Minimalist Cryptography for Low-Cost RFID Tags

Ari Juels

RSA Laboratories
Bedford, MA 01730, USA
e-mail: ajuels@rsasecurity.com

Abstract. A radio-frequency identification (RFID) tag is a small, inexpensive microchip that emits an identifier in response to a query from a nearby reader. The price of these tags promises to drop to the range of \$0.05 per unit in the next several years, offering a viable and powerful replacement for barcodes.
The challenge in providing security for low-cost RFID tags is that they are computationally weak devices, unable to perform even basic symmetric-key cryptographic operations. Security researchers often therefore assume that good privacy protection in RFID tags is unattainable.
In this paper, we explore a notion of *minimalist cryptography* suitable for RFID tags. We consider the type of security obtainable in RFID devices with a small amount of rewritable memory, but very limited computing capability. Our aim is to show that standard cryptography is not necessary as a starting point for improving security of very weak RFID devices. Our contribution is threefold:

1. We propose a new formal security model for authentication and privacy in RFID tags. This model takes into account the natural computational limitations and the likely attack scenarios for RFID tags in real-world settings. It represents a useful divergence from standard cryptographic security modeling, and thus a new view of practical formalization of minimal security requirements for low-cost RFID-tag security.
2. We describe protocol that provably achieves the properties of authentication and privacy in RFID tags in our proposed model, and in a good practical sense. Our proposed protocol involves no computationally intensive cryptographic operations, and relatively little storage.
3. Of particular practical interest, we describe some reduced-functionality variants of our protocol. We show, for instance, how static pseudonyms may considerably enhance security against eavesdropping in low-cost RFID tags. Our most basic static-pseudonym proposals require virtually no increase in existing RFID tag resources.

Key words: authentication, privacy, pseudonyms, RFID tags

1 Introduction

A passive *radio-frequency identification* (RFID) tag is a microchip that is capable of transmitting a static identifier or serial number for a short distance. It is typically activated by a query from a nearby reader, which also transmits power for the operation of the tag. Several varieties of RFID tag are already familiar in daily life. Examples include the small plaques mounted on car windshields for the purpose of automated toll payment, the theft-detection tags attached in shops to consumer goods such as clothing, and the proximity cards used to control physical access to buildings. More expensive RFID tags can execute advanced cryptographic and other functions, but we concern ourselves in this paper with the inexpensive variety geared to serve as a next-generation successor to barcodes.

The cost of rudimentary RFID tags promises to drop to roughly \$0.05/unit in the next several years [34], while tags as small as 0.4mm $\times$ 0.4mm, and thin enough to be embedded in paper are already commercially available [43]. Such improvements in cost and size augur a rapid proliferation of RFID tags into many areas of use. Indeed, Wal-Mart has issued a directive to its top one hundred suppliers requiring deployment of RFID at the pallet level [10], while The Gillette Company has recently placed an order for half a billion tags for use in supply-chain and retail environments [12]. A goal of researchers in RFID tag development is to see them serve ubiquitously as a replacement for barcodes. This change promises more flexible and intelligent handling of consumer goods and devices. Here are just a few enticing possibilities: Microwave ovens that can read the tags on packages and cook food without explicit instructions, refrigerators that can recognize expired and depleted foodstuffs, and closets that can inventory their contents (and perform a Web search for custom fashion advice).

The impending ubiquity of RFID tags, however, also poses a potentially widespread threat to consumer privacy [27]. If RFID tags are easily readable, then tagged items will be subject to indiscriminate physical tracking, as will their owners and bearers. Researchers have recognized this problem for some time [21, 35], and have yet to propose a truly satisfactory remedy. The issue has also seen recent attention in the popular press, whose negative news coverage forced the clothing retailer Benetton to withdraw plans for embedding RFID tags in its items of apparel [6, 37]. Corporate privacy is similarly problematic, as RFID tags can facilitate corporate espionage by revealing information about the operation of supply chains.

Auto-ID Labs and EPC Global (together formerly known as the Auto-ID Center) have been leading institutions in the development and standardization of RFID tags. Their initial RFID-chip designs are geared toward general corporate and consumer use. So as to permit inexpensive manufacture, they carry only the most basic functionality, emitting a static, 96-to-256-bit identifier on receiving a reader query [34]. Auto-ID Center chip designs give recognition to importance of privacy by permitting an RFID tag to be "killed," i.e., rendered permanently inoperable on receiving a short, specially designated key [35]. Other design proposals propose a pair of complementary "sleep" and "wake" commands that allow a chip to be rendered inoperable on a temporary basis. Thus, for example, a supermarket might deploy RFID tags to facilitate tracking of shipments and monitoring of shelf stocks. To protect the privacy of customers, checkout clerks might "kill" the tags of purchased goods. Alternatively, to permit tag use in the home, a consumer might furnish a secret "sleep" key at the time of checkout. This key could be used to put tags to sleep when the consumer leaves the supermarket, and to reawaken them for later use.

There are many environments, however, in which simple measures like use of "kill" or "sleep" commands are unworkable or undesirable for privacy enforcement. Consumers may wish RFID tags in their possession to remain active, or may simply find it inconvenient to manage their wake/sleep patterns. Businesses may have concerns about unauthorized monitoring of tags before they are "killed." We enumerate a few examples here of important uses and privacy concerns for which "kill" or "sleep" commands are unsatisfactory:

– **Access delegation**: A consumer may wish certain tags in her possession to be permanently active so as to enable reading by other parties. For example, a consumer might wish to use

RFID tags for effortless physical access control,[1] for theft-protection of belongings, for wireless cash and fidelity cards, and so forth. New and clever consumer applications are already beginning to emerge. For example, a Prada store in New York City tracks the RFID tags of items held by customers in order to display related accessories on nearby screens [2]. Function creep promises to result in many more uses unimagined or unimaginable today.

- **Consumer use**: As mentioned above, RFID readers may eventually be inexpensive enough and RFID tags prevalent enough to make a range of smart appliances practical in the home. In the shorter term, there are other consumer benefits, like the ability of consumers to return RFID-tags items to shops without the need for a receipt.
- **Industrial espionage**: Industrial espionage is a likely concern prior to the "killing" of tags. This is true, for example, in a retail environment, where a competitor capable of reading tags in shops or warehouses may gather business intelligence regarding the turnover rate of stocks, the shopping patterns of customers, and so forth.
- **Banknote tracking**: If tags are embedded in banknotes, then they must be permanently accessible to law enforcement agencies. One straightforward approach to enforcing privacy would be to distribute banknotes in a "sleep" state, and to assign a "waking" key to law enforcement. This is problematic in that to awaken banknote tags, a law enforcement reader must transmit the key, rendering it easily vulnerable to capture. Keys cannot be assigned on a fixed per-banknote basis, because in that case a banknote would have to emit a unique identifier in order to enable law enforcement to determine the correct key for that banknote. Thus a given awakening key would potentially have to be associated with a wide batch of banknotes, in which case one would expect privacy to be swiftly and broadly compromised.

RFID tags that promiscuously emit static serial numbers pose another serious problem, namely that of authentication. Such tags may be easily cloned by an attacker that has read access: The attacker need merely read the RFID tags of passersby to harvest their identifiers for later re-use. This is highly problematic for a number of the current and projected uses of RFID tags, most notably physical access to buildings via passive RFID tokens, and inventory tracking (especially with an eye to protection against counterfeiting). Privacy protection and the problem of authentication are thus intimately related, a fact highlighted by our investigations in this paper.

One of the most advanced of the current generation of small, inexpensive RFID tags is the Atmel TK5552 [11]. This tag has 992 bits of storage and a data transmission rate of about 100kB / sec. It permits both reading and writing to the contents of its memory. The Atmel TK5552, however, costs as much as \$1.00 per unit. Projections on the likely resources in several years of RFID tags with cost in the vicinity of \$0.05 include several hundred bits of memory and somewhere between 5,000 and 10,000 logical gates [33], of which a considerable fraction will be required for basic tag functions. Such RFID tags may be expected to perform some basic computational operations, but not conventional cryptographic ones. At best, they may include security functions involving static keys, such as keyed reads and keyed writes, i.e., essentially just PIN-controlled data accesses.

Remark: One might take the view that Moore's law will ensure greater processing power on tags in the coming years, and thus that cryptographic functionality will eventually be available

[1] Smartcards with RF-enabled chips are in fact in use for this purpose today, but generally only function in very close proximity to readers.

in five-cent tags. There is a competing phenomenon in this case, though: Users of low-end RFID tags are more concerned to see prices drop and RFID tags become more widespread than to see functionality increase. This means that cryptographic functionality in basic tags may be some time in coming.

1.1 Our work: minimalist cryptography

Our goal in this paper is to elaborate for RFID tags a notion of *minimalist cryptography*. We first seek to characterize common adversarial capabilities in the special security environment that RFID tags present. As a complementary endeavor, we investigate security designs and key management involving severely restricted computing resources. Our main goal is to show that standard cryptographic functionality is not needed to achieve stronger security in RFID tags.

To begin with, we present a security model for an adversary that we consider representative of real-world attack scenarios for RFID. This is an important new contribution of our work. As we show, despite the limited capabilities of RFID tags, RFID systems offer the security architect a special advantage. Like normal users, adversaries in an RFID-system are physically constrained: They must have physical proximity to RFID tags in order to read (and therefore attack) them. Such adversaries are necessarily weaker than in a traditional cryptographic setting. They also have more complex restrictions on their palette of attacks. The model we propose aims to capture these distinct adversarial characteristics. This model may not be perfect, but it aims to undercut some of the standard cryptographic assumptions that may not be appropriate for real-world deployments.

A fortunate feature of our security model is the fact that it is possible to design protocols without reliance on traditional cryptographic primitives. This turns out to be essential in the setting we consider. As explained above, low-cost RFID tags in particular are incapable of performing the most basic cryptographic operations – even those involving symmetric-key primitives. Such RFID tags cannot in fact withstand strong adversarial attacks of the kind usually considered in cryptographic security models, where an adversary has general "oracle" access, i.e., a largely unrestricted ability to interact with participating entities.

Given the features of our proposed model, we show how privacy and authentication may be considerably improved in low-cost RFID tags with only a small enhancement of their capabilities – which we refer to as minimalist cryptography. We propose a scheme that may be implemented in RFID tags with just several hundred bits of memory and read/write enablement, that is, in tags roughly comparable to the Atmel TK5552 or to the $0.05-per-unit tags anticipated in the near future. We refer to this scheme as *pseudonym throttling*.

Pseudonym throttling is conceptually simple approach to RFID-tag authentication in which an RFID tag stores a short list of random identifiers or pseudonyms (known by authorized verifiers to be equivalent). Each time the tag is queried, it emits the next pseudonym in the list, cycling to the beginning when the list is exhausted. Combined with this feature is the physical imposition of a low query-response rate in the tag. By using hardware-based delays, tags may be made to emit identifiers at a relatively low prescribed rate. (Indeed, delay-based throttling has already seen practical demonstration: Alien Technologies incorporates a throttling mechanism into its current generation of inexpensive RFID tags to prevent guessing of "kill" codes, i.e., the PINs used to disable tags in retail environments [32].)

Alternatively, in higher-end RFID tags that permit user involvement, a user might need to press a button to initiate reading of the tag: This would constitute a different form of throttling. Given the presence of a throttling mechanism, an attacker can only track an RFID tag with a high likelihood of success if she has access to it for a long, continuous period of time, or at many different times. In the latter case, moreover, the ability of the attacker to link pseudonyms is limited, as the tag continually changes appearance.

