Noname manuscript No.
(will be inserted by the editor)

Privacy Analysis of Forward and Backward Untraceable RFID Authentication Schemes

Raphael C.-W. Phan[†] · **Jiang Wu**[⋆] · **Khaled Ouafi** · **Douglas R. Stinson**[*]

Abstract In this paper, we analyze the first known provably secure RFID authentication schemes that are designed to provide forward untraceability and backward untraceability: the L-K and S-M schemes. We show how to trace tags in the L-K scheme without needing to corrupt tags. We also show that if a *standard* cryptographic pseudorandom bit generator (PRBG) is used in the S-M scheme, then the scheme may fail to provide forward untraceability and backward untraceability. To achieve the desired untraceability features, we show that the S-M scheme can use a *robust* PRBG which provides forward security and backward security. We also note that the backward security is stronger than necessary for the backward untraceability of the S-M scheme.

1 Introduction

Radio Frequency Identification (RFID) is an automated object identification technology. An RFID system consists of RFID tags, RFID readers and a back-end server. A tag is a tiny microchip containing identification information. A reader queries a tag, receives responses from the tag via radio signal, and queries the back-end server which maintains a database of tags. The server retrieves and returns to the reader the detailed information of the tag. RFID tags are being deployed in many consumer, financial and governmental applications, for instance respectively in supply chain [7, 23, 33], in contactless credit cards [13], and in e-passports [16, 14, 18].

RFID technology raises significant privacy issues. For example, since a tag automatically responds to queries via radio signal, sensitive data may be leaked to unauthorized readers. Even when the data is encrypted, the location of the tag may still be traced. There has been considerable research on these privacy issues and numerous privacy-preserving RFID authentication schemes have been proposed. For a survey, see [15].

In view of the pervasiveness and inconspicuous nature of these tiny RFIDs, privacy for RFID tag users is a major concern that could potentially impede the public's long-term adoption of RFID-enabled applications. To the best of our knowledge, formal treatments of strong privacy for RFID protocols include the work of Avoine [2], Juels and Weis [17], Damgård and Østergaard [11], Le, Burmester and de Medeiros [19]; Vaudenay [34, 35, 27] and Lim and Kwon [20]. The difference in these models lie basically in the power of the adversary's tag-corruption ability with respect to the strong privacy notion; here strong refers to the adversarial ability to corrupt tags thereby revealing the internal state. Such adversarial capability is standard consideration in security protocols like those for authentication and key establishment [5]. In the context of the RFID schemes, corruption also

[†]Part of work done while the author was with LASEC, EPFL.

[⋆]Research supported by an NSERC post-graduate scholarship. Work done while the author was with David R. Cheriton School of Computer Science, University of Waterloo.

[*]Research supported by NSERC discovery grant 203114-06.

Electronic & Electrical Engineering, Loughborough University, United Kingom
Email: r.phan@lboro.ac.uk
· The Institute of Electronics, Communications and Information Technology, Queen's University Belfast, United Kingdom
Email: j.wu@ecit.qub.ac.uk
· Security & Cryptography Lab (LASEC), EPFL, CH-1015, Switzerland
Email: khaled.ouafi@epfl.ch
· David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada
Email: dstinson@uwaterloo.ca

captures tag ownership transfer, e.g. when a merchandise item with an embedded RFID tag is purchased from a store, ownership of that tag needs to be transferred from store to the buyer in order to maintain the privacy of both parties: privacy of the store's dealings with that tag before the purchase should not be leaked to the buyer, and equally, privacy of the buyer's dealings with the tag after purchase should not be leaked to the store.

To capture these privacy requirements, two strong privacy notions, termed *backward untraceability* and *forward untraceability*, were proposed in [24] and [20] (here we use the terminology from [20]). Backward untraceability means that, if the adversary reveals the internal state of a tag at time τ, the adversary is not able to tell whether a transaction before time τ involves the tag. Forward untraceability is probably the strongest notion of privacy that can be achieved. It deals with adversaries who, even knowing the internal state of a tag at time τ, cannot assert whether the same tag was involved in a transaction that occured at time $\tau + \delta$ (for some $\delta > 0$). It is worth to note that in some privacy models, this notion has been proven to be impossible to achieve [35]. So, in order to make achieving forward untraceability possible, it was proposed in [20] to weaken the notion and add the condition that the adversary does not eavesdrop on the tag continuously after time τ. Summarizing here, backward untraceability assures that the privacy of the store's transactions with the tag before purchase, is maintained even against the new owner (in this case, the buyer); while forward untraceability assures that the privacy of the buyer's transactions with the tag after purchase is maintained even against the previous owner (in this case, the store).

