A State-Aware RFID Privacy Model with Reader Corruption

Kaleb Lee, Juan Gonzalez Nieto, and Colin Boyd

Information Security Institute, Queensland University of Technology,
GPO Box 2434, Brisbane, Queensland 4001, Australia
{leekj ,j.gonzaleznieto, c.boyd}@qut.edu.au

Abstract. A number of security models have been proposed for RFID systems. Recent studies show that current models tend to be limited in the number of properties they capture. Consequently, models are commonly unable to distinguish between protocols with regard to finer privacy properties. This paper proposes a privacy model that introduces previously unavailable expressions of privacy. Based on the well-studied notion of indistinguishability, the model also strives to be simpler, easier to use, and more intuitive compared to previous models.

1 Introduction

RFID tags are small microchips with an antenna attached, usually embedded within a plastic or paper package. Tags communicate wirelessly when interrogated by an RFID reader. The readers are much larger computing devices which are normally networked to a back-end database. RFID tags are starting to be commonly used for supply-chain and inventory applications to replace the existing barcode systems. Other applications of RFID tags include traffic transit, building and vehicle access, payment cards and national passports.

The last ten years have witnessed an explosion of interest in the area of radio frequency identification (RFID) security. The combination of mobility, wireless communications and low-power hardware presents unique security challenges in the design of authentication protocols, particularly with respect to ensuring privacy. Informally, in the context of RFID systems, privacy means that only authorised parties (tag readers) are able to identify and track RFID tags. Many RFID security models have been proposed in the past[2, 5, 17, 8, 10, 11, 12, 16, 18], however there is no commonly agreed "good" privacy model for analysing RFID protocols. As shown in the recent surveys of Chunhua *et al.* [6] and Coisel and Martin [7], existing privacy models have significant limitations in terms of the classes of protocols that can be analysed within the models and the strength of the privacy notions considered, particularly in relation to the corruption of tags. This paper proposes a new privacy model for RFID that overcomes the limitations of existing models. The new model is simple enough so that it can be widely adopted for security analysis by protocol designers.

Stateful and Stateless Protocols. Proposed RFID protocols can be separated into two categories, stateless and stateful, depending on secret data management mechanism employed by the protocol. Stateful protocols are protocols which make use of updatable information stored in the tag, referred to as the state, whereas stateless protocols do not make use of any updatable state [1]. A state typically consists of secrets that are used for tag authentication and are updated when protocol sessions are completed successfully. Traditionally, stateful protocols have been considered to be more efficient but less private compared to their stateless counterparts, this is mainly due to the ability to rely on state updates for privacy features rather than solely on the encryption schemes implemented within the tag. However, stateful protocols can attain certain privacy notions that are otherwise unachievable. Naturally, their strengths and weakness should be reflected in a privacy model.

An example of the different properties between the two types of protocols is the vulnerability of stateful protocols to *desynchronisation*. Desynchronisation commonly occurs when the last message of the protocol is not received, blocked by an adversary for example, resulting in the internal state of one of the parties not being updated. Next time the tag and reader engage in a protocol run, either the protocol fails because of the non-matching states, or an additional resynchronisation stage is executed. Both cases are likely observable by a third party; therefore de-synchronisation may be seen as resulting in side-channel information leakage, which can be used by an attacker to trace the tag. A security model for stateful protocols must take into account desynchronisation. This vulnerability does not affect stateless protocols. The model presented in this paper introduces a new definition of *stateful privacy*, a notion that coexists with a stateless notion of privacy.

Reader Corruption. Existing security models for RFID authentication model readers and the back-end database as a single entity and assume that the reader and backend database are implemented securely, so that the adversaries cannot interfere with them. This assumption is based on the availability of greater resources for their protection. Typically the back-end database is at least a workstation-class system, which is hosted in a central physically secured location. As discussed by Avoine et al. [4], assuming that the back-end database cannot be directly attacked is unwarranted in practice. After all, these are servers that are commonly connected to the Internet, where vulnerabilities and hacking attacks are commonplace. In the new model presented here, readers and the back-end database are also treated as a single entity, refered to as reader, but adversaries are afforded with corruption powers that expose the secrets in the back-end database. It must be noted that while Avoine et al.[4] discussed the importance of reader corruption, they do not consider it within a formal security model. Interestingly, it turns out that stateful protocols are naturally suited to resist attacks involving corruption of the reader.

Side-Channel Leakage. Traditionally, side-channel information refers to physically observable characteristics of a system, such as power consumption and heat, which can be used by an attacker to compromise security. Unfortunately

most side-channel sources are independent of the underlying protocol and are difficult to include in a security model. The new model focuses on side-channel information which is dependent on protocols and is easily observed without sophisticated means. One notable example is the result of a protocol session, where the results of a protocol can be observed though physical means, e.g. a door opening. There has been other recent work showing that execution time can leak information [3, 9]. As it is likely that more aspects of side channel will be discovered in the future, it would be impractical to capture each distinct leakage scenario independently. The proposed model does not focus on any specific trait, but rather the main cause of side-channel information leakage: de-synchronization.

