# Anonymous RFID authentication supporting constant-cost key-lookup against active adversaries

M. Burmester*

Department of Computer Science,
Florida State University,
Tallahassee, FL 32306, USA
E-mail: burmester@cs.fsu.edu
*Corresponding author

B. de Medeiros

Google Inc., 1600 Amphitheatre Pkwy,
Mountain View, CA 94043
E-mail: breno@google.com

R. Motta

Department of Computer Science,
Florida State University,
Tallahassee, FL 32306, USA
E-mail: motta@cs.fsu.edu

**Abstract:** In the absence of sufficiently optimised public key constructions, anonymous authentication for Radio-Frequency Identification Devices (RFIDs) requires state synchronisation between tags and a trusted server. Active adversaries disrupt this synchrony, making a recovery strategy necessary. In some protocols, tags recover by replaying previously used values, thus compromising unlinkability of their transcripts; other schemes require servers to search through the set of issued keys, incurring costs that are not constant with the number of legitimate tags. This article describes an approach based on a lightweight trapdoor one-way function from modular squaring. The solution exploits the fact that synchrony can be recovered even if tags are endowed with only the ability to perform public-key operations, whilst the trusted server is capable of trapdoor computations. The construction is provably secure and generic, transforming any anonymous, challenge-response RFID authentication protocol into another that is robust against active adversaries and supports constant key-lookup cost.

**Keywords:** anonymous authentication; lightweight cryptography; provably secure protocols; Radio-Frequency Identification Devices; RFID; RFID security; scalable security; unlinkability.

**Reference** to this paper should be made as follows: Burmester, M., de Medeiros, B. and Motta, R. (2008) 'Anonymous RFID authentication supporting constant-cost key-lookup against active adversaries', *Int. J. Applied Cryptography*, Vol. 1, No. 2, pp.79–90.

**Biographical notes:** Mike Burmester is a Professor in the Department of Computer Science, Florida State University. His main research interests are cryptography, network security, security of pervasive/ubiquitous systems, MANETs and sensor networks.

Breno de Medeiros is a Software Engineer at Google Inc., and holds a courtesy appointment at Florida State University. He has published research in applied cryptography, network security and information privacy issues.

Rossana Motta is a PhD student in the Department of Computer Science, Florida State University. She began her PhD in 2007. Her primary interests are RFID security and network applications.

## 1 Introduction

Radio-Frequency Identification Devices or tags (RFIDs), with their limited computational capabilities and constrained power source, represent the extreme low-end of computational devices that are both endued with a native communication interface and used for identification, verification, integrity and security functions.

Embedded RFIDs enable objects to be identified by radio waves, without physical contact and without need for line-of-sight alignment. The flexibility of this

technology holds great promise for novel applications, and increasingly RFID tags are being deployed in situations where their proper operation must be assured to a medium or high level of confidence. Well-known examples are the use of RFIDs to harden identification documents against forgery (in particular passports), or to provide access control to physical resources and/or secure locations. The deployment of RFIDs in consumer goods and identification documents give rise to considerations of privacy, in particular as the devices are designed to operate in promiscuous mode and are able to communicate wirelessly over distances that often exceed the 'sphere of personal privacy'. Thus, in addition to authenticity and integrity requirements, it is often desirable, and sometimes required, that RFIDs provide anonymised identification services. This has motivated the design of lightweight anonymous authentication protocols for RFIDs.

A considerable body of research has been developed to provide solutions to the anonymous authentication problem in RFID (Sharma, Weis and Engels, 2003; Henrici and Müller, 2004; Avoine and Oechslin, 2005; Burmester, van Le and de Medeiros, 2006; Molnar, Soppera and Wagner, 2006; Tsudik, 2006). However, currently available solutions either do not provide robust security guarantees, or suffer from scalability issues when the number of tags issued by the system is very large. The principal reason leading to this conflict between requirements is the small circuit footprint available on RFID tags, which has so far limited implementation of cryptography in tags to symmetric-key algorithms. In the anonymous setting, symmetric-key approaches introduce the difficulty that the server must first decide which tag (and corresponding key) should be used to validate a tag's authentication transcript. This difficulty is worsened if the system has a large number of tags, creating vulnerabilities to denial-of-service attacks and potentially raising threats to privacy through timing attacks.

In this article, we focus on the worst-case complexity (time and computation) of identifying tags, by searching for matches in the symmetric-key database of the back-end server. More specifically, we consider the ratio between the costs of:

1 authenticating the response of a tag against a single tag identity

2 authenticating the response of a tag in an anonymous interaction (when the identity of the tag is not known *a priori*).

This ratio we call the 'key-lookup cost'. In the worst case, for anonymous RFID authentication, the key-lookup cost is linear in the number of tags (the server has to exhaust the symmetric-key database to find a match). Molnar, Soppera and Wagner (2006) presented an anonymous RFID protocol that achieves logarithmic key-lookup, by using a binary tree of symmetric-keys (the tree of secrets), and assigning to each tag the keys of a root-to-leaf path: the response of a tag is then linked to this path, and this link is used to identify the tag (only $2 \log T$ checks are needed, where $T$ is the number of tags). Burmester, van Le and de Medeiros (2006) use a different approach, in which the key-lookup is constant for tags that have not been previously interrogated by rogue readers (invoked by the adversary), but otherwise it is linear.

### *1.1 New approach and results*

In Burmester, de Medeiros and Motta (2008), we introduce a new approach that leads to a reconciliation of privacy and availability requirements in anonymous RFID authentication: a generic compiler that maps each challenge-response RFID authentication protocol into another that supports key-lookup operations in constant-cost. If the original protocol were to satisfy anonymity requirements, the transformed one inherits these properties. The compiler is described in detail in Section 3.2; the result improves the prior best bound on worst-case key-lookup cost of $O(\log n)$, by Molnar, Soppera and Wagner (2006). The compiler never degrades the security and privacy properties enjoyed by the original protocol (compilee). It can be used to construct schemes with constant-cost-key-lookup that do not require the use of shared hierarchical keys (e.g. as in Molnar, Soppera and Wagner (2006)) and that, therefore, do not suffer from enhanced exposure to key leaks.

In order to instantiate the protocol, we make use of a lightweight one-way trapdoor function. Until recently, it was not believed that trapdoor functions could be implemented within the limitations of current tag architectures. Recently, Shamir demonstrated a construction of a one-way function based on modular squaring (and provably as secure as factoring), called SQUASH (for 'SQUaring hASH'). SQUASH is fully amenable to implementation in RFIDs (Shamir, 2007, 2008). We observe that, at a moderate additional cost, it is possible to implement a variant of SQUASH that supports trapdoor functionality (Section 3.3). This trapdoor variant is the tool that makes practical realisations of the compiler possible.

A security proof of the compiler is provided in Section 3.5.2. We also show that any RFID authentication protocol that simultaneously provides guarantees of privacy protection and of worst-case constant-cost key-lookup must also imply 'public-key obfuscation', at least when the number of tags is asymptotically large (Section 4).

We also consider relaxations of the privacy requirements and show that, if limited linkability is to be tolerated, then simpler approaches can be pursued to achieve constant key-lookup cost (Section 5).