Pseudonym throttling is simple and practical, but has a shortcoming: The small storage capacity of RFID tags permits only a small list of pseudonyms, and hence only limited privacy protection. Our full protocol allows pseudonyms in an RFID tag to be refreshed by authorized verifiers. In consequence, an additional feature required of our pseudonym-throttling scheme is authentication between tags and verifiers. Our proposed protocol provides such authentication with carefully conceived properties. This is a useful contribution of our work, and interrelated with our exploration of privacy. Given its range of security features, our full pseudonym-throttling protocol necessarily involves multiple flows, and is thus more complex than mere identifier emission. Adhering to the design principle of minimalist cryptography, our protocol involves operations no more computationally intensive than rudimentary memory management, string comparisons, and a basic XOR. To achieve privacy, we propose a special scheme involving composition of one-time pads across protocol sessions.

We emphasize that writeable memory of the type needed for our full-blown protocol may or may not ultimately be more expensive than the logic required to perform standard cryptographic operations. Our main goal here is to demonstrate how a different allocation of resources – namely a shift in favor of memory rather than computation – can subserve important security goals in a new way. This is particularly true as resource costs change on a regular basis. Given emerging breakthroughs in non-organic storage media, re-writeable memory may ultimately prove very inexpensive [**?**]

There is also some practical offshoot of our investigations with more immediate potential. We describe a few reduced-functionality variants of our basic pseudonym scheme that require very little supplementation of existing tag resources. These simple variants help offer security against the real-world threat of passive eavesdropping. Although the effective read distance of RFID tags is fairly short, the readers themselves broadcast tag identifiers for long distances – indeed, up to as much as a kilometer. Our simple techniques help address this problem.

1.2 Related work in data security

There is a considerable body of research on the design of lightweight public-key encryption and digital-signing algorithms – largely intended for use in smart cards and similarly small computational devices. These algorithms include identification or digital-signature schemes such as the classic Guillou-Quisquater algorithm [18] and also newer algorithms like the NTRU cryptosystem [19]. Even the most lightweight of these many schemes, e.g., [41], is likely to be well beyond the capabilities of small RFID tags for quite some time to come.

A related line of research considers the use of symmetric-key algorithms as a lightweight mechanism for authentication, again with the aim of enhancing security functionality on small devices. Perrig [30], for example, shows how hash chains may be used in conjunction with time-synchronization to achieve much the same functionality as that offered by public-key authentication techniques. RSA SecurID^{TM} is a popular device for remote authentication that relies on time-synchronization and use of a hash function to process a secret key shared with a server [23]. Remote keyless entry (RKE) systems are the protocols used to enabling consumers to unlock automobiles remotely using small fobs. (See [15] for a concise description.) Generally proprietary, these algorithms typically rely on the use of pseudo-random number generation to produce authentication codes that change from session to session. As already explained, however, even compact symmetric-key algorithms are in general too resource-intensive for incorporation into the current generation of low-cost RFID tags.

The use of pseudonyms as a privacy-preserving tool has seen wide ranging theoretical and practical application in computing environments. Cryptographic treatment of pseudonyms has focused on an early concept due to Chaum [9, 26], in which an entity presents different pseudonymous credentials to different verifiers and is capable of proving statements about these credentials in a privacy-preserving manner. Such pseudonymous and anonymous systems exploit the full flexibility of public-key cryptography, and as such tend to be rather heavyweight.

The pseudonym schemes in actual use are much more rudimentary. Pseudonyms are in common use in subscription-based systems on the Internet. For example, users can easily obtain e-mail accounts bearing pseudonymous identifiers. More strongly enforced pseudonymous network interfaces are achievable with the use of mix networks [8]. The now-defunct Freedom service of Zero Knowledge Systems, for example, permitted users to adopt any of a range of pseudonyms for Internet transactions [16]. A characteristic of such systems is their context dependence. In other words, users select from among a small set of persistent pseudonyms so as to achieve a consistent persona in a particular environment. Thus, in contrast with many other pseudonym systems, there are three important points of distinction in our approach to RFID-tag privacy: (1) The need for protocols that involve consistent rotation of pseudonyms without reference to external information, since RFID tags have only minimal awareness of transactional context; (2) The need for external generation of pseudonyms. Without access to cryptographic functions or a source of randomness, RFID tags cannot select their own pseudonyms, while most privacy-preserving systems have the benefit of direct selection of pseudonyms by pseudonymous entities; and (3) The fact that pseudonyms on an RFID tag must be linkable by an authorized verifier, a feature more common in public-key-based cryptographic pseudonym systems than in the lightweight pseudonym systems in practical use.

The threat of invasive physical tracking of people by means of remotely-readable device identifiers has already begun to manifest itself in everyday life. For example, users of EZPass, a system for toll payment employing RFID transponders in cars, may have their records subpoened in divorce cases in New York State for the purpose of proving claims of marital infidelity [42]. Jakobsson and Wetzel have pointed out a similar, but potentially much more pervasive threat to privacy in the authentication protocol for Bluetooth, a communication scheme that promises to proliferate into a wide range of electronic devices [20]. They propose frequent rotation of pseudonyms as a countermeasure.

The problem of security modeling for RFID-tag systems may be viewed as similar in flavor to that for ad-hoc wireless networks. This is true both in terms of the restricted power of participating devices and in terms of the rapid changes in their physical and therefore logical relationships. There is little formal work on security modeling particular to the special characteristics of ad-hoc networks, although it is an emerging area of interest. Of particular note is the “resurrecting duckling” idea of Stajano and Anderson [40], who consider secure authentication between devices in ad-hoc networks. As we do here, they examine the way that physical proximity may be treated as an element in security modeling.

1.3 Related work on RFID

Researchers have from the outset recognized the possibility of privacy threats from physical tracking in the deployment of RFID tags [35]. Several recent papers have proposed ways of addressing the problem. Juels and Pappu [21] consider a purported plan by the European Central Bank to embed RFID tags in Euro banknotes [1]. They propose a privacy-protecting scheme in which RFID tags carry ciphertexts on the serial numbers of banknotes. These ciphertexts are subject to re-encryption by computational devices in shops, thereby rendering multiple appearances of a given RFID tag unlinkable. The Juels/Pappu scheme, however, assumes a single verifying entity – namely a law-enforcement organization – and is not obviously extensible to the multi-verifier systems likely in commercial and consumer environments. A scheme of Golle, Jakobsson, Juels, and Syverson [17] builds on this idea with a primitive known as universal encryption, essentially a special extension of the El Gamal cryptosystem [13] in which re-encryption is possible without knowledge of public keys. The Golle *et al.* approach is geared toward general consumer use, as it does not require a centralized verifying entity. It has the drawback, though, of requiring an infrastructure of agents capable of performing public-key-based re-encryption for privacy protection of RFID tags.

Weis, Sarma, Rivest, and Engels [45] also propose a collection of privacy-enforcement ideas for RFID tags in general environments. First, they identify the problem of attacks based on eavesdropping rather than active tag queries. Recognizing that transmission on the tag-to-reader channel is much weaker than that on the reader-to-tag channel, they propose protocols in which tag-identifying information is concealed on the stronger channel. They also propose privacy-preserving schemes for active attacks. One scheme involves the use of a hash function to protect the key used for read-access to the tag. Another includes use of a pseudo-random number generator to protect tag identities. In a nutshell, their idea is for the tag to output the pair $(r, PRNG(ID, r))$, where r is a counter, ID is the secret tag identifier and $PRNG$ denotes a pseudo-random number generator. A verifier must perform an expensive brute-force lookup in order to extract the ID from such an output. The authors note that this drawback probably limits applicability of the idea to small systems. They also note that it is unclear how and when adequate pseudo-random number generators can be deployed on inexpensive RFID tags.

Juels, Rivest, and Szydlo [22] describe a privacy-protection tool they call a "blocker" tag. This is an RFID tag that can obstruct reading of tag identifiers within a certain numerical range by simulating the presence of RFID tags bearing *all* identifiers in that range. This is accomplished through non-standard interaction with the "tree-walking" or ALOHA protocols employed in current tag-reading standards [24, 34]. So as not to serve as a purely disruptive mechanism, the blocker may be accompanied by a form of privacy "zoning," according to which only the reading of a certain subset of identifiers is disrupted. Thus, tag identifiers may be "zoned" while in the possession of manufacturers and retailers such that their reading may take place without impediment. Before tags are placed in the hands of consumers, their "zoning" may be changed so that identifiers cannot be read in the presence of a "blocker," thereby enforcing the privacy of consumers. At the same time, the "blocker" concept offers more flexible privacy options than tag disablement: By deactivating a "blocker" or removing it from the vicinity of her tags, a consumer can still make use of tags. While a practical and attractive proposal for businesses and consumers alike, the "blocker" tag has limited applicability. For example, it does not address the problem of industrial espionage: In most portions of the supply chain, it is impractical to block tags, because they must be readable for industrial use. In contrast, the privacy protection of our proposal functions under general conditions, and requires no special action on the part of the user. Our proposal has a slightly different aim than the blocker, however: It permits reading of a tag by an authorized verifier, while the blocker prohibits reading categorically within the limits of its policy.

A rather different, complementary perspective on privacy for RFID tags is that of Garfinkel [14], who elaborates a policy for consumer privacy-protection in the form of a proposed "RFID Bill of Rights." Proposed there are: The right of the consumer to know what items possess RFID tags and the right to have tags removed or deactivated upon purchase of these items, the right of the consumer to access of the data associated with an RFID tag, the right to access of services without mandatory use of RFID tags, and finally the right to know to when, where, and why the data in RFID tags is used.

1.4 Organization

In section 2, we outline our security model for privacy and authentication in RFID tags (relegating formal details to the appendices). We describe our scheme for RFID-tag privacy in section 3. In section 4, we discuss practical deployment issues and introduce some reduced-functionality variants of our scheme with potential for short-term, real-world application. We conclude in section 5 with some discussion of future research directions. In appendix A, we present our formal security model. We provide formal security definitions in appendix B, followed by statements and proofs of concrete security bounds in appendix C. Finally, in appendix D we briefly discuss extensions of our protocols assuming the availability of on-board pseudo-random number generation.

2 A Security Model for RFID Tags

Given the very basic functionality of RFID tags, it is natural to consider an adversary in an RFID-tag system whose capabilities are quite limited. In most cryptographic security definitions, as for IND-CCA security on public-key encryption schemes [4], an adversary is presumed to be able to experiment extensively with elements of the system in the course of mounting an attack. In particular, the adversary is regarded as capable of submitting a large number of "oracle" queries, that is, exploratory inputs to the cryptographic operations composing the system. (In asymptotic analyses, the number of such oracle queries is polynomially bounded in the security parameters for the system; in concrete analyses, the bound on queries aims to reflect the limits of current computing ability, and may be on the order of, say, 2^{80} for local computation. Smaller bounds, e.g., 2^{30} may be imposed for practical modeling where interaction with, e.g., an actual signing or decrypting party is involved.)

In modeling an RFID system, it is natural to treat both tags and tag-verifiers as oracles. Given the limited computing ability of tags, however, a practical system cannot feasibly withstand an adversary that can submit a large number of arbitrarily ordered queries to all oracles in the system. Moreover, a high degree of adversarial power would not accurately reflect the physical characteristics of an RFID-tag system. Both readers and tags operate only at short range, and tags may in many cases be highly mobile. Thus, the collection of "oracles" available to an adversary at a given time is likely to be small in practice.

We seek to model the limitations on adversarial power in an RFID-tag system by the following key assumption: *An adversary may only interact with a given tag on a limited basis before that tag is able in turn to interact in a protected manner with a valid verifier.* We refer to this protected interaction as a *refresh*. In particular, a refresh is a privacy and integrity-protected session between a verifier and tag in which the verifier may update keying data in the tag. A refresh models the use of a tag with a legitimate reader outside the range of the adversary. In our security model, we impose two restrictions on adversarial interaction with tags between refreshes:

Limited successive tag queries: We assume that an adversary may interact with targeted RFID tags only a relatively small number of times in rapid succession prior to a refresh. This restriction would follow naturally from use of the throttling mechanism that we propose. Suppose, for example, that an RFID tag only permits reading once every several seconds. Given that an RFID-tag typically has a read range of at most a few meters, a rogue reader would have difficulty in harvesting more than, say, one or two pseudonyms from most passersby; tags might easily store half-a-dozen or so pseudonyms, however.[2] An attacker bringing a reader into a monitored environment like a shop or warehouse might similarly face difficulties in attempting prolonged intelligence gathering.

