Untraceability in the applied pi-calculus*

Myrto Arapinis Tom Chothia Eike Ritter Mark Ryan

School of Computer Science
University of Birmingham, UK
{m.d.arapinis, t.chothia, e.ritter, m.d.ryan}@cs.bham.ac.uk

Abstract

The use of RFID tags in personal items, such as passports, may make it possible to track a person's movements. Even RFID protocols that encrypt their identity may leak enough information to let an attacker trace a tag. In this paper we define strong and weak forms of untraceablility, and illustrate these definitions with a simple example. We formally define these concepts in the applied pi-calculus which in some cases makes it possible to automatically check if an RFID tag running a particular protocol is untraceable.

1 Introduction

Radio Frequency IDentification (RFID) systems consist of tags, readers and a database, as depicted in Figure 1. As the tag is small enough to be implanted into almost any item, and the data on the tag can be read wirelessly, RFID technology has proven useful in many situations including stock control, payment and identification systems. However, as RFID tags cannot be switched off and will answer any request without asking for the agreement of their bearer they have raised new security concerns. For instance, can a person's movements be traced using the RFID tags implanted in the items they are carrying? As early RFID tags responded to any signal broadcast to them and replied with a unique identifier (as depicted in Figure 2), Benetton's proposal to place RFID tags in clothes caused a public outcry for precisely this reason [5]. Similar traceability concern have also affected the New York area E-Zpass system [9].

There has been a lot of work on checking the security and authenticity properties of RFID protocols

*This work has been partially supported by the EPSRC project *Verifying Interoperability Requirements in Pervasive Systems* (EP/F033540/1)

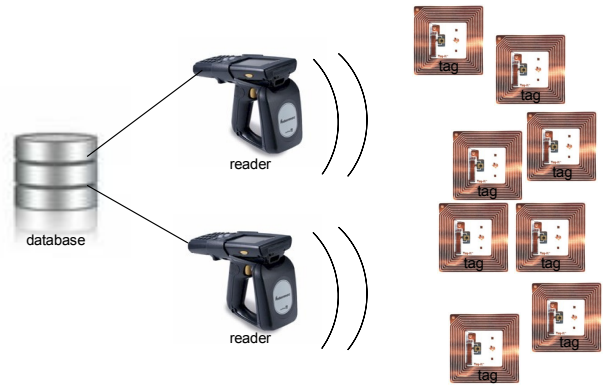


Figure 1. RFID **architecture**

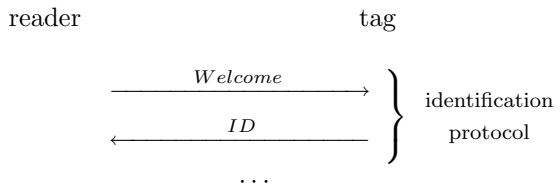


Figure 2. Typical tag-reader interaction

but there has been relatively little work on checking if an attacker can trace a particular tag. The protocol sketched in Figure 2 is clearly insecure and traceable. Suppose now that the tag encrypts its ID with a key shared between the tag and the reader; the ID would be be kept secure. However, if the encryption is deterministic the message is the same each time and the attacker can trace the tag by simply looking for this bit string.

Traceability is a threat that is particularly relevant to RFID protocols because RFID tags are the only technology that people regularly carry on their person and cannot turn off. Such security threats have led

to the development of RFID tags that encrypt their communication and authenticate the readers, such as the protocols used in the *e-passport* [11] and on transit system payment cards. However, even these protocols often turn out to be broken (such as the Mifare classic tag used for the London Underground [13]) or allow the user to be tracked (such as the Nike+iPod sports kit [17]). This shows the need for methods of checking the correctness of RFID protocols before they are implemented.

The aim of our work is to aid in the checking of RFID protocols for traceability attacks. We identify two levels of untraceability. *Strong untraceability* holds if an attacker cannot tell the difference between an RFID system in which all tags are different and a system in which some tags appear twice. *Weak untraceability* holds if an attacker cannot identify two particular runs of a protocol as having involved the same tag. Intuitively, an attacker learns nothing from a strongly untraceable system, whereas the attacker in a weakly untraceable system may learn some information about the tags but still cannot trace one particular tag.