## 2 Privacy at odds with availability

Fundamental requirement of all practical systems are robustness and availability. Indeed, the need to provide minimal guarantees of continuity of service and to tolerate

denial-of-service attacks often trumps other security requirements at a functional level, including privacy concerns.

There are several attack strategies against RFID systems that target availability. For instance, jamming attacks seek to overwhelm the communication medium with noise; such attacks can be detected and mitigated by mechanisms at the physical layer (Sharma, Weis and Engels, 2003). In this article, we focus instead on mechanisms that support availability at the protocol level (RFID application layer). A related threat (at the protocol level) to availability in wireless communication protocols is represented by network storms, when the number of transmissions exceeds the capacity of the system to process them, thus restricting availability. Storms are typically a result of design and protocol failures, e.g. if a system relies on network flooding for message transmission, this can result in storms if the local node density is high. In an RFID system, readers may not be able to process all tag responses, when the number of tags in the vicinity is too large.

In terms of adversarial threats, an attack may exploit the complexity of a key-lookup operation by having a rogue tag make faulty responses on behalf of several 'virtual' tags. It may not be easy to detect such attacks, because they cannot be distinguished from non-adversarial faulty responses. If the back-end server spends more resources trying to disambiguate fake responses than the adversary spends on generating them, then the RFID system is inherently flawed. Note that the cost of triggering such an 'RFID storm' is restricted to the cost of selecting EPC channels (EPC Global), one for each response, since the faulty response can be generated by simply updating a counter (or some similar mechanism). Furthermore, the rogue tag is not subject to the usual tag constraints (e.g. it can have its own power supply). Therefore, it is important to design RFID systems with scalable key-lookup, as systems with large number of tags may require a constant-cost of key-lookup to achieve resilience against attacks that target availability (denial-of-service attacks).

In devising a scheme to reduce the cost to key-lookup, one should not overlook a (in some ways complementary) class of attacks against availability, termed disabling attacks, which target state synchronisation requirements. More precisely, strong authentication (in the symmetric-key setting) requires that RFID tags and the back-end server share secrets (e.g. keys and other state information). In the case of privacy-preserving protocols, mutable information (e.g. a changing pseudonym) must be used by the back-end server to recognise the tag in the absence of fixed identification values. These represent shared dynamic states that must be maintained in synchrony by tags and the back-end server. Disabling attacks seek to break this synchronicity.

Among RFID protocols that provide strong privacy guarantees, some have limited ability to tolerate attacks against availability. For instance, in the Ohkubo–Suzuki–Kinoshita (2003) protocol, the attacker may use invalid timestamps to disable tags temporarily or permanently. Other solutions use hierarchic key structures to speed up lookup time, but are consequently more vulnerable to key-exposure threats as the higher-level keys are shared among many tags. For instance, in the Molnar, Soppera and Wagner (2006) a tag is identified if the key of its sibling is compromised. The key of a parent will link it to one of two possible tags, and the key of a $t$-ancestor to one of $2^t$ possible tags. Yet, other protocols require linear search among all issued keys to authenticate a response, an approach that is infeasible for large numbers of tags. An improvement over always requiring an exhaustive search is to employ an optimistic approach (Burmester, van Le and de Medeiros, 2006; van Le, Burmester and de Medeiros, 2007). In this case, the server is normally able to recognise the tag in constant time based on a pseudonym value, but when this value becomes de-synchronised, the server can recover the tag identity through linear search among the valid (issued) keys. All protocols that (in some circumstances) require a linear search suffer from scalability issues as the number of tags in the system increases.

In what follows, we describe how to systematically modify RFID protocols to achieve constant-time effort to authenticate a tag, resolving scalability issues and reconciling privacy and availability requirements of RFID applications.

## 3 A scalability-providing compiler

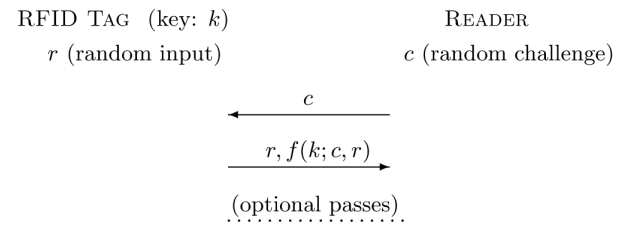
In this section, we provide a high-level description of our approach, a compiler that can transform many challenge-response RFID protocols to achieve scalability, supporting constant-cost for RFID key-lookup.

### *3.1 A generic challenge-response RFIDs protocol*

Figure 1 illustrates a typical challenge-response RFID authentication protocol. In the first pass, the reader produces a challenge $c$ that could include a timestamp, a random nonce, or other information as specified by the protocol. In the second pass, the tag evaluates and broadcasts the result of computing a function $f(k; \cdot,\cdot)$ on the challenge and (possibly) on additional input $r$ generated by the tag. The value $r$ could embed a tag nonce and either an identifier (if privacy is not a concern), or a (mutable) pseudonym to facilitate tag recognition without leaking its identity. The dotted line in Figure 1 indicates optional passes for added functionality, such as mutual authentication, key-update for forward-security, etc.

The security and efficiency provided by such generic protocols are highly related to the choice of the function $f$, which is keyed with a symmetric-key $k$ unique to the tag and known to the back-end server (not depicted above).

**Figure 1** A generic challenge-response RFID protocol




Even restricting our attention only to protocols that provide privacy, there are many possibilities for implementation of the above protocol, providing different security guarantees. For instance, if unlinkable privacy is desired, the outputs of the function $f$ must be indistinguishable from random. Protocols may further differ on the method for pseudonym update, and may provide for additional features – for instance, protocols may only support authentication when the reader has on-line access to the back-end server, or conversely, may be suitable for reader-server batch communication.[1]

In the following, we assume that we deal with a generic RFID protocol as in Figure 1 that suffers from lack of scalability of key-lookup. We also assume that the protocol must achieve privacy, since otherwise scalability may be easily achieved by having the tag reveal its identity to the server, thus allowing for immediate key-lookup.

### 3.2 The compiler

Let $h(\cdot)$ stand for a trapdoor one-way function, where the trapdoor $t$ is known only to the back-end server (the Verifier). We fix the protocol in Figure 1 to achieve key-lookup scalability by changing the form of the tag's response to

$$h(\mathrm{id}, r), f(k; c, r) \tag{1}$$

where $f$ is as in Figure 1 and id is a tag identifier. Note that $r$ in the original protocol is a random nonce, selected from $\{0,1\}^n$ uniformly at random, where $n$ is a security parameter. When receiving the above response (relayed by the reader), the back-end server (the Verifier) can use its knowledge of the trapdoor to recover id. It may, then apply the steps of the (non-scalable) version of the protocol.

In order for the compiler to preserve the security properties of the original protocol, the function $h(\cdot,\cdot)$ must satisfy the following property.

