Changing Hands Together: A Secure Group Ownership Transfer Protocol for RFID Tags

Yanjun Zuo
University of North Dakota
Grand Forks, ND USA
yanjun.zuo@und.nodak.edu

Abstract

In a Radio Frequency Identification (RFID) enabled system, RFID tags often need to change hands from one owner to another during their lifetime. In this paper, we investigate the application in a group context, i.e., transferring the ownership of a group of tags in one session. This is different from any previous work which focuses on either individual tag ownership transfer or grouping-proofs for simultaneous presence of multiple tags. We are the first to integrate these two aspects and develop a secure and private protocol for tag group ownership transfer. Our protocol resolves the problem of dual ownership which appeared in some previous work, where two entities possess the authentication information of the same tag for a certain period during the ownership transfer process. We evaluate our protocol using theoretical proofs and technical analysis.

1. Introduction

RFID uses radio signals for automatic item identification and data capture. Low-cost RFID tags have been designed for mass distribution and found their applications in various fields including manufacturing, business automation application, citizen identification, hospital patient care, and supply chain management. This technique is expected to play important roles in the future pervasive computing. Many retailers and wholesalers use RFID systems to manage product shipments and inventory tracking. Major retail chains such as Wal-Mart and Target have mandated that all suppliers introduce RFID. As next generation technology, RFID will be a substitution for an optical code system in the near future.

In general, there are three types of players in an RFID system: tags, readers, and a backend server. A tag is a small and cheap device which is combined in IC chip and an antenna for radio communications. It is physically attached to an item with a unique ID. A reader is a device that can recognize the presence of RFID tags and read the information supplied by them. To obtain data from a tag, a reader first queries the tag and then forwards the received identity information to the backend server, which maintains a database of tag entries. After being authorized, the reader can obtain data about the tag. One of the major advantages of RFID is its capability to offer automatic, large scale, and contactless data collection.

In this paper, we consider a specific application of RFID systems and the related security and privacy issues: RFID tag group ownership transfer, i.e., transferring the ownership of a group of tags in one session. It is based on two separated aspects of RFID: tag ownership transfer and grouping-proofs.

Ownership of tags may be changed frequently during their lifetime. For example, in a supply chain, tags are initially created and attached to products by manufactures. The tagged items are then handed over to retailers, and finally consumers buy tagged objects. Each of those transactions represents a product ownership transfer. In terms of RFID tags, ownership transfer means the new owner takes over the tag authorization from the current owner. Then the new owner should possess all the necessary secret information to identify the tag, authenticate itself to the tag, modify the tag’s configurations, and even pass the control of the tag to other entities.

RFID tag ownership is considered complete if all of the following conditions are satisfied [1-4]: (1) new owner privacy (forward privacy): the old owner will not be able to identify and control the tag after it passes the ownership to a new owner; (2) old owner privacy (backward privacy): the new owner will not be able to use any information it has on the tag to track back past interactions between the tag and its previous owners; and (3) authorization recovery: the present owner should be able to transfer its ownership temporarily to the previous owner in some situations such as after-sales service for an RFID tagged object.

---

1 This work was supported in part by US AFOSR under grant FA 9550-09-1-0215.
It seems straightforward to develop a scheme to transfer the ownership of a tag to its new owner. For example, the current owner simply passes all the relevant secret information about the tag to the new owner via a secure channel. As we will see in Section 2, this simple method has a serious flaw, i.e., the windowing problem. It allows dual ownership of a tag in a certain period during the ownership transfer process when both the old and new owners possess the authentication information of the tag. This is an undesirable feature in many multi-party environments, where the ownership of an item must be clearly defined and controlled for accountability purposes.

Another aspect of our protocol is RFID tag grouping-proofs, i.e., generating evidence of simultaneous presence of two or more tags in a reader’s range. Juels [5] pointed out that there are several practical scenarios where grouping-proofs could substantially expand the capacities of RFID enabled systems. In an environment that requires a high level of security such as airports, it may be necessary to couple an identifier, such as an electronic passport, with a physical person or with any of his/her belongings, such as their bags. Similarly, one may want to enforce that access to certain resources only be granted if appropriate groups of objects are present.

The proposed protocol in this paper is for RFID tag group ownership transfer. It is to ensure that a group of products (tags) are transferred in term of their ownership from the current owner to a new owner in one session as a whole instead of individually. There are several situations where transfer of ownership of RFID tags must be conducted on a group basis. In a supply chain, some products need to be shipped together in groups from one entity to another. For example, safety regulations may require that drugs be shipped accompanied by information leaflets [6]. If those grouped products are purchased by a patient, ownership transfer must be conducted as a group. The patient should only purchase both the drugs and the accompanying information leaflets. Missing any of the grouped tags should not be considered as a valid ownership transfer. In some other similar applications, we also need to ensure the atomic characteristic of a group ownership transfer: the tag ownership should be either transferred entirely as a group or aborted completely (there should be no such situation where some tags are transferred in terms of ownership while others are not).

It is possible to transfer the ownership of a group of RFID tags one by one. However, this is not only inefficient and time consuming, but also cannot ensure the simultaneous presence of multiple tags so that the tags’ ownership can be transferred as a group. Indeed, one cannot exclude the possibility that one tag is scanned by a malicious reader and then the result is combined with tag evidence generated in an earlier time. Then the combined result may fool a server that two tags appeared in the same time. To prevent this kind of interleaving attacks, grouping-proofs are necessary for a valid group ownership transfer.

Our protocol involves a group of tags being scanned by an RFID reader in the same session. This enables the tags to generate a proof of simultaneous presence and ownership transfer. The new owner sets the secret information to be shared with each tag in the group in the session. The ownership of all the tags is transferred simultaneously.