1.1 Our Work

We analyze in this paper the privacy issues of the first known provably secure RFID authentication protocols that have been designed to provide the strong privacy notions of backward and forward untraceability, namely the scheme by Lim and Kwon [20] at ICICS '06, and the scheme by Song and Mitchell [31] at WISEC '08. The L-K scheme is a nice unconventional design in the sense that it achieves both forward and backward untraceability in the face of tag corruption, while typical protocols only provide backward untraceability. That paper also defined a provable security model for backward and forward untraceability. The S-M scheme is designed to provide the same features as the L-K scheme, but with improved performance in memory space, computation time and communication overhead.

We present the only known privacy analysis of the L-K scheme with respect to a general untraceable privacy model detailed in Section 4, yet without needing a corrupt query. As proposing a new privacy model is not the aim of this work, the model we describe can be seen as an adaptation of security models for authenticated key exchange protocols [5]. Concretely, we describe how to trace a tag within the L-K scheme even without requiring state corruption [17, 34, 20, 19, 35], and further discuss why the S-M scheme is resistant to this problem.

In both the L-K scheme and the S-M scheme, a tag needs to generate random numbers. When a pseudorandom bit generator (PRBG) is used in a security protocol to generate random numbers, it is important to know whether a *standard* cryptographic PRBG suffices for the protocol or a *robust* PRBG is necessary. In some cases, a protocol designer may overlook the fact that a standard PRBG may not provide forward and backward security, which is necessary for certain properties of the protocol that it underlies. In the same work [31] proposing the S-M scheme, the definition for a PRBG refers to the standard PRBG. In Section 6, we show that if a standard PRBG is used in the schemes, then the schemes may fail to provide forward untraceability and backward untraceability. We also construct an example of a robust PRBG for the S-M scheme to ensure its desired backward and forward untraceability features. This is the only known backward and forward traceability analysis of the S-M scheme.

Since RFID tags are ubiquitously deployed, privacy in the sense of untraceability is a serious issue. This contrasts with recent security attacks shown on RFID schemes in literature [6, 28, 36] that attempt to impersonate readers or tags which are essentially just denial of service attacks in the sense they permanently desynchronize a tag from a reader causing the tag to no longer be able to interact with the reader in future. Such latter attacks do not pose problems to privacy.

2 Pseudo-Random Bit Generators

Random bit generation is very important process in cryptography. Therefore, cryptographic devices and parties often implement *pseudorandom bit generators* (PRBG). Although not as perfect as true number generators, PRBG provide an amount of security that can be used to build many cryptographic primitives. A standard (cryptographic) PRBG is a deterministic algorithm which, when given a truly random binary input of length n, outputs a binary sequence of length polynomial in n, say $p(n)$, which "appears" to be random. The input to the PRBG is called the seed, the output of the PRBG is called a pseudorandom bit sequence. A standard PRBG is

secure, if when given the first $l < p(n)$ bits of the PRBG's output, it is infeasible in polynomial time (in n) to predict the next bit of the output ([21], [32]).

Formally, given a polynomial funtion p, a secure pseudo-random bit generator is a function $\mathsf{PRBG} : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ such that for any polynomialy bounded (in n) adverary $\mathcal{A}$ who outputs a bit after receiving the sequence seq_{l-1} of the $l-1$ first bits from the output of PRBG ($l <= p(n)$) and either the l-th bit x from the output of PRBG or a random $\bar{x} \in \{0,1\}^{p(n)}$, the advantage:

$$\left|\Pr[\mathcal{A}(\mathsf{seq}_{l-1}, x) \rightarrow 1] - \Pr[\mathcal{A}(\mathsf{seq}_{l-1}, \bar{x}) \rightarrow 1]\right|$$

is negligible in n[1].

Several instantiations of PRBG were proposed. One example is to use a block cipher with a secret key working in counter mode: given $E_k()$, a block cipher with secret key k, the output is computed by outputing the encryptions $s_i = E_k(i), i = 1, 2, \cdots$. Such a PRBG is a secure PRBG if the underlying block cipher is secure. Another example of standard PRBG is a keyed hash function with a secret key in output-feedback mode. If $h_k()$ denotes a keyed hash function with secret key k and an initial input s_0, the output is the concatenation of the successive digests $s_{i+1} = h_k(s_i), i = 0, 1, \cdots$.

PRBGs are essential to the privacy of an RFID system. Ma et al. [22] prove that an RFID system achieves privacy if and only if the tags can compute pseudorandom functions (PRFs). Because PRFs and PRBGs can be constructed from each other [12], [1], it holds that an RFID system can achieve privacy if and only if the tags have PRBGs.

The security model for standard PRBGs assumes that a PRBG is a black box. When PRBGs are used in applications where the adversary has access to the internal information of a PRBG, stronger security properties for PRBGs are required.

In order to deal with key exposure, a stronger security notion was proposed for PRBG by Barak and Halevi [3] called *robust* PRBG. Robust PRBG provide additional security beyond a standard PRBG. In [3], Barak and Halevi propose a formal model and an architecture for robust PRBG, which satisfy the following properties of *forward security* and *backward security*[2].