*Assumption 1 (Unlinkable anonymity)*. The probability ensembles $\{h(\mathrm{id}_1, r)\}_{r\in\{0,1\}^n}$ and $\{h(\mathrm{id}_2, r)\}_{r\in\{0,1\}^n}$ must be computationally indistinguishable, for any pair of identities $\mathrm{id}_1$, $\mathrm{id}_2$.

The need for Assumption 1 should be obvious, as otherwise it allows for distinguishing between two identities, breaking unlikable anonymity of the underlying protocol.

It is possible to show that the compiled protocol provides at least the same anonymity and unforgeability guarantees as achieved by the initial protocol while guaranteeing constant key-lookup cost.[2] In the following, we shall first describe a lightweight implementation of a one-way trapdoor function (Section 3.3), and then demonstrate the explicit efficiencies (Section 3.4) and security guarantees (Section 3.5) achieved by the compiler when applied to a family of protocols (Burmester, van Le and de Medeiros, 2006; van Le, Burmester and de Medeiros, 2007) that is secure under a very robust security model (universal composability).

### 3.3 A lightweight one-way trapdoor function

The greatest challenge in making the above compiler practical is to design very efficient one-way trapdoor functions with the required properties. First, we point out that since the RFID tags never need to perform operations using the trapdoor – from the perspective of a tag, $h$ is simply a one-way function – this asymmetry can be exploited to obtain more efficient schemes. There are several alternatives that could be used to implement the function efficiently. The most interesting approach is based on SQUASH, proposed by Shamir (2007, 2008). It is based on modular squaring (where the modulus $N$ is reasonably large, say 1,024 bits), and requires just a few hundred Gate-Equivalents (GEs) for computation and another several hundred GEs for readonly storage. Because only arithmetical operations are involved, this approach can be implemented very efficiently, while from a security point of view it is as hard as integer factorisation (Shamir, 1994).

### 3.4 Applications and implementations

The compiler can be applied to practically any anonymous RFID protocol to establish constant key-lookup. In particular, to the lightweight RFID protocols O-TRAP and O-FRAP presented in Burmester, van Le and de Medeiros (2006) and van Le, Burmester and de Medeiros (2007), for strong Universal Composability (UC) security with constant key-lookup. Next, we discuss efficiency aspects of implementing the protocols that result from application of the compiler to the O-TRAP scheme.

*The authenticator* $f$. In the O-TRAP/O-FRAP family of protocols, $f$ is realised by a Pseudo-Random Function (PRF), which in practice can be implemented using a variety of well-known constructions. Efficiency vs. security trade-offs in this architecture are easily achieved, as key-size and pseudo-randomness (estimated as the logarithm of the PRF cycle) can be chosen to the granularity of individual bits. Here, we discuss two implementation strategies based on different PRF instantiations.

Using a well-known technique by Goldreich, Goldwasser and Micali (1986), it is possible to build a PRF that makes a call to a Pseudo-Random Number Generator (PRNG) per bit of input processed. In turn, a very efficient PRNG implementation can be achieved using linear feedback shift registers like the self-shrinking generator (Coppersmith, Krawczyk and Mansour, 1994). This results in a small number of bit operations per input and output bits. The entire footprint of this implementation has been estimated to require only 1,435 logic gates (within 517 clock cycles and 64B memory), achieving 128-bit security (Lee and Hong, 2006).

Block ciphers can similarly be used to implement PRFs through a number of standard constructions – their concrete security was analysed in Bellare et al. (1997). Recently, highly optimised implementations of the Advanced Encryption Standard (AES) block cipher (Daemen and Rijmen, 2002) have been achieved, and these are suitable for RFID architectures (Feldhofer, Wolkerstorfer and Rijmen, 2005). An RFID architecture using this implementation was proposed in Feldhofer, Dominikus and Wolkerstorfer (2004), with footprint equal to 3,400 GEs and mean current consumption equal to 8 μA, assuming a clock rate of 100 kHz and within 1,032 clock cycles.

*The obfuscator h*. The Rabin (1979) cryptosystem is a public-key encryption scheme that uses modular squaring with composite modulus $N$, to encrypt data. The public key is $N$ and the private key is the factorisation of $N$. In particular, if $x$ is the plaintext then the ciphertext is $y = x^2 \bmod N$. To decrypt $y$ the factors of $N$ are used: there are four quadratic residues of $y$, one of which is $x$.

In Shamir's adaptation, modular squaring is replaced with integer squaring: $h(x) = x^2 + kN$ where $k$ a random number larger than $N$, e.g. $\log k - \log N$ should exceed the security parameter (Shamir, 1994, 2007, 2008). It is not difficult to show that inverting $h$ is as hard as factoring composite numbers (Shamir, 2007, 2008).

*Cost of obfuscating*. Let $x = \text{Expand}(\text{id}||r)$, where Expand(•) is an invertible Non-Linear Feedback Shift Register (NLFSR). The NLFSR is necessary to map id$||r$ into a full-length $N$-residue, as the security proof of Rabin's scheme is for full-length inputs. The NLFSR is also used for this purpose by the SQUASH construction. To compute $h(x) = x^2 + kN$, for a 1,024-bit wide $N$, the square $x^2$ and the product $kN$, are computed separately on-the-fly, using a 128 bit register, and then combined (with carries buffered for the next iteration). To evaluate individual bits of $x^2$ and $kN$, it is sufficient to convolute $x$ with itself, and $k$ with $N$, using the 128 bit register, and invoke a PRNG to generate the bits of $x$ and $k$ on-the-fly. Nine invocations will be needed. The cost of implementing $h$ is then:

1 512 NOT gates for read-only storage of the 1,024-bit modulus $N$.

2 A PRNG and buffers for the computations.

Using an estimate of several hundred GEs for an optimised SQUASH implementation (Shamir, 2007, 2008), the circuit complexity of $h$ is less than 1,000 GEs.

A difference between the above scheme and SQUASH is that Shamir proposes not to store $N$, but to keep simply the seed to reconstruct it from a small-footprint PRNG. To construct such $N$, one must try several seeds, evaluate the corresponding value for $N$, and then perform some factorisation and primality tests to ascertain with high confidence that $N$ is a product of two large primes. However, this method does not lead to the factorisation of $N$ (the trapdoor) and can only be used as a one-way function. Instead, we propose to generate $N$ in the usual manner (e.g. as for use with an RSA cryptosystem). The observation is that we can compress the storage requirement for $N$ by half, because the 1-valued bits can be implemented as straight-through wires.

In addition, in Shamir's (2007, 2008) description, as the SQUASH function is used to implement a MAC algorithm, it is not necessary to generate or transmit all the bits of $h(x)$, but simply a sufficiently wide window containing the middle bits of the result. However, for our purposes it is necessary to evaluate and transmit all the bits of $h(\cdot)$. This modification has no impact in the size of the circuit footprint. However, it does require a larger number of clock cycles for evaluation: the middle bits in the evaluation of a square are affected only by the lower $n/2$ blocks, so require only approximately 1/4 of the effort to evaluate all the bits.

