Ninth IEEE International Symposium on Parallel and Distributed Processing with Applications Workshops

Untraceable and Serverless RFID Authentication and Search Protocols

Zeen Kim, Jangseong Kim, Kwangjo Kim
Dept. of Info. and Comm. Eng.
KAIST
Daejeon, Korea
Email: zeenkim,jskim.withkals,kkj@kaist.ac.kr

Imsung Choi
Samsung Electronics
Suwon, Korea
shaki@samsung.com

Taeshik Shon
Division of Info. and Comp. Eng.
Suwon, Korea
tsshon@ajou.ac.kr

***Abstract*—So far, conventional RFID protocols provide the security and privacy protections by utilizing the central database model where readers should maintain the persistent connection between the readers and the central database. Recently, severless RFID protocols [4], [5] have been proposed to provide more flexible RFID service by removing the need of this connection. In this paper, we first point out the tracing vulnerability of the existing serverless RFID protocols. To address this vulnerability, we suggest a novel method which generates a unique access list for each reader based on groups of tags and multiple pseudonyms. We then propose untraceable and serverless RFID authentication and search protocols with this method. In comparison with [4], our protocols provide more resilient protection to the tracing vulnerability. Moreover, our protocols show less computation overhead than [4].**

I. Introduction

Radio Frequency Identification (RFID) is a remote identification method for storing and retrieving data with cooperation of readers and tags. RFID technology has emerged as a promising technology to replace the bar-code technology [1], [2]. Compared with the bar-code technology [3], RFID technology has distinct advantages such as a unique identification and automation [6]. Although the advantages of deploying RFID technology are obvious, it is slowly adopted due to the concerns about security and privacy. Such concerns include the illegal tracing for tags and readers which violates the privacy of the reader and tag holders [7]. It is urgent to resolve the security and privacy problems of RFID technology because if these problems are not adequately solved, large scale deployment of RFID system is unlikely to happen.

Conventional work [11], [12], [13], [14] tried to address the security and privacy problems of the RFID technology by utilizing the central database model. This model consists of three entities: readers, tags, and a central database. When a reader authenticates a tag in the central database model, the reader first queries the tag and then forwards a reply of the tag to a central database. The central database authenticates the reader and verifies the reply. If these verifications are all positive, the central database sends the tag information to the reader. Otherwise, it does not. The central database approach provides the security and privacy protections, but it depends on a reliable and persistent connection between a reader and the central database.

This connection is infeasible in some situations due to the denial of service attack or the restrictive communication capability of the central database [4], [8]. Without this connection, a reader cannot obtain the tag information, even though the reader and tags are legitimate. A simple solution for this problem is to download the information which is required to authenticate tags onto readers in advance. However, unlike a static server, readers can be stolen or lost due to the portable and mobile nature of readers. If an adversary steals a reader, the adversary can obtain the information which is required to authenticate tags from the compromised reader. The adversary then can make fake tags with this information. Because a fake tag has the information of a legitimate tag, the reader cannot distinguish between the fake tag and the legitimate tag.

Recently, serverless RFID authentication protocols [4], [5] have been proposed to provide more flexible RFID service than conventional work based on the central database model. These protocols provide the security and privacy protections without assumption for a connection between a reader and the central database. In these protocols, each reader generates a unique access list for tags with a unique identity of the reader. Because this access list of readers does not include the secret information of tags, the adversary cannot make the fake tags even though he compromises a legitimate reader.

RFID authentication protocols allow readers to query a tag at one session. This operation is not proper in a situation where a reader finds a specific tag among multiple tags. For instance, in a large book store stocked with RFID embedded books, a customer wants to find a book among a large collection of books. RFID authentication protocols cannot support this desire efficiently, because the customer should check each book. For this reason, serverless RFID search protocols [4] were introduced to provide efficient service at this situation. In these search protocols, a reader broadcasts a search query to multiple tags. If the intended tag of the reader exists among them, it will reply to the reader. Due to the limited broadcast range of readers, we can determine the approximate locality of the intended tag by directing the reader at different locations, (*i.e.*, different book shelves).