Our major contribution in this paper is to present a comprehensive security and privacy framework for RFID group ownership transfer. To our best knowledge, we are the first to address the needs to transfer the ownership of tags as a group in a multi-party environment that involves generating evidence of simultaneous presence of a group of tags. Our protocol eliminates the possibility that two or more entities possess the tag authorization information in any time period during a tag ownership transfer process as presented in some previous work.

The rest of the paper is organized as follows. Section 2 introduces related work. Section 3 discusses the threat model and system assumptions. Section 4 presents the group ownership transfer protocol. Section 5 concludes this paper.

2. Related Work

2.1 RFID Tag Ownership Transfer

In the literature, relatively few research papers have been devoted to RFID tag ownership transfer. Molnar, et al., [4] presented a time-limited reader delegation and tag ownership transfer protocol. Lim, et al., [7] proposed a tag ownership transfer scheme, which is an extension of their reader-tag authentication and reader delegation protocol. Other protocols developed specifically for RFID tag ownership transfer include [1, 2, 8, and 9]. We will not discuss those protocols one by one but point out a concern which is common in those protocols. We believe this concern is a key issue that must be addressed in RFID tag ownership transfer. The procedure as specified by those protocols for RFID tag ownership transfer can be abstracted in three steps: (1) the old owner first updates the secrets shared with each tag. This is to ensure that any new owner will not be able to link the tag to its past transactions; (2) the old owner passes the updated tag secrets to the new owner via a secure channel; and (3) the new owner authenticates the tag using the received secrets and then applies the new secrets it generated to reset the tag. This is to ensure that the old
owner will not be able to authenticate the tags in the future. However, there is a time window after step (2) but before the new owner updates the secrets in step (3). During this period, both the old and new owners possess the information necessary to authenticate the tag. This basically means dual ownership for the tag during that time gap. We call it the windowing problem of RFID tag ownership transfer. This problem could damage the accountability of tag owners in a multi-party system such as a supply chain. In those systems, products change hands frequently. The ownership of any tagged product must be clearly defined and tightly controlled at any moment. In a situation, a malicious old owner could run a race attack, e.g., change product codes or introduce new codes, before the above step (3) is taken, and later place the blame for faulty products on the other since the other party also owns the valid secret keys. Conversely, a malicious new owner could perform some actions and then put the responsibilities to the old owner. Both situations could cause an infringement of tag ownership as well as tag security and privacy. The windowing problem leads to undesirable consequences in a situation where accountability is a key issue.

Our protocol resolves the windowing problem and leaves no opportunity for dual ownership at any moment during an ownership transfer process. The current and new owners will never simultaneously possess the same set of valid secrets of any tag in a group. Unlike the previous work, our protocol does not require separated steps for the old and new owners to update the secret keys of a tag.

2.2 RFID Grouping-Proofs

The Yoking-proofs [5] is the first protocol to explicitly address RFID grouping-proofs, i.e., a proof of the simultaneous presence of two tags. In the protocol, a reader scans the tags sequentially and each tag uses information received from peer tags to generate an evidence of its presence. Burmester, et al., [6] proposed a set of robust grouping-proofs protocols for RFID tags. Their work inspires some system design ideas for our group ownership transfer protocol. For instance, the protocols in [6] assign a specific tag to play the role of “initiator," which transmits some random numbers or a random password. This has the security benefit of curtailing reflection attacks. A group secret key is shared by all the tags belonging to the same group. This is to facilitate an early detection of non-group members. To avoid interleaving attacks, the challenge of authorized tags will include a nonce. We adopt those features in our protocol to enhance security and privacy. However, the fundamental difference between the protocols presented in [6] and ours is that the former are essentially server-tag mutual authentication schemes to prove the simultaneous presence of RFID tags and ours enables simultaneous group ownership transfer.

3. Threat Model and System Assumptions

Our protocol assumes that each tag possesses a unique ID stored in the tag as well as in the backend server. Each tag also possesses the basic cryptographic abilities such as hashing functions, pseudorandom functions, and bit-wise XOR operations.

The threat model assumes a Byzantine adversary, where all the entities (tags, reader, backend server, and the adversary) have polynomially bounded resources. The adversary can intercept all the communications between a tag and a reader and also be able to modify the contents of the messages. We do not consider physical attacks to readers or tags.

Each tag’s owner is represented by a server with a back-end database that maintains the information about the tags that the server owns. A server connects its reader(s) via a secure channel. Then the reader communicates with the RFID tags that the server manages. Since the server and its reader(s) are so tightly related, we consider they share all the necessary information to authenticate each tag.

In addition to tags, readers, and servers in the RFID system, we require an additional entity: a trusted server TS which the old and new owners both trust. We noticed that some earlier work [7 and 9] also requires a similar trusted entity to facilitate RFID system functions. In some multi-party environments such as supply chains or virtual organizations, there is a need for a system-wide authorized agent that can monitor and facilitate activities such as product transfer from an upstream partner to a downstream one. In our protocol, we assume that TS shares a secret key with each partner in the system. In addition, TS is required to share a secret $s_i$ with each tag $T_i$. This secret is set when the tagged item enters the system such as a supply chain for the first time. $s_i$ is exclusively shared between TS and $T_i$ and even the owner of $T_i$ does not possess this secret. We consider a supply chain differs from other applications since it needs to support frequent product ownership transfers. Each entity only “temporarily” owns the tagged item. $s_i$ is neither used to identify the tag $T_i$ nor used to secure communications. It is just to facilitate group ownership transfer and ensure new owner privacy.