In these contexts, a standard PRBG is given a state. As its output is computed using this state and the n-bit seed, each output bit of such a PRBG is a function over $\{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\} \times \{0,1\}^n$ that computes the output bit and the new state of the PRBG. We denote st_i the state after outputing the i-th bit. Note that the state can be updated using additional data.

In the context of backward security, the past output of the PRBG looks random to an adversary, even if the adversary learns the internal state at a later time.Using the same notation as for standard PRBG, this formally translates into saying that for any polynomial time algorithm $\mathcal{A}$ the advantage

$$\left|\Pr[\mathcal{A}(\mathsf{seq}_{l-1}, x, \mathsf{st}_l) \rightarrow 1] - \Pr[\mathcal{A}(\mathsf{seq}_{l-1}, \bar{x}, \mathsf{st}_l) \rightarrow 1]\right|$$

is negligible.

Conversely, forward security deals with the unpredictability of the future output of the PRBG to an adversary with knowledge of the current state, provided that the PRBG is later refreshed with data of sufficient entropy. More formally, for any polynomial time algorithm $\mathcal{A}$, we require that:

$$\left|\Pr[\mathcal{A}(\mathsf{seq}_{l-1}, x, \mathsf{st}) \rightarrow 1] - \Pr[\mathcal{A}(\mathsf{seq}_{l-1}, \bar{x}, \mathsf{st}_{i,i<l}) \rightarrow 1]\right|$$

is negligible.

Similar properties and constructions can be found in [4] from NIST.

3 The Forward and Backward Untraceable RFID Schemes

3.1 The L-K Scheme

At ICICS '06, Lim and Kwon [20] proposed an RFID protocol that offers untraceable privacy (UPriv) both before and after tag corruption. This is a major feat, since other existing RFID schemes are only able to treat backward untraceability, i.e. a corrupted tag cannot be linked to any past completed sessions.

The initialization phase is as follows:

[1] A function $f(s)$ is said to be negligible if there exists a constant c such that $f(s)$ is $O(s^{-c})$.

[2] In [3], forward security means that past output is secure, while backward security means that future output is secure. Here we switch the two names to be consistent with the terminology we use in this paper and those in [20, 24].

1. The reader chooses a random secret K_i for each tag $\mathcal{T}_i$; it evaluates $m-1$ evolutions of $K_i^0 = K_i$, i.e. $K_i^j = g(K_i^{j-1})$ for $1 \leq j \leq m-1$; where g is a pseudorandom function. It computes $t_i^j = ext_{l_2}(K_i^j)$ for $0 \leq j \leq m-1$, where l_2 is some bit length, $ext_l(x)$ is an extraction function returning l bits of x.
2. The reader also chooses a random u_i for each tag $\mathcal{T}_i$ and computes a key chain $\{w_i^j\}_{j=0}^{n-1}$ of length n, such that $w_i^n = u_i$ and $w_i^j = h(w_i^{j+1})$ for $0 \leq j \leq n-1$, where h is a pseudorandom function.
3. The tag stores $\langle w_{i,T}, K_i \rangle$ where $w_{i,T} = w_i^0$ and initializes a failure counter $c_i = 0$.
4. The reader creates two tables L_1, L_2 for $\mathcal{T}_i$ in its database, where L_2 is empty and L_1 has entries of the form $\langle s_i, \{t_i^j\}_{j=0}^{m-1}, u_i, n_i, w_{i,T}, w_{i,S} \rangle$ where $n_i = n$ and $w_{i,S} = w_i^1$ thus $w_{i,T} = h(w_{i,S})$.

After initialization, a normal protocol session is illustrated in Fig. 1, where f is a pseudorandom function. Refer to [20] for further details.

Reader $\mathcal{R}$		**Tag $\mathcal{T}_i$**
Database: $\{\ldots, (K_i, \text{tables } L_1, L_2), \ldots\}$		**Secret:** $w_{i,T}, c_i, K_i$
pick r_1	$\xrightarrow{r_1}$	
		calculate $t_i = ext_{l_2}(K_i)$;
		pick r_2;
	$\xleftarrow{t_i, r_2, \sigma_1}$	compute $\sigma_1 = ext_{l_1}(f(K_i, r_1 \|\| r_2))$.
check $\exists t_i^j : (t_i^j = t_i) \wedge (t_i^j \in \{t_i^k\}_{k=0}^{m-1}) \wedge$ $\langle K_i, \{t_i^k\}_{k=0}^{m-1}, u_i, n_i, w_{i,T}, w_{i,S} \rangle \in (L_1 \cup L_2)$.		
calculate $K_i' = g(K_i)^j, \sigma_1' = ext_{l_2}(f(K_i', r_1 \|\| r_2))$ and check that $\sigma_1' = \sigma_1$.		
calculate $\sigma_2 = f(K_i', r_2 \|\| r_1) \oplus w_{i,S}$.	$\xrightarrow{\sigma_2}$	$w_{i,S} = f(K_i, r_2 \|\| r_1) \oplus \sigma_2$.
for $k = 0 \ldots m-j-1$ calculate: $\hat{t}_i^k = t_i^{j+k+1}$;		check $h(w_{i,S}) = w_{i,T}$.
for $k = m-j \ldots m-1$ calculate:		If yes:
$\hat{K}_i = g(K_i'), \hat{t}_i^k = ext_{l_2}(g(\hat{K}_i)^{k-m+j})$;		$c_i = 0; w_{i,T} = w_{i,S}$;
update $\hat{K}_i, \{\hat{t}_i^k\}_{k=0}^{m-1}$ in L_2;		$K_i = g(K_i \oplus (w_{i,T} \|\| r_1 \|\| r_2))$.
calculate $K_i = g(K_i \oplus (w_{i,S} \|\| r_1 \|\| r_2))$;		else
$t_i^j = ext_{l_2}(g(K_i)^j)$ for $j = 0 \ldots m-1$;		$c_i = c_i + 1$;
$n_i = n_i - 1, w_{i,T} = w_{i,S}, w_{i,S} = h(u_i)^{n_i}$;		if $c_i < m$
update $\langle K_i, \{t_i^k\}_{k=0}^{m-1}, n_i, w_{i,T}, w_{i,S} \rangle$ in L_1.		update $K_i = g(K_i)$.