The above serverless RFID authentication and search protocols tried to provide security and privacy protections against several attacks that include tracing for readers and tags.

However, these protocols still are vulnerable to tracing for readers and tags. In this paper, we review the serverless RFID protocols and point out the tracing vulnerability in the analysis. To address this vulnerability, we suggest a novel method to generate a unique access list for each reader with groups of tags and multiple pseudonyms. We then propose RFID authentication and search protocols based on this method. Our protocols provide more resilient protection to the tracing vulnerability for readers and tags compared with [4]. Moreover, our protocols show less computation overhead than [4].

Contribution

We make the following contributions in this paper: First, we point out the tracing vulnerability of the existing serverless RFID protocols. Second, we suggest a novel method to make a unique access list for each reader with groups of tags. Third, we propose untraceable and serverless RFID authentication and search protocols based on the method.

Organization

The remainder of this paper is organized as follows: The next section describes the security requirements of serverless RFID protocols. In Section III, we review existing serverless RFID protocols and analyze them with respect to security requirements. We propose authentication and search protocols based on the novel method to make a unique access list in Section IV. We analyze the security of our protocols and evaluate the performance of our protocols in Section V and Section VI, respectively. Finally, we conclude this paper in Section VII.

II. Security Requirements of Serverless RFID Protocols

The following security requirements of serverless RFID protocols are basically the same as those well defined in the conventional RFID protocols except protection against the physical attack.

A. Basic Privacy

Basic privacy means that $\mathcal{A}$ cannot obtain the tag information. If someone obtain the tag information and find out the contents of the tagged items without a legitimate reader, we consider that the basic privacy is violated. Severless RFID protocols should guarantee that only legitimate readers are able to identify tags and obtain the tag information.

B. Untraceability

Untraceability means that $\mathcal{A}$ cannot recognize readers and tags which he has already seen, at another time or in another place.

Tag Untraceability: When a reader requests authentication to a tag, the tag replies to the reader. $\mathcal{A}$ queries to a tag and then obtain a reply from the tag. If $\mathcal{A}$ can couple this reply with the reply already seen, we consider that tags are traceable. Serverless RFID protocols should protect tags from this traceability.

RFID search protocols have an additional untraceability problem. In RFID search protocols, the reader broadcasts a search query to multiple tags and the only intended tag might reply to this query. $\mathcal{A}$ can utilize this property to trace the tag by replaying the succeeded search query. RFID search protocols should provide a countermeasure against this traceability.

Reader Untraceability: The untraceability of readers is similar with the untraceability of tags. If $\mathcal{A}$ can link a reader to the reader previously seen by eavesdropping on communications of the reader, we consider that readers are traceable. Serverless RFID should protect readers form this traceability.

C. Cloning Attack

Cloning attack is an attack which cheats legitimate readers to believe that a fake tag is legitimate. To make the fake tag, $\mathcal{A}$ obtains a response by querying or eavesdropping and then makes a fake tag by placing the response on a empty tag. If the legitimate reader believes this fake tag is legitimates, $\mathcal{A}$ succeeds the attack. The serverless RFID protocols should protect the legitimate readers from this attack.

D. Physical Attack

Due to the mobile nature of readers, $\mathcal{A}$ can compromise a legitimate reader. With information from the compromised reader, $\mathcal{A}$ tries to launch several attacks. In this paper, we focus on that $\mathcal{A}$ with the compromised reader cannot make the fake tags to cheat legitimate readers.

III. Analysis of Existing Severless RFID Protocols

In this section, we review the existing serverless RFID prtocols [4], [5] and analyze them according to the above mentioned security requirements. We show that existing severless RFID protocols are vulnerable to tracing. For the sake of simplicity, we denote Tan *et al.* as TSL and Han *et al.* as HDC in this paper. Table I shows the notations used in this paper.