We assume that a group of items whose ownership to be transferred has been specified. Each item is attached with an RFID tag, say $T_i$, which shares a group secret key $k_{group}$ with the group members and an individual key $k_i$ with its owner.
Figure 1: Phase One and Phase Two of the Proposed Group Ownership Transfer Protocol – for Two Tags

(a) \( S_{\text{current}} \triangleq M_1, S_{\text{current}} \triangleright k_{\text{current}, \text{new}}, S_{\text{current}} \equiv S_{\text{current}} \iff k_{\text{current}, \text{new}} \rightarrow S_{\text{new}}, S_{\text{current}} \equiv \emptyset(M_1), S_{\text{current}} \equiv \#(k_{\text{current}, \text{new}}) \)

(b) \( S_{\text{current}} \triangleright \text{ID}, \text{||(\text{Ki},XOR\text{Kk} \_\text{mask})}} || E[k_{\text{current},TS} k_i \_\text{mask}] || (k_{\text{group, new}} \_\text{XOR} k_{\text{group, mask}}) || E[k_{\text{current},TS} k_{\text{group, mask}}] \)

(c) \( T_i \triangleq \text{XOR}(f(k_i, N_T), M'), T_i \triangleright k_i, T_i \triangleright N_T \)

\( T_i \triangleq (k_{\text{new, Xor}} k_i \_\text{mask}) \)

Figure 2: Logical Postulate Formula for Protocol Correctness Proof
4. Group Ownership Transfer Protocol

There are three phases of the proposed group ownership transfer protocol (GOT) for RFID tags (See Figure 1). We assume there are $n$ tags in the group whose ownership is to be transferred from the current owner $S_{\text{current}}$ to a new owner $S_{\text{new}}$. For simplicity we illustrate our protocol with two tags. But the protocol can be extended to any number of tags. Table 2 shows the messages exchanged.

Table 1: Symbol Notations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{\text{current}}$</td>
<td>The server of the current owner</td>
</tr>
<tr>
<td>$S_{\text{new}}$</td>
<td>The server of the new owner</td>
</tr>
<tr>
<td>$TS$</td>
<td>The trusted server in the system</td>
</tr>
<tr>
<td>$R_{\text{current}}$</td>
<td>The tag reader of $S_{\text{current}}$</td>
</tr>
<tr>
<td>$T_i$</td>
<td>A tag in the group</td>
</tr>
<tr>
<td>$k_i$</td>
<td>The tag’s secret key</td>
</tr>
<tr>
<td>$k_{\text{group}}$</td>
<td>The group secret key</td>
</tr>
<tr>
<td>$s_i$</td>
<td>The secret key shared between tag $T_i$ and $TS$</td>
</tr>
<tr>
<td>$OT$</td>
<td>Ownership transfer request from $S_{\text{current}}$</td>
</tr>
<tr>
<td>$\rightarrow$</td>
<td>A message is transferred from the left to right</td>
</tr>
<tr>
<td>$</td>
<td></td>
</tr>
<tr>
<td>$\oplus$</td>
<td>Bit-wise mapping XOR operator</td>
</tr>
<tr>
<td>${}$</td>
<td>A set of elements</td>
</tr>
<tr>
<td>$ID_i$</td>
<td>The identification of $T_i$</td>
</tr>
<tr>
<td>$M_i$</td>
<td>A message to be exchanged</td>
</tr>
<tr>
<td>$f(k, M)$</td>
<td>Pseudorandom function taking seed $k$ and message $M$</td>
</tr>
<tr>
<td>$E[k, M]$</td>
<td>Message $M$ encrypted with key $k$ using a standard cryptographic function, i.e., AES</td>
</tr>
<tr>
<td>$NR$</td>
<td>Random nonce generated by $S_{\text{current}}$</td>
</tr>
<tr>
<td>$NT$</td>
<td>Random nonce generated by $TS$</td>
</tr>
<tr>
<td>$h(.)$</td>
<td>One-way hashing function</td>
</tr>
</tbody>
</table>

(1) **Phase One**: Group RFID tags identification

In this phase, $S_{\text{current}}$ authorizes a request for group ownership transfer from $S_{\text{new}}$. Then it instructs the reader $R_{\text{current}}$ to scan and verify all the group tags are present. If any member is missing, the process stops. Otherwise, the following steps are taken.

**Step 1.1**: $S_{\text{current}} \rightarrow S_{\text{current}}: OT ||\text{Credentials} || G_{id}$

$S_{\text{new}}$ submits a request $OT$ to $S_{\text{current}}$ for ownership transfer over a group of tags with an identification $G_{id}$. $S_{\text{new}}$ provides its credentials along with $OT$. The identification of the group of tags is assumed to be available for the system partners. For security reasons $G_{id}$ will not be used as a secret to encrypt messages. Rather, a secret group key is shared among the group members and used to authenticate the group members.

**Step 1.2**: $S_{\text{current}}$ evaluates the ownership transfer request $OT$ based on $S_{\text{new}}$’s credentials and its internal policies. If the based business transaction is authorized, the ownership transfer request will be honored. Next, $S_{\text{current}}$ confirms that all the tags in the group are present so that their ownership can be transferred together in the same session without any tag missing. To do that, $R_{\text{current}}$ scans the tags in its field and collects their IDs.

Table 2: Message Notations

<table>
<thead>
<tr>
<th>$M_i$</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ID</td>
<td></td>
</tr>
<tr>
<td>$M_i \oplus f(NR, c)</td>
<td></td>
</tr>
<tr>
<td>$M_i \oplus f(k_{\text{group}}_{\text{mask}})$</td>
<td></td>
</tr>
<tr>
<td>$M_i \oplus f(k_{\text{group}}_{\text{mask}})$</td>
<td></td>
</tr>
<tr>
<td>$M_i \oplus f(k_{\text{group}}_{\text{mask}})$</td>
<td></td>
</tr>
</tbody>
</table>

There are several approaches to identify RFID tags, collect tag IDs [10-14], and monitor missing tags [15]. All of these approaches assume that RFID tags resolve collision using a type of ALOHA to support effective and efficient communications between a reader and multiple tags at the same time. A general way to collect the tag IDs in the field of a reader has several steps. The reader first broadcasts the number of available time slots (as used in a ALOHA type scheme). Each tag then independently picks a time slot to reply. If one slot is chosen by one tag and the tag ID is sent, the transmission is successful. The reader can identify the tag. However, when multiple tags pick the same slot, a collision occurs and the process must be restarted again. This approach exploits the fact that a low cost RFID tag picks a reply slot in a deterministic fashion calculated using its ID and a random number as input. Given the random number and a frame size, a tag will always pick the same slot to reply. So, the reader will be able to detect the tags in its field. Since the focus of this paper is not on tag ID collection, we focus only on efficient communications between a reader and multiple tags at the same time. The reader then forwards the collected tag IDs to $S_{\text{current}}$, and $S_{\text{current}}$ stores all the IDs of the tags and the frame size, it will be able to determine the present tags in the group whose ownership is to be transferred. After $S_{\text{current}}$ confirms that all the group members are currently present, $S_{\text{current}}$ retrieves the corresponding secret key $k_i$ for each tag $T_i$ in the group as used in the following steps.

