Grouping-Proof Protocol for RFID Tags: Security Definition and Scalable Construction

Dang Nguyen Duc* and Kwangjo Kim*
*Department of Information and Communication Engineering, KAIST

Abstract

In this paper, we propose a grouping-proof protocol for RFID tags based on secret sharing. Our proposed protocol addresses the scalability issue of the previous protocols by removing the need for a RFID reader to relay messages from one tag to another tag. We also present a security model for a secure grouping-proof protocol and prove the security of our protocol.

I. Introduction

Grouping-proof protocols allow multiple RFID tags to be scanned at once such that their co-existence is guaranteed. One typical application of a grouping-proof protocol is to scan tags that are supposed to stay together. Juels [2] proposed the first protocol of this kind called yoking-proof. The protocol allows a RFID reader to produce a co-existence proof of two RFID tags. The proof can then be verified by a verifier which shares the secret keys with tags. Unfortunately, Saitoh and Sakurai [3] showed that yoking-proof is vulnerable to replay attack and proposed a timestamp-based version of yoking-proof to withstand the attack. A true grouping-proof protocol which supports simultaneous scanning of more than two tags was also proposed in [3]. Other improved variants of yoking-proof were also proposed in [4, 5, 6, 7]. In this paper, we first point out that all of the previous grouping-proof protocols in [2, 3, 4, 5, 6, 7] suffer from a scalability problem. We also point out the lack of proper security notion in previous works. In particular, no previous work addresses mafia fraud attack [1] which is simply a relay attack in which an attacker relays messages exchanged between a reader and tags. As noted in [8], all of grouping-proof protocols for RFID are insecure against this attack. Another issue when defining a security model for grouping-proof protocol is that the verifier has no knowledge of what or how many tags are actually in the communication range of a reader. Therefore, we cannot achieve security at all if a reader is allowed to behave maliciously in an arbitrary way. For example, a reader can deliberately avoid scanning some tags resulting in an invalid co-existence proof. In this paper, we present a security model for secure grouping-proof protocols which takes the
above issues into account. We then propose a scalable grouping-proof protocol for RFID by using a $(n, n)$-secret sharing scheme. We then prove the security of our protocol.

II. Related Work

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_i$</td>
<td>Secret key of tag $T_i$</td>
</tr>
<tr>
<td>$\text{MAC}_K[\cdot]$</td>
<td>Message authentication code with secret key $K$</td>
</tr>
<tr>
<td>$P$</td>
<td>A co-existence proof of multiple tags</td>
</tr>
<tr>
<td>$R$</td>
<td>Reader</td>
</tr>
<tr>
<td>$\text{SK}_K[\cdot]$</td>
<td>Symmetric encryption with secret key $K$</td>
</tr>
<tr>
<td>$TS$</td>
<td>Timestamp</td>
</tr>
<tr>
<td>$T_1, T_2, \ldots, T_n$</td>
<td>$n$ RFID tags</td>
</tr>
<tr>
<td>$\mathcal{D}$</td>
<td>Verifier (back-end database)</td>
</tr>
</tbody>
</table>

Table 1. Notations

Saitoh and Sakurai’s Grouping-Proof. In [5], the authors proposed a protocol which allows the simultaneous scanning of more than two tags. The protocol requires an additional entity called pallet tag. The pallet tag acts as a representative of all RFID tags that are in the same package with the pallet tag. A co-existence proof $P$ is also subject to timestamp verification by the verifier to prevent replay attack. The protocol is depicted in Fig. 2.

![Fig. 2. Saitoh and Sakurai’s Grouping-Proof](image)

III. Scalability Issues of Previous Grouping-Proof Protocols

The design of yoking-proof suffers from a serious scalability issue. The reason is that a reader needs to relay messages from one tag to another so that a tag can sign the random numbers that were generated by the other tags. As a result, if the reader wants to produce a co-existence proof of $n$ tags, it will need to relay $n(n-1)$ messages among $n$ tags. This same problem also appears in other variations of yoking-proof and in [2, 3, 4, 5, 6, 7]. The grouping-proof protocol by Saitoh and Sakurai does not use the same design of yoking-proof. However, it requires a pallet tag capable of performing symmetric
encryption. In addition, the reader still needs to relay \( n \) messages from all tags to the pallet tag in order to scan \( n \) tags at once.

**IV. Security Model for Grouping-Proof**

We now describe our security model for a secure grouping-proof protocol. First of all, we realize that for a grouping-proof for RFID tags protocol, the primary goal of an adversary is to inject some tags (possibly genuine) into a valid co-existence proof while the tags are not actually in the communication range of the reader. In addition, the adversary might also want remove some tags from a valid co-existence proof. It is also assumed that the reader can behave maliciously but does execute the protocol correctly. When reporting a co-existence proof to the verifier, a malicious reader may try to replace some tags in the proof with different tags, add a tag to the proof or remove a tag from the proof. We now define a set of oracles that provide information to the adversary:

- **The reader(\()\) oracle**: This oracle simulates a reader during a protocol session.

- **The corrupt-reader(\()\) oracle**: This oracle corrupts a reader and returns the current state of the reader. The adversary is also allowed to control the reader after this oracle is called.

- **The tag(\()\) oracle**: This oracle simulates a tag during a protocol session.

- **The verify(\()\) oracle**: This oracle takes a co-existence proof \( P \) as input and returns 1 if \( P \) is valid and 0 otherwise.

We now define the security notion for a secure grouping-proof protocol via the following game between a challenger and the adversary:

- The challenger sets the verifier and a reader and tags up.