Contribution This paper proposes a new privacy model for RFID systems. Although the model is designed for passive RFID systems, it is capable of analyzing active systems. Thus it can be used for protocols that are initiated by the tag, and it does not limit the number of message rounds.

- The first distinct contribution of the model is the consideration of stateful protocols. While previous work typically considered stateless protocols to be stronger than stateful protocols, our results suggest that neither is always stronger than the other. Thus both types of protocols have their merits.
- The second contribution is the introduction of reader corruption. In addition to allowing reader corruption, the model distinguishes between corruption of tag volatile and non-volatile memory, achieving previously unobtainable notions of privacy. This separation reflects the more sophisticated skills required to extract information from volatile memory than from non-volatile sources. Furthermore, two flavours of privacy, weak and strong, are defined, based on the ability of the adversary to corrupt challenge tags, which creates more meaningful expressions of privacy.
- Lastly, the paper presents a stateful protocol based on the stateless public-key protocol of Vaudenay [18] highlighting the uniqueness of the stateful definitions.

2 Model components

2.1 RFID System

The setup of an RFID system is simulated using the following two setup algorithms:

- (`rpk`, `rsk`) ← `SetupReader(rpd)`
  On input of a security parameter `rpd`, outputs public/secret key pair `rpk, rsk`. In cases where public-key pairs are not utilized, `rpk` can be considered as an empty string (ϕ).
- (`tpk`, `tsk`, `K`, `S`) ← `SetupTag(T, rpk)`
  On input of identifier `T` and the reader's public key `rpk`, a secret `K`, and a public and private key pair (`tpk, tsk`) are generated. State `S` is initialized according to the protocol specification. `K` is a fixed long term secret, whereas

`S` is an updatable secret. It is common for protocols to make use of only one of `K` or `S`; unused values are considered to be empty (ϕ). The values generated are stored by the reader and/or tag according to the protocol specification.

Definition 1 (Session). *A session is an instance of a protocol execution at a party. Each party stores a session identifier π for every initiated session which is unique within the party. When two session ids π_R and π_T refer to the same protocol execution, they are referred to as* `corresponding` *sessions. Each session also holds a* `Result` *value indicating if a session has* `completed` *or has* `failed`*.*

Definition 2 (Active Session). *Sessions are labeled as either* `active` *or* `inactive`*. It is only possible for a tag to have one* `active` *session at any point in time. Volatile memory is assumed to be erased when a session is labeled* `inactive` *unless otherwise specified by the protocol.*[1] *It is possible for readers to have multiple active concurrent sessions with different tags.*

2.2 Adversarial Oracles

As usual, the interaction between an adversary $\mathcal{A}$ and a RFID system is modelled by oracles that the adversary is allowed to query:

- $(\pi_T, \pi_R, t) \leftarrow$ `Execute`(T)
  This oracle models the situation where an adversary is eavesdropping on the communication between the reader and tag T. $\mathcal{A}$ receives the protocol execution transcript t with the corresponding session identifiers π_T and π_R. A *completed* acknowledgement will be stored as `Result` if the protocol run was successful, otherwise *failed* will be stored.
- $(\pi_R, \pi_T)/(\phi, \pi_T)/(\pi_R, \phi) \leftarrow$ `Initiate`$(T, R/T/R)$
  $\mathcal{A}$ initiates a protocol session with either with both tag T and reader R, or only T or R, returning the new corresponding session identifiers π_R and π_T or only session identifiers π_T or π_R. Any previous sessions of T or R, and corresponding sessions are marked as *inactive*. If `Result` of the previous session is not labeled, it will be labeled as $failed$. Unlike the `Execute` oracle, this oracle allows the adversary to create an incomplete an session between a reader and/or a specific tag.
- *completed*/*failed*/$\phi \leftarrow$ `Result`(π),
  $\mathcal{A}$ retrieves the value of *Result* of protocol session π. If there is yet to be a value of *Result*, ϕ is returned.
- $r/\phi \leftarrow$ `SendTag`(π_T, T, m)
  Sends message m to session π_T of tag T. $\mathcal{A}$ is returned a response message r or ϕ as per protocol specification. In normal protocol interactions a response message r would be returned, however there is also the possibility of a 'null' response ϕ if the message sent was invalid, or the session had already completed or failed. This oracle models the ability of the adversary to send messages to RFID tags.