We rely on this assumption to help enforce privacy protection in our proposed protocol.

Limited interleaving: We assume a restriction on the ability of an adversary to mount man-in-the-middle attacks between tags and legitimate readers. This assumption reflects the following adversarial constraints in real-world scenarios:

- *Stationary attacker*: A sophisticated adversary has the potential to mount a full-blown man-in-the-middle attack. Such an adversary might, for example, maintain a physical presence in proximity to a legitimate reader and alter, eavesdrop on, or inject messages to and from tags. There are two complementary impediments to such an attacker, one innate to many RFID-tag environments, another part of our proposal in this paper:
  1. Mobility of tags: In many cases, it is operationally inconvenient for an adversary to interact for an extended period of time with tags in the vicinity of legitimate readers. For example, if a reader were stationed so as to regulate physical access to a building or to permit automated checkout at a supermarket, then the mobility of users (and consequently of tags) would help ensure only a limited number of protocol flows for attack by the adversary.
  2. Throttling: Part of our proposal in this paper, throttling helps restrict the number of successive adversarial queries. It may be thought of as a defensive measure exercised

[2] Other throttling schemes are possible of course. For example, a tag might permit the reading of two pseudonyms a few seconds apart (in case of an initial read failure), but restrict access to others for a number of minutes. This would render attack even more difficult. Care is required to minimize the risk of denial-of-service attacks. We do not explore the issue of delay scheduling in detail here.

by stationary or lightly mobile tags against a persistent attacker. (In a sense, throttling boosts or simulates the natural security properties of mobile tags.) Moreover, in the face of a passive attack, a reader can help implement its own throttling policy by, e.g., refusing to initiate sessions with a particular tag in rapid succession.[3]

– *Mobile attacker*: An attacker might scan RFID tags and then use harvested information to interact with readers. Such an attacker, however, has only a limited ability to perform a man-in-the-middle attack, since this requires shuttling back and forth between tags and legitimate readers. (Indeed, our proposed scheme achieves secure authentication against an attacker of this kind irrespective of the amount of interleaving.)

We rely on the assumption of limited interleaving to help enforce both privacy and authentication properties in our proposed protocol.

We reiterate that our assumptions *do not characterize the strongest possible type of adversary.* One can easily envisage a sophisticated adversary violating these assumptions to a greater or lesser degree – particularly if targeting a single or small number of RFID tags or individuals. Our goal in this paper is to achieve good, practical security by defending against a broad, real-world class of attacks. Viewed another way, we try to minimize security vulnerabilities in this constrained environment, but do not expect to eliminate them.

Remark: Our model does not explicitly capture one important feature of RFID systems. While tags may be feasibly read at only a short distance, it is possible to eavesdrop on readers from a considerably larger distance, as they are powered broadcast devices. Thus, a passive attacker can in principle harvest reader-to-tag data more easily than tag-to-reader data. Our model does characterize this situation if it is assumed that an adversary eavesdrops only intermittently – or, more realistically, that tags are read by different readers at different times, and therefore not always near readers monitored by the adversary. More importantly, in the protocol we propose here, an eavesdropper on reader-to-tag transmissions does not receive tag identifiers. Therefore, such an eavesdropper has no way of determining which data correspond to which tags.

3 Our Proposed Scheme

As explained above, our proposed protocol relies upon rotation by a tag through multiple pseudonyms, which we denote by $\alpha_1, \alpha_2, \ldots, \alpha_k$. These pseudonyms, however, do not themselves serve as the sole means of authentication for tags. If a tag authenticated itself to a verifier merely by releasing a key α_i, then an adversary could clone a tag very simply as follows. The adversary would query the target tag, obtaining α_i; the adversary would then separately interact with the verifier, using the key α_i to simulate a valid tag. Indeed, this is precisely the type of cloning attack to which standard RFID tags with static identifiers are vulnerable, e.g., current EPC designs [34]. Any single-flow protocol is necessarily vulnerable to such an attack.

[3] A more sophisticated adversary might make use of two communicating devices: One simulating a valid tag near a reader, and another simulating a reader near a valid tag. This type of adversary can straightforwardly perform a full man-in-the-middle attack on any type of RF system that does not involve explicit user participation. Even a system employing sophisticated cryptography cannot defend against such an attack.

To prevent this type of attack in our protocol, a tag only authenticates to a verifier after the verifier has itself authenticated to the tag. The verifier authenticates to a tag by releasing a key β_i; this key β_i is unique to a given pseudonym α_i. Once the verifier has authenticated to the tag, the tag authenticates itself to the verifier by releasing an authentication key γ_i. Like β_i, this authentication key γ_i is unique to an identifier α_i. Briefly stated, we propose a kind of challenge-response protocol, but one that is carefully interwoven with pseudonym rotation.

In order to maintain the integrity of a tag over an extended period of time and in the face of multiple probing attacks by an adversary, we take the approach in our protocol of having the verifier update the $\{\alpha_i\}$, $\{\beta_i\}$, and $\{\gamma_i\}$ values in an RFID tag after successful mutual authentication between tag and verifier. This introduces a new problem, however: An adversary can eavesdrop on or tamper with the secrets used in this update process. Our strategy for addressing this problem is to update values using one-time pads that have been transmitted across multiple authentication protocols. Thus an adversary that only eavesdrops periodically is unlikely to learn the updated $\{\alpha_i\}$, $\{\beta_i\}$, and $\{\gamma_i\}$ values.

Updating tag values in this way provides integrity protection as an important side-benefit. An adversary without knowledge of the one-time pads used during a update cannot, for instance, mount a swapping attack involving the substitution of keys from one compromised tag into another tag.

3.1 One-time pads in our scheme

The one-time pad is, of course, a simple, classical form of encryption. (See, e.g., [28] for discussion.) We briefly recall the underlying idea. If two parties share a secret one-time pad δ, namely a random bitstring of length l, then one party may transmit an l-bit message M secretly to the other via the ciphertext $M \oplus \delta$, where $\oplus$ denotes the XOR operation. It is well known that this form of encryption provides information-theoretic secrecy.

In our scheme, the verifier transmits one-time padding data that the tag uses to update its shared $\{\alpha_i\}$, $\{\beta_i\}$, and $\{\gamma_i\}$ values. Provided that an eavesdropper does not obtain the padding data, she achieves no knowledge of the updated tag values. Although this procedure does not explicitly involve encryption by means of one-time pads, it is essentially equivalent to encryption. We may think of the pads as keys used to "encrypt" and thereby update the $\{\alpha_i\}$, $\{\beta_i\}$, and $\{\gamma_i\}$ values.

Additionally, we introduce a special twist into our use of the one-time pad. Our scheme involves composition of one-time pads across multiple verifier-tag sessions. This has the effect of retaining secrecy in the face of partial adversarial eavesdropping (or tampering). Suppose, for instance, that pads from two different verifier-tag sessions are XORed with a given tag value κ in order to update it. Then even if the adversary intecepts the pad used in one session, it may be seen that she will learn no information about the updated value of κ.

Application of a one-time pad requires only the lightweight computational process of XORing. Like encryption based on the one-time pad, updating tag values via one-time padding also provides information-theoretic security. While this latter property renders security proofs for our system somewhat simpler, it is not a motivation for our choice. Indeed, one-time padding results in less communications efficiency than that achievable with standard cryptographic encryption tools like block or stream ciphers. The problem, as we have already explained, is that standard cryptographic primitives require more computational power than is available in a low-cost RFID tag. This is the real motivation behind our use of one-time pads.

As explained above, we employ a strategy of updating tag values using pads from multiple authentication sessions. Let κ be some value stored in a tag, i.e., $\kappa \in \{\alpha_i\} \bigcup \{\beta_i\} \bigcup \{\gamma_i\}$. Let m be a parameter governing the resistance of the protocol to adversarial eavesdropping (the protocol analog of the value r in our security model in the appendix). For every value κ, we maintain in the tag a vector $\Delta_\kappa = \{\delta_\kappa^{(1)}, \delta_\kappa^{(2)}, \ldots, \delta_\kappa^{(m)}\}$ of one-time pads. The pad $\delta_\kappa^{(1)}$, which we refer to as the *live* pad, is used to update the tag value κ. In particular, to update κ, the tag computes $\kappa \leftarrow \kappa \oplus \delta_\kappa^{(1)}$.

Prior to update of κ, the pads in Δ_κ are updated with new padding material received from the verifier. Let $\tilde{\Delta}_\kappa = \{\tilde{\delta}_\kappa^{(1)}, \tilde{\delta}_\kappa^{(2)}, \ldots, \tilde{\delta}_\kappa^{(m)}\}$ be a vector of newly generated one-time pads received from the verifier in our protocol. The vector Δ_κ is updated as follows. The live pad $\delta_\kappa^{(1)}$ is discarded – as it has already been used to update κ. The indices of all other pads in Δ are then shifted downward, i.e., in increasing index order, we set $\delta_\kappa^{(i)} = \delta_\kappa^{(i+1)}$ for $1 \leq i \leq n-1$. We set $\delta_\kappa^{(m)} = 0^l$, i.e., we fill the last, missing element in the vector with a '0' bitstring. (Alternatively, it is possible to rotate the discarded, previously live pad to the last position in the vector.[4]) Finally, we "overlay" the newly received vector $\tilde{\Delta}_\kappa$ on the existing vector Δ_κ, by performing an element-wise XOR. That is, we let $\delta_\kappa^{(i)} = \delta_\kappa^{(i)} \oplus \tilde{\delta}_\kappa^{(i)}$.

As a result of these manipulations, the vector Δ_κ consists of a set of m one-time pads with decreasing levels of backward secrecy. After the completion of a session, the live pad $\delta_\kappa^{(1)}$, for instance, consists of the XOR of independent pads from the previous m successfully completed sessions. At the other end of the spectrum, the value $\delta_\kappa^{(m)}$ is constituted of only a single pad, namely the one just transmitted in the most recent session. This is why we update κ using the strongest pad in Δ_κ, namely the live one, and then strengthen and "promote" the other pads in Δ_κ by overlaying a vector of newly transmitted ones.

This approach provides information-theoretic security guarantees. In particular, an adversary that has knowledge of only $m-1$ of the last m pad-transmissions from the verifier has no knowledge at all about $\delta_\kappa^{(1)}$. Thus, when the live pad is employed to update κ, such an adversary learns no information whatever about the new value of κ.

The drawback to this approach is that the transmission cost to maintain pads is lm bits per session. In other words, the communications costs in our protocol are linear in the length of individual tag values *and* in the number of consecutive authentication sessions relative to which we wish to achieve security against the adversary. Given that there are $3k$ tag values, this translates into a total cost of $3klm$. This cost is less than ideal, but still permits a range of practical parameterizations, as we discuss below in section 4.

We use the notation $\mathsf{update}(\Delta_\kappa, \tilde{\Delta}_\kappa)$ to denote the function that updates Δ_κ and "overlays" it with $\tilde{\Delta}_\kappa$. We let $\mathsf{pad}(\kappa, \Delta_\kappa)$ denote the update of κ using the live pad $\delta_\kappa^{(1)}$ – again, the one with the strongest backward security. For brevity of notation, we let ABC denote the set of values $\{\alpha_i\} \bigcup \{\beta_i\} \bigcup \{\gamma_i\}$. We let Δ_{ABC} denote padding vectors for all values κ in the set ABC.