- In the first phase of the game, the adversary collects information via 4 oracles: reader(\(),\) tag(\(),\) corrupt-reader(\())\) and verify(\()\). These oracles are simulated by the challenger.

- In the second phase of the game, the challenger gives the adversary a valid proof \( P \) of \( n \) tags as a challenge. The adversary's goal is to either remove a tag from \( P \) or add a new tag to \( P \) or replace a tag in \( P \) with a different one. In this phase, the adversary is also given access to the corrupt-reader(\()) oracle after the challenge proof \( P \) is constructed. However, the tag(\()) oracle is not provided to the adversary after the adversary has seen \( P \). This is to reflect our assumption that relay attack is not possible. The adversary should output a new proof \( P' \) which satisfies one of its goals.

- The adversary wins the game if verify(\( P' \)) returns 1. That is, \( P' \) is a valid co-existence proof.

**Definition 1.** A grouping-proof protocol is said to be secure if the winning probability of the adversary in the above game is negligible.
V. Our Proposed Grouping-Proof Protocol

We now propose our grouping-proof protocol for multiple RFID tags. To solve the scalability problem, we use a $(n, n)$-secret sharing scheme which allows one to split one secret $x$ into $n$ so-called shared secrets such that $x$ can only be reconstructed from the shared secrets only if all of $n$ shared secrets are provided. This property is used in our proposed protocol so that each tag can sign its own random number to prove its presence. The random numbers are shared secrets generated by a secret sharing scheme. If the original secret generated by a verifier can be recovered from signed shared secrets that are backscattered by tags, then the proof of co-existence of tags is verified. A $(n, n)$-secret sharing scheme can be implemented as follows:

- Given a secret $x$, a dealer chooses $(n-1)$ random numbers $y_1, y_2, ..., y_{n-1}$ as the first $(n-1)$ shared secrets.
- The last shared secret $y_n$ is computed by $y_n = x \oplus y_1 \oplus y_2 \oplus ... \oplus y_{n-1}$.

It is easy to see that it is impossible to recover $x$ without any of $y_1, y_2, ..., y_{n-1}$. In addition, for each randomly chosen $x$, a shared secret of $y_i$ is also random. This property is important to prevent replay attack as a shared secret is used as a challenge in our proposed protocol. The details of our grouping-proof protocol is depicted in Fig. 2.

Remark. Note that, it is important to stress that we do use timestamp in our protocol to prevent a malicious reader from abusing $x$ (i.e., the malicious reader can take $x$ and use shared secrets of $x$ on different tags at different locations and times). However, the way which timestamp is used in our protocol is very different from in previous protocols. More specifically, we do not use timestamp as a challenge to a tag. Instead, only the verifier maintains timestamp for each interrogation session. This allows us to leave “time-to-live of $x$” to the security model. Indeed, the fact that a co-existence proof must be received within the lifespan of $x$ fits in the assumption that a reader always executes the protocol correctly until reporting a proof to the verifier. Therefore, we can ignore the use of timestamp in the security proof of our protocol.

We now analyse the success probability of an adversary attacking our protocol in the following theorem.

Theorem 1. Let $\alpha$ be success probability of an adversary attacking the underlying MAC scheme. Let $\varepsilon$ be the success probability of an adversary that attacks our
proposed grouping-proof protocol, we have:

\[ e = O(a + 2^{\sqrt{a}}) \]

where \( l \) is the bit length of \( x \).

**Proof:** Due to lack of space, we give a sketch of proof in this paper. We distinguish three types of adversaries: Type-I adversary that replaces a tag in a proof \( P \) with another tag, Type-II adversary that adds a tag to a proof \( P \) and Type-III adversary that remove a tag from a proof \( P \). For each type of adversary, we consider two cases: a MAC \( m_e \) in a new proof \( P' \) has not been queried in the first phase of the adversary and otherwise. In the first case, the adversary is essentially a MAC forger. Therefore, its advantage is bounded by \( a \). In the latter case, the adversary's hope is that a shared secret is generated twice, once in the challenge proof and once in the querying phase. Clearly, this probability is bounded by \( 2^{\sqrt{a}} \).

As a result, we obtain the proof. \( \square \)

We compare performance of different grouping-proof protocols in Table 2.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Number of Relaying Messages</th>
<th>Cost of Generating Proof</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yoking-Proof</td>
<td>( n(n - 1) )</td>
<td>2( n(n - 1) ) MACs 1 Encryption</td>
</tr>
<tr>
<td>2nd Protocol [5]</td>
<td>( n )</td>
<td>( n ) MACs 1 Encryption</td>
</tr>
<tr>
<td>Protocol in [7]</td>
<td>( n(n - 1) )</td>
<td>2( n(n - 1) ) MACs 1 Encryption</td>
</tr>
<tr>
<td>1st Protocol [8]</td>
<td>( n(n - 1) )</td>
<td>2( n(n - 1) ) MACs 1 Encryption</td>
</tr>
<tr>
<td>Protocol in [10]</td>
<td>( n(n - 1) )</td>
<td>4( \text{exp}(n-1) f(.) ) Evaluations</td>
</tr>
<tr>
<td>Proposed Protocol</td>
<td>0</td>
<td>( n ) MACs</td>
</tr>
</tbody>
</table>

Table 2. Comparison Table

**VI. Conclusion**

In this paper, we present a grouping-proof for RFID tags protocol based on secret sharing. Our protocol solves the scalability issue of previous protocols by avoiding relaying messages among tags by a reader.

We also define a security model for a secure grouping-proof protocol and show that our proposed grouping-proof protocol satisfies the proposed security notion.

**References**