[1] `Corrupt Memory` will return $M = \phi$

- $r/\phi \leftarrow$ `SendReader`$(\pi_R,\ m)$
  Sends message m to session π_R of the reader R. $\mathcal{A}$ is returned a response message r or ϕ as per protocol specification. Similar to the `SendTag` oracle, in normal protocol interactions a response message r would be returned, however there is also the possibility of a 'null' response ϕ if the message sent was invalid, or the session had already completed or failed. This oracle models the adversary transmitting a message to the RFID reader.
- $(K,\ S) \leftarrow$ `CorruptTag`(T),
  $\mathcal{A}$ obtains the long term secret `K` and session state S of T. Note that we assume that tags will continue to function after corruption. This oracle models the adversary's ability to extract secret data from a tag through specialized methods such as physical extraction.
- $M \leftarrow$ `CorruptMemory`(π_T, T)
  The adversary is given the memory state M of tag T. M is the contents of the temporary($Volatile$) memory of T used when computing the output of `SendTag`(π_T, T). For example, this can include input and output of values of hash functions and any generated/received nonce depending on protocol specification.
- $Y/N \leftarrow$ `Sync`(T)
  This query invokes a protocol-specific function F that determines if T and R are in sync. This oracle models side-channel leakage of information due to de-synchronized states. This oracle is aimed at stateful protocols where it is possible for the updatable states to be out-of-sync. Stateless protocols always return Y.
- $db \leftarrow$ `CorruptReader`()
  $\mathcal{A}$ obtains all information, including secrets, stored by R. This oracle models the ability for an adversary to obtain a snapshot of the database by compromising the back-end system.

2.3 Adversary Classes

We consider the following classes of adversaries, depending on the oracle queries they have access to:

Passive adversaries can only eavesdrop on communications between parties and are not able to communicate with the tag or reader. Thus they are allowed access to the `Execute` oracle, but not to the `Initiate`, `SendTag` and `SendReader` oracles.

Active adversaries can not only access the tag but also the reader. Thus they can access all four oracles `Execute`, `Initiate`, `SendTag` and `SendReader`.

Destructive adversaries are not allowed to interact with tag T after calling `CorruptTag`(T). This class of adversaries is similar to the destructive adversary defined in [18].

Wide adversaries are not allowed to access the oracle `Result` to determine whether a protocol session was successful or not. A wide adversary corresponds to a real life attacker who may be unable to interact with protocol

parties to observe the outcome of the protocol (e.g. whether a door opens). Adversaries are assumed to be wide unless otherwise specified.

Narrow adversaries are allowed to access the oracle `Result` to determine whether a protocol session was successful or not, representing attackers which are close by the tag and the reader it interacts with.

Corrupt 1 (C1) adversaries are those who have access to the oracles `CorruptTag` and `Sync`. They represent adversaries knowledgeable in the area of RFID technology in particular the internal workings of an RFID tag.

Corrupt 2 (C2) adversaries are those who have access to oracles `CorruptMemory` and `CorruptReader`. They represent adversaries with different skill-sets than `Corrupt 1` where specific knowledge of RFID might not be required but are capable of compromising different aspects of the system.

Note that not all types of adversaries are exclusive of each other. Thus, for example, we consider adversaries that are passive and C1, passive and C2, and so on. A noteworthy distinctionwith respect to previous models is that corruption powers are considered to be independent powers that can be possessed by any adversary type rather than abilities of strictly more powerful adversaries. A summary of the types of adversaries is given in table 1.

Passive	***Active***	***Corrupt 1***	***Corrupt 2***	***Narrow***	***Wide***	***Destructive***

`Execute`	`Execute`
	`Initiate`
	`SendTag`
	`SendReader`
		`CorruptTag`
		`Sync`
			`CorruptMemory`
			`CorruptReader`
				No `Result`	`Result`	No interaction with T after `CorruptTag`(T)

Table 1: Adversary classes

2.4 Privacy definitions

Privacy notions are defined based on two games played between the adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. The first game applies to stateless protocols, whereas the second one applies to stateful ones.

Stateless Game:

***Phase* 1**

- Tags, $T_1, T_2, \ldots, T_n$ and reader R are simulated by $\mathcal{C}$.
- $\mathcal{A}$ interacts with $\mathcal{C}$ via oracle queries.
- $\mathcal{A}$ selects two tags T_i, T_j.
- $\mathcal{C}$ selects $c \in \{0,1\}$
- T_i and T_j are reassigned as T_a and T_b respectively if $c = 0$, else T_i and T_j are reassigned as T_b and T_a.

***Phase* 2**

- $\mathcal{A}$ interacts with $\mathcal{C}$ though all oracles except for `Sync`, `CorruptTag`, and `CorruptMemory` on T_a and T_b.
- $\mathcal{A}$ stops interacting with $\mathcal{C}$ and outputs c'.
- $\mathcal{A}$ wins if $c' = c$.

Stateful Game:

In the *Stateful* privacy game, an extra oracle, `BlindExecute`, is introduced and used by $\mathcal{C}$ at the start of *Phase 2*.

- `BlindExecute`(T)
  A protocol session is executed between tag T and reader R with no output. This oracle models situations where a tag communicates with the reader in the absence of an adversary. The execution of the oracle is the same as `Execute` only with no output.
  In schemes where either or both parties store a previous secret for re-synchronization purposes, `BlindExecute` might be required to be called more than once for a scheme to remain private. However this additional requirement reduces the privacy provided by the scheme.

***Phase* 1** Same as in stateless game.

***Phase* 2**

- `BlindExecute`(T_a), `BlindExecute`(T_b).
- $\mathcal{A}$ interacts with $\mathcal{C}$ through all oracles.
- $\mathcal{A}$ stops interacting with $\mathcal{C}$ and outputs c'.
- $\mathcal{A}$ wins if $c' = c$.