We formalise our notions of strong and weak untraceability by defining them using observational equivalence in the applied pi-calculus [2]. This makes the definitions absolutely precise and in some cases allows a user to automatically check that a protocol is untraceable using the tool ProVerif [6].

In the next section we briefly describe the applied pi-calculus. We describe how RFID systems can be modelled in Section 3. Our main definitions of strong and weak untraceability are defined in Section 4. Finally we conclude and discuss further work in Section 5.

Related work While a number of papers discuss the privacy problems raised by RFID techonologies (see for example [14, 15, 20]), very few precisely define what they mean by untraceability. Avoine *et al.* in [4] were the firsts to give a formal definition of untraceability. Some other attempts to formalize untraceability then followed [3, 8, 16, 19]. All this work however is carried out in the computational model, which is poorly supported by automatic tools. The advantage of our work is that it is carried out in the symbolic setting which is supported, as already mentioned, by the ProVerif tool. These two settings being very different, it is difficult at this stage to compare our work with the ones mentioned before.

In the symbolic world, Deursen et al. [18] are, to the best of our knowledge, the only other people to define untraceability. They propose a formal definition of untraceability in a particular trace model. Their definition is similar to our definition of weak untraceability. However, the model they use to defined this property does not lend itself to automation, which is the main advantage of working in the symbolic setting instead of the computational.

2. The applied pi-calculus

The applied pi-calculus [2] is a language for describing concurrent processes and their interactions. It is based on the pi-calculus, but adds equations which make it possible to model a range of cryptographic primitives. Properties of processes described in the applied pi-calculus can be proved by employing manual techniques [2], or by automated tools such as ProVerif [6]. As well as reachability properties which are typical of model checking tools, ProVerif can in some cases prove that processes are observationally equivalent [7]. This capability is important for privacy-type properties such as those we study here. The applied pi-calculus has been used to study a variety of security protocols, such as those for private authentication [12] and for key establishment [1].

To describe processes in the applied pi-calculus, one starts with a set of *names* (which are used to name communication channels or other constants), a set of *variables*, and a *signature* Σ which consists of the function symbols which will be used to define terms. In the case of security protocols, typical function symbols will include pbk for constructing the public key $\mathsf{pbk}(k)$ associated to the secret key k, and aenc for asymmetric encryption, which takes a plaintext and a public key and returns the corresponding ciphertext, and adec for asymmetric decryption, taking a ciphertext and the corresponding private key and returning the plaintext. One can also describe the equations which hold on terms constructed from the signature. For example in the signature

$$\Sigma = \{\mathsf{pair},\ \pi_1,\ \pi_2,\ \mathsf{pbk},\ \mathsf{aenc},\ \mathsf{adec}\}$$

pbk, aenc, and adec are as described above, and pair is the function symbol that denotes the concatenation operation. π_1 and π_2 are respectively the projections on the first and second component of a pair. It is usual to consider for Σ the following equational theory

$$\begin{aligned} \pi_1(\mathsf{pair}(x,y)) &= x \\ \pi_2(\mathsf{pair}(x,y)) &= y \\ \mathsf{adec}(\mathsf{aenc}(x,\mathsf{pbk}(k)),k) &= x \end{aligned}$$

Terms are defined as names, variables, and function symbols applied to other terms. Plain processes are built up in a similar way to processes in the pi calculus,

except that messages can contain terms (rather than just names). In the grammar described below, M and N are terms, n is a name, x a variable and u is a metavariable, standing either for a name or a variable.