[4] This option is probably easier to implement. It also has the (slight) advantage of not causing a newly initialized tag to discard one of its original, secret values.

3.2 The protocol

As above, let k be a parameter denoting the number of pseudonyms stored in a given tag and let m denote the number of authentication sessions over which one-time pads are constructed; in other words, the higher the value of m, the stronger the eavesdropping-resistance of the system. For visual clarity in our protocol figure, we omit variable ranges and tag subscripts on variables for keys. The variables i and j, however, always span the ranges $\{1, 2, \ldots, k\}$ and $\{1, 2, \ldots, m\}$ respectively. We use $\in_R$ here and elsewhere to denote uniform random selection. In case of a message-delivery failure, we assume the input of a special symbol $\perp$ (leading to protocol termination). We assume initialization of all entities by a trusted party, who generates a key set ABC for every tag and distributes this to both the tag and the verifier. All counters are initialized at 0. Details of our protocol are provided in Figure 1.

```
Tag                                                    Verifier

$d \leftarrow (c \bmod k) + 1$
$c \leftarrow c + 1$
$\alpha' \leftarrow \alpha_d$       $\xrightarrow{\alpha'}$       if $\alpha'$ is valid $\alpha_i$ for some tag $T_x$ then
                                                           $tag \leftarrow x$
                                                           $\beta' \leftarrow \beta_i$
                                                           $\gamma \leftarrow \gamma_i$
                                                           mark $\alpha_i$ as invalid for $T_x$
                                                       else
                                                           output("reject") and abort
                                    $\xleftarrow{\beta'}$
if $\beta' \neq \beta_d$ then
    output("reject") and abort
$\gamma' \leftarrow \gamma_d$       $\xrightarrow{\gamma'}$
                                                       if $\gamma' \neq \gamma$ or $\gamma' = \perp$ then
                                                           output("reject") and abort
                                                       $\tilde{\Delta}_{ABC} \in_R \{\{0,1\}^l\}^{3km}$
                                    $\xleftarrow{\tilde{\Delta}_{ABC}}$
                                                       output($tag$, "accept")
$\{\mathsf{update}(\Delta_\kappa, \tilde{\Delta}_\kappa)\}_{\kappa \in ABC}$      $\{\mathsf{update}(\Delta_\kappa, \tilde{\Delta}_\kappa)\}_{\kappa \in ABC}$
$\{\kappa \leftarrow \mathsf{pad}(\kappa, \Delta_\kappa)\}_{\kappa \in ABC}$      $\{\kappa \leftarrow \mathsf{pad}(\kappa, \Delta_\kappa)\}_{\kappa \in ABC}$
```

Fig. 1. Full RFID-tag authentication protocol

Remarks: We assume no collisions among tag identifiers here – a property that can be enforced during tag initialization and updates with only a very slight skew from a uniform random distribution over identifiers.

Due to space limitations, we are forced to relegate formal security definitions and proofs for our proposed protocol to the paper appendices.

4 Practical Deployment

4.1 Pruning our scheme

The full-blown scheme we have proposed is practical for very low-cost tags only with the use of small security parameters. There are a number of strategies, however, for reducing the functionality of scheme while still retaining important properties.

To begin with, in real-world deployments, the moderate security afforded by relatively short keys $\{\beta_i\}$ and perhaps also short $\{\gamma_i\}$ keys would be acceptable in many cases. For example, if β_i and γ_i keys are a mere twenty bits each, then an adversary would have roughly a one-in-a-million chance of defeating the authentication protocol in a single try. Tag pseudonyms, i.e., the $\{\alpha_i\}$ keys, must be considerably longer to permit unique identification of tags and to avoid pseudonym collisions. We believe that 100-bit α values would suffice for this purpose in most environments. (It should be noted, however, that if a pseudonym collision occurs in the naming of a new tag, then different pseudonyms may be selected by the verifier. Such a naming strategy would probably permit a reduction in the lengths of α_i tags to around 80 bits.) In any event, large values of m or k are unlikely to be practical. Indeed, $m = 0$ (no updates via refresh) or 1 and $k = 4$ or 5 might be a reasonable choice for a real-world system.

A range of truncated versions of the protocol itself is also interesting. One example is a scheme that excludes the fourth flow from our protocol. In other words, the ABC values in the tag may remain the same throughout its lifetime. A much reduced variant might involve only the first flow in our protocol. This would mean that a tag merely cycles through a static set of pseudonyms, preferably with the benefit of throttling. This approach offers better privacy assurances than a system using static identifiers, but does not protect against cloning. (Such a degenerate case of our protocol also does not meet our security definitions unless the process of tag refresh in our model is replaced with elimination of a tag from the system.) Simple approaches like this might be especially attractive as a low-cost way of realizing privacy protection for RFID-enabled banknotes, weaker in some respects but involving much less overhead than the scheme proposed in [21]. Another, similarly useful truncation is one in which multiple identifiers $\{\alpha_i\}$ are stored in a tag, but only a single key β and single key γ for common use with all identifiers.

These and kindred approaches have the advantage of backward compatibility with existing RFID systems employing just a static identifier or challenge-response. In other words, a reader does not have to have awareness of the fact than an identifier is in fact a pseudonym: Only the verifying application on the back-end needs to. Such systems would merely have to include some application-level support for linkage of pseudonyms, but would not necessarily require any software or firmware adjustments at the level of the reader.

Another interesting, restricted case is that involving just one identifier, but with the challenge-response and pseudonym replacement protocols intact. This limited variant would be useful for cases in which consumers are borrowing RFID-tagged books from libraries or renting RFID-tagged videos. Use of a single pseudonym like this would not prevent physical tracking. But authenticated rotation of the pseudonym *would* help prevent the bigger problem of passersby being scanned to determine what books or videos they are carrying. Given plans by the San Francisco public library to implant RFID tags in books, and the resistance of civil libertarians in reaction to the USA Patriot Act [31], this seems like a potentially attractive solution.

4.2 Systems integration

As deployed by EPCglobal, RFID tags will carry what is known as "electronic product codes" or EPCs [34]. In addition to information about the product type, manufacturer, and so forth, EPCs will include unique item identifiers. These could be replaced with pseudonyms so as to support our proposed protocols. This would have the effect of providing privacy at the level of unique identifiers, and would therefore be useful in protecting against corporate espionage. (The approach would also be compatible with existing standards inasmuch as it would involve an extension to such standards, rather than a modification.) This would not solve problems of privacy invasion due to, e.g., identification of the types of objects carried by individual consumers. It may be worth instantiating pseudonyms into other fields as well, e.g., item type – or indeed into all fields other than those required for directing database lookups. This problem requires further study.

Although we consider only a single, centralized verifier in our model, it is easy to see that multiple verifiers – like multiple EPC systems – can co-exist side-by-side without real detriment to the privacy and authentication properties. Thus, for example, Shop X and Shop Y can each issue their own tags. Without explicit cooperation, neither will be able to track or otherwise interact with the tags of the other. Shop X will fail to recognize the pseudonyms in the tags generated by Shop Y, and may simply ignore them (and vice versa).

It is important to note that the verifier for our protocol need not store any data for a given tag T_x apart from its static identifier id_x and a counter value on the number of successful authentications for the tag. The α_i values for T_x may be obtained, for example, by encrypting $T_x \parallel z_x$ under a universal symmetric key K_α for the verifier, where z_x is a counter value on the number of pseudonyms issued for the tag T_x. When the verifier receives a pseudonym, it may decrypt it using K_α to obtain the corresponding static identifier. The β_i and γ_i keys may be similarly derived. For example, the verifier might compute β_i and γ_i as the encryption of α_i under universal secret keys K_β and K_γ respectively.

We offer one important caveat on our proposed protocol. The use of multiple flows, as we have pointed out, is essential to forestall cloning attacks. It is very likely, however, to diminish the effectiveness with which RFID tags may be read. It is not immediately clear how much of this performance degradation would occur, but we point out the existence of practical, moderate-cost RFID tags that already perform challenge-response protocols, such as those available from Texas Instruments and employed in the popular ExxonMobil Speedpass system [39, 44].

4.3 Denial-of-service and service failures

When an identifier α_i for a given tag is submitted in a protocol session, it is subsequently treated as invalid for the tag (except in the unlikely case that it recurs in a later refresh). This feature of the protocol is essential for preventing cloning: Once an attacker harvests the corresponding β_i and γ_i values for a tag, they are no longer valid for future sessions. Thus the attacker cannot use these values to clone the tag.

On the other hand, invalidation of α_i values opens up the possibility of a denial-of-service attack. An attacker that is able to harvest and submit all such values for a given tag to the verifier may render the tag inoperable. This is where our use of multiple pseudonyms is valuable, in that it increases the difficulty for an attacker in harvesting and maliciously submitting valid α_i values.

Another avenue for denial-of-service attacks arises from the refresh procedure. By tampering with data in the fourth flow of our protocol, an adversary can render a tag incapable of authenticating successfully, and thereby disable it. We do not consider this issue of active (and relatively sophisticated) denial-of-service attacks here. That is because there are more simple and effective physical means of achieving comparable denial-of-service attacks. An adversary with an electromagnetic weapon, for example, need not resort to cryptographic machinations in order to disable tags.

The reason why we propose loop-around in the rotation through pseudonyms in our protocol is to prevent failures resulting from benign reading of tags. It is to be expected that in normal use, tags will often be scanned by readers that are not associated with their designated verifier. (E.g., as it scans tags on its premises, Shop Y will inadvertently read tags of patrons carrying objects from Shop X.) Without indefinitely permissible rotation through pseudonyms, such benign scanning might cause a tag to exhaust its set of valid pseudonyms.

Finally, we note that because RFID tags are likely to undergo a great deal of wear and tear in the course of ordinary use, some of them are likely to fail in the field. Any RFID system must be equipped to handle such failures. Thus, denial-of-service attacks on small sets of tags are unlikely to pose a serious problem (and may not even be noticeable as such).

4.4 Variant schemes for pseudonym changes

In our main proposal above, a tag switches to a new pseudonym every time it is queried by a reader. There are a number of useful variant ideas for determining tag rotation through pseudonyms. We briefly consider three here:

1. **Per-scan pseudonym specification:** In this proposal, the reader specifies at the beginning of the scanning protocol an index i specifying which pseudonym α_i it would like the tag to emit. This obviously provides little protection against an active adversary, i.e., one that aims to gather information by surreptitiously scanning tags. It offers good protection against passive adversaries, however, as we explain. It also has the advantage of eliminating the need for tags to maintain state, e.g., a counter indicating which pseudonym to transmit. As mentioned above, the broadcast range of the reader in an RFID system is considerably greater than that of a tag – on the order of hundreds of meters, rather than several meters. Moreover, in the tree-walking singulation protocol, the reader transmission completely reveals tag identifiers. (ALOHA protocols typically reveal less information.) As a result, an attacker does not have to scan tags surreptitously to learn their identifiers. Instead, she may passively eavesdrop on a reader at a distance. This is an attractive avenue for corporate espionage, and thus a critical security risk.
   Per-scan pseudonym specification provides a defense against such passive eavesdropping. Let us consider a simple example in which tags contain two pseudonyms and in which challenge-response is not employed. Consider an adversary that eavesdrops on the RFID

readers at Warehouse A and Retailer B. Suppose the aim of the adversary is to learn whether there is a flow of items from A to B (and also, ideally, how large a flow). If Warehouse A's readers always direct tags to emit pseudonym α_1 and Retailer B's readers always direct tags to emit pseudonym α_2, then the eavesdropper cannot tell whether tags originating at A are in fact scanned at B, and will fail in her goal.