TABLE I
NOTATIONS

S	Trusted party, responsible for authenticating readers and deploying tags
R_i	RFID reader i
r_i	identity of RFID reader i
L_i	access list of RFID reader i
T_i	RFID tag i
T_{G_k}	tags in group k
id_i	identity of tag i for Reader
t_i	secret for RFID tag i
$h()$	one-way hash function
$\|\|$	concatenation
G_k	identity of group k
$S_{i,k}$	a pseudonym that reader i has corresponding group k
$\longrightarrow, \Longrightarrow$	unicast and broadcast, respectively
n_A	a random number made by entity A
$(a)_m$	the first m bits of a
e	number of entries in access list
g	number of groups
$\mathcal{A}$	adversary
$\|A\|$	the bit length of A

A. TSL's protocols and their analysis

TSL proposed a authentication protocol and three search protocols. Both the authentication and search protocols are based on a unique access list made in setup phase. we first describe how to make the access list and then explain the protocols.

Review

In setup phase, a reader R_i first obtains access list L_i from S. R_i receives L_i through secure channel that is shared by R_i and S. L_i is made as follows. S generates the identity of a tag T_j as $h(r_i||t_j)$ and then makes L_i combining the identity list of authorized tags and the corresponding tag information. Note that each reader has the unique access list because the identity of reader is unique.

To authenticate T_j, R_i first queries authentication request to T_j. T_j then makes a new random number n_{T_j} and answers it to R_i. Next, R_i makes a new random number n_{R_i} and sends it with r_i. T_j then computes id_j as $h(r_i||t_j)$ and sends $h(id_j)_m$ and $h(id_j||n_i||n_j) \oplus id_j$. R_i utilizes $h(id_j)_m$ to reduce the candidates for the search by matching it with pre-computed hash values of identities in L_i. Finally, R_i computes $h(id_{candidate}||n_i||n_j) \oplus id_{candidate}$ for each candidate and matches $h(id_j||n_i||n_j) \oplus id_j$ with them. If there is a match, the authentication is successful and R_i obtains the tag information of T_j.

TSL also proposed the three search protocols which protect tags from the tracing of search protocols mentioned in Section II. In the first search protocol, tags check whether a random number of the search query is recently used by matching the stored value. If it is right, tags refuse to reply. In the second search protocol, a reader sends $h(id_j)_m$ to tags and then the all tags that match it, will reply to confuse $\mathcal{A}$. Because multiple tags reply, $\mathcal{A}$ cannot identify the intended tag among them. In the third search protocol, tags that receive a search query will reply with some probability even if they are not the intended tag. Similar with the second search protocol, $\mathcal{A}$ cannot identify the intended tag because multiple tags reply.

Analysis

TSL's protocols satisfy the security requirements except untraceability. TSL's authentication protocol is vulnerable to tracing for both readers and tags. TSL's search protocols are vulnerable to tracing for readers.

In both TSL's authentication and search protocols, a reader should send its identity to tags. This identity is unique and is unchangeable until obtaining a new access list from S. If $\mathcal{A}$ eavesdrops on a communication of a reader and store the unique identity of the reader, he then can recognize the reader whenever the reader tries to authenticate or search a tag.

TSL's authentication protocol utilizes $h(id)_m$ to reduce the candidates for the search. $h(id)_m$ of a tag changes depend on a reader identity because $h(id)_m$ is made as $h(r_i||t)_m$. This property enables a following attack to trace a tag. If $\mathcal{A}$ repeatedly queries to a tag T_j with different reader identities, he then obtains a list of $h(id)_m$ responses for the different reader identities. With this list, $\mathcal{A}$ can recognize that a tag is T_j or not. For example, $\mathcal{A}$ repeatedly queries to a tag with the different reader identities. $\mathcal{A}$ then matches the replies of the tag with the responses of T_j. Because tags that match all responses of T_j are very few, $\mathcal{A}$ can determine whether this tag is T_j or not. $\mathcal{A}$ can make the responses as much as $2^{|r|}$, where $|r|$ is the bits of the reader identity. Whenever $\mathcal{A}$ has many responses of T_j, $\mathcal{A}$ can recognize T_j easily.