$P, Q, R, ::=$	
0	null process
$P \mid Q$	parallel composition
$!P$	replication
$\nu n.P$	name restriction
$\mathsf{if}\ M = N\ \mathsf{then}\ P\ \mathsf{else}\ Q$	conditional
$\mathsf{in}(u, x).P$	message input
$\mathsf{out}(u, N).P$	message output

Example Consider the following process $System$:

$$\begin{array}{lcl} System & = & Tag \mid Reader \\ Reader & = & \mathsf{out}(c, Welcome).\ \mathsf{in}(c, x) \\ Tag & = & \nu id.\ (\mathsf{in}(c, y).\mathsf{if}\ (y = Welcome) \\ & & \qquad \mathsf{then}\ \mathsf{out}(c, id)) \end{array}$$

The first component sends a $Welcome$ message to the second, and waits for the second to identify itself by sending its id. Accordingly the second component waits for a $Welcome$ message and identifies itself.

The operational semantics of processes in the applied pi-calculus is defined by structural rules defining two relations: *structural equivalence*, written $\equiv$ and *internal reduction*, written $\rightarrow$. Structural equivalence is the smallest equivalence relation on extended processes that is closed under α-conversion on names and variables, by application of evaluation contexts −an evaluation context is an extended process with a hole instead of some extended process− and satisfying some further basic structural rules such as $A \mid 0 \equiv A$, associativity and commutativity of $\mid$, *etc.* Internal reduction $\rightarrow$ is the smallest relation on extended processes closed under structural equivalence and application of evaluation contexts such that $\mathsf{out}(a, x).P \mid \mathsf{in}(a, x).Q \rightarrow P \mid Q$, and for any ground terms M and N, whenever $M \neq_E N$ we have

$$\begin{array}{rcl} \mathsf{if}\ M = M\ \mathsf{then}\ P\ \mathsf{else}\ Q & \rightarrow & P \\ \mathsf{if}\ M = N\ \mathsf{then}\ P\ \mathsf{else}\ Q & \rightarrow & Q \end{array}$$

Applied pi-calculus processes evolve by executing the actions mentioned above. We write $A \rightarrow A'$ to mean that process A evolves to A' by one step, and $A \rightarrow^* A'$ for finitely many steps.

Example Consider process P described in the previous example. We have

$$System \rightarrow \mathsf{in}(c, x) \mid \nu id.\ (\mathsf{if}\ (Welcome = Welcome)\ \mathsf{then}\ \mathsf{out}(c, id)).$$

This internal reduction expresses a communication on the channel c between the two components of the process P. In the reminder of the process y is replaced by $Welcome$.

Many properties of security protocols (including the properties we study in this paper) are formalised in terms of *observational equivalence* ($\approx$) between two processes. Intuitively, processes which are *observationally equivalent* cannot be distinguished by an outside observer, no matter what sort of test he makes. This is formalised by saying that the processes are indistinguishable under any context, *i.e.*, no matter in what environment they are executed.

Example Consider the following three processes P_1, P_2, and P_3:

$$\begin{array}{lcl} P_1 & = & \nu k.\ \mathsf{out}(c, \mathsf{aenc}(a, \mathsf{pbk}(k))) \\ P_2 & = & \nu k.\ \mathsf{out}(c, \mathsf{aenc}(\mathsf{pair}(a, b), \mathsf{pbk}(k))) \\ P_3 & = & \nu k.\ \mathsf{out}(c, \mathsf{pair}(\mathsf{aenc}(a, \mathsf{pbk}(k)), b)) \end{array}$$

Since an outside observer doesn't know the decryption key k, he cannot ditsinguish if it is the constant a or the pair $\mathsf{pair}(a, b)$ which is encrypted in P_1 and P_2 respectively. We thus have that $P_1 \approx P_2$. On the other hand, $P_1 \not\approx P_3$, since the test $\bullet \mid \mathsf{in}(c, x).\ \mathsf{if}\ \pi_2(x) = b\ \mathsf{then}\ \mathsf{out}(c, \text{"}success\text{"})$ distinguishes the two processes, *i.e.*, the observer can tell the processes apart by testing if the second component of the output message is b.