In fact, in this and related scenarios, per-scan pseudonym specification can provide *stronger* guarantees than simple pseudonym rotation. Suppose that tags do not include pseudonym refresh capability (as rewritable memory is expensive), but otherwise rotate through a small, static set of pseudonyms as in our initial proposal. If Warehouse A scans tags frequently, then the adversary may be able to harvest all tag pseudonyms $\alpha_1, \alpha_2, \ldots, \alpha_k$. The adversary may not be able to link pseudonyms, i.e., may not be able to determine which pseudonyms reside on the same tag. The adversary will, however, be able to determine whether tags are shipped from A to B, as it will be able to recognize identifiers originating from A.

It is possible to treat the full concatenated set of pseudonyms $\alpha = \alpha_1 \parallel \alpha_2 \parallel \ldots \parallel \alpha_k$ as a single, long identifier. Per-scan pseudonym specification, in this view, simply involves selective reading of portions of a tag identifier. In this view, one can see how to perform per-scan pseudonym specification in a manner that is backward compatible with existing tree-walking singulation protocols. Suppose we express α for a given tag as a concatenated pair of l-bit pseudonyms, i.e., $\alpha = \alpha_1 \parallel \alpha_2$. To return to our example above, Warehouse A might extract identifier α_1 from tags by executing the tree-walking protocol so as to traverse only half the depth of the corresponding identifier tree, i.e., down to depth l. Retailer B might extact identifier α_2 by singulating on only the lower half of the tree, i.e., at depth greater than l. Singulation on the lower half of the tree is supported in Auto-ID standards, e.g., [25], by the availability of a pointer to specify a starting position in identifiers where commands involve string matching. (A number of variants are possible in which, α_1 and α_2, for example, overlap slightly, resulting in slight leakage of linkage data, in which α_1 and α_2 represent tag data other than identifiers, etc.)

2. **Reader setting of pseudonym counters:** As a twist on our first idea, rather than specifying the pseudonym to be emitted by a tag, a reader might explicitly set the pseudonym counter in a tag. As a variant on our two-pseudonym example above, for instance, tags might contain a single, write-once bit b, which is set to $b = 0$ by default. When $b = 0$, the tag emits pseudonym α_1. When $b = 1$, the tag emits pseudonym α_2. To enforce privacy against a passive eavesdropper, then, Warehouse A may simply set $b = 1$ on the tags of items it is about to ship. The consequence will be that A reads α_1, while B reads α_2 for any given tag.

   One small advantage of this approach over per-scan pseudonym specification is that it eliminates the need for A and B to coordinate the determination of which pseudonyms they will read. This may be particularly useful in cases where tags have multiple pseudonyms that are read by multiple entities. Of course, b may be a full-blown counter, and may be re-writeable, rather than just write-once.

3. **Time-elapsed pseudonym changes:** Another possibility is to incorporate timing information into the determination of when to change pseudonyms. For example, a tag may increment its counter only if a certain minimum amount of time has elapsed since the previous reading has taken place. (This timing may be inexpensively enforced using the same hardware mechanisms as pseudonym throttling.) Suppose that Warehouse A scans

items frequently on its premises, but items are not scanned for some time while in transit. Timing-based pseudonym changes will ensure that Warehouse A always sees the same pseudonym for a given tag. The tag counter will be incremented, however, when the tag is scanned by Retailer B, with the result that Retailer B sees a new pseudonym.

This approach has the same advantage as per-scan pseudonym specification of not requiring tag state (apart from the simple and cheap hardware required to perform the timing). At the same time, it has the advantageous feature, like reader setting of counters, of not requiring coordination by entities of which pseudonyms they will scan, as pseudonym changes are automatically executed by the tag. Indeed, for this reason, timing approaches to pseudonym changes provide a useful defense against active, and not just passive adversaries. Additionally, timing information may be combined with information about the number of times a tag has been scanned in order to create relatively sophisticated policies governing the changing of pseudonyms. For example, a tag might implement a policy like the following: On either being scanned ten times or not having been scanned in the past hour, the tag rotates to its next pseudonym; otherwise it continues to emit the same pseudonym.

5 Conclusion: Further Research

Our investigation here has proceeded under the assumption that even standard symmetric-key cryptographic algorithms lie beyond the computational reach of RFID tags. Such algorithms still deserve investigation along the lines of [45]. We may, after all, anticipate greater future capabilities in RFID tags, as well as a broadening of the varieties and pervasiveness of computational devices in everyday surroundings.

One hardware-related problem is that of distributing pseudonyms efficiently to both tags and software applications. Pseudonyms might be determined at the time of manufacture, but it might also be convenient to make a master key for the pseudonyms of a particular tag readable via an optically or physically enabled channel, by analogy with [21, 40]. This would make registration and transfer of ownership more fluid. A comprehensive perspective on key management is thus important in RFID-tag system development.

Finally, security modeling is another line of research that deserves further attention. We feel that the model proposed here captures a range of the special characteristics of RFID-tag environments in an effective way. This model can no doubt benefit from refinement, however, particularly as real-world experience with RFID-tag systems evolves, and as it becomes possible to draw on analogous experience and results from the field of ad-hoc networking. The centralized verifier model that we work with in this paper, for instance, is valuable as a first step toward RFID-system characterization. Further development and understanding of RFID systems will certainly yield other useful models involving varying degrees and forms of decentralization.

Acknowledgements

Thanks to Markus Jakobsson, Burt Kaliski, Ravi Pappu, and Sanjay Sarma for their suggestions and comments on this work. Thanks also to Peter Trei for providing useful references.

References

1. Security technology: Where's the smart money? *The Economist*, pages 69–70. 9 February 2002.
2. Prada's smart tags too clever? *Wired News*, 27 October 2002.
3. M. Bellare, R. Canetti, and H. Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). In *STOC '98*, pages 419–428, 1998.
4. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In H. Krawczyk, editor, *CRYPTO '98*, pages 26–45. Springer-Verlag, 1998. LNCS no. 1462.
5. M. Bellare and P. Rogaway. Entity authentication and key distribution. In D.R. Stinson, editor, *CRYPTO '93*, pages 232–249. Springer-Verlag, 1993. LNCS no. 773.
6. Benetton undecided on use of 'smart tags'. *Associated Press*, 8 April 2003.
7. D. Boneh, N. Modadugu, and M. Kim. Generating RSA keys on a handheld using an untrusted server. In B.K. Roy and E. Okamoto, editors, *Indocrypt '00*, pages 271–282. Springer-Verlag, 2000. LNCS no. 1977.
8. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.
9. D. Chaum. Security without identifications: transaction systems to make Big Brother obsolete. *Communications of the ACM*, 28(10), 1985.
10. J. Collins. The cost of Wal-Mart's RFID edict. *RFID Journal*, 10 Sept. 2003.
11. Atmel Corporation. Atmel TK5552 data sheet, 2001. Referenced at http://www.atmel.com/atmel/products/prod227.htm.
12. D.M. Ewatt and M. Hayes. Gillette razors get new edge: RFID tags. *Information Week*, 13 January 2003. Referenced at http://www.informationweek.com/story/IWK20030110S0028.
13. T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.
14. S. Garfinkel. An RFID Bill of Rights. *Technology Review*, page 35, October 2002.
15. G. Goebel. *Codes, Ciphers, and Codebreaking.* 1 March 2002. Online publication. Referenced at http://www.vectorsite.net/ttcode.html.
16. I. Goldberg and A. Shostack. Freedom network 1.0 architecture, November 1999.
17. P. Golle, M. Jakobsson, A. Juels, and P. Syverson. Universal re-encryption for mixnets. Springer-Verlag, 2004. To appear.
18. L. Guillou and J. Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In C. G. Günther, editor, *EUROCRYPT '88*, pages 123–128. Springer-Verlag, 1988. LNCS no. 330.
19. J. Hoffstein, J. Pipher, and J.H. Silverman. NTRU: A ring based public key cryptosystem. In *ANTS III*, pages 267–288, 1998. LNCS no. 1423.
20. M. Jakobsson and S. Wetzel. Security weaknesses in Bluetooth. In D. Naccache, editor, *RSA-CT '01*, pages 176–191. Springer-Verlag, 2001. LNCS no. 2020.
21. A. Juels and R. Pappu. Squealing Euros: Privacy protection in RFID-enabled banknotes. In R. Wright, editor, *Financial Cryptography '03*, pages 103–121. Springer-Verlag, 2003. LNCS no. 2742.
22. A. Juels, R.L. Rivest, and M. Szydlo. The blocker tag: Selective blocking of RFID tags for consumer privacy. In V. Atluri, editor, *8th ACM Conference on Computer and Communications Security*, pages 103–111. ACM Press, 2003.
23. RSA Laboratories. What is SecurID?, 2003. Referenced at http://www.rsasecurity.com/rsalabs/faq/5-2-5.html.
24. Auto-ID Labs. 13.56 MHz ISM band class 1 radio frequency identification tag interference specification: Candidate recommendation, version 1.0.0. Technical Report MIT-AUTOID-WH-002, Auto-ID Labs, 2003. Referenced at http://www.autoidlabs.org.
25. Auto-ID Labs. 860 MHz-960 MHz class 1 radio frequency identification tag radio frequency and logical communication interface standard: Recommended standard, version 1.0.0. Technical Report MIT-AUTOID-TR-007, Auto-ID Labs, 2003. Referenced at http://www.autoidlabs.org.
26. A. Lysyanskaya, R. L. Rivest, A. Sahai, and S. Wolf. Pseudonym systems. In H.M. Heys and C.M. Adams, editors, *Selected Areas in Cryptography*, pages 184–199. Springer-Verlag, 1999. LNCS no. 1758.
27. D. McCullagh. RFID tags: Big Brother in small packages. *CNet*, 13 January 2003. Referenced at http://news.com.com/2010-1069-980325.html.

28. A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.
29. S. Micali, S. Even, and O. Goldreich. On-line/off-line digital signatures. *Journal of Cryptology*, 9(1):35–67, 1996.
30. A. Perrig, R. Canetti, J. D. Tygar, and D. Song. The TESLA broadcast authentication protocol. *Crypto-Bytes*, 5 (Summer), 2002.
31. Associated Press. Libraries eye RFID to track books: Privacy issues raised as San Francisco plans chips' use. 3 Oct. 2003.
32. RFID, privacy, and corporate data. *RFID Journal*, 2 June 2003. Feature article. Referenced at www.rfidjournal.com on subscription basis.
33. S. E. Sarma, S. A. Weis, and D.W. Engels. Radio-frequency-identification security risks and challenges. *CryptoBytes*, 6(1), 2003.
34. S.E. Sarma. Towards the five-cent tag. Technical Report MIT-AUTOID-WH-006, Auto-ID Labs, 2001. Referenced at http://www.autoidlabs.org/.
35. S.E. Sarma, S.A. Weis, and D.W. Engels. RFID systems and security and privacy implications. In B. Kaliski, editor, *CHES '02*, pages 454–469. Springer-Verlag, 2002. LNCS no. 2523.
36. A. Shamir and Y. Tauman. Improved online/offline signature schemes. In J. Kilian, editor, *CRYPTO '01*, pages 355–367. Springer-Verlag, 2001. LNCS no. 2139.
37. R. Shim. Benetton to track clothing with ID chips. *CNET*, 11 March 2003. Referenced at http://news.com.com/2100-1019-992131.html.
38. V. Shoup. On formal models for secure key exchange (version 4), 15 November 1999. Revision of IBM Research Report RZ 3120 (April 1999).
39. Texas Instruments radio frequency identification systems demonstrates retail RFID solutions, 23 April 2001. Referenced at http://www.ti.com/tiris/docs/news/news_releases/2001/rel4-23-01.shtml. Discusses ExxonMobil Speed-pass.
40. F. Stajano and R. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *7th International Workshop on Security Protocols*, pages 172–194. Springer-Verlag, 1999. LNCS no. 1796.
41. J. Stern and J. Stern. Cryptanalysis of the OTM signature scheme from FC'02. In R. Wright, editor, *Financial Cryptography '03*, pages 138–148. Springer-Verlag, 2003. LNCS no. 2742.
42. S. Stern. Security trumps privacy. *Christian Science Monitor*, 20 December 2001.
43. K. Takaragi, M. Usami, R. Imura, R. Itsuki, and T. Satoh. An ultra small individual recognition security chip. *IEEE Micro*, 21(6):43–49, 2001.
44. Texas Instruments. Radio frequency identification products sheet, 2003. Referenced at http://www.ti.com/tiris/docs/products/transponders/transponders.shtml.
45. S. A. Weis, S. Sarma, R. Rivest, and D. Engels. Security and privacy aspects of low-cost radio frequency identification systems. In *First International Conference on Security in Pervasive Computing*, 2003. To appear.