**Step 1.3**: $S_{\text{current}} \rightarrow S_{\text{new}}: ACK$
$S_{current}$ acknowledges $S_{new}$ about its request for tag ownership transfer. An acknowledgement message $ACK$ sent to $S_{new}$ includes the pseudonym IDs of the tags in the group.

(2) Phase Two: Group ownership transfer

In this phase, $S_{new}$ sets new secret keys for each tag, which are then secretly transmitted to each tag. After a tag $T_i$ authenticates $S_{current}$ and confirms that $S_{current}$ has authorized the ownership transfer, $T_i$ generates a message to authenticate itself to other group members. Then each tag updates its secret keys to be shared with $S_{new}$ and the group key to be shared with the group members. Finally, a proof for group ownership transfer is generated.

**Step 2.1:** $S_{new} \rightarrow S_{current}: E[k_{Scurrent_Snew}, M_1] || G_i$ $S_{new}$ chooses a new secret key $k_{i_{new}}$ to be shared with tag $T_i$ and a new group key $k_{group_{new}}$ to be shared among the members of the group. Since those new keys will be submitted to $S_{current}$ to be delivered to each tag, $k_{i_{new}}$ and $k_{group_{new}}$ are masked by random values $k_{i_{mask}}$ and $k_{group_{mask}}$, respectively, and these masks should not be known to $S_{current}$. This is to prevent $S_{current}$ from learning the new keys and ensure new owner privacy.

$S_{new}$ constructs a message $M_1$ and sends it to $S_{current}$ where $M_1 = \{ID_i \oplus (k_{i_{new}} \oplus k_{i_{mask}}) \mid E[k_{Snew_TS_ki_{mask}}(k_{i_{new}} \oplus k_{group_{mask}}) \mid E[k_{Snew_TS_ki_{mask}}(k_{group_{mask}}), \mid \mid \mid E[k_{Snew_TS_ki_{mask}}(k_{group_{mask}}), \mid \mid \mid E[k_{Snew_TS_ki_{mask}}(k_{group_{mask}}))\}$ where $1 \leq i \leq n$, and $k_{i_{mask}}$ and $k_{group_{mask}}$ are encrypted using the key shared between $S_{new}$ and $TS$, i.e., $k_{Snew_TS_{ki_{mask}}}$ Upon receipt of $M_1$, $S_{current}$ can only retrieve $k_{i_{new}} \oplus k_{i_{mask}}$, but it cannot learn anything about the new key $k_{i_{new}}$ since it does not know $k_{i_{mask}}$. For security reasons, $M_1$ is encrypted using a standard cryptographic function (such as AES, 3DES) with the key $k_{Scurrent_Snew}$ shared between $S_{new}$ and $S_{current}$.

**Step 2.2:** $S_{current} \rightarrow TS: E[k_{Scurrent_TS_M_2}]$ $S_{current}$ constructs a message $M_2$ and forwards it to $TS$. $M_2$ contains the following components: (1) the IDs of the current and the new owners, i.e., $ID_{Snew}$ and $ID_{Scurrent}$; (2) the received $E[k_{Snew_TS_ki_{mask}}(k_{i_{new}} \oplus k_{i_{mask}})]$ and $E[k_{Snew_TS_ki_{mask}}(k_{group_{mask}})]$ in message $M_1$ from $S_{new}$; and (3) the IDs of the tags in the group, i.e., $M_2 = ID_{Snew} || ID_{Scurrent} || E[k_{Snew_TS_ki_{mask}}(k_{i_{new}} \oplus k_{i_{mask}})] || E[k_{Snew_TS_ki_{mask}}(k_{group_{mask}})]$ where $1 \leq i \leq n$. We require that $S_{current}$ send the ownership transfer related messages to $TS$ directly because $TS$ needs to confirm that $S_{current}$ has authorized the group ownership transfer. $M_2$ is encrypted using the secret key $k_{Scurrent_TS}$, which is shared between $TS$ and $S_{current}$. Since $k_{i_{new}} \oplus k_{i_{mask}}$ is not forwarded to $TS$, $TS$ only knows $k_{i_{mask}}$ but not $k_{i_{new}} \oplus k_{i_{mask}}$ for each tag $T_i$. Furthermore, $S_{current}$ only knows $k_{i_{new}} \oplus k_{i_{mask}}$ but not $k_{i_{mask}}$. So, neither $S_{current}$ nor $TS$ alone can figure out the new secret key $k_{i_{new}}$ as set by $S_{new}$.

This kind of separation of duties provides an additional layer of security: any attempt to compromise the new keys goes beyond the ability of either $TS$ or $S_{current}$ alone. Also, $TS$ never possesses the ownership title of the tagged items.

