

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 two 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.

I. 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 in 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.

A. What is UC security?

The universal composability (UC) framework specifies a particular approach to security proofs, and guarantees that proofs that follow that approach remain valid if the protocol is composed with others (modularity) and under arbitrary concurrent protocol executions (including with itself).

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

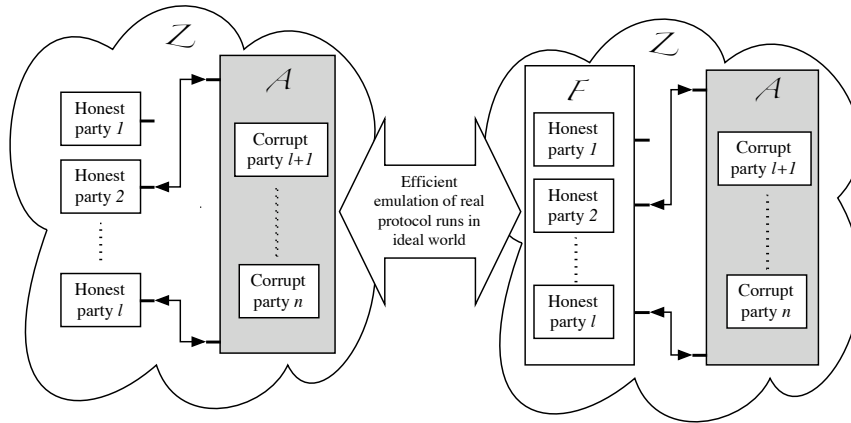


Fig. 1. Architecture of UC-security proofs.

The UC framework defines a real world simulation, an ideal world simulation, an emulation that translates runs from the real world to the ideal world, and an interactive environment \mathcal{Z} that captures whatever is external to the current protocol execution.

In the ideal world, the communications between the parties are intermediated by an ideal (incorruptible) functionality. Security follows not from cryptographic properties of the protocol, but because the channels between honest parties and the ideal functionality are absolutely secure and the functionality ensures the desirable system behavior. Because security in the ideal world does not depend on cryptographic assumptions and properties, protocols in the ideal world are concurrent and compositionally secure.

In the real world, there is no trusted party and honest parties communicate using the specified protocol rules. The adversary is only restricted by computational bounds (probabilistic polynomial-time) but can deviate from the protocol specification in any way, corrupt multiple parties and coordinate their malicious behavior, etc. Moreover, the adversary can interact with the environment arbitrarily, and in particular it is capable of manipulating all communication channels.

For UC security, it is necessary to show that for all adversaries, any environment \mathcal{Z} that can distinguish the real world protocol execution from its ideal world emulation in the presence of an adversary, can also use this distinguishing capability to break specific cryptographic assumptions. The main feature of the UC framework is that the UC security of a composite system can be derived from the UC security of its components without need for holistic re-assessment.

B. Why use UC security?

In this section, we argue that universally composable 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 HB+ ([1]), a variant of the HB protocol introduced in [2]. The HB protocol can be shown to be secure against passive adversaries by reduction to the “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 [1] 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 [3], by showing that rewinding techniques in the original security proof ([1]) 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 [4] and result in a total protocol break.

The second example is based on a new scheme, named YA-TRAP [5], 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 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 §III. The above examples provide evidence that comprehensive security models are relevant in the context of RFID applications.

Perhaps a more compelling argument for a UC model is that because RFIDs offer only very basic functionalities, they are more likely to be used as components of more complex systems, possibly ubiquitous. UC is the only security model that allows for a modular approach for the design of protocols, using pre-designed, off-the-shelf components.

In this paper we formulate protocol security in terms of indistinguishability between real and ideal protocol simulations. This formalism was informally proposed by Goldwasser et al. [6], and more properly specified by Beaver et al. [7], [8], [9]. Canetti was the first to consider computationally bounded adversaries in this setting, establishing the *universal composability framework* [10], [11], [12].

Formal modeling via real vs. ideal simulations is being increasingly used in the analysis of cryptographic protocols, including for authentication and key-exchange [13], [14], [15], for zero-knowledge proofs [16], [17], and for the universe of cryptographic primitives [18]. More recently, an RFID privacy-oriented protocol has been proven secure in a strong real/ideal setting [19].