Advantages and limitations of the applied pi-calculus An advantage of the applied pi-calculus is that we can combine powerful (hand) proof techniques from the applied pi-calculus with automated proofs provided by Blanchet's ProVerif tool. Moreover, the verification is not restricted to a bounded number of sessions and we do not need to explicitly define the adversary. We only give the equational theory describing the intruder. Generally, the intruder has access to any message sent on public, *i.e.*, unrestricted channels. These public channels model the network. Note that all channels are anonymous in the applied pi-calculus. Unless the identity or something like the IP address is specified explicitly in the conveyed message, the origin of a message is unknown. This abstraction of a real protocol is very appealing, as it avoids the need to model explicitly an anonymiser service. However, we stress that a real implementation needs to treat anonymous channels with care. Another advantage of the applied pi-calculus is its ability to model sophisticated cryptographic primitives by means of the equational theory. One limitation concerns modelling non-determinism or

probabilities, *e.g.* MIX-nets [10]. In the applied pi calculus, all non-determinism is controlled by the attacker. If MIX-nets are modelled non-deterministically, this gives the attacker unreasonably strong powers.

3. Modelling protocols

In this work, we do not consider all the processes of the applied pi-calculus. We thus first need to define which processes correspond to the RFID protocols[1] we have considered.

Definition 1 An RFID protocol P is a closed plain process such that

$$P \equiv \nu\tilde{n}.\ (DB \mid !R \mid !T)$$

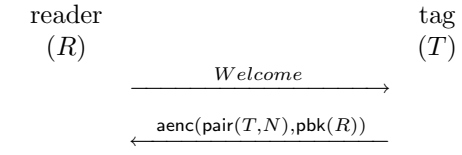
where

$$T \equiv \nu\tilde{m}.\ init.\ !main$$

for some processes *init* and *main*. Moreover, we consider P such that all channels occurring in it are ground, and private channels are never sent on any channel.

As we mentioned in the introduction, a system P is formed of readers (R), tag (T), and a database (DB). Intuitively, T is the process modelling one tag, and having T under a replication in P corresponds to considering a system with an unbounded number of tag. Each tag, initialises itself (this includes registering to the database DB and is modelled by *init* in T) and then may execute itself an unbounded number of times. Thus *main* models one session of the tag's protocol. R corresponds to one session of the reader's protocol, and DB to the database. We consider an unbounded number of readers, thus R is under a replication in P.

Example Let's consider the following RFID identification protocol



The reader continuously emits a welcome message and expects from a tag to answer with its identity T paired with a fresh nonce N, the whole asymmetrically encrypted with the reader's public key $\mathsf{pbk}(R)$. Moreover, the reader may (but not necessarily) send some message (lets say 0) to the database if it has seen two particular tags more then three times each. Although the reader's behaviour is somehow odd and the protocol doesn't satisfy important properties like authentication (it is subject to replay attacks), it is still (as we will see later) an interesting candidate for us. It will allow us to separate weak from strong untraceability. We thus start by modelling it in the applied pi-calculus. For this, we consider the signature Σ and the equational theory given in the first example of Section 2

$$\begin{array}{lcl}
DB & = & 0 \\
R & = & \mathsf{out}(c, Welcome).\ \mathsf{in}(c, x_1).\ \ldots.\ \mathsf{in}(c, x_6). \\
 & & \mathsf{if}\ (\pi_1(\mathsf{adec}(x_h, k)) = \pi_1(\mathsf{adec}(x_i, k)) = \pi_1(\mathsf{adec}(x_j, k)) \\
 & & \mathsf{and}\ \pi_1(\mathsf{adec}(x_\ell, k)) = \pi_1(\mathsf{adec}(x_m, k)) = \pi_1(\mathsf{adec}(x_n, k)) \\
 & & \mathsf{and}\ \{h, i, j, \ell, m, n\} = \{1, \ldots, 6\}) \\
 & & \mathsf{then}\ \mathsf{out}(db, 0) \\
T & = & \nu id.\ !(\nu n.\ \mathsf{in}(c, x). \\
 & & \mathsf{if}\ (x = Welcome) \\
 & & \mathsf{then}\ \mathsf{out}(c, \mathsf{aenc}(\mathsf{pair}(id, n), \mathsf{pbk}(k)))) \\
P & = & \nu k.\ (DB \mid !R \mid !T)
\end{array}$$