**Step 2.3:** $TS \rightarrow S_{current}: E[k_{Scurrent_TS_M_3}] || N_T$ In order to keep $S_{current}$ from learning the new keys $k_{i_{new}}$ and $k_{group_{new}}$ as set by $S_{new}$ and also allowing each tag $T_i$ to retrieve those keys, $TS$ masks each $k_{i_{mask}}$ and $k_{group_{mask}}$ using the corresponding secret it shares with $T_i$, i.e., $s_i$. Specifically, $TS$ generates $f(s_i, N_T) \oplus k_{i_{mask}}$ and $f(s_i, N_T) \oplus k_{group_{mask}}$, where $1 \leq i \leq n$. $N_T$ is a random nonce generated by $TS$. Then $TS$ constructs a message $M_3$ and sends it to $S_{current}$ where $M_3 = \{ID_i || f(s_i, N_T) \oplus k_{i_{mask}} || f(s_i, N_T) \oplus k_{group_{mask}}\}$. $M_3$ is encrypted using key $k_{Scurrent_TS}$ shared between $TS$ and $S_{current}$.

**Step 2.4:** $S_{current}$ generates a random nonce $N_R$ and transfers all the components in $M_3$ as well as other necessary information to $R_{current}$ so that $R_{current}$ can interact with each tag in the group. The following sub-steps represent the group authentication and new key update process. It is assumed that $T_i$ plays the role of an initiator.

**Step 2.4.1:** $R_{current} \rightarrow T_i: M_4 || N_R \& R_{current} \rightarrow T_j: M_5 || N_R$ $R_{current}$ needs to send the following information to tag $T_i$ and $T_j$ in a similar way: (1) $k_{i_{new}} \oplus k_{i_{mask}}$; (2) $f(s_i, N_R)$; (3) $k_{group_{new}} \oplus k_{group_{mask}}$; and (4) $f(s_j, N_R) \oplus k_{group_{mask}}$. The message to be sent from $R_{current}$ to each tag must allow the tags to verify the integrity of the above four components and also authenticate $R_{current}$ (and thus $S_{current}$). This can be accomplished by using the secret key shared between $R_{current}$ and each tag. For example, to send $k_{i_{new}} \oplus k_{i_{mask}}$, $R_{current}$ generates two messages: $\{f(k_{i_{new}} \oplus k_{i_{mask}}) \mid f(k_{i_{new}} \oplus k_{i_{mask}}) \mid f(k_{i_{new}} \oplus k_{i_{mask}}) \mid f(k_{i_{new}} \oplus k_{i_{mask}})\}$. We use the symbol $M_{ij}$ to denote these two messages and call it the verification code for $k_{i_{new}} \oplus k_{i_{mask}}$. Since tag $T_i$ possesses $k_i$ and $N_R$, it can verify and retrieve ($k_{i_{new}} \oplus k_{i_{mask}}$) as shown in the next step. In a similar way, $R_{current}$ generates the verification codes for the above messages (2), (3), and (4), respectively, and those verification codes are represented by $M_{ij_2}, M_{ij_3}$, and $M_{ij_4}$. Then the entire message to be transferred from $R_{current}$ to $T_i$ is denoted as $M_i = M_{ij} || M_{ij_2} || M_{ij_3} || M_{ij_4}$. $R_{current}$ also transfers a similar structure $M_j$ (by replacing each subscript $i$ with $j$) to $T_j$.

**Step 2.4.2:** $T_i \rightarrow R_{current}: M_6 || M_j \mid c$ When tag $T_i$ receives $M_i$ from $R_{current}$, it retrieves and verifies the four components as specified in the previous step: (1) $k_{i_{new}} \oplus k_{i_{mask}}$; (2) $f(s_i, N_T) \oplus k_{i_{mask}}$;
message proofs with unrelated tags. This capability is denied if a tag engages in grouping-accident paring may create challenges to real-world characterized as DoS vulnerability. To appreciate how waste of resources in many situations and could be, run. As pointed by Burmester, in an early stage rather than leaving it after the protocol

(3) $k_{\text{group}} \oplus k_{\text{group-mask}}$; and (4) $f(s_i, N_{T_i}) \oplus k_{\text{group-mask}}$. For component (1), $T_i$ first performs $f(k_i, N_{T_i} \oplus [k_i \oplus (k_{i_{\text{new}}} \oplus k_{i_{\text{mask}}})])$ and retrieves $k_{i_{\text{new}}} \oplus k_{i_{\text{mask}}}$. Then it plugs in this value to calculate the hash of $(k_{i_{\text{new}}} \oplus k_{i_{\text{mask}}} \oplus N_{T_i} || k_i)$ using its own $k_i$ and $N_{T_i}$. If the result matches the received $h(k_{i_{\text{new}}} \oplus k_{i_{\text{mask}}} \oplus N_{T_i} || k_i)$ in $M_{4s}$, $T_i$ authenticates $S_{\text{current}}$. If all the four components are verified, $T_i$ constructs two pseudonyms based on $N_k$, i.e., $M_6 = f(k_{\text{group}} \oplus N_k \oplus c)$ and $M_7 = f(k_i, N_k \oplus c)$, where $c$ represents a counter set by the initiator $T_i$. The value of $c$ is updated after each protocol run and hence prevents reply attacks. Message $M_6$ allows other group members to authenticate $T_i$ since only members in the group share the group key $k_{\text{group}}$. Message $M_7$ will be kept by $R_{\text{current}}$ as a part of the proof for group ownership transfer. Finally, $T_i$ sends both $M_6$ and $M_7$ to $R_{\text{current}}$.

**Step 2.4.3:** $R_{\text{current}} \rightarrow T_j$: $M_6 \parallel c$ $R_{\text{current}}$ keeps a copy of $M_6$ and then forwards it and $c$ to $T_j$.

**Step 2.4.4:** Message $M_6$ is for group member authentication. Since tag $T_j$ possesses the group key $k_{\text{group}}$, the nonce $N_k$ generated by $S_{\text{current}}$ and counter $c$ received from $R_{\text{current}}$, it can verify the received message $M_6$. It is preferable to verify group members in an early stage rather than leaving it after the protocol run. As pointed by Burmester, et al., [6], although unrelated tags can be detected later, it is undesirable to waste resources in many situations and could be characterized as DoS vulnerability. To appreciate how accident paring may create challenges to real-world applications, consider the following scenario. An RFID reader is configured to take temporary measures after a failed grouping-proof attempt, e.g., notify an assembly worker of a missing component in a shipment pallet. This capability is denied if a tag engages in grouping-proofs with unrelated tags.