B. HDC's protocol and its analysis

Review

HDC proposed a authentication protocol [5]. HDC modified TSL's Authentication protocol to support mutual authentication. HDC's authentication protocol mostly inherits from TSL's authentication. For this reason, we do not describe HDC's authentication protocol in detail.

Analysis

HDC's authentication protocol has comparable security and privacy protections with TSL's authentication protocol except one additional vulnerability. Because tags send $h(h(id))_m$ which is enough long to match the only tag in the access list, at each session, $\mathcal{A}$ can trace tags easily than TSL's authentication protocol. Note that we exclude HDC's authentication protocol from comparison with our authentication protocol due to the similarity with TSL's authentication protocol.

IV. Proposed Protocols

In this section, we suggest a novel method to make a unique access list for each reader in setup phase and propose serverless RFID authentication and search protocols based on this method.

A. Setup Phase

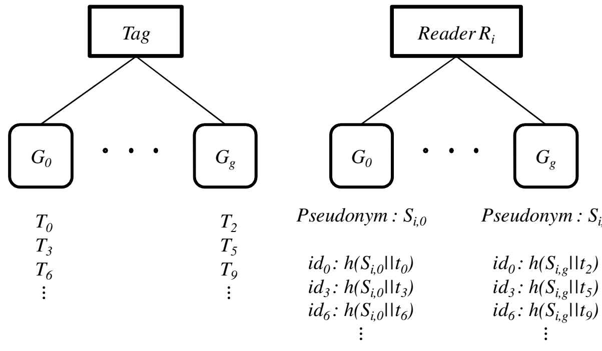
Tag
Reader R_i
G_0
G_g
G_0
G_g
T_0
T_3
T_6
T_2
T_5
T_9
Pseudonym : $S_{i,0}$
Pseudonym : $S_{i,g}$
id_0: $h(S_{i,0}||t_0)$
id_3: $h(S_{i,0}||t_3)$
id_6: $h(S_{i,0}||t_6)$
id_0: $h(S_{i,g}||t_2)$
id_3: $h(S_{i,g}||t_5)$
id_6: $h(S_{i,g}||t_9)$

Fig. 1. How to Make Access List

In setup phase, readers obtain a access list from S. For the sake of simplicity, we consider one session with a reader R_i and a tag T_j to explain our protocols. We assume that R_i and S can authenticate each other and share a secure channel to communicate.

We assume that entire tags are divided uniformly into several groups by S. Any tag therefore, is possessed into a group and stores its group identity. The way to manage groups will be discussed later.

When R_i requests its access list L_i, S first makes g random values where g is the number of tag groups. Next, S assigns these values as pseudonyms for groups of R_i. S then generates the tag identity for R_i by utilizing the assigned pseudonyms and the secret of tags. For example, in Fig 1, G_0 consists of T_0, T_3, T_6, *etc.* S then makes the tag identities as $h(S_{i,0}||t_0)$, $h(S_{i,0}||t_3)$, $h(S_{i,0}||t_6)$, *etc.* After generation of the tag identities for R_i, S generates L_i by combining the identities with the corresponding tag information. S then sends L_i and pseudonyms to R_i.

Note that R_i does not know the secret information of tags t. R_i only knows the outcome of $h(pseduonyms||t)$ which cannot be utilized to make fake tags. We assume that S assigns unique pseudonyms for each reader. Therefore, each reader has a unique access list. We also assume that the S cannot be compromised, and all authenticated readers are trusted. They will not reveal their access list to anyone else.