Fig. 1 The L-K Scheme

3.2 The S-M Scheme

We briefly recall the S-M scheme proposed at WISEC '08 [31]. $f_k()$ is a keyed hash function with key k. $h()$ is a hash function. $x \gg y$ denotes the operation that right rotate-shifts x by y bits and $x \ll y$ denotes the operation that left rotate-shifts x by y bits. l is the length of the parameters in the scheme.

For each tag $\mathcal{T}_i$, the reader $\mathcal{R}$ stores its identifier (u_i, t_i). u_i is a unique secret for T_i and $t_i = h(u_i)$. The value t_i is stored in $\mathcal{T}_i$. An authentication session takes place as follows.

1. The $\mathcal{R}$ sends a random challenge r_1 to $\mathcal{T}_i$.
2. $\mathcal{T}_i$ generates a random r_2, computes $M_1 = t_i \oplus r_2, M_2 = f_{t_i}(r_1 \oplus r_2)$, and sends (M_1, M_2) to $\mathcal{R}$.
3. $\mathcal{R}$ searches for a tag T_i such that t_i satisfies $M_2 = f_{t_i}(r_1 \oplus r_2)$ where $r_2 = M_1 \oplus t_i$, computes $M_3 = u_i \oplus (r_2 \gg l/2)$. $\mathcal{R}$ also updates u_i and t_i as follows: $u_{i(new)} = (u_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2$, $t_{i(new)} = h(u_{i(new)})$.
4. $\mathcal{R}$ forwards M_3 to $\mathcal{T}_i$.
5. $\mathcal{T}_i$ computes $u_i = M_3 \oplus (r_2 \gg l/2)$. If $h(u_i) = t_i$, then $\mathcal{T}_i$ updates t_i as follows: $t_i = h((u_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2)$.

To perform the above protocol, a tag needs to generate random numbers. [31] cites a definition for pseudo-random bit generators (PRBGs) which is just a standard PRBG; see Section 6 for details of why this definition is an issue.

4 Modelling RFID Untraceable Privacy

We now describe our general untraceable privacy (UPriv) model that will be the setting in which we use in later sections to demonstrate how to trace tags and thus show that the schemes do not achieve the notion of untraceable

privacy. In fact, the model defined herein can be seen as an alternative definition of the Juels-Weis model [17] in a style more in line with the Bellare et al. [5] models for authenticated key exchange (AKE) protocols, for which RFID protocols can be seen to have close relationship with. With this model as a reference, our emphasis throughout this paper is on the analysis of the privacy issues of RFID protocols.

A protocol party is a $\mathcal{T} \in Tags$ or $\mathcal{R} \in Readers$ interacting in protocol sessions as per the protocol specifications until the end of the session upon which each party outputs Accept if it feels the protocol has been normally executed with the correct parties. Adversary $\mathcal{A}$ controls the communications between all protocol parties (tag and reader) by interacting with them as defined by the protocol, formally captured by $\mathcal{A}$'s ability to issue queries of the following form:

- $\mathsf{Execute}(\mathcal{R}, \mathcal{T}, i)$ query. This models *passive* attacks, where adversary $\mathcal{A}$ gets access to an honest execution of the protocol session i between $\mathcal{R}$ and $\mathcal{T}$ by eavesdropping.
- $\mathsf{Send}(U_1, U_2, i, m)$ query. This query models *active* attacks by allowing the adversary $\mathcal{A}$ to impersonate some reader $U_1 \in Readers$ (resp. tag $U_1 \in Tags$) in protocol session i and send a message m of its choice to an instance of some tag $U_2 \in Tags$ (resp. reader $U_2 \in Readers$). This query subsumes the TagInit and ReaderInit queries as well as challenge and response messages in the Juels-Weis model.
- $\mathsf{Corrupt}(\mathcal{T}, K)$ query. This query allows the adversary $\mathcal{A}$ to learn the stored secret K' of the tag $\mathcal{T} \in Tags$, and which further sets the stored secret to K. It captures the strong privacy notions of *backward untraceability (a.k.a. forward privacy)* or *forward untraceability* depending on the time at which this query is allowed. Both notions measure the extent of the damage caused by the compromise of the tag's stored secret. This query is the equivalent of the SetKey query of the Juels-Weis model.
- $\mathsf{Test}_{\mathsf{UPriv}}(U, i)$ query. This query is the only query that does not correspond to any of $\mathcal{A}$'s abilities or any real-world event. This query allows to define the indistinguishability-based notion of *untraceable privacy* (UPriv). If the party has accepted and is being asked a Test query, then depending on a randomly chosen bit $b \in \{0, 1\}$, $\mathcal{A}$ is given $\mathcal{T}_b$ from the set $\{\mathcal{T}_0, \mathcal{T}_1\}$. Informally, $\mathcal{A}$ succeeds if it can guess the bit b. In order for the notion to be meaningful, a Test session must be *fresh* in the sense of Definition 2.

Definition 1 (Session Completion) *We say that a reader instance $\mathcal{R}_j$ and a tag instance $\mathcal{T}_i$ are partners if, and only if, both have output* Accept*$(\mathcal{T}_i)$ and* Accept*$(\mathcal{R}_j)$ respectively, signifying the completion of the protocol session.*

Definition 2 (Freshness) *A party instance is fresh at the end of execution if, and only if,*

1. *it has output* Accept *with or without a partner instance,*
2. *both the instance and its partner instance (if such a partner exists) have not been sent a* Corrupt *query.*

Definition 3 (Untraceable Privacy (UPriv)) UPriv *is defined using the game $\mathcal{G}$ played between a malicious adversary $\mathcal{A}$ and a collection of reader and tag instances. $\mathcal{A}$ runs the game $\mathcal{G}$ whose setting is as follows.*

Phase 1 (Learning): $\mathcal{A}$ can send any Execute, Send, and Corrupt queries.
Phase 2 (Challenge):

1. At some point during $\mathcal{G}$, $\mathcal{A}$ will choose a fresh session on which to be tested and send a Test query corresponding to the test session. Note that the test session chosen must be fresh in the sense of Definition 2. Depending on a randomly chosen bit $b \in \{0, 1\}$, $\mathcal{A}$ is given a tag $\mathcal{T}_b$ from the set $\{\mathcal{T}_0, \mathcal{T}_1\}$.
2. $\mathcal{A}$ continues making any Execute, Send, and Corrupt queries at will, subjected to the restrictions that the definition of freshness described in Definition 2 is not violated.

Phase 3 (Guessing): Eventually, $\mathcal{A}$ terminates the game simulation and outputs a bit b', as its guess of the value of b.

The success of $\mathcal{A}$ in winning $\mathcal{G}$ and thus breaking the notion of UPriv is quantified in terms of $\mathcal{A}$'s advantage in distinguishing whether $\mathcal{A}$ receives $\mathcal{T}_0$ or $\mathcal{T}_1$, i.e. it correctly guessing b. This is denoted by $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{UPriv}}(k)$ where k is the security parameter.

In relation to other models, note that the Le-Burmester-de Medeiros model [19] similarly allows the corruption of tags. The Vaudenay model [34, 35] is stronger than both the Juels-Weis and Le-Burmester-de Medeiros models in terms of the adversary's corruption ability. In more detail, it is stronger than the Juels-Weis model in the sense that it allows corruption even of the two tags used in the **Challenge** phase. It is stronger than the Le-Burmester-de Medeiros model in the sense that it considers all its privacy notions even for corrupted tags, in contrast to the Le-Burmester-de Medeiros model that only considers corruption for its forward privacy notion.

5 Privacy Analysis of the L-K Scheme

The gist of the L-K scheme is this: the tag updates its stored secret K_i in two possible ways. If the reader is successfully authenticated, it would update as $K_i = g(K_i \oplus (w_{i,T}||r_1||r_2))$. Else, the tag would update as $K_i = g(K_i)$, up to m times of unsuccessful authentications, after which the tag stops updating its K_i. This eventual non-updating allows the reader to catch up.

Our attack works in spite of the explicit design gist above, by using the basic principle where we intentionally desynchronize the tag from the reader by sending the tag into the future [20].