Definition 3 (Strong/Weak Game). *A game is said to be* ***`Strong`*** *if during Phase* 1 *the oracles* ***`CorruptTag`****(T_i/T_j) and/or* ***`CorruptMemory`****(T_i/T_j) has been called, for either one or both of the challenge tags, T_i, A non-***`Strong`** game is otherwise a* ***`Weak`*** *game.*

Since the *Strong* game requires the use of corruption oracles, the notion is not applicable for *Active* and *Passive* adversaries. Because *Strong* privacy implies *Weak* privacy, it is assumed that a game is *Strong* unless otherwise specified.

Definition 4 (Privacy). *A scheme is said to be A-G Private, if for any adversary of class* $A \in \{$*Wide, Narrow*$\} \times \{$*Passive, Active, Destructive*$\} \times \{\phi, C1, C2, C1+C2\}$ *playing privacy game* $G \in \{$*Weak, Strong*$\} \times \{$*Stateless, Stateful*$\}$ *have a winning probability of* $\frac{1}{2} + \epsilon$*, where* ϵ *(the advantage) is negligible.*

3 Comparison and Results

3.1 Comparison with Previous Models

A survey of proposed RFID models was recently conducted by Coisel et al. [7] who discussed the advantages and disadvantages of current models before selecting five protocols to be compared. The survey discussed features which should be

considered useful in future models. Most notable is the ability for an adversary to choose and corrupt challenge tags, the differentiation between *Narrow* and *Wide* adversaries, the ability for the adversary to play with all tags in the system, and the ability for the model to analyze all protocols. The model presented in this paper not only considers all the said abilities in addition to a novel notion of privacy, but does so without compromising flexibility.

Of particular concern was the conclusion of Coisel et al. [7] that none of the models was able to distinctively identify the privacy differences between the protocols. The difference between tree-based and SK-protocols was shown to be the most difficult to distinguish. Thus this section uses the same five protocols as a baseline for comparison with other models and shows that the proposed model can distinguish between all five. However due to space constraints, not all full proofs will not be shown. A summary of the results can be found in table 2.

Protocol	**Results**
SK-Protocol	*Weak Active+C1 Stateless*
OSK Protocol	*Weak Active+C1 Stateful*(*Weak Passive+C1 Stateful*)
O-FRAP	*Passive+C1+C2 Stateful Private*
Tree-Based	*Active Stateless*
PK-Based	*Active+C1 Stateless*

Table 2: Comparison of Models

SK-Based Protocol This well-known protocol begins with the reader generating a random nonce N_R. The tag, after receiving N_R, generates its own nonce N_T and uses a pseudorandom function to generate the message $H(K\|N_R\|N_T)$, where K is a unique secret of the tag. The reader than has to compute the same message for all secrets stored until a match is found, completing the protocol.

Theorem 1. *The SK-based protocol shown in Figure 4a is Weak Active+C1 Stateless Private.*

Proof (Sketch):
By simulating the pseudorandom function H a random oracle ensures that it is infeasible for an adversary to obtain K from $H(K\|N_R\|N_T)$. Given that K is randomly distributed, and that the key space $|K|$ is sufficiently large, the advantage of the adversary is: $Pr|A\ Adv| = \frac{1}{|K|-\#\ of\ \texttt{CorruptTag}\ calls}$.

OSK Protocol The OSK protocol makes use of two second pseudorandom functions, H and G. Every time the tag receives a random nonce N_R from the reader, the secret state S is updated with G. For each authentication attempt, the reader has to compute $\mathrm{H}(G^i(S'), N_R)$ for every tag in the system.

Theorem 2. *The OSK protocol[14] shown in table 3a is Weak Active+C1 Stateful Private if the number of consecutive failed sessions at $T_a/T_b < \delta$, otherwise the scheme is Weak Passive+C1 Stateful Private.*

Reader		Tag
S'		S
$N_R \in_R \{0,1\}^\alpha$		
	$\xrightarrow{N_R}$	$N_T \in_R \{0,1\}^\alpha$
		$m_T = H(S\|N_R)$
		$S = G(S)$
	$\xleftarrow{m_T}$	
Find $H(S\|N_R)$= H$(G^i(S'), N_R)$, $i < \delta$		
S=$G^i(S)$		

(a) OSK Protocol

Reader		Tag
$(ID^{-1}, K^{-1}, S^{-1})$, (ID, K, S)		ID, K, S
$N_R \in_R \{0,1\}^\alpha$		
	$\xrightarrow{N_R}$	
		$v = F(K, N_R, S)$
		$v = v_1\|v_2\|v_3\|v_4$
	$\xleftarrow{S, v_2}$	
Computes v_3'		$S = v_1$
	$\xrightarrow{v_3'}$	
		If v_3'=v_3, $K = v_4$

(b) O-FRAP Protocol

Table 3: OSK, and O-FRAP Protocol

Proof (Sketch):
Assuming that secrets S are chosen uniformly at random, the scheme can achieve *Weak* Privacy. Using a similar setup to the proof for SK-based protocol, there would be an additional random oracle G. Evidently the scheme would not be *Strong* private, since by obtaining S at any point in time will allow the adversary to obtain all subsequent updates of S though G. However, the reverse is not true. As it is possible to desynchronize the protocol beyond re-synchronization if the number of consecutive *failed* sessions at a tag is less than δ, by bounding the number of *failed* sessions it is possible for the scheme to be *Weak Active+C1 Stateful Private*. Without this bound the scheme would otherwise be *Weak Passive+C1 Stateful Private*.