Here, $init = 0$ and $main = \nu n.\ \mathsf{in}(c, x).\ \mathsf{if}\ (x = Welcome)\ \mathsf{then}\ \mathsf{out}(c, \mathsf{aenc}(\mathsf{pair}(id, n), \mathsf{pbk}(k)))$. In order to reduce the notational clutter we have introduced the following notation:

$$\mathsf{if}\ C_1\ \mathsf{and}\ C_2\ \mathsf{then}\ P \stackrel{def}{=} \mathsf{if}\ C_1\ \mathsf{then}\ \mathsf{if}\ C_2\ \mathsf{then}\ P.$$

Moreover, the condition $\{h, i, j, \ell, m, n\} = \{1, \ldots, 6\}$ expresses that we consider all the permutations for the values of h, i, j, ℓ, m, and n.

4 Formalising untraceability

4.1 Strong untraceability

Untraceability is sometimes defined as ensuring that an intruder tampering with the system thinks that each tag session is initiated by a different tag.

Definition 2 Let P be an RFID protocol. Let

$$P' \equiv \nu\tilde{n}.\ (DB \mid !R \mid !T')$$

where

$$T' \equiv \nu\tilde{m}.\ init.\ main$$

[1] RFID protocols only differ from usual protocols like the Needham-Schrœder protocol in the number of roles. We presently restrict ourselves to RFID protocols because traceability is particularly relevant to them. Although our definitions could be extended to k-party protocols for any k.

P preserves strong untraceability for tags if

$$P \approx P'$$

The intuitive idea behind this definition is as follows: each session of P should look to the intruder as if it was initiated by a different tag. In other words, an *ideal* version of the protocol, *w.r.t.* untraceability, would allow tags to execute themselves at most once. The intruder should then not be able to tell the difference between the protocol P and the ideal version of P'.

4.2. Weak untraceability

Let P be an RFID protocol. Informally, weak untraceability ensures that a tag can execute its protocol multiple times without an intruder being able to link these executions together (ISO 15408 definition). To formally define this in the applied pi calculus, we first need to define tagging the outputs of a process with a term.

Definition 3 Let P be a process such that all channels that appear in P are ground and let C_{pub} be the set of public channels appearing in P. Tagging the public outputs of P with N results in the process $\overline{P}^N = tag(P, N, C_{pub})$ with

$$\begin{array}{lcl}
tag(0, N, C) & = & 0 \\
tag(P_1 \mid P_2, N, C) & = & \\
 & & tag(P_1, N, C) \mid tag(P_2, N, C) \\
tag(!P_1, N, C) & = & !tag(P_1, N, C) \\
tag(\nu n.\ P_1, N, C) & = & \nu n.\ tag(P_1, N, C) \\
tag(\mathsf{in}(c, x).\ P_1, N, C) & = & \mathsf{in}(c, x).\ tag(P_1, N, C) \\
tag(\mathsf{out}(c, M).\ P_1, N, C) & = & \\
\quad \left\{ \begin{array}{ll} \mathsf{out}(c, \mathsf{pair}(M, N)).\ tag(P_1, N, C) & if\ c \in C \\ \mathsf{out}(c, M).\ tag(P_1, N, C) & otherwise \end{array} \right. & & \\
tag(\mathsf{if}\ M_1 = M_2\ \mathsf{then}\ P_1\ \mathsf{else}\ P_2, N, C) & = & \\
\quad \mathsf{if}\ M_1 = M_2\ \mathsf{then}\ tag(P_1, N, C)\ \mathsf{else}\ tag(P_2, N, C) & &
\end{array}$$