1. **Learning:** An adversary sends m number of queries r_1^j for $1 \leq j \leq m$ to the tag $\mathcal{T}_0$, and records the tag's response t_j for $1 \leq j \leq m$. Since the adversary is impersonating the reader, thus each time it will not pass the check by the tag, and so each time the tag would update its stored secret as $K_i = g(K_i)$, from which t_i will be derived in the next session.
2. **Challenge:** Query r_1^m to the tag $\mathcal{T}_b \in \{\mathcal{T}_0, \mathcal{T}_1\}$, and obtain its response t^*.
3. **Guess:** Check if $t^* = t_m$. If so, then the adversary knows this was the tag it queried during the **Learning** phase i.e. $\mathcal{T}_b = \mathcal{T}_0$. Else, it knows that $\mathcal{T}_b = \mathcal{T}_1$.

[20] remarked that once a tag is successfully authenticated by a reader, then the tag's stored secret K_i would be freshly randomized so that tracing of any kind is prevented. Yet, even after that point, our adversary can repeat the above attack step of the **Learning** phase by sending m arbitrary queries r_1^j for $1 \leq j \leq m$ to the tag again to desynchronize it and the same tracing attack applies. This attack can be performed after every successful tag-to-reader authentication. To solve the DoS problem, the designers included into the design a feature that unfortunately allowed our attack causing the tag to be traceable even without corruption, although the goal for their protocol was much stronger i.e. backward and forward untraceability even with corruption.

Unlike the L-K scheme, we note that the S-M scheme does not exhibit this problem because the transmitted messages M_1, M_2, M_3 are randomized by the ephemeral random value r_2 which is itself not publicly transmitted. This versus the L-K scheme where the transmitted t_i is a deterministic function of the stored secret K_i, the latter of which is no longer updated after m unsuccessful authentication attempts.

6 Strong Privacy Analysis of the S-M Scheme

6.1 Forward Untraceability

We review forward untraceability as defined in [20] and [31]: at time τ during the **Learning** phase, the adversary reveals the internal state of the tag T_i via the Corrupt query. At time $\tau' > \tau$, the tag performs a transaction with the server and the adversary does not eavesdrop on this transaction. Forward untraceability means that the adversary cannot tell if a transaction at time $\tau'' > \tau'$ involves the tag T_i.

We show that, if the PRBG used by the tag is not a robust PRBG, then the scheme does not achieve forward untraceability. To be concrete, we assume the block cipher based PRBG described in Section **??**. Suppose at time τ, the adversary reveals the internal state of a tag, including its value t_i and the internal state of its PRBG. At some time $\tau' > \tau$ during the **Challenge** phase, the tag has a transaction without being eavesdropped by the adversary. After time τ' and still during the **Challenge** phase, the adversary observes multiple transaction messages via Execute queries.

Given the internal state (k and i) of the PRBG that the adversary obtained at time τ, the adversary can compute a sequence of n future outputs of the PRBG, where n is an upper bound on the maximum number of times that the tag could have invoked its PRBG since time τ. If any value r in this sequence and any observed message (r_1, M_1, M_2, M_3) after τ' satisfies $f_{M_1 \oplus r}(r_1 \oplus r) = M_2$, then the adversary can deduce that the message involves the tag T_i. In addition to identifying the tag, the adversary can further compute the t_i (note that this is not possible with permanent desynchronization attacks [6, 28, 36]) used in the observed transaction as well as the updated t_i after this transaction: $u_i = M_3 \oplus (r \gg l/2)$, current $t_i = h(u_i)$, and updated $t_i = h((u_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r)$. Therefore, the adversary can launch more attacks, e.g., to impersonate the tag or clone the tag.

6.2 Backward Untraceability

We review backward untraceability as defined in [20] and [31]: at time τ, the adversary reveals the internal state of the tag T_i. Backward untraceability means that the adversary cannot tell if a transaction at time $\tau' < \tau$ involves the tag T_i.

Given the internal state (k and i) of the PRBG that the adversary obtained at time τ, the adversary can compute a sequence of i previous outputs of the PRBG. The adversary has also observed multiple transactions before time τ. If any r value in the computed output sequence and any observed message $(r_1, (M_1, M_2), M_3)$ satisfies $f_{M_1 \oplus r}(r_1 \oplus r) = M_2$, then it can deduce that the message involves the tag T_i.

6.3 Using Robust PRBG in the S-M Scheme

It is clear that, for the S-M scheme to achieve forward untraceability and backward untraceability, the PRBG used in the tags needs to be a robust PRBG, i.e. forward secure and backward secure. We can follow the construction in [3] or [4] to build a robust PRBG for the tags. An example of a robust PRBG is based on a block cipher working in counter mode. Let k be the secret key. Each time the PRBG is invoked, $s = E_k(i)$ is outputted as the random bits, the key is updated as $k = E_k(i + 1)$, and the counter i increases. The PRBG also refreshes its key as $k = k \oplus r_1$ each time a random challenge r_1 is received from the reader.

Now we briefly analyze the case when k and i are revealed at time τ. First we consider backward untraceability. If E is a secure block cipher, then it is infeasible to find the previous keys it used; without the previous keys, it is infeasible to distinguish its previous output from a random number. Therefore backward untraceability of the schemes is preserved.