*Total cost of scheme*. In total, assuming a PRG based implementation of $f$, the cost is $\sim 1{,}435 + 1{,}000 = 2{,}435$ GEs. In practice, this evaluation may be pessimistic, because some of the building block in the implementation of SQUASH, e.g. the NLFSR, might in practice share some circuit sub-routines with the implementation of the PRF for the O-TRAP protocol family. Using a non-modular implementation, it may be possible to reach a more compact implementation than reflected in the above estimates.

### *3.5 The security proof for the compiler*

In this section, we prove the result for the case where the generic challenge-response RFID protocol in Figure 1 realises one-way authentication with strong (unlinkable) privacy in the UC framework.

#### *3.5.1 Overview of the Universal Composability model*

The UC framework defines the security of a protocol $\pi$ in terms of the interactive indistinguishability between real-and ideal-world simulations of executions of $\pi$ (Canetti, 1995, 2000, 2001). Simulated protocol runs include a polynomial number of participants, each of which (either honest or adversarial) is represented by a Probabilistic Polynomial-Time (PPT) machine.[3]

In the real-world executions, honest parties are represented by PPT machines that execute the protocol as

specified. These honest parties are fed private inputs at the beginning of the simulation, where also a unique 'session identifier' *sid* is assigned. Adversarial parties are also represented by PPTs, but can deviate from the protocol in an arbitrary fashion.

The adversarial parties are controlled by a PPT adversary $A$ that has full knowledge of their state, controls the communication channels of all parties (honest and adversarial), and interacts with the environment $\mathcal{Z}$ in an arbitrary way, and in particular eavesdrops on all communications. $\mathcal{Z}$ is itself a PPT, but at the start of a protocol run simulation, it is fed as input a non-uniform advice string. This input represents $\mathcal{Z}$'s ability to capture state from concurrent sessions of other protocols (or the same protocol). Indeed, as an unbounded length string can encode Turing machines representing arbitrary protocols, $\mathcal{Z}$ can simulate any polynomial number of sessions of other protocols concurrently.

Also, because $\mathcal{Z}$'s advice is non-uniform and is not under control of the simulator, and the interaction between $A$ and $\mathcal{Z}$ is arbitrary, it is not possible to 're-wind' the UC model executions (as the context of the execution, represented by the subset of $\mathcal{Z}$'s input string that is sampled, cannot be reproduced).

Note that the UC framework itself controls the assignment of session identifiers and prevents any part with session identifier $sid' \neq sid$ from communicating with a party with session identifier $sid$. This guarantees isolation of sessions, while capturing interaction between sessions through the non-uniform advice that the environment $\mathcal{Z}$ receives at the beginning of simulation, as explained above.

The honest parties in ideal-world executions are controlled by an ideal functionality $\mathcal{F}$, a trusted party that guarantees the correct execution of the protocol – in particular, $\mathcal{F}$ emulates all honest parties. The ideal-world adversary is controlled by $\mathcal{F}$ to reproduce as faithfully as possible the behaviour of the real adversary.

We say that $\pi$ realises $\mathcal{F}$ in the UC framework, or UC-realises $\mathcal{F}$, if no PPT environment $\mathcal{Z}$ can distinguish (with better than negligible probability) real- from ideal-world protocol runs.

### 3.5.2 The proof

We shall show that if a challenge-response protocol of the form in Figure 1 realises one-way authentication with strong privacy in the UC framework, as defined for instance, in van Le, Burmester and de Medeiros (2007), then the compiled protocol will maintain the same security level.

Let $\pi$ be the compiled protocol, with pre-$\pi$ the original protocol as in Figure 1. We shall assume that pre-$\pi$ UC-realises anonymous one-way authentication. As observed earlier, a session identifier *sid* is created for each protocol simulation; this captures the state of the trusted server's database, i.e. the set of valid keys assigned to honest tags and moreover binds tag identities to keys: $\mathrm{id}_i \leftarrow k_i$ , where $k_i$ is used by the Verifier (trusted server) to authenticate tag with identity $\mathrm{id}_i$ (see also van Le, Burmester and de Medeiros, 2007).

In this article, we consider only non-adaptive security, i.e. the set of honest keys remains constant throughout the execution of the protocol. This allows us to consider simpler and more efficient protocols. However, we note that at a moderate increase in complexity it is possible to consider variants of the protocols that achieve forward-security in the presence of dynamic corruption. For more details, see, van Le, Burmester and de Medeiros (2007).

Each honest party (tag) also receives its key (shared with the trusted server) as part of its private input at the beginning of a protocol simulation. In addition, $\mathcal{F}$ takes the following actions in the idealised protocol executions:

- generates perfect random challenges ($c$) for honest readers on behalf of the trusted server
- receives challenges on behalf of honest tags
- generates responses on behalf of honest tags
- decides which tag responses are authentic on behalf of the Verifier.

In the real world, the protocol is instantiated with $f(\cdot,\cdot,\cdot)$, which must satisfy the cryptographic definition of a PRF. This follows from the assumption that pre-$\pi$ is UC-secure in the sense of van Le, Burmester and de Medeiros (2007). By construction, recall that $h(\cdot,\cdot)$ satisfies Assumption 1.

To prove that $\pi$ is secure we show that each behaviour securely provided by $\mathcal{F}$ can also be achieved in the real-world through $\pi$. That is, we simulate the operation of $\pi$ with access to $\mathcal{F}$ by the real-world operation of the protocol that does not rely on $\mathcal{F}$. We summarise the key features of $\pi$, that represent the real-world protocol runs:

- the challenges of the Verifier are received through a reader
- the responses of the tags are mediated by a reader, that may be adversarial
- the adversary controls the adversarial readers and may, modify or interrupt any channels at will – but cannot temper with the contents of the channels connecting honest readers to the Verifier.

The main difference between the real- and ideal-world is that the values produced by $\mathcal{F}$ are generated as truly random, as opposed to pseudo-random. More precisely, at the beginning of the simulation, $\mathcal{F}$ chooses a special identity id* among the set of all possible identities. Later, whenever $\pi$ produces a response on behalf of the tag with

identity id, and on input the challenge $c$, the functionality $\mathcal{F}$ generates a random value $r$, and checks if it has an entry (function_value; id, $c$, ($r$, $F$)) in its database, for some $F$ in the output space of $f$. If so, $\mathcal{F}$ sets $\rho \leftarrow F$. If not, it selects a new value $F$ at random (in the output space of $f$), and sets $\rho \leftarrow F$, entering (function_value; id, $c$, ($r$, $F$)) into its database. Then, it selects $H$ according to the probabilistic ensemble $\{h(\mathrm{id}^*; \cdot)\}$, and enters the record (identity, id, $r$) in its database (but not the value $H$). Finally, it returns the values $\rho$,$H$ as the tag's response in the ideal-world.