A Formal Security Model

We describe here the formal model underlying our security definitions. We first lay out a broad but not fully specified description, and then fill in missing details in section A.1.

We consider a single, centralized verifier, denoted by V, and a set of n tags, denoted by $T_1, T_2, \ldots, T_n$. All entities possess state. V communicates with tags by means of an authentication protocol AP that comprises a fixed number of flows. For convenient notation, tags are presumed to initiate the protocol. (Indeed, readers or verifiers may indeed be thought of as always transmitting, with actual protocol initiation exercized by tags.)

In our security model, we let $\mathcal{V}$ and $\mathcal{T}_x$ denote oracles for V and T_x respectively. While tag oracles engage in sessions sequentially, $\mathcal{V}$ may maintain an arbitrary number of concurrent sessions. Let j be a unique session identifier; in practice, j might correspond to a session identifier determined by the verifier; it need not be explicitly employed in protocol flows, and is primarily useful here for modeling. An oracle query of the form $\mathcal{V}(j, flow, \cdot)$ denotes a query to be input to $\mathcal{V}$ (in general the output of some tag oracle $\mathcal{T}_x$); here, $flow$ indicates the protocol-flow number in AP to which the message corresponds, while $\cdot$ represents a protocol message. Similarly, an oracle query $\mathcal{T}_x(j, flow, \cdot)$ represents input to $\mathcal{T}_x$ (output by $\mathcal{V}$) , with values analogously defined. Any query representing an out-of-order flow is ignored by the receiving oracle.

On conclusion of a session, the oracle $\mathcal{V}$ outputs either $(ID, \text{“accept”}, j)$ to indicate successful authentication of a tag carrying identifier ID in session j, or else outputs $(\text{“reject”}, j)$ to indicate a failed authentication attempt in session j. Similarly, the tag may be viewed for purposes of security modeling as outputting either $(\text{“reject”}, j)$ or $(\text{“accept”}, j)$. (In practice, a tag does not output messages of this kind.)

We assume a global initialization phase in which a trusted party – e.g., $\mathcal{V}$ – may set all oracle states privately, i.e., without adversarial eavesdropping. In practice, this represents the programming of devices by manufacturers or system administrators.

A session between $\mathcal{V}$ and a given tag $\mathcal{T}_x$ may be (prematurely) *terminated* in our model. In this case, a special query $\perp(j)$ may be input to either $\mathcal{V}$ and $\mathcal{T}_x$; $\mathcal{V}$ and $\mathcal{T}_x$ are assumed on such input to abort the session immediately. (A terminated session models a break in contact, i.e., a message-delivery failure between a tag and verifier in the real world.) A *refresh* in our model is formally defined as a complete, integrity and privacy-protected invocation of the authentication protocol AP between $\mathcal{V}$ and some tag oracle $\mathcal{T}_x$.

We denote the adversary in our model by $\mathcal{A}$. The adversary may submit oracle queries in an arbitary order and may initiate concurrent protocol sessions, but we assume for simplicity that $\mathcal{A}$ may not make multiple simultaneous oracle queries. In addition to oracle queries, the adversary may cause any tag to initiate a new session, via an *initalization* query; in this case the tag terminates any previously active session and outputs a first flow for the new one.

Two security parameters, q and r, govern adversarial and oracle interaction as follows. After $\mathcal{A}$ has submitted q initialization queries to an oracle $\mathcal{T}_x$, for a security parameter q, the oracle $\mathcal{T}_x$ terminates any existing session and engages in a refresh with $\mathcal{V}$ (which we assume is completed instantaneously). Thus, the parameter q effectively determines the rate at which an adversary may be presumed to query tags in immediate succession. (Note that if there are more than q pseudonyms contained in tag memory, then the tag is presumed not to have to reuse pseudonyms in our model.)

The security parameter r represents a cap on the interleaving of tag and verifier queries by the adversary, i.e., the ability of the adversary to mount man-in-the-middle attacks. In particular, for each $\mathcal{T}_x$, we may view a counter as being maintained that records the number of queries have been sent to $\mathcal{T}_x$ and $\mathcal{V}$ in alternation. This counter is never incremented more than r times prior to a refresh. In particular, if this counter has been incremented by r subsequent to the last system initialization or the last refresh with $\mathcal{V}$, then any active session for $\mathcal{T}_x$ is terminated and a refresh performed between $\mathcal{T}_x$ and $\mathcal{V}$. Note that although refreshes are assumed to be integrity and privacy protected, they may nonetheless result in an authentication failure if the adversary has tampered with previous communications between the tag and verifier, and thereby corrupted the keys they use during the refresh process.

It is useful to observe that if r is greater than the number of flows in the protocol AP, then the adversary is capable of simulating the ability to eavesdrop and actively tamper with at least one full session in AP entirely as desired. Thus, our security definitions are capable of capturing a broad range of natural and powerful adversarial behaviors. We let $\mathcal{T}_x^{[q,r]}$ denote a tag oracle with the described query parameters. We let s in our experiments denote the total number of adversarial queries to all oracles. Excluding refreshes, in all cases in these experiments $\mathcal{A}$ is responsible for delivering all messages. In other words, the adversary actually *is the communications medium.* This is similarly the case in standard cryptographic models for key exchange, e.g., [3, 5, 38].

What is rather different about our security model is the nature of the security parameters q and r. In standard cryptographic security definitions, the probability of adversarial success is typically expressed as a function of the security parameters; in our model here, the security parameters determine a system reaction to adversarial queries.

A.1 Further details

We now provide a more succinct and precise specification of our security model. We let the symbol $\cdot$ here denote a protocol-specific value that lies outside the model. The session identifiers used throughout are for notational convenience, and need not figure directly in an actual protocol. A global value counter J, initially set to 0, records the total number of initiated sessions.

Recall that q and r are oracle parameters denoting the number of successive tag queries and tag / verifier interleavings permitted to an adversary prior to invocation of a refresh. Recall that s is the total number of adversarial queries to all oracles. The value l is a security parameter governing key lengths in the system. The variable n denotes the number of tag oracles in the system. The values k and m parameters specific to our proposed protocol: k is the number of pseudonyms in a tag, while m is the number of one-time pads used for key updates (corresponding to the level of backward secrecy).

A tag oracle $\mathcal{T}_x$ in our model maintains state (j_x, f_x, χ_x), denoting a session identifier, flow identifier, and input-query count respectively for the tag. The notation $f_x = \vdash$ here indicates that no session is currently active.

$\mathcal{T}_x$ takes as input three possible queries:

- init: This query initializes a new session and terminates any previously active one. If $f_x \neq \vdash$ (there is a session in progress), then $\mathcal{T}_x$ outputs ("reject", j_x). Otherwise, $J \leftarrow J + 1$, and $\mathcal{T}_x$ sets $j_x \leftarrow J$, $f_x \leftarrow 1$, and $\chi_x \leftarrow \chi_x + 1$. $\mathcal{T}_x$ then outputs $\mathcal{V}(j_x, 1, \cdot)$.
- $\mathcal{T}_{x'}(j', f', \cdot)$: This query represents a protocol input. If $j' \neq j_x$ or $x' \neq x$ or $f' \neq f_x + 1$, then the query is ignored. Otherwise, $\mathcal{T}_x$ sets $f_x \leftarrow f_x + 2$ and yields one of three possible outputs: $\mathcal{V}(j_x, f_x, \cdot)$ or ("reject", j_x) or ("accept", j_x). In the latter two cases $\mathcal{T}_x$ sets $f_x \leftarrow \vdash$.
- $\perp(j)$: This symbol denotes (premature) termination of session j. If $j = j_x$, then $\mathcal{T}_x$ sets $f_x \leftarrow \vdash$.

$\mathcal{V}$ maintains a counter g_j of the current protocol flow for session j (in addition to any other session state); we assume $g_j = 0$ for all j at initialization. The oracle $\mathcal{V}$ takes as input three basic forms of query:

- $\mathcal{V}(j', 1, \cdot)$: This query represents a session initialization by a tag. If $g_{j'} \neq 0$, then the query is ignored. Otherwise, $\mathcal{V}$ sets $g_{j'} \leftarrow 1$.

– $\mathcal{V}(j', g', \cdot)$ for $g' > 1$: If $j' > J$ or $g' \neq g_{j'} + 1$ then the query is ignored. Otherwise $\mathcal{V}$ sets $g_j \leftarrow g_j + 2$ and yields one of three possible outputs: $\mathcal{T}_x(j, g_{j'}, \cdot)$ for some x or (“reject”, j') or (ID, “accept”, j') for some identifier ID. In the latter two cases $\mathcal{V}$ sets $g_{j'} \leftarrow \vdash$.
– $\perp(j)$: This symbol denotes (premature) termination of session j. If $j \leq J$, then $\mathcal{V}$ sets $g_j \leftarrow \vdash$.

For each tag $\mathcal{T}_x$, a global counter ρ_x is maintained. The counter records the number of interleaved queries sent to $\mathcal{T}_x$ and $\mathcal{V}$. In particular, if $\mathcal{T}_x$ receives a query and ρ_x was last incremented in response to a query to $\mathcal{V}$, then ρ_x is incremented. Likewise, if $\mathcal{V}$ receives a query and ρ_x was last incremented in response to a query to $\mathcal{T}_x$, then ρ_x is incremented. If $\rho_x = r$, or $\chi_x = q$ then a refresh is invoked between $\mathcal{T}_x$ and $\mathcal{V}$. This may be viewed as a global command of the following form:

– $\mathsf{refresh}(x)$: A full, privacy-protected protocol invocation is performed. The query init is sent to $\mathcal{T}_x$. All queries generated by $\mathcal{T}_x$ and all queries generated by $\mathcal{V}$ of the form $\mathcal{T}_x(j, \cdot, \cdot)$ are then faithfully delivered via a private channel until counters $f_x = \vdash$ and $g_j = \vdash$, i.e., until both $\mathcal{T}_x$ and $\mathcal{V}$ have reached an “accept” or “reject” state. The counters ρ_x and χ_x are set to 0. The adversary remains inactive during a refresh.

As explained above, the adversary $\mathcal{A}$ in our model may submit queries to any oracles in an arbitrary order and may invoke $\mathsf{refresh}$ at any time. We assume a temporal ordering on queries, i.e., $\mathcal{A}$ may not make simultaneous queries. All oracle outputs are passed to the adversary, rather than to another oracle. In consequence, $\mathcal{A}$ can modify, withhold, reorder, or replay messages at will. In other words, refreshes aside, the adversary has complete control of the communications medium among oracles.