Our contribution:

- A universally composable security model for RFID applications, described in Section §III, that provides availability, authentication, and anonymity guarantees. The model accommodates the analysis of practical protocols, such as O-TRAP & YA-TRAP+, described in sections §II and §V, respectively.
- A security proof for O-TRAP within the proposed UC-type security framework, in Section §IV. 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 extendability of O-TRAP and YA-TRAP+ to accommodate kill-keys while avoiding vulnerability to side-channel attacks, on Section §V.

II. O-TRAP: an optimistic, secure anonymous RFID authentication protocol

The protocol we now describe will be referred to as O-TRAP, which stands for Optimistic, Trivial (R)FID Authentication Protocol. The protocol is *optimistic* in the sense that the security overhead is minimal when the adversary is passive (eavesdropper) or inactive.

An RFID authentication system has three components: tags T_i , readers R_j , and a trusted server \mathcal{S} . 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.³ 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 is more communication-efficient than multiple unicast challenges. The server periodically updates the value of the common, random challenge. During each period a tag may be authenticated at most once.

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{S} . Tags are issued with individual private keys K_i which they share only with the trusted server \mathcal{S} . 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{S} . 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{S} are linked by a secure communication channel (reliable and authenticated).

A. Protocol description

We now describe O-TRAP, a 2-pass¹ optimistic protocol that authenticates RFID tags anonymously. Figure 2 shows the protocol messages and how authentication checks are performed.

Fig. 2. O-TRAP.

<ol style="list-style-type: none"> 1. \mathcal{S} sends to R_j, who broadcasts: r_{sys}^t 2. $T_i \rightarrow R_j \rightarrow \mathcal{S} : r_i, h = H_{K_i}(r_{sys}^t, r_i)$
<p>T_i updates $r_i \leftarrow H_{K_i}(r_i)$</p> <p>$\mathcal{S}$ accepts T_i (as $T_{i'}$) if:</p> <ul style="list-style-type: none"> • $\exists K_{i'} : (r_i, K_{i'}) \in L \wedge h = H_{K_{i'}}(r_{sys}^t, r_i)$, or • $\exists K_{i'} \in \mathcal{K}$ s.t. $h = H_{K_{i'}}(r_{sys}^t, r_i)$. <p>$\mathcal{S}$ updates $r_{K_{i'}} \leftarrow H_{K_{i'}}(r_i)$ in lookup table L (Fig. 3).</p>

In this protocol, each reader R_j broadcasts a random string r_{sys}^t obtained from the server \mathcal{S} , and updated at regular intervals. Each tag T_i (containing key K_i) in the range of R_j uses the same r_{sys}^t , but will combine it with a locally generated string r_i , and sends (broadcasts) to the reader R_j : r_i and $h = H_{K_i}(r_{sys}^t, r_i)$. We require

¹In the first step the reader broadcasts the same challenge to all tags. This is a cheaper than a regular communication pass.

that $\{H_K(\cdot)\}_K$ be a pseudo-random function (PRF) family [20], e.g., a keyed hash function. T_i computes the value of local string r_i by computing the pseudo-random function of its previous value, stored locally. The server \mathcal{S} also updates the value r_i in a local key lookup table—see Fig. 3.

Fig. 3. The key lookup table L .

<i>strings</i>	r_1	r_2	\cdots	r_n
<i>keys</i>	K_1	K_2	\cdots	K_n

From this table, and the value r_i sent by T_i , \mathcal{S} can find a corresponding key $K_{i'}$ and check that the value h is that same as $H_{K_{i'}}(r_{sys}^t, r_i)$. 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 PRFs computations. However, if the tag has most recently interacted with a malicious reader, the stored values will be out-of-sync. Then \mathcal{S} will have to exhaustively search through all keys $K \in \mathcal{K}$ to find the correct value and resynchronize. Note that in the presence of active attacks, 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 PRFs to authenticate itself. In the case of a passive adversary, this is also the protocol cost for the trusted server.

Theorem 1: O-TRAP guarantees availability, anonymity, and secure authentication in the security framework defined in Section §III, 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.

III. 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 trusted 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), and an emulation that translates real world protocol runs to ideal world ones. Security in the ideal world can be readily seen to follow from the behavior of the ideal functionality in simulations. Finally, we show security in the real world by proving that the environment \mathcal{Z} cannot distinguish between real and ideal world simulations, for any adversary.

Adversaries are allowed to schedule the actions of honest parties, to eavesdrop in all communications, and to interact with the environment in an arbitrary manner (Byzantine adversaries).

