A New Framework for RFID Privacy

Robert H. Deng, Yingjiu Li, Moti Yung, Yunlei Zhao

ESORICS 2010

Outline

- Introduction.
- Model of RFID Systems.
- Adaptive Completeness and Mutual Authentication.
- zk-Privacy: Formulation, Clarifications and Comparisons.
- An RFID Protocol within Our Framework.
- Future Works

Introduction

- RFID tags are low-cost electronic devices, from which the stored information can be collected by an RFID reader efficiently (from tens to hundreds of tags per second) at a distance (from several centimeters to several meters) without the line of sight.
- RFID technology has been widely used in numerous applications, ranging from manufacturing, logistics, transportation, warehouse inventory control, supermarket checkout counters, to many emerging applications.
- As a key component of future ubiquitous computing environment, however, RFID technology has triggered significant concerns on its security and privacy as a tag's information can be read or traced by malicious readers from a distance without its owner's awareness.

- It is critical to investigate formal RFID security and privacy frameworks that are fundamental to the design and analysis of robust RFID systems [JW07,V07,DO08,PV08, HMZH08,NSMS08,MLDL09,NSMS09].

- However, due to high system complexity, it turns out to be full of subtleties in developing rigorous and precise RFID system models.

- In this work, we develop a new definitional framework for RFID security and privacy in a rigorous and precise manner. Our framework is based on a zero-knowledge formulation [GMR85], and incorporates the notions of adaptive completeness and mutual authentication.

- We make detailed justification and clarifications, and make comparisons with existing frameworks. Along the way, we also clarify certain confusions and rectify several defects in the existing frameworks.

Model of RFID Systems

- We consider the basic scenario of RFID systems, comprising of a single legitimate reader R and a set of ℓ tags $\mathcal{T} = \{\mathcal{T}_1, ..., \mathcal{T}_\ell\}$. We assume reader R is secure.
- An RFID system $(R, \mathcal{T})$ is setup by a procedure Setup(κ, ℓ).
  - Setup(κ, ℓ) generates the public system parameter σ_R, the reader secret-key k_R and initial internal state s_R^1 for R. It may also setup an initial database DB^1 for R to store information for identifying and authenticating tags.
  - For each i, $1 \leq i \leq \ell$, this procedure generates the public parameter $\xi_{\mathcal{T}_i}$ and the initial secret-key $k_{\mathcal{T}_i}^1$ for a tag $\mathcal{T}_i$ and sets the tag's initial internal state $s_{\mathcal{T}_i}^1$.
- We use $para = (\sigma_R, \xi_1, \cdots, \xi_\ell)$ to denote the public system parameters.

Protocol $\pi(R, \mathcal{T}_i)$

- A tag $\mathcal{T}_i$, $1 \leq i \leq \ell$, exchanges messages with the reader R through a protocol $\pi(R, \mathcal{T}_i)$.
- W.l.o.g., we assume the protocol run of π is always initiated by R and π consists of $2\gamma + 1$ rounds. Each protocol run of π is called a session.
- We assume each tag interacts with the reader sequentially, but multiple tags can interact with the reader "concurrently".
- To allow and distinguish concurrent sessions (at the side of the reader R), we associate each session of protocol π with a unique session identifier *sid*.
  - In practice, *sid* is typically generated by the reader when it is invoked to send the first-round message.

Random Coins, and Internal State and Secret-Key Updates

- Each uncorrupted player uses fresh and independent random coins (generated on the fly) in each session, *in case it is an randomized algorithm*. We assume that the random coins used in each session are erased once the session is completed (whether successfully finished or aborted).
- We assume that the update process of new internal state and secret-key, by an uncorrupted tag in a session run, automatically overwrites (i.e., erases) its old internal state and secret-key.