If $M_6$ is verified, $T_j$ knows that it is interacting with the tags in the same group. Then, it performs the following operations to update its new keys to be shared with $S_{\text{new}}$ and the group members, respectively:

a) Apply $s_j$ (the secret shared between $T_j$ and $TS$) to retrieve $k_{j_{\text{mask}}}$ and $k_{\text{group-mask}}$ by performing XOR operations on $f(s_j, N_{T_j})$ and the received messages $f(s_j, N_{T_j} \oplus k_{j_{\text{mask}}})$ and $f(s_j, N_{T_j} \oplus k_{\text{group-mask}})$ in message $M_6$, respectively.

b) Apply $k_{j_{\text{mask}}}$ and $k_{\text{group-mask}}$ to retrieve the new keys:

$k_{j_{\text{new}}} = (k_{j_{\text{new}}} \oplus k_{j_{\text{mask}}}) \oplus k_{j_{\text{mask}}}$

$k_{\text{group-new}_{j}} = (k_{\text{group-new}} \oplus k_{\text{group-mask}}) \oplus k_{\text{group-mask}}$

c) $T_j \rightarrow R_{\text{current}}$: $M_9 \parallel M_8$

$T_j$ applies $k_{\text{group-new}_{j}}$ and $k_{i_{\text{new}}}$ and $k_i$ to generate three pseudonyms based on $N_k$ and $c$ and packs them into two messages $M_8$ and $M_9$, where $M_8 = f(k_{\text{group-new}_{j}}, N_k \oplus c)$ $k_{i_{\text{new}}} \oplus (N_k \oplus c)$ and $M_9 = f(k_{\text{group-new}_{j}}, N_k \oplus c)$. $M_8$ will be part of the proof for group ownership transfer. It can be used to verify that $T_j$ has updated its secret key shared with $S_{\text{new}}$ and also provides evidence of the message generated by using $T_j$’s old secret key. $M_9$ will be used to verify that all the group members have updated their group key. Finally, the two messages are transferred to $R_{\text{current}}$.

**Step 2.4.5:** $R_{\text{current}} \rightarrow T_j$: $M_8$ $R_{\text{current}}$ keeps message $M_8$ and forwards $M_9$ to $T_i$. $T_i$ performs the following functions:

a) Update the group key $k_{\text{group-new}_{i}}$ similar to step 2.4.4

b) Verify that the received message $M_8$ matches the calculated $f(k_{\text{group-new}_{i}}, N_k \oplus c)$ using its own updated group key $k_{\text{group-new}_{i}}$; the nonce $N_k$ generated by $S_{\text{current}}$ and $c$

c) If $M_8$ is verified, update key $k_{i_{\text{new}}}$ as shared with $S_{\text{new}}$ in a similar way to step 2.4.4

d) $T_i \rightarrow R_{\text{current}}$: $M_{10}$ $T_i$ constructs a message $M_{10}$ and transfers it to $R_{\text{current}}$ as part of the proof for group ownership transfer, where $M_{10} = f(k_{i_{\text{new}}, N_k \oplus c}) \oplus f(k_{\text{group-new}_{i}}, N_k \oplus c)$. Then, $T_i$ updates $c$ by $c = c + 1$.

**Step 2.4.6:** $R_{\text{current}} \rightarrow TS$: $M_{11}$ $R_{\text{current}}$ constructs a group ownership transfer proof message $M_{11} = GT_{ij} = [f(k_{\text{group}} \oplus N_k \oplus c), f(k_{\text{group-new}_{j}}, N_k \oplus c), f(k_{\text{group-new}_{i}}, N_k \oplus c), f(k_{i_{\text{new}}}, N_k \oplus c), f(k_{j_{\text{new}}}, N_k \oplus c), f(k_{\text{group-new}_{j}}, N_k \oplus c), f(k_{\text{group-new}_{i}}, N_k \oplus c), f(k_{i_{\text{new}}}, N_k \oplus c), f(k_{j_{\text{new}}}, N_k \oplus c), f(k_{\text{group-new}_{j}}, N_k \oplus c), f(k_{\text{group-new}_{i}}, N_k \oplus c)]$. $GT_{ij}$ is then forwarded to $S_{\text{new}}, TS$, and $S_{\text{current}}$ for verification.

Figure 1 shows the phase one and phase two of the proposed protocol. Like [16], a timer is set for each tag in order to restrict the time that a valid proof can be generated. This can help prevent interleaving attacks where messages from different protocol runs are combined to construct a valid proof of simultaneous presence of multiple tags while those tags really were not present at the same time.

(3) **Phase Three:** Verification phase

At this stage, it is supposed that all the tags in the group have already updated their secret keys as set by the new owner. As the final step of a complete group ownership transfer process, $S_{\text{new}}$ conducts a challenge-response protocol with the group of tags as a whole (using a grouping-proofs protocol, e.g., [6, 17-18]) or with individual tags using a tag-reader authentication protocol (e.g., [19-22]). In the case that some tags’ secret keys cannot be verified (due to arbitrary failures in tag key updating or communication problems during the group ownership transfer process), or some tags’ proofs are missing (e.g., the tags’ timers are expired), the entire process must be aborted. The system can be rolled back by allowing $S_{\text{current}}$ to use its old keys to reset the tags (with the help of $TS$). Requiring that all
the tags update their keys in the same session ensures the atomic of a group ownership transfer (either all successful or nothing at all).