A value ($\rho$,$H$) is authentic in the ideal world, against challenge $c$, if there is a record of it in the database maintained by $\mathcal{F}$; more precisely, if $H$ can be inverted to find values id*,$r$'', where the entries (identity,id′,$r'$) and (function_value; id', $c$', ($r$', $F$')), with $\mathrm{id}' = \mathrm{id}'$, $r'' = r'$, $c = c'$, and $\rho = F'$. Observe that this specification of $\mathcal{F}$ makes several security guarantees self-evident:

- Unforgeability is guaranteed because entries in the database are added by $\mathcal{F}$, not the adversary.
- Freedom from replays follows from the fact that $\mathcal{F}$ generates new challenges with either protocol, with overwhelming probability. Therefore, an earlier answer will have a negligible probability of validating against the new challenge (i.e. the probability that $\mathcal{F}$ will re-create the same random value with two distinct entries).
- Privacy (unlinkable anonymity): Indeed, all the tag values are produced according to the same distribution, irrespective of tag identity. So the adversary can at best guess the identity of an honest tag.

In the real world, since the compiled protocol $\pi$ may fail in a variety of ways we must ensure that no combination of such failures may enable the environment $\mathcal{Z}$ to distinguish between real and ideal protocol runs (with non-negligible probability). Note that the real and ideal-world protocols could differ as follows:

1 A match (valid response) occurs in the real-world, while in the ideal-world the match is unsuccessful.

2 A mismatch (invalid response) occurs in the real-world, while in the ideal-world the match is successful.

3 The adversary is able to distinguish tag identities in the real-world, while that is clearly impossible in the ideal-world.

*Case 1*. This occurs when the adversary in the real-world is able to modify some values in the channels (via reflection, reply, modification, etc.) forcing the Verifier to accept responses not produced by tags. However, since the outputs of $f$ observed by the adversary are pseudo-random, the adversary can only have a negligible probability of forcing such an outcome without re-using the outputs of $f$. It is possible for the adversary to re-use the outputs of $f$ with different values for $H$. However, by re-using $f$-values, the adversary (with overwhelming probability) commits to the inputs of $f : k$ (an unknown value, and hence the tag identity id, through the Verifier's binding), $c$ and $r$. This is because as $f$ is pseudo-random, any modification of the inputs would result in a value that, to the adversary, appears completely unrelated to the original value $f$. Since, the values $c$ and $r$ completely characterise the transcript from the perspective of the honest parties – as it binds the never repeated inputs generated by the tag and the Verifier – then, the entire transcript, as well as each of its individual components, is not re-playable.

*Remark*. We do not claim that the transcript is non-malleable. In particular, it may be possible for the adversary to produce, from $H = h(\mathrm{id}, r), a\ \mathrm{value}\ H' \neq H$ so that the inverse of $H'$ also equals the pair (id, $r$). This does not allow the attacker to replay the transcript (because the values $c$ and $r$ are unchanged and bound in $F = f(k; c||r)$), nor to re-use parts of it. (As id is bound to $k$, and $r$ is the common value committed in both $F$ and $H$.)

*Case 2*. This occurs when the randomly generated values in the ideal-world, corresponding to the evaluations of $h(\cdot,\cdot)$ and $f(\cdot; \cdot, \cdot)$ on different inputs, lead to a coincidental match. Since, the values generated by $\mathcal{F}$ are random and independent, the chance of coincidence is negligible.

*Case 3*. If this were to occur, consider a partially idealised world, wherein $\mathcal{F}$ idealises $f$ – i.e, substitutes evaluation of $f$ by recording of entries (function_value; id, $c$, ($r$, $F$)) as above – but evaluates $h(\cdot,\cdot)$ as in the real protocol. No adversary $A$ could distinguish this partially idealised world from the real-world, because if such an adversary existed, it is straightforward to show that it could be used to distinguish outputs of $f$ from random values, violating the assumption that $f$ is pseudo-random. Similarly, no adversary $A$ could exist that distinguishes this partially idealised world from the ideal-world above. Otherwise, such an adversary could be easily shown to distinguish between distributions $\{h(\mathrm{id}_1, r)\}_{r \in \{0,1\}^n}$ and $\{\mathrm{id}_2, r)\}_{r \in \{0,1\}^n}$, for distinct identities $\mathrm{id}_1$ and $\mathrm{id}_2$, with non-negligible probability. Indeed, the difference between the partially idealised and the ideal-world is that in the partially idealised world, the released $H$ values are sampled on the distributions corresponding to the true tag identities, while in the ideal-world, they are all sampled from the distribution corresponding to the fixed identity id*. Such an adversary would violate Assumption 1.

It follows that when the challenge-response protocol UC-realises one-way authentication with strong privacy, then the compiled protocol will maintain this security level. We get scalable key-lookup because the server can invert the function $h$. □

## 4 Implications of constant key-lookup

We show that when the number of tags $T$ is large, anonymous, unlinkable RFID authentication with constant key-lookup implies public-key obfuscation. To achieve this, we clarify in greater detail what we mean by constant-cost of key-lookup.

The security parameter $n$ serves as a natural constraint on algorithmic efficiency. More specifically, a feasible algorithm is characterised by having its cost factor dominated by some polynomial $p(n)$. Let DB($n$) be the number of tags in the server database, in some instance of the RFID scheme, with security parameter $n$. We require that the algorithm that lists all entries in DB($n$) be feasible, since the database must be constructible – hence, the size of DB($n$) is bound above by some polynomial $p_{\mathrm{DB}}(n)$. Note that the cost to the (honest) server to invalidate an answer from the adversary, by exhaustion in the database, is $O(\mathrm{DB}(n) \times \mathrm{val}(n))$, where val($n$) is the cost to authenticate an honest tag, if the server knows its identity in advance. Therefore, if the strategy (in the worst-case) is to use exhaustive search for the key, then the key-lookup cost lookup($n$) is $O(\mathrm{DB}(n))$ and it is only constant when DB($n$) is constant. By contrast, our definition of scalable key-lookup requires that the lookup cost lookup($n$) is constant whenever the size of DB($n$) is bounded by a polynomial in $n$.

To simplify our argument, we make the following assumptions on the obfuscator $h$ and the authenticator $f$:

1 $h$ (id, $r$), as in Section 3

2 $f(k; c, r)$ is pseudo-random.

where $r$ is randomly selected from $\{0,1\}^n$.

The proof is straightforward. Let $T$ stand for the number of valid tags, where $T = T(n)$ is an increasing (at most polynomial) function of the security parameter $n$. Suppose that a tag's response $h(k, r)$, $f(k; c, r)$, to the server's challenge $c$ identifies the tag to the server with overwhelming probability, say $1 - \varepsilon$, where $\varepsilon$ is negligible (in $n$). Then, it is easy to see that the obfuscator $h$ will identify the tag to the server with non-negligible probability. Indeed, the contribution of the authenticator $f$ to the identification of the tag in constant key-lookup time is asymptotically smaller than 1 by a non-negligible amount. More specifically, if the server can only check the authenticator of a constant number $l$ of tags for a possible match, then it will succeed with probability bounded by $\varepsilon' = \ell / T$. It follows that the obfuscator $h$ will identify the tag to the server independently of $f$ with probability bounded below by $1 - \varepsilon - \varepsilon'$ or $1 - \ell / T - \varepsilon$ which is non-negligible if $n$ is large enough, since $\ell / T$ must eventually approach 0 – as $\ell$ is constant, and $T$ is not, as functions of $n$.