O-FRAP The Optimistic Forward-secure RFID entity Authentication Protocol (O-FRAP), shown in table 3b, was proposed by Le et al. in [17], and can be referred to for a more detail description on the computation of v_3'. In general, the reader using N_R (ID, K, S), or previous values (ID^{-1}, K^{-1}, S^{-1}), re-computes v allowing it to check the values of v_3. A pseudorandom function F is used in the protocol, where the output v from input K, N_R, S is separated into four sections v_1, v_2, v_3 and v_4.

Reader		Tag
K		K
$N_R \in_R \{0,1\}^\alpha$		
	$\xrightarrow{N_R}$	$N_T \in_R \{0,1\}^\alpha$
	$\xleftarrow[H(K\|N_R\|N_T),N_T]{}$	

(a) SK Protocol

Reader		Tag
rsk, K, ID_T		rpk, K, ID_T
$N_R \in_R \{0,1\}^\alpha$		
	$\xrightarrow{N_R}$	
		$\mathrm{m}_T = \mathrm{ENC}_{rpk}(N_R\|ID_T\|K)$
	$\xleftarrow{m_T}$	

(b) Public Key Protocol

Table 4: SK and Public Key Protocol

Theorem 3. *The O-FRAP protocol shown in table 3b is Passive+C1+C2 Stateful Private.*

Proof (Sketch):
As it is possible to desynchronize the protocol with an *active* adversary[15], the scheme cannot be *Active-Stateful* private. However, assuming a *passive* adversary, even when the adversary has knowledge of (ID, K, S) and ID^{-1}, K^{-1}, S^{-1}, it is not possible to associate the secrets with the challenge tags after two calls of `BlindExecute`. Thus the protocol is *Passive+C1+C2 Stateful(2) Private.*

Tree-based Protocol The tree-based protocol is proposed by Molar et al. [13] and is very similar to the SK-based protocol shown in table 4a. Instead of each tag sharing a single unique secret with the reader, however, each tag shares a unique *set* of secrets. The secrets are also shared among tags, thus the corruption of a tags can lead to compromise of uncorrupted tags.

Theorem 4. *The tree-based protocol is Active Stateless Private*

Proof (Sketch):
The proof is similar to that of the SK-protocol described in section 4a, only in tree-based protocols the corruption of a large number non-challenge tags would allow the adversary to win the *Weak Active+C1 Stateless Private* with non-negligible advantage. Thus the protocol can only achieve *Active Stateless* Privacy. However, if given a bound on the number of times `CorruptTag` can be called, *Weak Active+C1 Stateless* Privacy can be achieved.

Public Key Protocol The protocol shown in table 4b is presented by Vaudenay in [18]. The protocol uses a public-key pair rsk and rpk.

Theorem 5. *Scheme in Figure 4b is Active+C1 Stateless Private if* (ENC_{rpk}, DEC_{rsk}) *is IND-CCA2 secure.*

Proof (Sketch):
The proof follows very similarly to that shown in Theorem 6. The only minor difference is the omission of the PUF, thus the resulting adversarial advantages will be the same.

3.2 Other Results

Impossibility of Stateless Protocols to Achieve Strong-Stateful Privacy This section describes an attack within the any Strong-Stateful game that allows the adversary to trivially win thus showing the impossibility for stateless protocols to achieve any notions of *Strong* `Stateful` privacy. The adversary $\mathcal{A}$ proceeds as follows:

- ***SetupReader(rpd)***
- $T_\alpha \leftarrow$***SetupTag(*** $\boldsymbol{T_\alpha}$***)***
- $T_\beta \leftarrow$***SetupTag(*** $\boldsymbol{T_\beta}$***)***
- $(K_\alpha, S_\alpha) \leftarrow \texttt{CorruptTag}(T_\alpha)$
- $T_a, T_b \leftarrow T_\alpha, T_\beta$
- $(K_a, S_a) \leftarrow \texttt{CorruptTag}(T_a)$
- if $(K_a, S_a) = (K_\alpha, S_\alpha)$ $b' = 0$, else $b' = 1$
- output b'

Evidently $\mathcal{A}$ wins with probability 1. The above attack assumes a $C1$ adversary, however, a similar attack can be launched for a $C2$ adversary using `CorruptReader`.