5. Analysis

5.1. Correctness analysis

The goal of the correctness analysis is to prove that a new secret key and a new group key as provided by the new owner $S_{new}$ have been successfully set on each tag in the group. Since our protocol is designed to ensure the atomic of group ownership transfer, we only prove that the new keys have been updated on one randomly selected tag in the group, say $T_i$. To do that, we prove that $T_i$ believes $S_{new}$ conveys the keys to the Trust Server $TS$, $TS$ conveys the keys to the current owner $S_{current}$, and finally $S_{new}$ conveys the keys to $T_i$.

The following proof is based on the GNY logic [23].

We want to show that after an execution of our protocol, $T_i$ would believe that the secret information it received comes from $S_{new}$ and the messages are fresh. To apply the GNY logic, we first translate some of the protocol steps to generic types below:

$$2.1) \quad S_{current} \lll E[k_{S_{current}S_{new}}, M_1] \implies S_{new} \equiv S_{current} \oplus E(k, .), \text{and} \quad S_{new} \lll E[k_{S_{current}S_{new}}, M_1] \implies S_{current} \equiv S_{current} \oplus E(k, .).$$

$$2.2) \quad T_i \lll E[k_{S_{current}TS}, M_2] \implies S_{current} \equiv T_i \ominus E(k, .), \text{and} \quad S_{current} \lll E[k_{S_{current}TS}, M_2] \implies T_i \equiv S_{current} \ominus E(k, .).$$

$$2.3) \quad S_{current} \lll E[k_{S_{current}TS}, M_3] \implies S_{current} \equiv E(k, .), \text{and} \quad S_{current} \lll E[k_{S_{current}TS}, M_3] \implies T_i \equiv S_{current} \ominus E(k, .).$$

$$2.4.1) \quad T_i \lll M_1 \implies R_{current} \equiv T_i \ominus E(k_{R_{current}}, M_2) \implies R_{current} \equiv T_i \ominus E(k_{R_{current}}, M_2).$$

The goals to prove are given below:

$$G_1) \quad T_i \equiv S_{new} \equiv \#(k_{i_{new}} \oplus k_{i_{mask}})$$

$$G_2) \quad T_i \equiv S_{new} \equiv \#(k_{i_{new}} \oplus k_{i_{mask}})$$

$$G_3) \quad T_i \equiv S_{new} \equiv \#(k_{group_{new}} \oplus k_{group_{mask}})$$

$$G_4) \quad T_i \equiv S_{new} \equiv \#(k_{group_{new}} \oplus k_{group_{mask}})$$

Goal $G_1$ says that $T_i$ believes $S_{new}$ conveys the message $k_{i_{new}} \oplus k_{i_{mask}}$. Goals $G_2)$, $G_3)$ and $G_4)$ can be interpreted similarly. If these goals are proved, then $T_i$ will believe that $k_{i_{new}}$ and $k_{group_{new}}$ are legitimate new keys set by $S_{new}$. To prove $G_1$ we need to prove the following two sub-goals:

$$G_1.1) \quad S_{current} \equiv S_{new} \equiv \#(k_{i_{new}} \oplus k_{i_{mask}})$$

$$G_1.2) \quad T_i \equiv R_{current} \equiv \#(k_{i_{new}} \oplus k_{i_{mask}})$$

$G_1.1)$ says that $S_{current}$ believes $S_{new}$ conveys the message $k_{i_{new}} \oplus k_{i_{mask}}$ to it and $G_1.2)$ says that $T_i$ believes $R_{current}$ conveys the message $k_{i_{new}} \oplus k_{i_{mask}}$ to it. Since no message contamination is possible during the message transmission without being detected, if the two sub-goals are proved, then goal $G_1)$ can be approved transitivity. Furthermore, according to GNY logical postulate rule I6, if entity $P$ believes that $Q$ once conveyed formula $X$ and $P$ believes that $X$ is fresh, then $P$ is entitled to believe that $Q$ possesses $X$. In our scenario, tag $T_i$ would believe that $S_{new}$ possesses $k_{i_{new}}$ and $k_{group_{new}}$.

To approve $G_1)$, two steps are necessary. In the first step, we apply the GNY logical postulate rule I1. Suppose that for $S_{current}$ all of the following conditions hold: (1) $S_{current}$ receives a formula $M_1$, which is encrypted with key $k_{S_{current}S_{new}}$ and marked with a not-originated-here symbol; (2) $S_{current}$ possesses $k_{S_{current}S_{new}}$; (3) $S_{current}$ believes $k_{S_{current}S_{new}}$ is a suitable secret for itself and $S_{new}$ (4) $S_{current}$ possesses $M_1$ is recognizable; and (5) $S_{current}$ believes $k_{S_{current}S_{new}}$ is fresh. Then $S_{current}$ is entitled to believe that (1) $S_{new}$ once conveyed $M_1$; (2) $S_{new}$ possesses $k_{S_{current}S_{new}}$. The proof is shown in Fig. 2 (a).

In the second step, we apply the GNY logical postulate rule P3. Suppose that $S_{current}$ possesses a formula $M_1 = \{ID_i || (k_{i_{new}} \oplus k_{i_{mask}}) || E[k_{S_{new}TS}, k_{i_{mask}}] || (k_{group_{new}} \oplus k_{group_{mask}}) || E[k_{S_{new}TS} k_{group_{mask}}]\}$, then it is capable of possessing any one of the concatenated components of that formula such as $k_{i_{new}} \oplus k_{i_{mask}}$. The proof is shown in Fig. 2 (b). Combining the above two steps, $G_1)$ is approved.

To approve $G_2)$, we revisit the message $M_2 = M_2 || M_3 || M_4$ sent by $R_{current}$ to tag $T_i$ in step 2.4.1. Consider $M_{i+1}$ which contains two messages $f(k_i N_{i}) \oplus (k_{i_{new}} \oplus k_{i_{mask}})$ and $h((k_{i_{new}} \oplus k_{i_{mask}} \oplus N_{i}))$. We let $M' = k_{i_{new}} \oplus k_{i_{mask}}$ and rewrite $f(k_i N_{i}) \oplus (k_{i_{new}} \oplus k_{i_{mask}})$ as $XOR(f(k_i, N_{i}), M')$. This can be interpreted that $M'$ is encrypted using an XOR encryption with the key it possesses, then it is considered to have been told the decrypted contents of that formula. The proof is shown in Fig. 2 (c).