Since, we are assuming that every RFID tag in our challenge-response protocol can obfuscate its identity, but only the back-end server can disambiguate it, $h$ must be a public-key one-way function (it must have a trapdoor that only the back-end server possesses).

## 5 Mitigating privacy for availability

In this section, we weaken the requirement for unlinkable privacy while maintaining scalability for the back-end server. We first observe that for tag responses to be linked certain patterns must be detectable. This can happen in different ways. For example, the adversary may succeed in detecting patterns after having corrupted some tags. Alternatively, the adversary may destabilise tags so that they cannot be recognised by the back-end server in scalable time, thus forcing them into using responses with detectable patterns (e.g. re-using pseudonyms), or forcing the server into a linear key-lookup search.

The Molnar–Soppera–Wagner (2006) protocol discussed earlier is an example of an anonymous RFID protocol for which tag responses may be linked if some tags get corrupted. Most of the other anonymous RFID protocols proposed in the literature rely on state synchronisation (see, e.g. Ohkubo, Suzuki and Kinoshita, 2003; Henrici and Müller, 2004; Juels, 2004; Dimitriou, 2005; Burmester, van Le and de Medeiros, 2006; Tsudik, 2006; van Le, Burmester and de Medeiros, 2007). State synchronisation protocols require an extra pass to confirm state changes, and are subject to the well-known 'two generals' 'attack' problem: the server (tag) can never be certain of the next state of the tag (server) when the adversary controls the communication channels. These protocols are prone to disabling attacks. In the worst case, tags cannot be directly recognised by the server, which must then run a linear search through the key-lookup database for each disabled tag. To mitigate this DoS attack, tags may re-use earlier pseudonyms (if these are still recognisable), or as a last resort, reveal their identity (not their secret key).

A desirable privacy compromise is to minimise the loss of privacy. For example, to restrict linkability to those periods when the tag is attacked. One of the most effective disabling attacks is the entrapment attacks, in which the tag is prevented from communicating with authorised readers and can only be interrogated by the adversary. Entrapment is not necessarily physical although it does imply the ability to locate or track, since tags communicate autonomously in wireless mode.

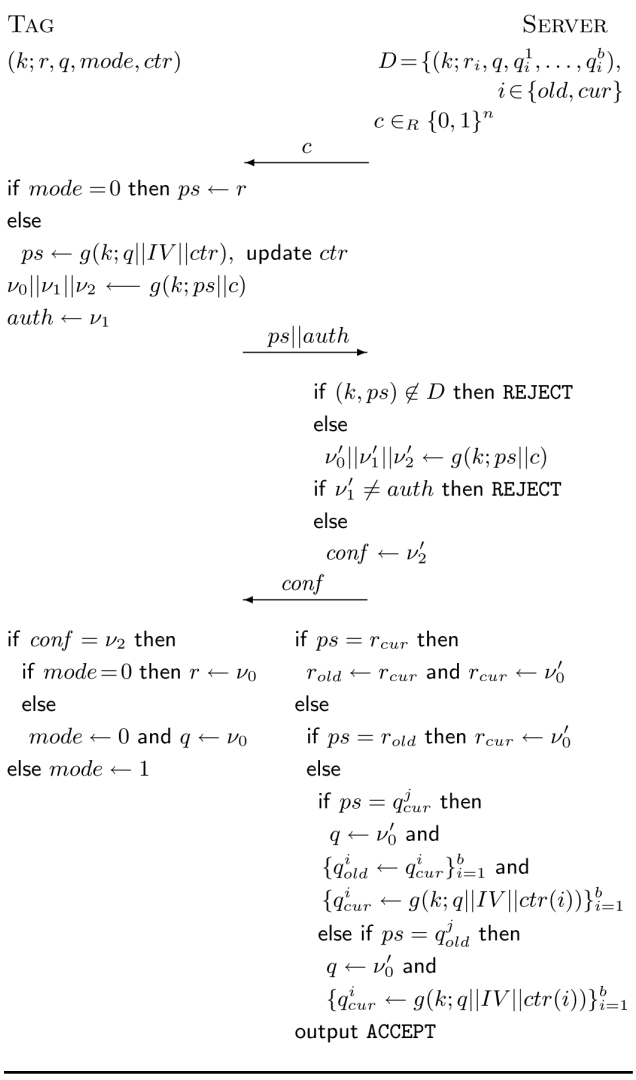
Attacks of this kind on privacy prompt us to revisit the definition of identification. The traditional cryptographic approach is to define identification as entity authentication (Menezes, van Oorschot and Vanstone, 1996). However, in more general applications it is clear that identification has a broader scope and interpretation. Entities are typically identified by their attributes, which may not involve authentication. For example, for a criminal anonymity means not being visually identified by a witness: if the criminal is visually identified then he/she will most likely be found guilty in a court of law even though the witness may not have checked the criminal's ID while the crime was committed. Therefore, it is necessary to interpret RFID privacy in the context of its application.

### *5.1 An anonymous RFID authentication protocol with constant key-lookup*

We present a scalable solution for RFID authentication in which anonymity is established by synchronising states. Our solution will allow a determined adversary to link tags during an entrapment attack, but this will not extend beyond such attacks. More specifically, although the adversary may succeed in linking tags during an entrapment session, this information will be independent for each entrapment session, thus minimising the loss of privacy.

The protocol is described in Figure 2, and has three passes. It is based on O-FRAP, a protocol proposed by van Le, Burmester and de Medeiros (2007), which is an optimistic, forward-secure RFID authentication protocol. To simplify the description of our protocol in this article, we shall drop the requirement for forward-secrecy: however, the required changes to recapture this functionality are straightforward (and are discussed in Section 5.5).

**Figure 2** An anonymous RFID authentication protocol that supports constant key-lookup




### *5.2 Trusted setup and key-lookup database*

Each tag is initially assigned a unique tuple ($k$; $r$, $q$) of random values of bit-length $n$, the security parameter, that is stored in non-volatile memory: $k$ is its secret key, $r$ a one-time pseudonym and $q$ a seed for generating entrapment pseudonyms. The tag is also given a boolean variable mode and a cyclic counter ctr that takes $c$ distinct values, $c$ a small constant. The protocol uses an appropriate PRF $g$ to generate values for the pseudonyms, the authenticators and for confirmation. For each tag, the server stores in a key-lookup database DB a tuple: $(k; r_i; q; q_i^1, \ldots, q_i^b)$, $i \in \{\text{old, cur}\}$ with $q$ a seed and $q_i^j$, $0 \le j \le b$, pseudonyms used during entrapment attacks. Initially: $q = q_0$, $r = r_i = r_0$, and $q_i^j = g(k; q_0 \,\|\, IV \,\|\, ctr\,(j))$, $i \in \{\text{old, cur}\}$, $1 \le j \le b$, where $q_0$, $r_0$ are random values, IV is an initial vector and ctr($j$) is the $j$th value of ctr. The server stores pairs (old, cur) of values in DB to maintain state coordination with the tags. We assume that DB is indexed by each of the $2c + 2$ pseudonyms so that the cost of disambiguating a pseudonym is constant.