A. 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 $K_i, i = 1, \dots, n$, choosing keys at random from $\{0, 1\}^\tau$, where τ is a security parameter and $n = \text{poly}(\tau)$. In this paper, we do not consider dynamic corruption of tags; we plan to consider this issue and its impact on forward-secrecy requirements in the full version of this paper. Instead, the adversary is initialized with a subset of the valid keys $K_{\ell+1}, \dots, K_n$, 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 , that models the behavior of the honest tag with corresponding key K_i .

The initialization also requires a special type of oracle $\mathcal{O}_{i \vee j}, 1 \leq i < j \leq \ell$, which we call *ambivalent*. The key of an ambivalent oracle $\mathcal{O}_{i \vee j}$ is selected randomly from $\{K_i, K_j\}$, with equal probability. The role of the ambivalent oracles will soon be made clear.

As the simulation starts, each tag oracle or ambivalent oracle independently initializes values $r_i, r_{i \vee j}$ at random.

The server \mathcal{O}_S generates a random value r_{sys}^0 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_{sys}^t with a new random value, where t counts how many periods have completed before the current one.

B. 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} .

During the real simulation the adversary \mathcal{A} has access to all the oracles \mathcal{O}_i , $1 \leq i \leq \ell$, and can interact with these via calls that will be specified below. We shall refer to these interactions as conversations. Crucial to the UC approach is the notion of a simulation environment \mathcal{Z} . \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 calls the adversary can make to the tag oracles and ambivalent oracles in the real protocol simulation, together with the oracle responses.

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_{sys}^t . 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.

SEND(i, m): \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($i \vee j, m$): $\mathcal{O}_{i \vee j}$ responds with the pair $r_{i \vee j}, h = H_{K_\nu}(m, r_{i \vee j})$, where K_ν is the key of $\mathcal{O}_{i \vee j}$, and then updates $r_{i \vee j} \leftarrow H_{K_\nu}(r_{i \vee j})$.

SEND(\mathcal{S}, 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 $h = H_{K_i}(r_{sys}^t || r)$, 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 $h = H_{K_i}(r_{sys}^t || r)$. If such a key K_i is found, it replies to \mathcal{A} with 1, or 0 otherwise. \mathcal{O}_S privately outputs the identity i of the tag if the authentication is successful with key K_i , else it outputs nothing. This output is not observed by the environment \mathcal{Z} .

The role of the identity-ambivalent oracles: The ambivalent oracles $\mathcal{O}_{i \vee j}$ enable \mathcal{A} to interact with parties whose identity is one of two possible choices. Ambivalent oracles are used in attacks against anonymity, where \mathcal{A} 's objective is to determine if $\mathcal{O}_{i \vee j}$ represents \mathcal{O}_i or \mathcal{O}_j .

Rules of engagement: The adversary \mathcal{A} can engage in arbitrary and concurrent conversations with the oracles during a communication session, subject to the following concurrency constraints:

- For each tag T_i the adversary can make only one SEND(i, m) call per server period; and
- if SEND(k, m) and SEND($i \vee j, m$) calls are combined in a period then k must be different from both i and j .

The reason for these constraints is that we do not allow single tags to execute concurrently in the same period, and we do not want that \mathcal{A} may disambiguate the ambivalent oracles simply on the basis of availability conflicts.

C. Security definitions

We now formally define the security goals of anonymous authentication protocols.

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 that session. This should remain true even if \mathcal{A} has interacted with T_i or \mathcal{S} 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 $adv_{AVLB}^{\mathcal{A}, T_i}$ of \mathcal{A} in this game against T_i is the maximum probability that \mathcal{S} rejects T_i in any session:

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

and $adv_{AVLB}^{\mathcal{A}}$ is defined as the maximum of the $adv_{AVLB}^{\mathcal{A}, T_i}$ 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 ([21]) 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 §V.

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{S} believes \mathcal{A} to have authenticated itself as tag T_i in ses ; and (b) the duration interval 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 $\mathcal{O}_{i \vee j}$ 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 $adv_{\text{AUTH}}^{\mathcal{A}, T_i}$ of the adversary against authentication is simply the probability that it succeeds.