B. Authentication Protocol

1) $R_i \longrightarrow T_j$: Authentication Request
2) $R_i \longleftarrow T_j$: n_{T_j}, G_k
3) R_i : Generates n_i and computes $h(id||n_{R_i}||n_{T_j})$ for all tags in G_k
4) $R_i \longrightarrow T_j$: $n_{R_i}, S_{i,k}$
5) $R_i \longleftarrow T_j$: $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$
6) R_i : Matches hashed values in G_k with $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$

First, R_i queries authentication request to T_j. After receiving this request, T_j generates a random number n_{T_j}, and sends n_{T_j} and G_k to R_i. R_i then generates a random number n_{R_i} and sends n_{R_i} and $S_{i,k}$ to T_j. Simultaneously, R_i computes $h(id||n_{R_i}||n_{T_j})$ for all tags in G_k in advance. Next, T_j computes id_j as $h(S_{i,k}||t_j)$. T_j also computes $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$. T_j then sends it to R_i.

After receiving $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$, R_i matches it with hashed values. If there is a match with $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$ and R_i then obtains the tag information of T_j. Otherwise, there is no entry in G_k that matches $h(id_j||n_{R_i}||n_{T_j})$. In this case, the tag T_j might be a fake tag, since it is unable to generate a correct id_j. It might be a tag that is not authorized to access for R_i, thus not appearing in L_i. Since the different random numbers n_{R_i} and n_{T_j} will be used, $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$ will changes at each session.

C. Search Protocol

```
1) $R_i \Longrightarrow T*$ : $G_k, S_{i,k}, n_{R_i}$
2) If group identity = $G_k$ :
3)     $R_i \longleftarrow T_j$ : $n_{T_j}, h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$
4)     $R_i \longleftarrow T_{G_k}*$ : $n_{T_{G_k}}, h\left(h(S_{i,k}||t_{G_k})||n_{R_i}||n_{T_{G_k}}\right)$
5) Else :
6)     T do not reply
7) $R_i$ : Computes $h\left(id_j||n_{R_i}||n_{T_{G_k}}\right)$ for each reply and matches received values with them
```

When R_i searches T_j among a large collection of tags, R_i first finds out the group id of T_j, G_k in L_i. Next, R_i generates n_{R_i} and queries search request with G_k, $S_{i,k}$, and n_{R_i}. After receiving this search request, tags check that their group identity is G_k. If the group identity is G_k, tags T_{G_k} will reply to this search query. Otherwise, the tags do not reply. If the intended tag T_j exist among T_{G_k}, T_j makes n_{T_j}, and sends n_{T_j} and $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$ to R_i. Other T_{G_k} also generate $n_{T_{G_k}}$ and compute $h\left(h(S_{i,k}||t_{G_k})||n_{R_i}||n_{T_{G_k}}\right)$, and send them to R_i.

After receiving these messages, R_i computes the hash values $h\left(id_j||n_{R_i}||n_{T_{G_k}}\right)$ for each reply. R_i then matches them with $h\left(h(S_{i,k}||t_{G_k})||n_{R_i}||n_{T_{G_k}}\right)$ for corresponding $n_{T_{G_k}}$. If there is a match, R_i knows that T_j is among T_{G_k}. Otherwise, there is no T_j among T_{G_k}.

D. Management of Group

How to manage groups is the important issue in our protocols. In our authentication protocol, tags should send their group identity to a querying reader. In our search protocol, tags also reply to a search query for a specific group. Thus, anyone can know the group identity of tags by querying or eavesdropping. $\mathcal{A}$ tries to utilize the group identity for tracing. For example, $\mathcal{A}$ knows that a person has a tag T_j of which a group identity is G_k. T_j will answer G_k at each query. $\mathcal{A}$ can trace the person if there is no tag of which a group identity is G_k around him.

To protect this kind of tag tracing, one solution is to group all tags equally over entire service area. It can be achievable easily. When S deploys tags, S assigns identities of whole groups to the tags in rotation. $\mathcal{A}$ then cannot utilize group identity of tags for tracing, because multiple tags that have same group identity exist at a position. For example, when $\mathcal{A}$ tries to trace a tag T_j by querying, multiple tags will answer G_k. Thus $\mathcal{A}$ cannot identify T_j among them.