Next we consider forward untraceability. Suppose in a certain transaction after τ, the adversary does not observe the messages, including r_1. Then the adversary does not know the updated key and thus he cannot predict the future outputs of E. Therefore, forward untraceability of the schemes is preserved.

Remark. We note that the backward security of the robust PRBG is sufficient but not necessary for the backward untraceability of the S-M scheme. Suppose the standard PRBG based on keyed hash function given in Section 2 is used in the S-M scheme. Given s_i and k at time τ, the adversary can tell if a given value x was generated by the PRBG by computing $x_1 = h_k(x), x_2 = h_k(x_2), \cdots$ and checking if s_i appears in the sequence. Therefore, this PRBG does not provide backward security. But given s_i and k at time τ, the adversary cannot compute any previous output of the PRBG; thus the backward traceable attack described above for a block cipher based PRBG does not work for the keyed hash based PRBG. However, we note that that the L-K scheme in [20] does need a robust PRBG to ensure its backward untraceability because the output of the PRBG is sent in plaintext.

7 Conclusion

We analyzed the only known provably secure RFID authentication schemes that were designed to provide forward untraceability and backward untraceability. To this end, we described in an alternative manner the privacy models that capture untraceable privacy (UPriv) notion. Against this background, we showed how to trace tags in the L-K scheme, without requiring to corrupt tags.

Further, we showed that if a standard PRBG is used in the S-M scheme, then the scheme may fail to provide forward untraceability and backward untraceability. To achieve these untraceability features, the scheme can use a robust PRBG which provides forward security and backward security.

In some sense, these results support the case [8, 9, 10, 26] that while provable security is the right approach to design and analysis of protocols, more careful analysis and interpretation of provable security models and proofs are needed to ensure the right definitions [29] and underlying assumptions are put in place. As a commonly accepted model addressing privacy and security in RFID has to be established and many RFID protocols are proposed without providing any formal security proof, these results strengthen the need for such a model to facilitate better design of RFID protocols that offer both privacy and security.

Acknowledgement

The authors would like to thank Ian Goldberg for referring us to the paper [3]. We thank the anonymous reviewers for their suggestions that improve the paper, notably to highlight the application context of backward and forward untraceability, give detailed treatment of underlying PRBG functions and security properties, contrast between denial of service and privacy attacks, and highlight more clearly our contributions.