The state of each tag is controlled by the variable mode: if the tag is subject to an attack mode = 1, otherwise mode = 0. More specifically, mode←1 whenever the tag fails to receive confirmation to its response from a challenge, while mode←0 when confirmation has been received. When in mode = 1, the tag employs entrapment pseudonyms computed on-the-fly by evaluating $g(k; q \,\|\, IV \,\|\, ctr)$. Since, there is only a constant number of such values (eventually they recycle) the tag has to defend itself against fly-by attacks by rogue readers that seek to exhaust these values. The simplest defense is to use a time-delay mechanism as described in van Le, Burmester and de Medeiros (2007). This will extend the recycle time by a few orders and thwart such attacks, but may fail to deal with entrapment attacks, for which eventually the tag responses will be linked. However, un-linkability will be restored the moment the tag gets mutually authenticated by the server: on receiving confirmation from the back-end server the tag will update its seed $q$.

### *5.3 Protocol description*

We refer to Figure 2. In the first pass, the tag is challenged by the server with a random $c$. If the tag received confirmation in its previous interaction (mode = 0) then, it will update its pseudonym ps←$r$ and compute three values $v_0$, $v_1$ and $v_2$ by inputting ($k$, ps, $c$) to the PRF $g$: $v_0$ is kept for later use as a pseudonym update; $v_1$ = auth is an authenticator and $v_2$ = conf is used for confirmation. The tag responds with ps||auth. If the tag has not previously received confirmation (mode = 1) then, it uses a different pseudonym in its response, computed on-the-fly with seed $q$.

The server uses the key-lookup database DB to disambiguate ps, and then checks auth. If correct, it sends conf to the tag. Then, the server proceeds with pseudonym

updates, that have to be synchronised with those of the tag: $ps = r_{cur}$ corresponds to the case when the tag is not under attack; $ps = r_{old}$ and $ps = q_i^j$, $i \in \{old, cur\}$, correspond to cases when the tag did not receive confirmation, with the last one indicating that the tag was (also) previously interrogated by an unauthorised reader (an entrapment attack). In this case, the server will use a new seed $q \leftarrow v_0$ to update the pseudonyms $q_{cur}$, to support unlinkability between entrapment attacks.

If the tag receives confirmation, then it will update the pseudonym $r$ if in mode = 0, otherwise it updates the seed $q$. What distinguishes this protocol from O-FRAP (van Le, Burmester and de Medeiros, 2007) is that, at all times in this protocol, the values $r$, $q$ stored by the tag in its non-volatile memory are synchronised with those stored by the server in DB. Consequently, the tag can be identified with constant key-lookup.

### 5.4 Security considerations

The protocol in Figure 2 addresses disabling attacks by weakening the requirement for unlinkable privacy. However, linkability is restricted to entrapment attacks in which the tag is either physically restricted or closely tracked. During such attacks, it is reasonable to assume that the tag is monitored and therefore, to some extend, already identified or located.

Our protocol is based on O-FRAP (van Le, Burmester and de Medeiros, 2007) that is proven secure in the UC framework. From a security point of view, the main difference with our protocol is its functionality: it uses entrapment pseudonyms that will eventually recycle. However, these pseudonyms remain pseudo-random until they get exhausted.

*Theorem 1*. Let $m$ be the maximum number of uninterrupted interrogations that the adversary can make to a tag.

1 If $m$ is constant then: when $b \geq m$, the protocol in Figure 2 realises one-way authentication with strong privacy in the UC framework and supports constant key-lookup.

2 If $m$ is unbounded then: the protocol in Figure 2 realises one-way authentication with linkable privacy in the UC framework and supports constant key-lookup.

The proof is straightforward and is based on the proof for O-FRAP that realises anonymous RFID authentication with strong privacy in the UC framework (van Le, Burmester and de Medeiros, 2007). We only need to show that during an entrapment attack no PPT environment $\mathcal{Z}$ can distinguish real – from ideal-world protocol executions. During such attacks, in the real-world the pseudonyms of a tag are pseudo-random, provided the number of uninterrupted adversarial interrogations does not exceed the threshold $b$, which is the case when $b \geq m$. In the ideal-world, the functionality generates random pseudonyms. So the environment $\mathcal{Z}$ cannot distinguish between these.

As pointed out in Section 3.5.2 protocols in the UC framework may fail in a variety of ways so we must ensure that no combination of failures will enable the environment $\mathcal{Z}$ to distinguish between real and ideal protocol runs. Again, the significant ways that could cause the real-and ideal-world protocols to differ involve the possibility of a match (valid response) in the real-world while in the ideal-world the match is unsuccessful, or a mismatch (invalid response) in the real-world, while in the ideal-world the match is successful. The first case occurs when the adversary in the real-world is able to modify some values in the channels, forcing the Verifier to accept responses not produced by tags. However, since the protocol flows observed by the adversary are pseudo-random, the adversary can only have a negligible probability of forcing such an outcome without re-using flows. By re-using flows the adversary commits (with overwhelming probability) to an unknown value of the key $k$ (and hence the other private tag values). Since, we are assuming that it is hard to invert $g$ ($g$ is a PRF), the probability of succeeding is negligible. The second case occurs when the randomly generated values in the ideal-world that correspond to parsings of evaluations of $g(\cdot;\cdot,\cdot)$ on different inputs in the real-world lead to a coincidental match. Since, the values generated by $\mathcal{F}$ are random and independent, the chance of coincidence is negligible. It follows that the protocol realises one-way authentication with strong privacy in the UC framework when $b \geq m$, $m$ constant.

Next, suppose that $m$ is not constant. Then, whenever the number of uninterrupted adversarial interrogations exceeds the period of the counter ctr, the pseudonyms will be linkable. However, this is restricted to the particular entrapment session. As soon as the adversarial interrogation is interrupted, the tag will update the seed $q$ and the pseudonyms will become pseudo-random. □

### 5.5 Implementation and extensions

Our protocol requires only the use of a PRF which as pointed out in Section 3.4 can be implemented with a PRNG. This allows for very efficient implementations. In particular, the protocol can be adapted to conform with the EPC Gen2 standards EPC Global /www.epcglobalinc.org/standards/EPCglobal. However, this protocol does not support forward-secrecy. To capture this functionality, we can adapt it so that as in O-FRAP (van Le, Burmester and de Medeiros, 2007) the key is updated whenever the pseudonym is. The tag will require additional non-volatile memory for key update storage, but otherwise the same basic circuit can be used. One can also capture key-exchange, by using O-FRAKE (van Le, Burmester and de Medeiros, 2007), which is a key-exchange extension of O-FRAP.

## 6 Conclusions

In order for RFID systems that support strong security and privacy to become a reality, a well-rounded practical solution that also considers threats to availability, and which supports scalability, is needed. In this article, we have introduced a scalability compiler that transforms a challenge-response RFID authentication protocol into a similar RFID protocol that shares the same functionality and security as well as provides scalability for the back-end server (constant lookup time even in the presence of active adversaries).

We have described a particular instantiation of the compiler and illustrated its application to a family of authentication protocols with strong security features. In particular, we have shown how to achieve security and privacy with constant lookup cost within the universally composable security model. The compiler requires only several hundred additional GEs of circuit area. Moreover, the compiler preserves other properties like suitability for batch authentication with delayed verification by the trusted server.