Reader		Tag
$\texttt{rsk}, \texttt{ID}_R, \texttt{ID}_R^{-1}$		$\texttt{rpk}, \texttt{S}_T$
$N_R \in_R \{0,1\}^\alpha$		
	$\xrightarrow{N_R}$	
		$N_T \in_R \{0,1\}^\alpha$
		$S_T^{+1} \in_R \{0,1\}^\beta$
		$ID_T = \mathsf{PUF}(S_T)$
		$ID_T^{+1} = \mathsf{PUF}(S_T^{+1})$
		$\mathrm{m}_T = \mathrm{ENC}_{rpk}(N_R \Vert ID_T \Vert ID_T^{+1} \Vert N_T)$
	$\xleftarrow{m_T}$	
$\mathrm{DEC}_{rsk}(m_T) \rightarrow$ if $ID_T = ID_R, ID_R^{-1} = ID_R$ $ID_R = ID_T^{+1}$		
	$\xrightarrow{N_T}$	
		if $\overline{N}_T = N_T$, $S_T = S_T^{+1}$

Fig. 1: Active+C1 Stateful Private Protocol

Stateful Public-Key Protocol The scheme shown in Figure 1 is a stateful protocol using both public-key and a Physical Unclonable Function PUF. A PUF is a hardware pseudorandom function which is unique to each tag, thus it is not assumed to be public. After each successful protocol execution, the state S is updated.

Theorem 6. *Scheme in Figure 1 is Active+C1 Stateful Private if* (ENC_{rpk}, DEC_{rsk}) *is CCA2 secure.*

Proof:
The section will show that using an adversary that can win the Stateful Active+C1 Game with non-negligible advantage, it is possible to construct an adversary that can break CCA2. In the proof 2 games are played by 3 parties: the challenger, C, CCA Adversary, A^{CCA} and A^{Game}, an adversary that can break the Active+C1 Game with non-negligible advantage. A CCA2 game is played between C and A^{CCA}, and a Stateful Active+C1 Game is played between A^{CCA} and A^{Game}.
To model the *Physical Unclonable Function*(PUF), a *random oracle* is simulated by the challenger, only that A does not have direct access to this oracle.The PUF is simulated by the challenger where on input a value $s_i \in_R \{0,1\}^{\gamma}$, generate and returns a unique random value $o_i \in_R \{0,1\}^{\chi}$. Values s_i and o_i are then stored on a table. All values of s and o are to be unique within the table. On the event that a previous s_j is used as input, the previously assigned o_j is returned. As only C has access to PUF, there are no oracles that allow A to interact with the PUF.

To begin the proof, ***SetupReader(rpd)*** is executed, followed by the execution of ***SetupTag(T)*** p unique times, where $p \geq 2$. The public key of the $IND-CCA2$ Game is used for the public key of the protocol(rpk). The simulation of oracles by A^{CCA} to A^{Game} are as follows:

- `Initiate`(T), A^{CCA} generates values π_R and π_T. π_R and π_T are marked as *Active*.Both values are stored and returned to A^{Game}.
- `Execute`(T), If T exists, A^{CCA} generates values π_R ,π_T, and N_R. Values $N_R \in_R \{0,1\}^{\alpha}$, $N_T \in_R \{0,1\}^{\alpha}$ and $S_T^{+1} \in_R \{0,1\}^{\beta}$ are generated. S_T and ID_T^{+1} are passed to the PUF with outputs ID_T and ID_T^{+1} respectively. The message m_T=$(N_R \| ID_T \| ID_T^{+1} \| N_T)$ is encrypted using `rpk`. The corresponding S_T and S_T^{+1} are also retrieved. If $ID_R = ID_T$, then ID_R^{-1} is replaced by $\overline{ID_R}$ and ID_R is replaced by $\overline{ID_R^{+1}}$, if $\overline{ID_R} = ID_R^{-1}$ then only ID_R is replaced by $\overline{ID_R^{+1}}$. *Result* of π_R and π_T are marked as Y. (π_R ,π_T, [N_R, m_T, N_T]) are returned to A^{CCA}.
- `Initiate`(T), A^{CCA} generates values π_R and π_T. Both values are stored and returned to A^{Game}.
- `SendTag`(π_T, T, N_R), If π_T exists and that it is *Active*, the corresponding S_T is retrieved, otherwise ϕ is returned. Values $N_T \in_R \{0,1\}^{\alpha}$ and $S_T^{+1} \in_R \{0,1\}^{\beta}$ are generated. S_T and ID_T^{+1} are passed to the PUF with outputs ID_T and ID_T^{+1} respectively. The message $(N_R \| ID_T \| ID_T^{+1} \| N_T)$ is encrypted using `rpk` and returned as m_T.
- `SendTag`(π_T, T, N_T), If π_T exists and that it is *Active*, A^{CCA} the corresponding S_T and S_T^{+1} are retrieved, else ϕ is returned. If N_T corresponds to the N_T used in π_T, the value of S_T is replaced by the value of S_T^{+1}. Y is recorded in the *Result* and S_T^{+1} is then removed. N is otherwise recorded as the *Result*.