In addition to it, to determine the number of groups is also considered. There is a trade-off between the tag anonymity and the reader anonymity based on the number of groups. If the number of groups is few, the tag anonymity is well satisfied because many tags exist in one group, while the reader anonymity can be violated because readers can use few pseudonyms which is same with the number of groups. If the number of groups is large, the reader anonymity is well satisfied, but the tag anonymity will be violated because the number of one group is small.

To determine the number of groups is depend on the taste of S. If S wants to guarantee the tag anonymity well, S will employ small groups. When the high reader anonymity is required, S will use large number of groups. In our protocols, we assume that S employs the proper number of groups to satisfy both the tag anonymity and the reader anonymity adequately.

V. Security Analysis

In this section, we analyze our protocols depend on security requirements mentioned in Section II. For each requirement, we describe the attack and explain how the our protocols defend against the attack. We first analyze our authentication protocol and then analyze our search protocol. We denote a legitimate reader and tag as R_i and T_j respectively. T_j has G_k as its group identity. A fake tag that impersonate T_j is depicted as $\hat{T}_j$.

A. Authentication Protocol

Basic Privacy: In the basic privacy attack, $\mathcal{A}$ has list of valuable products and $\mathcal{A}$ tries to find these tags by querying every tag in service area. We assume that $\mathcal{A}$ does not possess a legitimate reader. In authentication protocol, when a reader R_i queries T_j, T_j replies new response $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$ at each session. $\mathcal{A}$ therefore, cannot identify which tag is the specific tag without t_j or a legitimate reader. Only legitimate readers can identify tags.

Tag Untraceability: Under this attack, $\mathcal{A}$ tries to trace a tag T_j. We assume that $\mathcal{A}$ does not have a legitimate reader and can always query to T_j. In our authentication protocol, T_j sends G_k for each query and $\mathcal{A}$ can try to utilize it to trace T_j. We assume that *S* groups tags uniformly over entire area. Therefore, multiple tags of which the group identity is G_k can exist at same position. $\mathcal{A}$ then cannot identify T_j among multiple tags that send G_k as the reply to a query. T_j also sends $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$, but it cannot be utilized for tracing because it is generated newly at each session. Our authentication protocol is also secure against the attack described in Section III because T_j answers same G_k to each authentication request.

Reader Untraceability: Under this attack, $\mathcal{A}$ recognize a reader by its unique identity. We cannot provide complete protection against this attack because readers should send their identity to tags. However, our authentication protocol is more resilient to tracing for readers compared with [4], [5], because our authentication protocol utilizes the multiple pseudonyms. $\mathcal{A}$ should know all pseudonyms of a reader to trace the reader completely.

Cloning Attack: In the cloning attack [9], $\mathcal{A}$ first queries T_j and obtains a response. $\mathcal{A}$ then makes a fake tag $\hat{T}_j$ by storing the response. Next, $\mathcal{A}$ attempts to pass off $\hat{T}_j$ as legitimate, $\mathcal{A}$ succeeds if R_i believes that $\hat{T}_j$ is T_j.

In our authentication protocol, T_j replies newly generated $h\left(h(S_{i,k}||t_j)||n_{R_i}||n_{T_j}\right)$ at each session. R_i generates a new random number n_{R_i} and $\mathcal{A}$ cannot predict it. Therefore, $\mathcal{A}$ then cannot create a $\hat{T}_j$ that cheats legitimate readers.

Physical Attack: Under this attack, we assume that $\mathcal{A}$ compromises R_i physically. For physical attack, we only concern an attack that $\mathcal{A}$ tries to make fake tags to cheat legitimate readers. We assume that readers do not have any tamper-proof device.