B Security Definitions

Authentication security: Our definition of authentication security for protocol AP characterizes the ability of $\mathcal{A}$ to clone valid-looking tags in an RFID system. We define security here in terms of an experiment $\mathbf{Exp}^{auth}$ in which $\mathcal{A}$ interacts with the verifier and with tags for an arbitrary period of time determined by $\mathcal{A}$, but with a total number of oracle queries bounded by a parameter s. In this initial “test” phase, the adversary may interact with all oracles, with refreshes taking place according to the security parameters q and r. On the conclusion of the “test” phase, all sessions are terminated, and $\mathcal{A}$ interacts with $\mathcal{V}$ in a new session. In this subsequent “cloning” phase, during which the adversary has no oracle access to tags, the goal of $\mathcal{A}$ is to cause $\mathcal{V}$ to accept. Acceptance in the “cloning” phase denotes a successfully mounted adversarial attack against the authentication protocol. In particular, a successful adversary is capable of creating a freestanding tag that can cause the verifier to accept at least once. Let l be a security parameter governing protocol key lengths. Formally, then:

```
Experiment $\mathbf{Exp}^{auth}_{\mathcal{A}}(AP)$; $[q, r, s, l, n]$
    initialize $\{\mathcal{T}_x\}_{i=x}^{n}$ and $\mathcal{V}$;
    while $state$ = “test”
        $state \leftarrow \mathcal{A}^{\{\text{access to all oracles } \{\mathcal{T}_x[q,r]\}_{x=1}^{n} \text{ and } \mathcal{V}\}}(\text{“test”})$;
    $\mathcal{A}$("clone") queries $\mathcal{V}$ until $\mathcal{V}$ yields some output $\gamma$;
    if $\gamma = (ID$, "accept", $j)$ for any $ID$ then
        output '1';
    else
        output '0';
```

Our concrete definition of authentication security is given by the expression:
$\mathbf{Adv}_{\mathcal{A}}^{auth}(AP); [q, r, s, l, n] = \mathsf{pr}[\mathbf{Exp}_{\mathcal{A}}^{auth}(AP); [q, r, s, l, n] = \text{'1'}]$.

Remark: According to our definition, a system may in fact provide strong authentication security and yet still permit the adversary to spoof the verifier into accepting adversarial authentication requests during the "test" phase of the experiment $\mathbf{Exp}^{auth}$. This models the ability of an adversary to mount a strong attack prior to attempting to clone a tag.

Privacy: The privacy of AP may be defined by an experiment $\mathbf{Exp}^{priv}$ involving the same pattern of oracle accesses on the part of $\mathcal{A}$ as in our experiment $\mathbf{Exp}^{auth}$. There is in fact a strong interrelationship between authentication and privacy in AP. This is reflected in the following observation. If verifier interaction with tags is unauthenticated, and AP supports verifier updates of pseudonyms, then $\mathcal{A}$ can pose as the verifier and update pseudonyms as desired. In this case, $\mathcal{A}$ can violate any natural privacy definition, as it knows the pseudonyms of tags. Attacks of this kind are captured by our characterization of privacy in the following definition.

Briefly stated, the experiment is as follows: The adversary interacts as desired with all system elements during the initial "test" phase. All active sessions are then terminated. During the subsequent portion of the experiment, the adversary attempts to break the privacy of the underlying protocol. The adversary chooses a pair of target tag oracles, which it specifies according to their identifiers. The selected tag oracles are presented to the adversary in a random order. The task of the adversary is to determine this order with probability significantly better than a random guess.

```
Experiment $\mathbf{Exp}_{\mathcal{A}}^{priv}(AP); [q, r, s, l, n]$
    initialize $\{\mathcal{T}_x\}_{x=1}^{n}$ and $\mathcal{V}$;
    while $state$ = "test"
        $state \leftarrow \mathcal{A}^{\{\text{access to all oracles } \{\mathcal{T}_x[q,r]\}_{x=1}^{n} \text{ and } \mathcal{V}\}}$("test");
    $(\tau_0, \tau_1) \leftarrow \mathcal{A}$("select target tags");
    $b \in_U \{0, 1\}$;
    $b' \leftarrow \mathcal{A}^{\{\text{access to oracles } \mathcal{T}_{\tau_b}^{[q,r]}, \mathcal{T}_{\tau_{1-b}}^{[q,r]}\}}$("guess tag");
    if $b = b'$ and $\tau_0, \tau_1 \in \{1, 2, \ldots, n\}$ then
        output '1';
    else
        output '0';
```

Our concrete characterization of the privacy level of AP is given by the expression:
$\mathbf{Adv}_{\mathcal{A}}^{priv}(AP); [q, r, s, l, n] = |\, \mathsf{pr}[\mathbf{Exp}_{\mathcal{A}}^{priv}(AP); [q, r, s, l, n] = \text{'1'}] - 1/2 \,|$. (We subtract 1/2 here as the adversary can trivially guess bit b successfully with probability 1/2.)

C Security Analysis

We are now ready to give concrete bounds on $\mathbf{Adv}^{auth}_{\mathcal{A}}(AP)$ and $\mathbf{Adv}^{priv}_{\mathcal{A}}(AP)$. As we see, the security of our system depends most critically on the key-length parameter l. This parameter determines the probability with which an adversary may guess unknown keys in the system. The number of tags in the system n and the number of valid tag identifiers k play lesser but similar roles in system security, as they determine the density of the overall space of tag identifiers. The bounds we present here are good, but not tight. To simplify our proofs, we assume that successful guessing of *any* key by the adversary results in defeat of the security properties of our protocol. This is not of course always the case; for example, the adversary may successfully guess an identifier α_i but still be unable to falsify a successful authentication for lack of knowledge of the corresponding key γ_i. Thus, it is possible that the bounds we prove may be tightened somewhat. Our aim is to prove the following two theorems.

Theorem 1. *Suppose that $m > \lfloor r/2 \rfloor$. Then for any parameter set q, r, s, l, n, we have $\mathbf{Adv}^{auth}_{\mathcal{A}}(AP); [q, r, s, l, n$ $\frac{skn}{2^l}$.*

Theorem 2. *Suppose that $m > \lfloor r/2 \rfloor$ and $k > q$. For any parameter set q, r, s, l, n such that $n \geq 2$, we have $\mathbf{Adv}^{priv}_{\mathcal{A}}(AP); [q, r, s, l, n] \leq \frac{skn}{2^l}$.*

For the proofs of these two theorems, we begin by defining a special adversary $\mathcal{A}^*$ with restricted capabilities. This adversary must deliver all tag-to-verifier and verifier-to-tag messages faithfully, that is, to the correct recipient and without any modification. $\mathcal{A}^*$ must deliver messages corresponding to a specific tag in their correct order. $\mathcal{A}^*$ may, however, substitute the termination message $\perp(j)$ for a given message of the form $\mathcal{T}_x(j, \cdot, \cdot)$ or $\mathcal{V}(j, \cdot, \cdot)$ in session j, i.e., $\mathcal{A}^*$ may indicate a refusal to deliver the correct message. Like $\mathcal{A}$, the adversary $\mathcal{A}^*$ may cause a tag to initiate a session, i.e., yield a first-flow output at any time that the tag is not already engaged in an active session. Thus, $\mathcal{A}^*$ may be regarded essentially as an honest-but-curious adversary with the added ability to cause explicitly designated delivery failures. Our proof strategy is to show that a real-world adversary $\mathcal{A}$ can effectively do little more than the special adversary $\mathcal{A}^*$.

Recall that we assume a temporal ordering on oracle queries (and thus on oracle outputs). Thus the full behavior of $\mathcal{A}^*$ in our model may be specified by a transcript σ^* consisting of a sequence of triples $\{M_1^*, M_2^*, \ldots, M_u^*\}$ of the following form: $M_v^* = (type \in \{query, output\}, ID \in \{\mathcal{T}_x\}_{x=1}^n \bigcup \{\mathcal{V}\}, msg)$. Here *type* specifies whether the message *msg* is an oracle query or oracle output, while ID specifies the oracle from which the message originates or to which it is transmitted.[5]

A transcript σ for $\mathcal{A}$ assumes essentially the same form, except that a "query" transcript entry assumes the form $M_v = (query, ID, msg, status)$, where *status* is a bit indicating whether or not the query is accepted by the oracle ID. We regard a query as accepted if it does not result in the oracle outputting a "reject".

We say that a transcript σ^* for $\mathcal{A}^*$ is a *simulation* of transcript σ for $\mathcal{A}$ if the following process renders the two transcripts identical:

[5] Note that ID is the identity of the oracle to which the message was *actually* delivered. While *msg* carries an oracle identifier, $\mathcal{A}$ may choose to misdeliver *msg*.

1. Every (rejected) transcript entry $M_v = (query, ID, msg, 0)$ in σ is replaced with $M_v^* = (query, ID, \perp(j))$, where j is the session identifier in msg. This is because a rejected query may be simulated by $\mathcal{A}^*$ in the form of a termination message, i.e., a termination message induces the same state in oracles as the rejected query.
2. Every transcript entry $M_v = (query, ID, msg, 1)$ in σ is replaced with $M_v^* = (query, ID, msg)$. The '1' – indicating query acceptance – is simply removed to harmonize notation.
3. Every transcript entry $M_v = (query, ID, msg, 1)$ or $M_v^* = (query, ID, msg)$ where $msg = (\cdot, 4, \cdot)$ is replaced with the symbol '*'. Since fourth-flow messages are key updates, and never rejected in our proposed protocol, they may always be successfully tampered with by $\mathcal{A}$ in a real protocol execution. Thus we do not treat them as simulable by $\mathcal{A}^*$, and remove them from transcripts.

Our first lemma aims to show that with high probability in either experiment $\mathbf{Exp}^{auth}$ or $\mathbf{Exp}^{priv}$, an arbitrary adversary $\mathcal{A}$ can only produce a transcript that is simulable by $\mathcal{A}^*$. Briefly stated, we show that for an adversary $\mathcal{A}$ to deliver an invalid message in any of the first three flows such that the message is accepted, $\mathcal{A}$ must effectively guess a protocol key. Thanks to the use of one-time pads, these keys are randomly distributed in the view of the adversary. Hence the probability that the adversary can guess a key successfully is small. Since a query by $\mathcal{A}$ consisting of an incorrectly guessed key can be simulated by a termination message $\perp(j)$ in a transcript generated by $\mathcal{A}^*$, we conclude that $\mathcal{A}^*$ can simulate a transcript for $\mathcal{A}$.

Lemma 1. *Suppose that $m > \lfloor r/2 \rfloor$. For any such parameter set q, r, s, l, n, the following holds for our proposed protocol AP. Let σ be a transcript generated by $\mathcal{A}$ in experiment $\mathbf{Exp}^E_{\mathcal{A}}(AP); [q, r, s, l, n]$ for $E \in \{auth, priv\}$. The probability over coin-flips of $\mathcal{V}$ that σ is not simulable by $\mathcal{A}^*$ in experiment $\mathbf{Exp}^E_{\mathcal{A}^*}(AP); [q, r, s, l, n]$ is bounded above by $\frac{skn}{2^l}$.*

Proof: Let $\sigma_v = \{M_1, M_2, \ldots, M_v\}$ define a partial transcript produced by $\mathcal{A}$. We prove the lemma by induction over v, with the inductive assumption that σ_v is simulable. (This is clearly true for $v = 0$.) We consider the probability that $\mathcal{A}$ is capable of generating a transcript entry M_{v+1} such that $\sigma_{v+1} = \{M_1, M_2, \ldots, M_{v+1}\}$ is non-simulable.

An entry of the form $M_{v+1} = (\cdot, \cdot, \cdot, 0)$ is always simulable by some termination message $\perp(j)$; similarly an entry of the form $M_{v+1} = (output, \cdot, \cdot, \cdot)$ is always simulable under the inductive assumption: As M_v is simulable, it induces a valid oracle output that corresponds to M_{v+1}. Thus we may assume that $M_{v+1} = (query, ID, msg, 1)$ for some session identifier j, oracle ID, and message msg.