- `SendReader`(π_R, m), If π_R exists the corresponding S_T and S_T^{+1} are retrieved. m_T is forwarded to C for decryption, returning $(N_R \| \overline{ID_R} \| \overline{ID_R^{+1}} \| \overline{N_T})$. If $ID_R = ID_T$, then ID_R^{-1} is replaced by $\overline{ID_R}$ and ID_R is replaced by $\overline{ID_R^{+1}}$ before $\overline{N_T}$ is returned. If $\overline{ID_R} = ID_R^{-1}$ then only ID_R is replaced by $\overline{ID_R^{+1}}$ before $\overline{N_T}$ is returned. If $\overline{ID_R} \neq ID_R or ID_R^{-1}$ then ϕ is returned.
- `CorruptTag`(T), values (K, S) and ID_T corresponding to T is retuned. If T does not exist, ϕ is retuned.
- `Sync`, A^{CCA} retrieves and computes ID_T before evaluating if $ID_T = ID_R$. If $ID_T = ID_R$, Y is returned, else N is returned.

A^{CCA} and A^{Game} plays Phase 1 of the *Stateful* privacy game as specified. Eventually A^{Game} select challenge tags T_i and T_j. A^{CCA} removes T_i and T_i from the system and are reassigned as T_a and T_b[2]. During phase 2, for one instance of `Execute`$(T_{[a/b]})$ or `SendTag` $(T_{[a/b]})$, A^{CCA} picks challenge message with C where $m_0 = N_R \| ID_{T_a} \| K$ and $m_1 = N_R \| ID_{T_b} \| K$ and returns the response from C to A^{Game} as described above. The remainder of the game is simulated by C^{CCA} as in Phase 1. At the end of the game A^{Game} outputs b'_{Game}, A^{CCA} selects $b'_{CCA} = b'_{Game}$.

$$Adv|A^{CCA}| = (Pr[A^{Game} win | b'_{Game} = b_{CCA}]\frac{1}{2} + Pr[A^{Game} win | b'_{Game} \neq b_{CCA}]\frac{1}{2}) - \frac{1}{2}$$

Given the probability of A^{Game} winning the *Stateful* Privacy game equals $\frac{1}{2} + \epsilon$, where ϵ is non-negligible. Thus for half the time when $b'_{Game} = b_{CCA}$, $Adv|A^{CAA}| = Adv|A^{Game}| = \epsilon$. However, the at other times where $b'_{Game} \neq b_{CCA}$ there would be an error in simulation. Thus three cases can be considered:

- C^1: A^{Game} gives maximum advantage: $Pr[A^{Game} win | b'_{Game} \neq b_{CCA}] = 1$, $Adv|A^{CCA}| = \frac{1}{4} + \frac{\epsilon}{2}$
- C^2: A^{Game} gives minimum advantage: $Pr[A^{Game} win | b'_{Game} \neq b_{CCA}] = 0$, $Adv|A^{CCA}| = \frac{1}{4} + \frac{\epsilon}{2}$
- C^3: A^{Game} gives random advantage: $Pr[A^{Game} win | b'_{Game} \neq b_{CCA}] = \frac{1}{2}$, $Adv|A^{CCA}| = \frac{\epsilon}{2}$

Thus, $Adv|A^{CCA}| \geq \frac{\epsilon}{2}$. This concludes the proof.

Theorem 7. *Scheme in Figure 1 is also Destructive+C1+C2 Stateful Private if (ENC_{rpk}, DEC_{rsk}) is CCA2 secure.*

Proof (Sketch):
The proof is very similar to the one shown above with the additional simulation of the two oracles below:

[2] For simplicity it is assumed that A^{CCA} always picks 0. Thus $b_{Game} = 0$

- `CorruptMemory` (π_T, T), A^{CCA} checks if π_T of T is *Active*. If π_T is *Active*, A^{CCA} returns $(ID_T, N_T, S_T^{+1}, ID_T^{+1}, m_T)$.
- `CorruptReader`, The contents of the database is given to A^{Game}. This would include `rsk`, (ID_R, ID_R^{-1}) for all T.

As ID_R^{-1} is also stored in the database, `BlindExecute` would need to be called twice for the protocol to be private. The remainder of the proof remains identical to the one above.

4 Discussion, Conclusion and Future Work

Recently concerns were raised in the ability of current RFID models to capture various privacy properties. It is suggested that current models do not offer sufficient notions of privacy need to analyze the privacy differences in proposed protocols. The model proposed in this paper not only aim to address the concerns raised, but also novel notions of privacy not offered in current models. By offering a new corruption model, the ability for reader corruption and a stateful notion of privacy, the model introduces both stronger and weaker notions of privacy compared to current models. Extending on the analysis from Coisel et al. [7] the model was able to distinguish the privacy properties of the five protocols, which was not possible in the eight chosen models.

This paper also presents a stateful protocol based on Vaudenay's public key protocol capable of achieving the strong notion of *Active+C1 Stateful* and *Destructive+C1+C2* privacy. Due to space constraints, however, little has been presented on notions of privacy involving reader corruption, and there has been little discussion on the weaker notions of privacy. Also open is the problem of achieving the strongest notions of privacy, namely *Active+C1+C2 Stateless* and *active+C1+C2 Stateful* privacy. But as the *Active+C1* notion of privacy is likely to require public-key cryptography, such protocols are unlikely to be practical for RFID. Nevertheless, the model is capable of analyzing said protocols when the need arise. Finally, it would be of interest to further explore the privacy implications between different levels of *strong* and *weak* privacy and between *stateful* and *stateless* privacy.