Session Numbers, and Session Outputs

- Given a security parameter κ, we assume that each tag $\mathcal{T}_i$ takes part in at most s (sequential) sessions in its life time with R, and thus R involves at most $s\ell$ sessions, where s is some polynomial in κ. In practice, the value s can be a fixed constant (e.g., $s = 2^{28}$[BBEG09]).
- The output of reader R in a session sid is a bit o_R^{sid}, which indicates either acceptance ($o_R^{sid} = 1$) or rejection ($o_R^{sid} = 0$). The output of a tag $\mathcal{T}_i$ in a session sid is a bit $o_{\mathcal{T}_i}^{sid}$, which indicates either acceptance ($o_{\mathcal{T}_i}^{sid} = 1$) or rejection ($o_{\mathcal{T}_i}^{sid} = 0$) of the current session run by $\mathcal{T}_i$.
- We assume the session output bits are *public*, and can particularly be accessed by the adversary $\mathcal{A}$. The reason is that, in reality, such outputs can be publicly observed from the behaviors of protocol participants during/after the protocol run or can be learnt by some other side channels.

Adversary

An adversary $\mathcal{A}$, against an RFID system $(R, \mathcal{T})$, is given access to the following four oracles $\mathcal{O} = \{O_1, O_2, O_3, O_4\}$:

- O_1: InitReader()
- O_2: SendT($\mathcal{T}_i, \hat{m}$)
- O_3: SendR($\widehat{sid}, \hat{\alpha}$)
- O_4: Corrupt($\mathcal{T}_i$): Adversary $\mathcal{A}$ obtains the secret-key and internal state information (as well as the random coins) currently held by $\mathcal{T}_i$. Once a tag $\mathcal{T}_i$ is corrupted, all its actions are controlled and performed by $\mathcal{A}$.

- An adversary is a (t, n_1, n_2, n_3, n_4)-adversary, if it works in time t and makes oracle queries to O_μ without exceeding n_μ times, where $1 \leq \mu \leq 4$.
- We denote by $A^{\mathcal{O}}(R, \mathcal{T}, para)$ a PPT algorithm A that, on input of some system public parameter *para*, concurrently interacts with R and the tags in $\mathcal{T}$ via the four oracles in $\mathcal{O}$, where $(R, \mathcal{T})$ is setup by $\mathsf{Setup}(\kappa, \ell)$.

Adaptive Completeness and Mutual Authentication

Adaptive Completeness

- Roughly speaking, adaptive completeness says that, after any attacks (*particularly the desynchronizing attacks*) made by the adversary $\mathcal{A}$, the protocol execution between the reader R and any uncorrupted tag is still complete (e.g., being able to recover from desynchronization).
  - In other words, after undergoing arbitrary attacks, the uncorrupted parties of the RFID system still can recover *whenever the attacks stop*.
  - Formal formalization is referred to the paper.

Matching Sessions

- For a successfully completed session run by a tag $\mathcal{T}_i$, its matching session is defined to be the successfully completed session with the *identical* session transcript at the side of reader R.

- But, the matching-session for a successfully completed session run by R with transcript $trs = (trs', c_{2\gamma+1})$, where trs' denotes the transcript of the first 2γ rounds and $c_{2\gamma+1}$ denotes the last round message sent by R , its matching session can be any session at the side of an uncorrupted tag $\mathcal{T}_i$:
  - a successfully finished session of the identical transcript str;
  - a completed but aborted session of the session transcript $(str', c'_{2\gamma+1})$, where $c'_{2\gamma+1} \neq c_{2\gamma+1}$.
  - an incomplete ongoing session with partial transcript sid', where $\mathcal{T}_i$ is waiting for the last-round message.
- This treatment takes into account the following *unpreventable* "cutting-last-message" attack: : a CMIM adversary $\mathcal{A}$ relays the messages being exchanged by R and $\mathcal{T}_i$ until receiving the last-round message $c_{2\gamma+1}$ from R; after this, $\mathcal{A}$ sends an arbitrary message $c'_{2\gamma+1}(\neq c_{2\gamma+1})$ to $\mathcal{T}_i$ (which typically causes $\mathcal{T}_i$ to abort the session), or, just drops the session at the side of $\mathcal{T}_i$ without sending $\mathcal{T}_i$ the last-round message.

Formulating Mutual Authentications

Experiment $\mathbf{Exp}_{\mathcal{A}}^{auth}[\kappa, \ell]$

1. run Setup(κ, ℓ) to setup the reader R and a set of tags $\mathcal{T}$; denote by *para* the public system parameters;
2. $trans \leftarrow \mathcal{A}^{\mathcal{O}}(R, \mathcal{T}, para)$.