Case 1 : Suppose that $msg = \mathcal{V}(j, 1, \alpha)$, i.e., represents the first-flow output of a tag in response to the init command. Since we assume that the associated query is accepted, α must be a currently valid identifier for some tag $\mathcal{T}_x$. If α was emitted as output by $\mathcal{T}_x$, then it must already have been delivered to $\mathcal{V}$ as a query; otherwise M_{v+1} would not be simulable. On the other hand, α could not have been delivered since the last refresh by $\mathcal{V}$ of $\{\alpha_i\}$ values for tag $\mathcal{T}_x$; if it had been, then α would have been marked as invalid.

Let us consider a given valid identifier α' for $\mathcal{T}_x$. W.l.o.g., let us assume that the initialization procedure for tags in our experiments is immediately followed by m refreshes for each tag (as this does change the initial distribution over keys), i.e., we represent α as the composition of a collection of one-time pads. Let us also assume that a newly initialized tag does not discard the live pad on its first update.[6] Then it must be the case that $\alpha' = \Delta'_1 \oplus \Delta'_2 \oplus \ldots \oplus \Delta'_m$ for a sequence of one-time pads successively selected uniformly and independently at random by $\mathcal{V}$ for $\mathcal{T}_x$.

We will now prove by contradiction that at least one of these one-time pads, Δ'_z, must have been transmitted to $\mathcal{T}_x$ during the course of a refresh. Suppose not. Then $\mathcal{V}$ must have computed all of these one-time pads in response to some accepted third-flow query $\mathcal{V}(j, 3, \gamma)$ for $\mathcal{T}_x$ in our protocol AP. All of these queries must, under our inductive assumption, have been faithfully transmitted outputs of $\mathcal{T}_x$. Each such query is therefore accompanied by at least two interleaved protocol flows between $\mathcal{V}$ and $\mathcal{T}_x$, i.e., is part of an invocation of AP that results in ρ_x being incremented by 2. Thus $\rho_x \geq 2m$. Since $m > \lfloor r/2 \rfloor$, it follows that $\rho_x > r$. This, however, is a contradiction, as $\rho_x = r$ results in the invocation of a refresh. Hence, for at least one Δ'_z of the one-time pads composing α, it must be that Δ'_z was transmitted to $\mathcal{T}_x$ during a refresh.

Since Δ'_z is selected by $\mathcal{V}$ uniformly at random and assumed to be transmitted to $\mathcal{T}_x$ over a private, integrity-protected channel, it follows that α' is distributed uniformly at random in the view of $\mathcal{A}$. Thus the probability that $\mathcal{A}$ can compute M_{v+1} for $msg = \mathcal{V}(j, 1, \alpha)$ is at most the probability that $\mathcal{A}$ can guess one of the currently valid identifiers for the full family of tags $\{\mathcal{T}_x\}_{x=1}^n$. Since each tag has at most k valid identifiers at a given time, the success probability of $\mathcal{A}$ for Case 1 may be straightforwardly bounded above by $nk/2^l$.

Case 2: Suppose that $msg = \mathcal{T}_x(j, 2, \beta)$ or $msg = \mathcal{V}(j, 3, \gamma)$ i.e., represents a second-flow or third-flow message in AP. Our analysis in this case is very much like that for Case 1. The difference lies in the fact that in a given session, a tag will accept only a single second-flow value β; similarly $\mathcal{V}$ will accept only a single third-flow value γ. (This is due to the fact that after the first flow, both oracles participating in a given session work on the basis of a fixed tag identity.) Thus, based on an argument analogous to that in Case 1, the probability that $\mathcal{A}$ can compute an entry M_{v+1} yielding a non-simulable transcript is at most $1/2^l$.

Note that termination of sessions at the end of the "test" phase in our experiments is equivalent to the transmission of a termination symbol $\perp(j)$ to all oracles for all active sessions, and does not change the above analysis.

As the number of oracle queries made by $\mathcal{A}$ is assumed to be bounded above by s, the lemma follows straightforwardly $\square$

We now present theorems characterizing the authentication and privacy characteristics of our proposed protocol AP. In fact, the theorems go farther than this, and provide concrete bounds on $\mathbf{Adv}^{auth}_{\mathcal{A}}(AP); [q, r, s, l, n]$ and $\mathbf{Adv}^{priv}_{\mathcal{A}}(AP); [q, r, s, l, n]$. These derive directly from and are identical to that in Lemma 1.

Theorem 1. *Suppose that $m > \lfloor r/2 \rfloor$. For any parameter set q, r, s, l, n, we have $\mathbf{Adv}^{auth}_{\mathcal{A}}(AP); [q, r, s, l, n] \leq \frac{skn}{2^l}$.*

[6] This requirement is essentially to correct an awkward boundary condition. We do not want to discard a secret value that has not yet been used. An alternative is to assume that a tag does not discard the old live pad during an update, but instead rotates it into the last position of the vector.

Proof: Under the bound of Lemma 1, let us assume that the behavior of $\mathcal{A}$ yields a transcript simulable by $\mathcal{A}^*$. At the conclusion of the "test" phase of the experiment $\mathbf{Exp}^{auth}$, all active sessions are terminated.

If a session for a given tag $\mathcal{T}_x$ was terminated immediately subsequent to the first flow α, then $\mathcal{V}$ will not have received α. In this case, $\mathcal{A}$ may deliver message α to $\mathcal{V}$ and have it accepted. $\mathcal{A}$ may not, however, deliver the resulting challenge β to $\mathcal{T}_x$, because tags are not active during the "cloning" part of the experiment. Hence, since the behavior of $\mathcal{A}$ is simulable, $\mathcal{A}$ cannot obtain the correct corresponding third flow γ, and may not cause $\mathcal{V}$ to accept the session.

If some given session between a tag $\mathcal{T}_x$ and verifer was terminated after the first flow, or if no session was in progress, then $\mathcal{A}$ is incapable of delivering a first flow α that will be accepted for $\mathcal{T}_x$. Hence $\mathcal{A}$ cannot cause $\mathcal{V}$ to output an "accept" message. $\square$

Theorem 2. *Suppose that $m > \lfloor r/2 \rfloor$ and $k > q$. For any such parameter set q, r, s, l, n such that $n \geq 2$, we have $\mathbf{Adv}_{\mathcal{A}}^{priv}(AP); [q, r, s, l, n] \leq \frac{skn}{2^l}$.*

Proof: By the same reasoning as in the proof of Lemma 1, we see that any key α_i, β_i, or γ_i not yet emitted by a tag subsequent to a refresh is distributed uniformly at random in the view of the adversary. It therefore suffices to show that a tag never emits the same key twice prior to update of the key during a refresh.

Let us consider a given key triple $(\alpha_i, \beta_i, \gamma_i)$ for some tag $\mathcal{T}_x$. As all sessions are terminated on conclusion of the "test" phase, $\mathcal{A}$ must initiate new sessions in the "guess tag" phase. Therefore, $\mathcal{T}_x$ will only emit β_i or γ_i in a session where it emits α_i as the identifier. Hence, it suffices to show that no identifier is emitted twice prior to update during a refresh.

Now, $\mathcal{T}_x$ emits α_d for the counter $d = (c \bmod k) + 1$. Since c can only be incremented in response to an init query, it follows that $\mathcal{T}_x$ can only emit α_d twice after receiving k init queries. Given the condition $k > q$, however, this cannot happen without a refresh occurring – at which point α_d is updated. $\square$

Remark: In the experiment $\mathbf{Exp}_{\mathcal{A}}^{priv}(AP)$, the adversary is not permitted to query $\mathcal{V}$ during the second, "guess tag" phase. If the adversary were indeed able to do so, then the adversary could break the privacy of AP as follows. During the "test" phase, for a target tag $\mathcal{T}_x$, the adversary replaces the one-time pads in the fourth flow of the protocol with random pads. This renders $\mathcal{T}_x$ incapable of successful protocol completion: With overwhelming probability, the tag will abort upon receipt of a β value. The adversary is then able to distinguish $\mathcal{T}_x$ in the "guess tag" phase of the experiment, as the tag will abort the protocol. On the other hand, if $\mathcal{A}$ is *passive*, then $\mathbf{Exp}_{\mathcal{A}}^{priv}(AP)$ may be modified to permit queries to $\mathcal{V}$ during the "guess tag" phase with no loss in privacy. As part of this modification, naturally the value x in messages of the form $\mathcal{T}_x(j, f, \cdot)$ must be suppressed.

D RFID Tags with Pseudorandom Number Generation

In this appendix, we briefly consider how our proposed scheme might be adapted – and strengthened – for RFID tags capable of performing symmetric-key cryptography. Even if a tag cannot perform a full-blown primitive of this kind, symmetric-key crypto might still be deployable in some form. Time-sharing might provide one avenue for more rapid adoption of symmetric-key crypto. For example, an RFID tag might execute one or several rounds of a cipher on every query, storing partial results and releasing output only when all rounds of the cipher have been fully computed. The resource requirements of cipher implementation would then be significantly reduced. Alternatively, given the small number of outputs yielded by an RFID tag over a normal lifetime, a weak PRNG might be acceptable for some of the uses we propose here.

We focus in particular here on the possibility of on-chip pseudorandom number generation, the most likely first deployment of strong cryptography in RFID tags [45]. Let us suppose that $f_{\kappa_x}(i)$ represents the (suitably long) output of a pseudorandom number generator for index i, where κ_x is a secret, random seed unique to $\mathcal{T}_x$.

In this case, an RFID tag could of course generate its own pseudonyms $\alpha_1 = f(1)$, $\alpha_2 = f(2), \ldots, \alpha_k = f(k)$, rather than storing them. The tag could generate β and γ values analogously. In principle, then, there need be no bound on k. If the verifier and tag are perfectly synchronized, then they may maintain a common value counter d_x unique to $\mathcal{T}_x$, and may share the seed κ_x. To determine which tag an incoming value α corresponds to, the verifier could simply perform a lookup in a list of current α values for all tags, namely $\{f_{\kappa_x}(d_x)\}$.

Desynchronization, however, would very likely be concern in real applications (and might provide a mechanism for denial-of-service attacks). One possible approach to preventing desynchronization would be for the verifier to maintain a list not just of current α values, but also of values from several future timesteps.

This proposal is similar in flavor to that of Weis *et al.* [45] involving tag release of $(r, PRNG(ID, r))$. An advantage of our proposal as just described, however, is that it would permit a certain amount of synchronization, but would still not leak any counter values. (Counter values themselves may serve as identifiers permitting privacy infringement.)

A more comprehensive approach would be to establish a bound of moderate size on k, e.g., $k = 100$. Given storage of the full set of k pseudonyms for each tag the verifier could feasibly search through the full list of possible pseudonyms.[7] Upon successful verifier-to-tag authentication, refresh would be quite straightforward. Suppose that for counter value d, an RFID tag computes $\alpha_d = f(bk + d)$, where b is a "base" value. The last flow from the verifier might simply be an "ack" indicating that b should be incremented.

It is worth noting that registration of a tag $\mathcal{T}_x$ by a verifier or transfer of ownership to a new verifier would require transfer of κ_x and preferably also of counter values. Alternatively, as mentioned in the body of the paper and in [21, 45] these values might be transferred with the aid of an optical or contact channel. For example, κ_x might be printed in scannable optical form on the RFID tag. On receiving κ_x – or after some other appropriate form of authentication – a tag might release its current counter value. A new owner might presumably also provide a new seed value κ_x to a tag through a process involving authentication via the old one.

Finally, we remark that f here may be a stream cipher for the schemes we describe here, and need not be directly indexable. A tag would only have to store state permitting computation of $f(bk + 0)$ in our scheme involving rotation through pseudonyms.

[7] For this and the previous approach, a simple efficiency enhancement is possible: The verifier can store a small set of pseudonyms corresponding to the counter value d_x (and a few successors) in a primary list, and other pseudonyms in a secondary list. This permits a form of "optimistic" search, in which it may be expected that search in the primary list will generally yield a successful result.