If $\mathcal{A}$ compromises R_i, he obtains L_i. L_i contains the identity list and the corresponding tag information, but does not contains the secret information of tags. $\mathcal{A}$ can authenticate tags of L_i and obtains the tag information, but $\mathcal{A}$ cannot generate $\hat{T}$ to cheat other readers because readers have a unique access list.

B. Search Protocol

Security analysis of authentication protocol can directly applies to our search protocol but an additional problem describe in Section II. Only the intended tag can reply to a search query of a reader. $\mathcal{A}$ then can recognize this tag by replying the search query. Thus, the countermeasure for this problem is mandatory in search protocols.

We assume that *S* groups the entire tags uniformly over entire service area. Therefore, multiple tags of all groups exist at a position. When R_i wants to find T_j of which group identity is G_k, R_i broadcasts search query with G_k, $S_{i,k}$, and n_{R_i}. There are multiple tags of which group identity is G_k among tags which receive search query, and these tags then reply to search query. $\mathcal{A}$ cannot identify which tag is the intended tag among them. Only R_i can know which tag is the intended tag by verifying replies.

VI. Performance Evaluation

In this section, we evaluate the performance of our protocols compared with TSL's protocols [4]. The evaluation metrics are communication cost and the number of hash operations. Note that we consider a session to evaluate. Table II depicts the overall evaluation comparison. η and θ mean tags that do not reply and tags that reply among tags that receive a search query, respectively. γ denotes the number of replying tags for a search query. We assume that the bits of random numbers and request messages are $|n|$, and the bits of pseudonyms of readers, the group identity, and the identity of tags are $|id|$.

A. Authentication Protocol

The overall communication costs of our authentication protocol are similar with the overall communication costs of TSL's authentication protocol. In our authentication protocol, a reader sends a request message, a random number, and a pseudonym at each session. The communication cost of the reader is $2{\cdot}|n| + |id|$ for one session. This communication cost is same with the communication cost of reader of TSL's authentication protocol. A tag sends a random number, its group identity, and one hash value. The communication cost of the tag is $|n| + 2{\cdot}|id|$. It is similar with $|n| + |id| + |m|$ which is communication cost of reader in TSL'authentication.

At each session, a tag computes two hash operations in our protocols, while a tag should compute three hash operations in TSL' authentication protocol. A reader computes hash values as much as the number of tags in one group, $e \times g^{-1}$ and matches the reply of the tag with these hash results. The overhead of hash operations and searching can be different depend on the parameters. If we assume that $e \times g^{-1}$ and $e \times 2^{-m}$ are same, our authentication protocol can provide less computation overhead and faster operation. In our authentication protocol, R_i can start to compute the hash computations quickly than TSL's authentication protocol (*i.e.*, after receiving G_k from T_j

TABLE II
PERFORMANCE EVALUATION

	Our Authentication	Authentication [4]	Our Search	Search 1 [4]	Search 2 [4]	Search 3 [4]
Communication cost of reader	$2\cdot\lvert n\rvert + \lvert id\rvert$	$2\cdot\lvert n\rvert + \lvert id\rvert$	$\lvert n\rvert + 2\cdot\lvert id\rvert$	$\lvert n\rvert + \lvert id\rvert + \lvert m\rvert$	$\lvert n\rvert + 2\cdot\lvert id\rvert$	$\lvert n\rvert + 2\cdot\lvert id\rvert$
Communication cost of tag	$\lvert n\rvert + 2\cdot\lvert id\rvert$	$\lvert n\rvert + \lvert id\rvert + \lvert m\rvert$	$\lvert n\rvert + \lvert id\rvert$	$\lvert n\rvert + \lvert id\rvert$	$\lvert n\rvert + \lvert id\rvert$	$\lvert n\rvert + \lvert id\rvert$
Hash operation of reader	$e \times g^{-1}$	$e \times 2^{-m}$	γ	$\gamma + 1$	$\gamma + 1$	$\gamma + 1$
Hash operation of tag	2	3	$\eta : 0, \theta : 2$	$\eta : 2, \theta : 3$	$\eta : 1, \theta : 2$	$\eta : 2, \theta : 3$

and generating n_{R_i}, R_i can compute hash values), while TSL's authentication protocol should wait until receiving $h(id_j)_m$. Moreover, TSL's authentication protocol should search the candidates to compute hash value by matching $h(id_j)_m$, while our authentication protocol can compute hash values after knowing the group identity of T_j.

