[1] G. Ateniese, J. Camenisch, and B. de Medeiros. Untraceable RFID tags via insubvertible encryption. In *Proc. ACM Conf. on Computer and Communication Security (ACM CCS 2005)*, pages 92–101. ACM Press, 2005.

[2] G. Avoine and P. Oechslin. A scalable and provably secure hash based RFID protocol. In *Proc. IEEE International Workshop on Pervasive Computing and Communication Security (PerSec 2005)*. IEEE Computer Society Press, 2005.

[3] D. Beaver. Foundations of secure interactive computing. In *Proc. Advances in Cryptology (CRYPTO 1991)*, volume 576 of *LNCS*, pages 377–391. Springer, 1991.

[4] D. Beaver. Secure multi-party protocols and zero-knowledge proof systems tolerating a faulty minority. *Journal of Cryptology*, 4:2:75–122, 1991.

[5] D. Beaver and S. Goldwasser. Multiparty computation with faulty majority. In *Proc. Advances in Cryptology (CRYPTO 1989)*, volume 435 of *LNCS*, pages 589–590. Springer, 1989.

[6] M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Proc. Advances in Cryptology (CRYPTO 1993)*, volume 773 of *LNCS*. Springer, 1994.

[7] R. Bird, I. Gopal, A. Herzberg, P. Janson, S. Kutten, R. Molva, and M. Yung. Systematic design of two-party authentication protocols. In *Proc. Advances in Cryptology (CRYPTO 1991)*, LNCS. Springer, 1991.

[8] R. Canetti. *Studies in Secure Multiparty Computation and Application*. PhD thesis, Weizmann Institute of Science, Rehovot 76100, Israel, June 1995.

[9] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13:1:143–202, 2000.

[10] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proc. IEEE Symp. on Foundations of Computer Science (FOCS 2001)*, pages 136–145. IEEE Press, 2001.

[11] R. Canetti and M. Fischlin. Universally composable commitments (extended abstract). In *Proc. Advances in Cryptology (CRYPTO 2001)*, volume 2139 of *LNCS*, page 19. Springer, 2001.

[12] R. Canetti and J. Herzog. Universally composable symbolic analysis of cryptographic protocols (the case of encryption-based mutual authentication and key exchange). Technical Report E-print Report # 2004/334, International Association for Cryptological Research, 2004.

[13] R. Canetti and H. Krawczyk. Universally composable notions of key exchange and secure channels (extended abstract). In *Proc. Advances in Crypotology (EUROCRYPT 2002)*, volume 2332 of *LNCS*, page 337. Springer, 2001.

[14] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *Proc. ACM Symp. on Theory of Computing (STOC 2002)*, volume 34, pages 494–503. ACM Press, 2002.

[15] C. Chatmon, T. van Le, and M. Burmester. Anonymous authentication with RFID devices. Technical Report TR-060112, Florida State University Computer Science Dept., 2006. Submitted for publication.

[16] T. Dimitriou. A lightweight RFID protocol to protect against traceability and cloning attacks. In *Proc. IEEE Intern. Conf. on Security and Privacy in Communication Networks (SECURECOMM 2005)*. IEEE Press, 2005.

[17] T. Dimitriou. A secure and efficient RFID protocol that can make big brother obsolete. In *Proc. Intern. Conf. on Pervasive Computing and Communications, (PerCom 2006)*. IEEE Press, 2006.

[18] H. Gilbert, M. Rodshaw, and H. Sibert. An active attack against HB+ – a provably secure lightweight authentication protocol. Technical report, International Association for Cryptological Research, 2005.

[19] O. Goldreich. *The Foundations of Cryptography*, volume 1. Cambridge University Press, 2001.

[20] D. Hofheinz, J. Müller-Quade, and R. Steinwandt. Initiator-resilient universally composable key exchange. In *Proc. European Symp. on Research in Computer Security (ESORICS 2003)*, volume 2808 of *LNCS*, pages 61–84. Springer, 2003.

[21] N. J. Hopper and M. Blum. Secure human identification protocols. In *Proc. Advances in Crypotology (ASIACRYPT 2001)*, volume 2248 of *LNCS*. Springer, 2001.

[22] A. Juels and S. A. Weis. Authenticating pervasive devices with human protocols. In *Proc. Advances in Cryptology (CRYPTO 2005)*, volume 3621 of *LNCS*, page 293. Springer, 2005.

[23] J. Katz and J. S.Shin. Parallel and concurrent security of the HB and HB+ protocols. In *Proc. Advances in Crypotology (EUROCRYPT 2006)*, LNCS. Springer, 2006.

[24] P. Laud. Formal analysis of crypto protocols: Secrecy types for a simulatable cryptographic library. In *Proc. ACM Conf. on Computer and Communication Security (ACM CCS 2005)*, pages 26–35. ACM Press, 2005.

[25] Y. Oren and A. Shamir. Power analysis of RFID tags. Invited talk, RSA Conference, Cryptographer's Track (RSA-CT 2006). Available at `http://www.wisdom.weizmann.ac.il/~yossio/rfid`, 2006.

[26] B. Pfitzmann and M. Waidner. Composition and integrity preservation of secure reactive systems. In *Proc. ACM Conf. on Computer and Communication Security (ACM CCS 2000)*, pages 245–254. ACM Press, 2000.

[27] B. Pfitzmann and M. Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *Proc. IEEE Symp. on Security and Privacy (S & P 2001)*, pages 184–200. IEEE Press, 2001.

[28] G. Tsudik. YA-TRAP: Yet another trivial RFID authentication protocol. In *Proc. IEEE Intern. Conf. on Pervasive Computing and Communications (PerCom 2006)*. IEEE Press, 2006.

[29] I. Vajda and L. Buttyan. Lightweight authentication protocols for low-cost RFID tags. In *Proc. Workshop on Security in Ubiquitous Computing (UBICOMP 2003)*, 2003.