Example If we consider the process $System$ given as an example in Section 2. Tagging its public outputs with the constant n results in the following process

$$\begin{array}{lcl}
\overline{System}^n & = & \overline{Tag}^n \mid \overline{Reader}^n \\
\overline{Tag}^n & = & \mathsf{out}(c, \mathsf{pair}(Welcome, n)).\ \mathsf{in}(c, x) \\
\overline{Reader}^n & = & \nu id.\ (\mathsf{in}(c, y).\mathsf{if}\ (y = Welcome) \\
 & & \qquad \mathsf{then}\ \mathsf{out}(c, \mathsf{pair}(id, n)))
\end{array}$$

Definition 4 Let P be an RFID protocol. From P we build the two following processes

$$Q \equiv \nu\tilde{n}.\ (DB \mid !R \mid !T \mid T_1 \mid T_2)$$

where T_1 and T_2 are the processes modelling two tag such that

$$\begin{array}{lcl}
T_1 & \equiv & \nu\tilde{m}.\ init.\ (!main \mid \overline{main}^{s_1} \mid \overline{main}^{s_2}) \\
T_2 & \equiv & \nu\tilde{m}.\ init.\ (!main \mid \overline{main}^{s_3})
\end{array}$$

and

$$Q' \equiv \nu\tilde{n}.\ (DB \mid !R \mid !T \mid T_1' \mid T_2')$$

where T_1' and T_2' are the processes modelling two tags such that

$$\begin{array}{lcl}
T_1' & \equiv & \nu\tilde{m}.\ init.\ (!main \mid \overline{main}^{s_1} \mid \overline{main}^{s_3}) \\
T_2' & \equiv & \nu\tilde{m}.\ init.\ (!main \mid \overline{main}^{s_2})
\end{array}$$

We consider s_1, s_2, and s_3 to be three distinct session identifiers that do not occur in P (*i.e.* three distinct public constants not appearing in P). P preserves weak untraceability for tags if

$$Q \approx Q'$$

The intuitive idea behind this definition is as follows: if a protocol satisfies untraceability, then an intruder shouldn't be able to tell the difference when two sessions s_1 and s_2 are executed by the same tag in the process P_1 sessions s_1 and s_2 are initiated by the tag T_1), or by two different tags (in the process P_2 sessions s_1 and s_2 are imitated by the tags T_1 and T_2 respectively).

4.3. Strong untraceability $\subsetneq$ Weak untraceability?

Let's consider again our toy protocol. This protocol satisfies weak but not strong untraceability. Intuitively, this is due to the fact that when an outside observer sees a message output on channel *db*, he knows that there are two tags that have executed themselves three times each and thus that it is not the case that each tag executes itself at most once. He can hence distinguish the actual protocol from its ideal version, violating the definition of strong untraceability. On the other hand this information doesn't allow him to really link two sessions of the same tag.

More formally, if we consider the evaluation context $\mathcal{C}[\bullet] = \bullet$, then in P the reader may emit a sound (*i.e.* a message on channel *beep*) according to the protocol. However, in the ideal version P' this will never happen since tags execute their protocol only once. Nevertheless, we formally proved using ProVerif that this protocol satisfies weak untraceability.

5. Conclusion

The main contribution of the paper is two definitions of untraceability in the applied pi-calculus. These

definitions can be used to automatically verify (with ProVerif) RFID protocols. We also proved that these two definitions are not equivalent by building an RFID protocol that is weakly untraceable but not strongly untraceable. Moreover, we proved that this protocol is weakly untraceable using the ProVerif tool, demonstrating that this definition can be suitable for automatic verification.

The next step for us, is to test our definitions on existing protocols, academic ones as well as real ones. We are planing amongst others to verify *w.r.t.* untraceability the protocols used in the *e-passport*. As much as possible, we will do so using the ProVerif tool, in order to conclude to the suitability of our definitions to automation.

In this paper, we conjecture that strong untraceability implies weak untraceability. We would now want to prove this formally. We will then look at untraceability of corrupted tags (namely backward and foraward untraceability). Moreover, untraceability is closely related to anonymity. In particular, one would expect that untraceability implies anonymity. This is something we would like to look at in the future.