We have also proven that one-way trapdoor functions have to be used to obfuscate identifiers, in RFID authentication protocols that support anonymity with constant lookup cost. Finally, by weakening the restriction on unlinkable privacy, we have described a provably secure anonymous RFID authentication protocol that supports scalable lookup and minimises the loss of privacy due to linkability.

## References


Avoine, G. and Oechslin, P. (2005) 'A scalable and provably secure hash-based RFID protocol', Paper presented in the *Proceedings* of the *IEEE International Conferences on Pervasive Computing and Communications (PerCom 2005)*, pp.110–114, IEEE Press.

Bellare, M., Desai, A., Jokipii, E. and Rogaway, P. (1997) 'A concrete security treatment of symmetric encryption', Paper presented in the *Proceedings* of the *IEEE Symposium on Foundations of Computer Science (FOCS 1997)*, IEEE Computer Society Press.

Burmester, M., de Medeiros, B. and Motta, R. (2008) 'Robust, anonymous rfid authentication with constant key-lookup', Paper presented in the Proceedings of the *ACM Symposium on Information, Computer and Communications Security (ASIACCS'08)*.

Burmester, M., van Le, T. and de Medeiros, B. (2006) 'Provably secure ubiquitous systems: universally compos-able RFID authentication protocols', Paper presented in the Proceedings of the *2nd IEEE/CreateNet International Conference on Security and Privacy in Communication Networks (SE-CURECOMM 2006)*, IEEE Press.

Canetti, R. (1995) *Studies in Secure Multiparty Computation and Application*. Rehovot 76100, Israel: PhD Thesis, Weizmann Institute of Science.

Canetti, R. (2000) 'Security and composition of multiparty cryptographic protocols', *Journal of Cryptology*, Vol. 13, pp.143–202.

Canetti, R. (2001) 'Universally composable security: a new paradigm for cryptographic protocols', Paper presented in the Proceedings of the *IEEE Symposium on Foundations of Computer Science (FOCS 2001)*, pp.136–145, IEEE Press.

Coppersmith, D., Krawczyk, H. and Mansour, Y. (1994) 'The shrinking generator', Paper presented in the Proceedings of the *Advances in Cryptology (CRYPTO 1993), LNCS*, pp.22–39, Springer.

Daemen, J. and Rijmen, V. (2002) *The design of Rijndael*. Secaucus, NJ: Springer-Verlag, New York, Inc.

Dimitriou, T. (2005) 'A lightweight RFID protocol to protect against traceability and cloning attacks', Paper presented in the Proceedings of the *IEEE International Conference on Security and Privacy in Communication Networks (SECURECOMM 2005)*, IEEE Press.

EPC Global. EPC tag data standards, vs. 1.3. Available at: //www.epcglobalinc.org/standards/EPCglobal_Tag_Data_ Standard_TDS_Version_1.3.pdf.

Feldhofer, M., Dominikus, S. and Wolkerstorfer, J. (2004) 'Strong authentication for RFID systems using the AES algorithm', Paper presented in the Proceedings of the *Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004)*, Vol. 3156 of LNCS, Springer.

Feldhofer, M., Wolkerstorfer, J. and Rijmen, V. (2005) 'AES implementation on a grain of sand', *IEE Proceedings on Information Security*, Vol. 152, pp.13–20.

Goldreich, O., Goldwasser, S. and Micali, S. (1986) 'How to construct pseudorandom functions', *Journal of the ACM*, Vol. 33, pp. 792–807.

Henrici, D. and Müller, P.M. (2004) 'Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers', Paper presented in the Proceedings of the *IEEE International Conference on Pervasive Computing and Communications*, pp.149–153.

Juels, A. (2004) 'Minimalist cryptography for low-cost RFID tags', Paper presented in the Proceedings of the *International Conference on Security in Communication Networks (SCN 2004)*, Vol. 3352 of LNCS, pp.149–164, Springer.

Lee, H. and Hong, D. (2006) 'The tag authentication scheme using self-shrinking generator on RFID system', *Transactions on Engineering, Computing, and Technology*, Vol. 18, pp.52–57.

Menezes, A., van Oorschot, P. and Vanstone, S. (1996) *Handbook of Applied Cryptography*. New York, NY: CRC Press.

Molnar, D., Soppera, A. and Wagner, D. (2006) 'A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags', Paper presented in the Proceedings of the *Workshop on Selected Areas in Cryptography (SAC 2005)*, Vol. 3897 of LNCS, Springer.

Ohkubo, M., Suzuki, K. and Kinoshita, S. (2003) 'Cryptographic approach to 'privacy-friendly' tags', Paper presented in the Proceedings of the *RFID Privacy Workshop*.

Rabin, M.O. (1979) *Digitalized Signatures and Public-key Functions as Intractable as Factorization. Technical Report TR-212*. Cambridge, MA: Massachusetts Institute of Technology.

Shamir, A. (1994) 'Memory efficient variants of public-key schemes for smart card applications', *EURO-CRYPT*, Vol. 1, pp.445–449.

Shamir, A. (2007) 'SQUASH: a new one-way hash function with provable security properties for highly constrained devices such as RFID tags', *In Invited Talk, International Conference on RFID Security (RFIDSec'07)*.

Shamir, A. (2008) 'SQUASH – a new MAC with provable security properties for highly constrained devices such as RFID tags', Paper presented in the Proceedings of the *Workshop on Fast Sofware Encryption (FSE'08)*, Springer.

Sharma, S., Weis, S. and Engels, D. (2003) 'RFID systems and security and privacy implications', Paper presented in the Proceedings of the *Workshop on Cryptographic Hardware and Embedded Systems (CHES 2002), LNCS*, pp.454–470, Springer.

Tsudik, G. (2006) 'YA-TRAP: yet, another trivial RFID authentication protocol', Paper presented in the Proceedings of the *IEEE International Conference on Pervasive Computing and Communications (PerCom 2006)*, IEEE Press.

van Le, T., Burmester, M. and de Medeiros, B. (2007) 'Universally composable and forward-secure RFID authentication and authenticated key exchange', Paper presented in the Proceedings of the *ACM Symposium on Information, Computer, and Communications Security (ASIACCS 2007)*, ACM Press.

## Notes

[1]In this case, the tag validation may be delayed until the next batch interaction (e.g. Tsudik, 2006), or may be immediate with (limited) delegation by the back-end server to the reader (Molnar, Soppera and Wagner, 2006).

[2]Strictly speaking, the compiler introduces the risk that compromise of the trapdoor invalidates forward-security properties (that might be enjoyed by the protocol pre-compilation). However, as commonly done in RFID authentication research, we do not consider security threats against the trusted server.

[3]The UC model allows to capture security notions for the case when the computational power of parties is described by other complexity classes, however we restrict ourselves to security definitions that can be expressed in the complexity class BPP: The adversary must have a success probability bounded away from 0 by $1/p(t)$, where $p$ is some polynomial and $t$ is the complexity parameter as some natural function of input length.