Denote by E_1 the event that *trans* corresponds to the transcript of a successfully completed session run by R in which R successfully identifies an *uncorrupted* tag $\mathcal{T}_i$, but this session has no matching session at the side of $\mathcal{T}_i$. Denote by E_2 the event that *trans* corresponds to the transcript of a successfully completed session run by some *uncorrupted* tag $\mathcal{T}_i \in \mathcal{T}$, and this session has no matching session at the side of R.

Then, roughly speaking, authentication from reader to tag (resp., from tag to reader) says that the probability of E_1 (resp., E_2) occurs is negligible.

zk-Privacy: Formulation, Justification and Comparisons

Notations

- Let $A^{\mathcal{O}}(R, \widehat{\mathcal{T}}, \mathcal{I}(\mathcal{T}_g), aux)$ be a PPT algorithm A that, on input $aux \in \{0,1\}^*$, concurrently interacts with R and a set of tags $\widehat{\mathcal{T}}$ via the four oracles $\mathcal{O} = \{O_1, O_2, O_3, O_4\}$, and has *blind access* to a *challenge* tag $\mathcal{T}_g \notin \widehat{T}$ via a special interface $\mathcal{I}$.
  - From the viewpoint of A, it does not know which tag it is interacting with. It is also required that A interacts with $\mathcal{T}_g$ via O_2 queries only.
- Clean tag: A tag $\mathcal{T}_i$ is called *clean*, if it is not corrupted, and is not currently running an incomplete session with the reader.
  - In other words, a clean tag is an uncorrupted tag that is currently at the status of waiting for the first-round message from the reader to start a new session.

zk-Privacy Experiment: Real World

Experiment $\mathbf{Exp}_{\mathcal{A}}^{zkp}[\kappa, \ell]$

1. run Setup(κ, ℓ) to setup the reader R and a set of tags $\mathcal{T}$; denote by *para* the public system parameter;
2. $\{\mathcal{C}, st\} \leftarrow \mathcal{A}_1^{\mathcal{O}}(R, \mathcal{T}, para)$, where $\mathcal{C} = \{\mathcal{T}_{i_1}, \mathcal{T}_{i_2}, \cdots, \mathcal{T}_{i_\delta}\} \subseteq \mathcal{T}$ is a set of *clean* tags, $0 \leq \delta \leq \ell$;
3. $g \in_R \{1, \cdots, \delta\}$, set $\mathcal{T}_g = \mathcal{T}_{i_g}$ and $\widehat{\mathcal{T}} = \mathcal{T} - \mathcal{C}$;
4. $view_{\mathcal{A}} \leftarrow \mathcal{A}_2^{\mathcal{O}}(R, \widehat{\mathcal{T}}, \mathcal{I}(\mathcal{T}_g), st)$;
5. output $(g, view_{\mathcal{A}})$.

zk-Privacy Experiment: Simulated World

Experiment $\mathbf{Exp}_{\mathcal{S}}^{zkp}[\kappa, \ell]$

1. run Setup(κ, ℓ) to setup the reader R and a set of tags $\mathcal{T}$; denote by *para* the public system parameter;
2. $\{\mathcal{C}, st\} \leftarrow \mathcal{S}_1^{\mathcal{O}}(R, \mathcal{T}, para)$, where $\mathcal{C} = \{T_{i_1}, T_{i_2}, \cdots, T_{i_\delta}\} \subseteq \mathcal{T}$ is a set of *clean* tags, $0 \leq \delta \leq \ell$;
3. $g \in_R \{1, \cdots, \delta\}$, and set $\widehat{\mathcal{T}} = \mathcal{T} - \mathcal{C}$;
4. $sview \leftarrow \mathcal{S}_2^{\mathcal{O}}(R, \widehat{\mathcal{T}}, st)$, where *sview* particularly includes all oracle answers to queries made by $\mathcal{S}$;
5. output $(g, sview)$.