References

1. M. Abdalla and M. Bellare, "Increasing the Lifetime of a Key: A Comparative Analysis of the Security of Re-keying Techniques," Advances in Cryptology - Asiacrypt '00, LCNS 1976, pp. 546–559, 2000.
2. G. Avoine, "Adversarial Model for Radio Frequency Identification," Cryptology ePrint Archive, report 2005/049, 20 February, 2005. Available at IACR ePrint Archive, http://eprint.iacr.org/2005/049.
3. B. Barak and S. Halevi, "A Model and Architecture for Pseudo-Random Generation with Applications to /dev/random," *Proceedings of CCS '05*, ACM, pp. 203–212, 2005.
4. E. Barker and J. Kelsey, "Recommendation for Random Number Generation Using Deterministic Random Bit Generators (Revised)," NIST Special Publication 800-90, March, 2007.
5. M. Bellare, D. Pointcheval and P. Rogaway, "Authenticated Key Exchange Secure against Dictionary Attacks," *Advances in Cryptology - EUROCRYPT '00*, LNCS 1807, pp. 139–155, 2000.
6. S. Cai, Y. Li, T. Li and R.H. Deng, "Attacks and Improvements to an RFID Mutual Authentication Protocol and its Extensions," *Proceedings of WiSec '09*, ACM, pp. 51–58, 2009.
7. CASPIAN, "Boycott Benetton," accessed 19 September 2007. Available online at http://www.boycottbenetton.com.
8. K.-K.R. Choo, "Refuting Security Proofs for Tripartite Key Exchange with Model Checker in Planning Problem Setting," *Proceedings of IEEE CSFW '06*, pp. 297–308, 2006.
9. K.-K.R. Choo, C. Boyd and Y. Hitchcock, "Examining Indistinguishability-based Proof Models for Key Establishment Protocols," *Advances in Cryptology - Asiacrypt '05*, LNCS 3788, pp. 585–604, 2005.
10. K.-K.R. Choo, C. Boyd and Y. Hitchcock, "Errors in Computational Complexity Proofs for Protocols," *Advances in Cryptology - Asiacrypt '05*, LNCS 3788, pp. 624–643, 2005.
11. I. Damgård and S. Østergaard, "RFID Security: Tradeoffs between Security and Efficiency',' *Topics in Cryptology - CT-RSA 2008*, LNCS 4964, pp. 318-332, 2008.
12. O. Goldreich, S. Goldwasser and S. Micali, "How to Construct Random Functions," *Journal of the ACM*, Vol. 33, No. 4, pp. 792–807, 1986.
13. T.S. Heydt-Benjamin, D.V. Bailey, K. Fu, A. Juels and T. O'Hare, "Vulnerabilities in First-Generation RFID-enabled Credit Cards," *Proceedings of Financial Cryptography '07*, LNCS 4886, pp. 2-14, 2008.
14. J.-H. Hoepman, E. Hubbers, B. Jacobs, M. Oostdijk and R.W. Schreur, "Crossing Borders: Security and Privacy Issues of the European e-Passport," *Proceedings of IWSEC '06*, LNCS 4266, pp. 152–167, 2006.
15. A. Juels, "RFID Security and Privacy: a Research Survey," *IEEE Journal on Selected Areas in Communications*, Vol. 24, No. 2, pp. 381–394, 2006.
16. A. Juels, D. Molnar and D. Wagner, "Security and Privacy Issues in E-Passports," *Proceedings of SecureComm '05*, pp. 74–88, 2005.
17. A. Juels and S.A. Weis, "Defining Strong Privacy for RFID," *Proceedings of PerCom '07*, pp. 342–347, 2007. Full version available at IACR ePrint Archive, http://eprint.iacr.org/2006/137, 7 April 2006.
18. E. Kosta, M. Meints, M. Hensen and M. Gasson, "An Analysis of Security and Privacy Issues Relating to RFID Enabled ePassports," *Proceedings of IFIP SEC '07*, IFIP 232, pp. 467-472, 2007.
19. T.V. Le, M. Burmester and B. de Medeiros, "Universally Composable and Forward-Secure RFID Authentication and Authenticated Key Exchange," *Proceedings of ASIACCS '07*, pp. 242–252, 2007. Full version titled "Forward-Secure RFID Authentication and Key Exchange" available at IACR ePrint Archive, http://eprint.iacr.org/2007/051, 14 February 2007.
20. C.H. Lim and T. Kwon, "Strong and Robust RFID Authentication Enabling Perfect Ownership Transfer," *Proceedings of ICICS '06*, LNCS 4307, pp. 1–20, 2006.
21. A.J. Menezes, P.C. van Oorschot and S.A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, 1996.
22. C. Ma, Y. Li, R.H. Deng and T. Li, "RFID Privacy: Relation between Two Notions, Minimal Condition, and Efficient Construction," *Proceedings of CCS '09*, ACM, pp. 54–65, 2009.
23. "Michelin Embeds RFID Tags in Tires," RFID Journal, 17 January, 2003. Available online at http://www.rfidjournal.com/article/articleview/269/1/1/.
24. M. Ohkubo, K. Suzuki and S. Kinoshita, "Cryptographic Approach to "Privacy-friendly" Tags," *Proceedings of RFID Privacy Workshop*, MIT, 2003.
25. M. Ohkubo, K. Suzuki and S. Kinoshita, "RFID Privacy Issues and Technical Challenges," *Communications of the ACM*, Vol. 48, No. 9, pp. 66–71, 2005.
26. K. Ouafi and R.C.-W. Phan, "Traceable Privacy of Recent Provably-Secure RFID Protocols," *Proceedings of ACNS '08*, LNCS 5037, pp. 479–489, 2008.
27. R.I. Paise and S. Vaudenay, "Mutual Authentication in RFID," *Proceedings of AsiaCCS '08*, ACM, pp. 292-299, 2008.

28. P. Rizomiliotis, E. Rekleitis and S. Gritzalis, "Security Analysis of the Song-Mitchell Authentication Protocol for Low-cost RFID Tags," *IEEE Communications Letters*, Vol. 13, No. 4, pp. 274–276, 2009.
29. P. Rogaway, "On the Role Definitions in and Beyond Cryptography," *Proceedings of ASIAN '04*, LNCS 3321, pp. 13–32, 2004.
30. B. Song, "RFID Tag Ownership Transfer," *Proceedings of RFIDsec '08*, July 2008.
31. B. Song and C.J. Mitchell, "RFID Authentication Protocol for Low-Cost Tags," *Proceedings of WISEC '08*, ACM, pp. 140–147, 2008.
32. D.R. Stinson, *Cryptography: Theory and Practice*, 3rd Edition, Chapman & Hall/CRC, Boca Raton, 2006.
33. "Target, Wal-Mart Share EPC Data," RFID Journal, 17 October, 2005. Available online at http://www.rfidjournal.com/article/articleview/642/1/1/.
34. S. Vaudenay, "RFID Privacy based on Public-Key Cryptography," *Proceedings of ICISC '06*, LNCS 4296, pp. 1–6, 2006.
35. S. Vaudenay, "On Privacy Models for RFID," *Advances in Cryptology - Asiacrypt '07*, LNCS 4833, pp. 68-87, 2007.
36. K.Y. Yu, S.M. Yiu and L.C.K. Hui, "RFID Forward Secure Authentication Protocol: Flaw and Solution," *Proceedings of CISIS '09*, IEEE, pp. 627–632, 2009.