If the following two conditions hold: $T_i$ applies $h(M' \oplus N_{i}) || k_i$ and the result matches the received $h((k_{i_{new}} \oplus k_{i_{mask}} \oplus N_{i})) || k_i$ in message $M_4$ and $T_i$ believes $S_{current}$ is the only entity to share the secret key $k_i$ with it, then $T_i$ is entitled to believe that $S_{current}$ conveys the message $k_{i_{new}} \oplus k_{i_{mask}}$.

Goals $G_2)$, $G_3)$ and $G_4)$ can be approved in a similar way. The only difference is that $TS$ has to be involved in the transitive proof for goal $G_2)$ and $G_3)$. Due to page limitation, we will not discuss the details.

5.2. Security and privacy analysis

Our protocol uses fresh nonce and shared secrets in both message exchange and reader-tag
authentication. It resists RFID security and privacy attacks such as tag information leakage, eavesdropping, tag impersonation, and replay attacks. Although we did not explicitly specify, our protocol can be extended to have each tag and the reader maintain both the current and old versions of secret keys. This will mitigate the problem of tag key desynchronization [6] and prevent denial of service attacks. Since this paper focuses on group ownership transfer, we next discuss the privacy features specific to RFID tag ownership transfer and RFID tag grouping-proofs. Table 3 gives the comparisons between our protocol, i.e., GOT, and the schemes described in Section 2 in term of ownership transfer characteristics.

New owner’s privacy is ensured in the proposed protocol. As we mentioned earlier, the new owner \( S_{\text{new}} \) chooses new secret keys at its discretion. In order to be secretly delivered to each tag, those new keys are masked using random numbers (in message \( M_j \)) which are set by \( S_{\text{new}} \) and can only be known to the trusted server (TS). Since the current owner \( S_{\text{current}} \) has no way to learn those masks, it cannot retrieve the new keys. This prevents the old owner from knowing the new keys of the tags after the tag ownership transfer. The protocol also protects old owner privacy. \( S_{\text{new}} \) only submits its proposed new secret keys at the beginning of a tag ownership transfer process and receives a proof at the end. Since \( S_{\text{new}} \) is not involved in any other in-between steps during the process where the tag keys are updated, it cannot learn anything about the previous keys of the tags. To support tag authorization recovery, we only need to switch the roles of \( S_{\text{new}} \) and \( S_{\text{current}} \) by letting the latter choose new keys and then running the protocol one more time.

**Table 3: Privacy of Ownership Transfer Protocols**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Old Ownership Privacy</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Authorization Recovery</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Free of Windowing Problem</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Group Ownership Transfer</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Our protocol supports the atomic of a group ownership transfer. One output of the protocol, i.e., a proof for a group ownership transfer, will indicate that a group of tags are present at the same time and their secret keys have been updated to share with the new owner. While any missing tag in a group can be detected as early as in **phase one**, any unsuccessful key update will be detected in **phase two** and **phase three**.

For any failure, the system is rolled back simply by allowing \( S_{\text{current}} \) and TS to reset the tag keys.

Particularly, our protocol closes the time window of dual ownership in any time period during an ownership transfer process when two owners possess the authentication secrets of a tag. The **windowing problem** exists if we let the old owner directly transfer all the relevant tag secret information to the new owner even via a secure channel. Theoretically, the two entities will both possess the tag authentication information immediately after the secret keys are passed by the old owner but before those keys are updated by the new owner.

Our protocol does not require that the backend server be available all the time, i.e., a fully-interactive mode of the grouping-proofs. **Phase one** of the proposed protocol can be conducted ahead of time and in a batch mode. Then **phase two** and **phase three** can be conducted even when the server is offline. Even when the backend server is not available, tags in a group can authenticate each other using the group key during a group ownership transfer process.

### 5.3. Performance analysis

While low-cost RFID tags are only capable of performing basic operations, the tag readers, servers, and TS have the abilities of applying standard cryptographic key encryption/decryption algorithms. Our protocol has modest computational requirements for RFID tags. Each tag only performs a few pseudorandom and bit-wise XOR functions as well as one hashing function in each protocol run (**phase two**). In terms of memory storage, each tag \( T_i \) only requires three \( I \)-bit non-volatile memory space to store \( k_s \), the key shared with the owner, \( k_{\text{group}} \) the key shared with other group members, and \( s_i \), the key shared with the trusted server. Note that a tag still needs other memory to store temporary variables (e.g., nonce and counter) in its computations. But, the required memory space is rather limited since only a hashing function and a few XOR and pseudorandom operations are carried out.

In terms of communication costs, only two messages are required for a tag to exchange with its group members in order to authenticate each other. Three messages for the initiator tag and two messages for other tags are required to authenticate with the reader in a protocol run. This level of communication loads is moderate for low-cost RFID tags.

### 6. Conclusion

The low cost and high convenience value of RFID tags have given them the potential for massive
deployment. With emerging applications in various multi-party environments, RFID tag group ownership transfer will become more and more important. No previous work has explicitly addressed the issue of transferring the ownership of a group of tags in one session: multiple tags are present simultaneously and either the entire group is transferred or no single tag should change hands. This paper presents a comprehensive protocol for RFID tag ownership transfer on a group basis. The protocol ensures new owner privacy, old owner privacy, and authorization recovery. Furthermore, it eliminates the possibility of dual ownership of an RFID tag in any time period during an ownership transfer process.

Acknowledgement
The author is thankful to Dr. Robert Herklotz for his support, which made this work possible.

References