zk-Privacy

- Roughly speaking, zk-privacy says that the output of real world experiment $(g, \mathit{view}_{\mathcal{A}})$ and the output of simulated world experiment (g, sview) are indistinguishable.
- Informally, an RFID protocol π satisfies zk-privacy, if what can be derived by interacting with the challenge tag $\mathcal{T}_g$ in the second-stage of $\mathcal{A}$ can actually be derived by $\mathcal{A}$ itself *without interacting with* $\mathcal{T}_g$. In this sense, the interaction between $\mathcal{A}_2$ and $\mathcal{T}_g$ leaks "zero knowledge" to $\mathcal{A}$.
- The formulation can be easily extended to capture forward and backward zk-privacy, where the experiment output also includes the final (resp., initial) secret-key and internal state of the challenge tag $\mathcal{T}_i$
  - For forward/backward zk-privacy, it is required that the challenge tag $\mathcal{T}_g$ should remain *clean* at the end of real world experiment $\mathbf{Exp}^{zkp}_{\mathcal{A}}$. But, $\mathcal{A}$ is allowed to corrupt the challenge tag after the end of $\mathbf{Exp}^{zkp}_{\mathcal{A}}$.

Why allow $\mathcal{A}_1$ to output an *arbitrary set* $\mathcal{C}$ of tags, and limit $\mathcal{A}_2$ to blind access to a challenge tag $\mathcal{T}_g$ chosen randomly from $\mathcal{C}$?

- The zk-privacy definition implies that adversary $\mathcal{A}$ cannot distinguish any challenge tag $\mathcal{T}_g$ from any set $\mathcal{C}$ of tags;
  - otherwise, $\mathcal{A}$ can figure out the identity of $\mathcal{T}_g$ in $\mathcal{C}$ from its view $view_{\mathcal{A}}$, while this tag's identity cannot be derived from any simulator's view *sview*.
- If $\mathcal{C}$ is removed from the definition of zk-privacy, it is possible for the adversary to distinguish any two tags under its attack, even if each of the tags can be perfectly simulated by a simulator.
  - A special case is that each tag has an upper-bound of sessions in its life time so that an adversary can distinguish any two tags by setting one tag to be run out of sessions in the learning stage.
- In addition, we do not restrict $\mathcal{C}$ to two tags so as to take into account the case that any number of tags may be correlated.

Why limit $\mathcal{A}_1$ to output of *clean* tags?

If $\mathcal{A}_1$ is allowed to output "unclean tags", $\mathcal{A}_2$ can trivially violate the zk-privacy. Consider that $\mathcal{A}_1$ selects two tags that are waiting for different round message (e.g., one tag is clean and the other is not), then $\mathcal{A}_2$ can trivially distinguish them by forwarding to $\mathcal{T}_g$ different round messages.

Why allow $\mathcal{S}$ to have access to oracles in $\mathcal{O}$?

Suppose that $\mathcal{S}$ simulates a tag $\mathcal{T}_i$ from scratch and $\mathcal{A}$ (run by $\mathcal{S}$ as a subroutine) requests to corrupt $\mathcal{T}_i$ in the middle of the simulation. Without oracle access, it is difficult or even impossible for $\mathcal{S}$ to continue its simulation and keep it consistent with its previous simulation for the same tag.

- For example, suppose $\mathcal{T}_i$ ever sent $H(sk, r)$ before corruption, and simulator S has to come up with the secret values (sk, r) upon corruption.

Why limit *sview* to include all oracle answers to queries made by $\mathcal{S}$?

This is to restrict $\mathcal{S}$ not to access the oracles in $\mathcal{O}$ more than $\mathcal{A}$ does.

- The indistinguishability between the simulated view *sview* and the real view $view_{\mathcal{A}}$ of adversary $\mathcal{A}$ in zk-privacy implies that for any (t, n_1, n_2, n_3, n_4)-adversary $\mathcal{A}$, with overwhelming probability, $\mathcal{S}$ cannot query O_1, O_2, O_3, O_4 more than n_1, n_2, n_3, n_4 times, respectively.

Why require $\mathcal{T}_g$ to remain clean at the end of $\mathbf{Exp}^{zkp}_{\mathcal{A}}$ for forward/backward privacy?

In general, forward/backward privacy cannot be achieved if the adversary is allowed to corrupt the challenge tag before the end of its sessions in $\mathbf{Exp}^{zkp}_{\mathcal{A}}$ (i.e., the tag is not clean at the moment of corruption).