References

[1] M. Abadi, B. Blanchet, and C. Fournet. Just fast keying in the pi calculus. *ACM Trans. Inf. Syst. Secur.*, 10(3):9, 2007.

[2] M. Abadi and C. Fournet. Mobile values, new names, and secure communication. *SIGPLAN Not.*, 36(3):104–115, 2001.

[3] G. Avoine. *Cryptography in Radio Frequency Identification and Fair Exchange Protocols*. PhD thesis, EPFL, Lausanne, Switzerland, December 2005.

[4] G. Avoine, E. Dysli, and P. Oechslin. Reducing time complexity in RFID systems. In B. Preneel and S. Tavares, editors, *Selected Areas in Cryptography – SAC 2005*, volume 3897 of *Lecture Notes in Computer Science*, pages 291–306, Kingston, Canada, August 2005.

[5] Boycott benetton. http://www.boycottbenetton.com/.

[6] B. Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *CSFW '01: Proceedings of the 14th IEEE workshop on Computer Security Foundations*, page 82, Washington, DC, USA, 2001. IEEE Computer Society.

[7] B. Blanchet. Automatic Proof of Strong Secrecy for Security Protocols. In *IEEE Symposium on Security and Privacy*, pages 86–100, Oakland, California, May 2004.

[8] M. Burmester, T. v. Le, and B. d. Medeiros. Provably Secure Ubiquitous Systems: Universally Composable RFID Authentication Protocols. In *Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm*, Baltimore, Maryland, USA, August-September 2006. IEEE.

[9] C. Caldwell. A pass on privacy? The New York Times, July 17, 2005.

[10] D. Chaum, P. Y. A. Ryan, and S. A. Schneider. A practical voter-verifiable election scheme. In *ESORICS*, pages 118–139, 2005.

[11] P. T. Force. Pki for machine readable travel documents offering icc read-only access. Technical report, International Civil Aviation Organization, 2004.

[12] C. Fournet and M. Abadi. Hiding names: Private authentication in the applied pi calculus. In *ISSS'02: Proceedings of the International Symposium on Software Security, volume 2906 of LNCS, pages 317–338, 2003.*, 2003.

[13] F. D. Garcia, G. Koning Gans, R. Muijrers, P. Rossum, R. Verdult, R. W. Schreur, and B. Jacobs. Dismantling mifare classic. In *ESORICS '08: Proceedings of the 13th European Symposium on Research in Computer Security*, pages 97–114, 2008.

[14] S. L. Garfinkel, A. Juels, and R. Pappu. RFID privacy: An overview of problems and proposed solutions. *IEEE Security and Privacy*, 3(3):34–43, 2005.

[15] A. Juels. RFID security and privacy: a research survey. *IEEE Journal on Selected Areas in Communications*, 24(2):381–394, 2006.

[16] A. Juels and S. A. Weis. Defining strong privacy for RFID. In *PERCOMW '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 342–347, Washington, DC, USA, 2007. IEEE Computer Society.

[17] S. Saponas, J. Lester, C. Hartung, and T. Kohno. Devices that tell on you: The Nike+iPod sport kit. Technical report, University of Washington, 2006.

[18] T. van Deursen, S. Mauw, and S. Radomirovi. Untraceability of RFID protocols. In *Information Security Theory and Practices. Smart Devices, Convergence and Next Generation Networks*, volume 5019. 5019 of *Lecture Notes in Computer Science*, page 115. Springer, 2008.

[19] S. Vaudenay. On privacy models for RFID. In *Advances in Cryptology ASIACRYPT 2007*, pages 68–87, 2007.

[20] S. A. Weis, S. E. Sarma, R. L. Rivest, and D. W. Engels. Security and privacy aspects of low-cost radio. In *Hutter, D., Müller, G., Stephan, W., Ullman, M., eds.: International Conference on Security in Pervasive Computing - SPC 2003, volume 2802 of LNCS, Boppard, Germany*, pages 454–469. Springer-Verlag, march 2003.