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

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

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

$$adv_{\text{ANON}}^{\mathcal{A}, i \vee j} := \text{Prob}[(i, j, 1) \leftarrow \mathcal{A} \mid \mathcal{O}_{i \vee j} = \mathcal{O}_j] \\ - \text{Prob}[(i, j, 1) \leftarrow \mathcal{A} \mid \mathcal{O}_{i \vee j} = \mathcal{O}_i],$$

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

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

D. 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_{sys}^t , and sending r_{sys}^t

to $S_{\mathcal{A}}$. It resets its database, so that it is empty at the start of the period.

SEND(i, m): \mathcal{F} generates a new random value r , and returns r to $S_{\mathcal{A}}$. If $m = r_{\text{sys}}^t$, then \mathcal{F} also stores the pair (i, r) in its database.

SEND($i \vee j, m$): \mathcal{F} generates random value r (returned to $S_{\mathcal{A}}$) and if $m = r_{\text{sys}}^t$, it stores the pair $(i \vee j, r)$ in its database.

SEND(S, m): First, \mathcal{F} checks if there exists a pair (i, m) or a pair $(i \vee j, m)$ 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 , $i = \ell + 1, \dots, n$, \mathcal{F} checks if $h = H(K_i, r_{\text{sys}}^t || r)$. 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 .

The rules of engagement for the ideal simulation are the same as those for the real simulation.

E. Security in the ideal simulation

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

Authentication: \mathcal{F} only returns 1 if the ideal adversary $S_{\mathcal{A}}$ makes a call **SEND(S, m)** in period t , where m was either computed as $m \leftarrow r||h$, where $h = H(K_i, r_{\text{sys}}^t || r)$, 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_{sys}^t)** by $S_{\mathcal{A}}$.

Anonymity: This 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.

IV. 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 emulating all oracle calls of \mathcal{A} to \mathcal{O}_i in the real world simulation by identically-named calls of $S_{\mathcal{A}}$ to \mathcal{F} in the ideal world. We then show that \mathcal{Z} cannot distinguish between values returned by \mathcal{F} in the

ideal simulation form values returned by oracles in the real world simulation.

First, note that REFRESH() calls 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 the 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{S}, 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 $h = H(K_i, r_{sys}^t||r)$, for some $K_i, i = 1, \dots, n$. We distinguish two subcases:

- (a) $i \leq \ell$ (honest tags), and
- (b) $i > \ell$ (adversarial tags).

A SEND(\mathcal{S}, 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 $h = H(K_i, r_{sys}^t||r)$, 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 $r, H(K_i, r_{sys}^t||r)$, with $i \leq \ell$, without obtaining this from an oracle call to \mathcal{O}_i or some $\mathcal{O}_{i \vee j}$ —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.

V. Extensions

In this section we describe a 2-pass optimistic RFID authentication protocol that addresses most of the drawbacks of the authentication protocols in [22], [23], [5], [24], [25]. The protocol is an extension of YA-TRAP—Yet Another Trivial Authentication Protocol, proposed by G. Tsudik [5].

We also discuss how to accommodate kill-keys in both O-TRAP and YA-TRAP+ without introducing side-channel vulnerabilities.

A. YA-TRAP+

We describe an optimistic extension YA-TRAP+ of YA-TRAP that addresses availability. 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 4.

Fig. 4. YA-TRAP+

<ol style="list-style-type: none"> 1. \mathcal{S} sends to R_j, who broadcasts (t, r_{sys}^t). 2. $T_i \rightarrow R_j \rightarrow \mathcal{S} : r_i, h_1 = H_K(00 t r_{sys}^t)$, if $t > t_i$. $r_i, h_1 = H_K(01 r_i r_{sys}^t)$, if $t \leq t_i$. 3. $\mathcal{S} \rightarrow R_j \rightarrow T_i : r_i, h_2 = H_K(10 r_i t)$, (optional)
<ol style="list-style-type: none"> 1. T_i sets $t_i = t$ if $t > t_i$. 2. \mathcal{S} accepts T_i as authentic only if $\exists K \in \mathcal{K}$: $(h_1 = H(K, 00 t r_{sys}^t))$ or $(h_1 = H(K, 01 r_i r_{sys}^t))$. 3. If $h_2 = H(K, 10 r_i t)$, T_i sets $t_i = t$ (optional)

Pass 3 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 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 §III, under the assumption that the keyed hash function is chosen from a pseudo-random function family $\{H_K(\cdot)\}_K$.