- Otherwise, the adversary is able to derive certain protocol messages from the tag's internal state, secret-key, random coins, and the partial session transcript

Comparison with Ind-Privacy [JW07]

- Some observations on ind-privacy:
  - The issue of "clean tags" was not explicitly clarified.
  - Any RFID protocol, *even if it just reveals the tag's secret-key*, trivially satisfies ind-privacy for special RFID systems consisting only a *single* tag (e.g., for a unique item of high value).
- ZK-privacy is *strictly* stronger than ind-privacy, which also an open question posed in [JW07] for developing stronger RFID privacy models.

Comparison with Models of [V07,PV08]

- In [V07,PV08], the simulator is not required to handle tag corruption queries by the adversary.
- In [V07,PV08], an adversary can corrupt any tag at any time (possibly in the middle of session). However, in such a case, forward/backward privacy may not be achievable.
- The matching session concept defined in [V07,PV08] is restricted to *identical* session transcript, without realizing subtleties such as the "cutting last message attacks".
- The notion of adaptive completeness is not defined in [V07,PV08], with no adversarial desynchronizing attacks being taken into account.
- In general, the privacy notions of [V07,PV08] and ind-privacy of [JW07] are incomparable (though for some concrete adversarial strategy, privacy notions of [V07,PV08] may imply ind-privacy), while zk-privacy is strictly stronger than ind-privacy.

Comparison with unp-privacy of [HMZH08,MLDL09]

- unp-privacy is formulated w.r.t. a special kind of 3-round RFID protocols, while zk-privacy is not confined to such.
- unp-privacy requires that messages from tags must be pseudorandom, which is over restricted to include naturally secure protocols (e.g., messages from tags may bear some constant values like protocol version number, etc), and may exclude the use of PKE in RFID-protocols, as ciphertexts of PKE are typically *not* pseudorandom.
- unp-privacy does not allow adversary to access protocol outputs. Otherwise, no protocol can satisfies unp-privacy.
- In summary, zk-privacy is more reasonable than unp-privacy in practice. It allows for more general protocol structure, more powerful adversary, and non-pseudorandom protocol messages.

An RFID Protocol Within Our Framework

Protocol Implementation

Reader R
$\{(I, k, ctr, ID)\}$

Tag $\mathcal{T}_i$
(k, ctr)

c →

$I = F_k^0(ctr||pad_1)$
$(r_0, r_1) = F_k(c||I)$
$r_{\mathcal{T}} = r_0 \oplus (ctr||pad_2)$
$ctr = ctr + 1$

← $I||r_{\mathcal{T}}$

r_R →

If $r_R = r_1$, accept R
Else reject

If find a tuple (I, k, ctr', ID), then
compute $(r_0, r_1) = F_k(c||I)$
If $ctr'||pad_2 = r_0 \oplus r_{\mathcal{T}}$, then
accept the tag, send back $r_R = r_1$
update $ctr' = ctr' + 1$ and $I = F_k^0(ctr'||pad_1)$
Else If $\exists(I', k, ctr', ID)$ s.t. $ctr||pad_2 = F_k^0(c||I) \oplus r_{\mathcal{T}}$ and $F_k^0(ctr||pad_1) = I$,
accept the tag, send back $r_R = F_k^1(c||I)$
update $ctr' = ctr + 1$ and $I' = F_k^0(ctr'||pad_1)$
Else reject

- We prove that the protocol is of adaptive completeness, mutual authentication, and zk-privacy within the new framework.
  - The analysis zk-privacy involving a series of hybrid experiments.
- In comparison with the protocol proposed in [MLDL09], our protocol adds mutual authentication, and is logically more precise.
  - Analysis of completeness and authentication was not conducted in [MLDL09], and the zk-privacy analysis of our protocol is much more complicated than the unp-privacy analysis in [MLDL09].
- We suggest that the methodology used in our analysis is of independent interest, which can be applied to analyze other RFID protocols (particularly those based on PRFs) within our new framework.

Future Works

- To analyze existing RFID protocols and design new protocols within the new framework.
- To extend our RFID privacy framework to more sophisticated and practical scenarios
  - reader corruption, tag cloning (or more feasibly, protocols to prevent swapping attacks), tag group authentication, anonymizer-enabled RFID systems, and tag ownership transfer.

Thanks