B. Search Protocol

Our search protocol has almost same communications cost with TSL's three search protocols [4]. In the communication cost of the reader, our search protocols same with TSL's second and the third search protocols as $|n| + 2 \cdot |id|$. This communication cost is not much bigger than the communication cost of TSL's the first search protocol, $|n| + |id| + |m|$. In all search protocols, the communication cost of the tag is same as $|n| + |id|$.

For the number of hash operations, our search protocol outperforms TSL's three search protocols. A reader computes one less hash operations than them. After receiving search query, tags check that the group identity in query is their group identity. If not, the tags will not compute any hash operation and sleep, while in TSL's three search protocols, all tags should compute one or two hash operations to confirm that search query is aim at itself.

VII. CONCLUSION

In this paper, we have analyzed the existing serverless RFID protocols [4], [5] and then have pointed out the tracing vulnerability in the analysis. To address this traceability, we have suggested a novel method to generate a unique access list for each reader by utilizing groups of tags and multiple pseudonyms of readers. We then have proposed serverless RFID authentication and search protocols based this method. Our protocols provide more resilient protection to tracing for readers and tags compared with the existing serverless RFID protocols. In performance comparison with TSL's protocols [4], our authentication protocol can provide less computation overhead. Our search protocol outperforms TLS's three search protocols in the overhead of hash computation.

REFERENCES

[1] M.R. Rieback, B. Crispo, and A. S. Tanenbaum, "The evolution of RFID security," *IEEE PerCom'06*, pp. 62-69, January-March 2006.

[2] C.C. Tan and Q. Li, "A robust and secure RFID-based pedigree system (short paper)," *ICICS*, 2006.

[3] Advanced Barcode Technology, http://bar-codes.com.

[4] C. C. Tan, B. Sheng, and Q. Li, "Secure and Severless RFID Authentication and Search Protocols," *IEEE Transaction on Wireless Communication*, vol. 7, no. 4, April 2008.

[5] S. Han, T. S. Dillon, and E. Chang, "Anonymous Mutual Authentication Protocol for RFID Tag Without Back-End Database," *MSN 2007*, LNCS 4846, pp. 623-632, 2007.

[6] A. Juels, "RFID security and privacy: A research survey," Manuscript, 2005.

[7] G. Avoine, "Radio frequency identification: adversary model and attacks on exisiting protocols", *Technical Report LASEC-REPORT-2005-001*, EPFL, Lausanne, Switzerland, September 2005.

[8] J. Kang and D. Nyang, "RFID authentication protocol with strong resistance against traceability and denial of service attacks," *ESAS 2005*, LNCS 3813, pp. 164-175, 2005.

[9] A. Jules, "Strengthening EPC tags against cloning," *WiSe'05*, 2005.

[10] D. Molnar and D. Wagner, "Privacy and secuirty in library RFID: Issues, practices, and architectures," *CCS'04*, 2004.

[11] T. Dimitriou, "A lightweight RFID protocol to protect against traceablity and cloning attcks," *SecureComm*, 2005.

[12] S. M. Lee, Y. J. Hwang, D. H. Lee, and J. I. L. Lim, "Efficeint authentication for low-cost RFID systems," *ICCSA*, 2005.

[13] M. Ohkubo, K, Suzuki, and S. Kinoshita, "Cryptographic approach to "privacy-friendly" tags," *RFID Privacy Workshop*, 2003.

[14] G. Tsudik, "YA-TRAP: Yet another trivial RFID authentication protocol," *PerCom*, 2006.