VI. Further Research

A. 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 $t > t_{max}^i$ —known to \mathcal{S} —and executes the extra optional step. The tag accepts t as valid, and becomes disabled. Note that the tag can update its time t to a non-authenticated value above t_{max}^i without becoming 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{S} .

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

B. 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. However, extending the UC security model to express issues of forward security in the presence of key compromise is more challenging. We plan to address this important issue in a future work.

C. 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.

VII. 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+ 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 forward security, DoS resilience, effects of tag capture, and/or to incorporate specific descriptions of side-channel information leakage.

References

- [1] A. Juels and S. A. Weis, “Authenticating pervasive devices with human protocols,” in *Proc. Advances in Cryptology (CRYPTO 2005)*, ser. LNCS, vol. 3621. Springer, 2005, p. 293.
- [2] N. J. Hopper and M. Blum., “Secure human identification protocols,” in *Proc. Advances in Cryptology (ASIACRYPT 2001)*, ser. LNCS, vol. 2248. Springer, 2001.
- [3] J. Katz and J. S. Shin, “Parallel and concurrent security of the HB and HB+ protocols,” in *Proc. Advances in Cryptology (EUROCRYPT 2006)*, ser. LNCS. Springer, 2006.
- [4] H. Gilbert, M. Rodshaw, and H. Sibert, “An active attack against HB+ – a provably secure lightweight authentication protocol,” International Association for Cryptological Research, Tech. Rep., 2005. [Online]. Available: <http://eprint.iacr.org/2005/237.pdf>
- [5] 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.
- [6] O. Goldreich, S. Micali, and A. Wigderson, “How to play any mental game,” in *19th Symposium on Theory of Computing (STOC 1987)*. ACM Press, 1987, pp. 218–229.
- [7] D. Beaver and S. Goldwasser, “Multiparty computation with faulty majority,” in *Proc. Advances in Cryptology (CRYPTO 1989)*, ser. LNCS, vol. 435. Springer, 1989, pp. 589–590.
- [8] D. Beaver, “Secure multi-party protocols and zero-knowledge proof systems tolerating a faulty minority.” *Journal of Cryptology*, vol. 4:2, pp. 75–122, 1991.
- [9] —, “Foundations of secure interactive computing,” in *Proc. Advances in Cryptology (CRYPTO 1991)*, ser. LNCS, vol. 576. Springer, 1991, pp. 377–391.
- [10] R. Canetti, “Studies in secure multiparty computation and application,” Ph.D. dissertation, Weizmann Institute of Science, Rehovot 76100, Israel, June 1995.
- [11] —, “Security and composition of multi-party cryptographic protocols,” *Journal of Cryptology*, vol. 13:1, pp. 143–202, 2000.
- [12] —, “Universally composable security: A new paradigm for cryptographic protocols,” in *Proc. IEEE Symp. on Foundations of Computer Science (FOCS 2001)*. IEEE Press, 2001, pp. 136–145.
- [13] R. Canetti and H. Krawczyk, “Universally composable notions of key exchange and secure channels (extended abstract),” in *Proc. Advances in Cryptology (EUROCRYPT 2002)*, ser. LNCS, vol. 2332. Springer, 2001, p. 337.
- [14] 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)*, ser. LNCS, vol. 2808. Springer, 2003, pp. 61–84.
- [15] R. Canetti and J. Herzog., “Universally composable symbolic analysis of cryptographic protocols (the case of encryption-based mutual authentication and key exchange),” International Association

- for Cryptological Research, Tech. Rep. E-print Report # 2004/334, 2004. [Online]. Available: <http://eprint.iacr.org/2004/334>
- [16] R. Canetti and M. Fischlin, “Universally composable commitments (extended abstract),” in *Proc. Advances in Cryptology (CRYPTO 2001)*, ser. LNCS, vol. 2139. Springer, 2001, p. 19.
 - [17] 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)*, vol. 34. ACM Press, 2002, pp. 494–503.
 - [18] 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)*. ACM Press, 2005, pp. 26–35.
 - [19] 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)*. ACM Press, 2005, pp. 92–101.
 - [20] O. Goldreich, *The Foundations of Cryptography*. Cambridge University Press, 2001, vol. 1.
 - [21] 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.
 - [22] 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.
 - [23] 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.
 - [24] —, “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.
 - [25] I. Vajda and L. Buttyan, “Lightweight authentication protocols for low-cost RFID tags,” in *Proc. Workshop on Security in Ubiquitous Computing (UBICOMP 2003)*, 2003.