References

[1] B. Alomair, A. Clark, J. Cuellar, and R. Poovendran. Scalable RFID Systems: a Privacy-Preserving Protocol with Constant-Time Identification. In *the 40th Annual IEEE/IFIP International Conference on Dependable Systems and Networks – DSN'10*, Chicago, Illinois, USA, June 2010. IEEE, IEEE Computer Society.

[2] G. Avoine. Adversary Model for Radio Frequency Identification. (LASEC-REPORT-2005-001), September 2005.

[3] G. Avoine, I. Coisel, and T. Martin. Time Measurement Threatens Privacy-Friendly RFID Authentication Protocols. In S. O. Yalcin, editor, *Workshop on RFID Security – RFIDSec'10*, volume 6370 of *Lecture Notes in Computer Science*, pages 138–157, Istanbul, Turkey, June 2010. Springer.

[4] G. Avoine, C. Lauradoux, and T. Martin. When Compromised Readers Meet RFID. In H. Youm and M. Yung, editors, *Workshop on Information Security Applications – WISA'09*, volume 5932 of *Lecture Notes in Computer Science*, pages 36–50, Busan, Korea, August 2009. Springer.

[5] M. Bruso, K. Chatzikokolakis, and J. den Hartog. Formal Verification of Privacy for RFID Systems. In *Computer Security Foundations Symposium – CSF 2010*, Edinburgh, United Kingdom, July 2010. IEEE.

[6] S. Chunhua, L. Yingjiu, Z. Yunlei, H. D. Robert, Z. Yiming, and Z. Jianying. A survey on privacy frameworks for RFID authentication. *IEICE Transactions on Information and Systems*, E95.D(1):2–11, January 2012.

[7] I. Coisel and T. Martin. Untangling rfid privacy models. Cryptology ePrint Archive, Report 2011/636, 2011. http://eprint.iacr.org/.

[8] R. H. Deng, Y. Li, M. Yung, and Y. Zhao. A New Framework for RFID Privacy. In D. Gritzalis, B. Preneel, and M. Theoharidou, editors, *15th European Symposium on Research in Computer Security – ESORICS 2010*, volume 6345 of *Lecture Notes in Computer Science*, pages 1–18, Athens, Greece, September 2010. Springer.

[9] I. Erguler, E. Anarim, and G. Saldamli. Unbalanced states violates RFID privacy. *Journal of Intelligent Manufacturing*, 23:1–9, 2012.

[10] J. Ha, S. Moon, J. Zhou, and J. Ha. A New Formal Proof Model for RFID Location Privacy. In S. Jajodia and J. Lpez, editors, *13th European Symposium on Research in Computer Security – ESORICS 2008*, volume 5283 of *Lecture Notes in Computer Science*, pages 267–281, Malaga, Spain, October 2008. Springer.

[11] A. Juels and S. Weis. Defining Strong Privacy for RFID. In *International Conference on Pervasive Computing and Communications – PerCom 2007*, pages 342–347, New York City, New York, USA, March 2007. IEEE, IEEE Computer Society.

[12] J. Lai, R. H. Deng, and Y. Li. Revisiting Unpredictability-Based RFID Privacy Models. In J. Zhou and M. Yung, editors, *Proceedings of the 8th International Conference on Applied Cryptography and Network Security – ACNS 2010*, volume 6123 of *Lecture Notes in Computer Science*, pages 475–492, Beijing, China, June 2010. Springer.

[13] D. Molnar and D. Wagner. Privacy and Security in Library RFID: Issues, Practices, and Architectures. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 210–219, New York, NY, USA, 2004. ACM.

[14] M. Ohkubo, K. Suzuki, and S. Kinoshita. Cryptographic Approach to "Privacy-Friendly" Tags. In *In RFID Privacy Workshop*, 2003.

[15] K. Ouafi and R. C.-W. Phan. Traceable Privacy of Recent Provably-Secure RFID Protocols. In S. M. Bellovin, R. Gennaro, A. D. Keromytis, and M. Yung, editors, *Applied Cryptography and Network Security, 6th International Conference, ACNS 2008, New York, NY, USA, June 3-6, 2008*, Lecture Notes in Computer Science, pages 479–489, Berlin, 2008. Springer.

[16] R.-I. Paise and S. Vaudenay. Mutual Authentication in RFID: Security and Privacy. In *ASIACCS'08*, pages 292–299, Tokyo, Japan, 2008. ACM Press.

[17] T. Van Le, M. Burmester, and B. de Medeiros. Universally Composable and Forward-secure RFID Authentication and Authenticated Key Exchange. In F. Bao and S. Miller, editors, *ACM Symposium on Information, Computer and Communications Security – ASIACCS 2007*, pages 242–252, Singapore, Republic of Singapore, March 2007. ACM, ACM Press.

[18] S. Vaudenay. On Privacy Models for RFID. In K. Kurosawa, editor, *Advances in Cryptology – Asiacrypt 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 68–87, Kuching, Malaysia, December 2007. Springer.