View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Open Repository and Bibliography - Luxembourg

A Scalable RFID Authentication Protocol Supporting Ownership Transfer and Controlled Delegation

Albert Fernàndez-Mir[1], Rolando Trujillo-Rasua[1], Jordi Castellà-Roca[1], and Josep Domingo-Ferrer[1]

Universitat Rovira i Virgili
UNESCO Chair in Data Privacy
Department of Computer Engineering and Mathematics
Av. Països Catalans 26,
E-43007 Tarragona, Catalonia
albert.fernandez,rolando.trujillo,jordi.castella,josep.domingo@urv.cat

Abstract. RFID systems allow fast and automatic identification of RFID tags through a wireless channel. Information on product items like name, model, purpose, expiration date, etc., can be easily stored and retrieved from RFID tags attached to items. That is why, in the near future, RFID tags can be an active part of our everyday life when interacting with items around us. Frequently, such items may change hands during their life-cycle. Therefore, beyond RFID identification protocols, there is a need for secure and private ownership transfer protocols in RFID systems. To ensure privacy to tag owners, the keys of tags are usually updated during the ownership transfer process. However, none of the previous proposals takes advantage of this property to improve the system scalability. To the best of our knowledge, we propose the first RFID identification protocol supporting ownership transfer that is secure, private and scalable. Furthermore, our proposal achieves other valuable properties related to ownership transfer, such as controlled delegation and decentralization.

Key words: RFID, Identification, Security, Privacy, Scalability, Ownership transfer, Controlled delegation

1 Introduction

A basic RFID scheme consists of a reader that uses radio waves in order to identify several small devices named *RFID tags*. This technology is gaining more and more momentum for identification, because it does not need physical or visual contact to identify tags. A few years ago, Gillette announced their plans of purchasing 500 million RFID tags for inventory control in their supply chain. RFID systems are not only useful to identify single items, but also to identify items in batches that logically should be placed together, *e.g.* a razor and a razor blade into a box.

A good use of the RFID technology by a company may enable a positive return of the investments in a short time [7]. However, although the price of a tag can be small, the fact that companies should often buy millions of these devices ties down their price to be not higher than 5 dollar cents [13]. That is why low-cost tags are, in general, more suitable for large-scale systems.

Even though this technology offers important advantages to the users, it also poses serious privacy and security risks to them. Considering that RFID tags respond to any reader query without the tag holder being aware of it, an adversary may be able to create a profile of a user by just reading the data of the tags in the user's possession. On the other hand, security measures must be carefully adopted because simply eavesdropping the communication channel between tags and readers could be enough to spoof a tag's identity. For example, if a tag attached to a Cohiba cigar box sends its identifier in plaintext, this identifier can be easily cloned in order to fake this expensive product in an inventory. Thus, it is necessary to develop secure schemes that prevent attackers from misusing the information managed in RFID systems. Notice that this point is earnestly suggested by the European Union in [15]. However, low-cost tags are very constrained devices that can only perform basic and simple cryptographic operations. Asymmetric-key cryptography is considered too expensive for low-cost tags; symmetric-key cryptography is more suitable for resource-constrained devices. Although several identification protocols using symmetric-key cryptography have been proposed for RFID systems, combining privacy and scalability through resource-constrained devices is still an issue [1].

In addition to the security and privacy problems, key management becomes an issue when the owner of a tagged item changes. Let us consider a manufacturer distributing RFID-tagged products to the point of sale. Later, these products are sold to buyers who can resell them to other buyers. In order to track RFID-tagged products across buyers, each reader could connect to the central server that manages all the tags' keys. However, this solution does not only cause bottlenecks and overloads on the server side, but also causes privacy issues when the product changes hands. The opposite solution is to share the keys of the tags among the different owners. But, with this scheme, privacy issues arise because previous and future owners of a tag are able to identify it even when it is not in their possession. Indeed, the tag key and secret information must be transferred from previous owners to new owners ensuring the security and privacy of past and future tag identifications *ownership transfer* [12]).

In some special cases, such as for after-sale service of an RFID-tagged object, the previous owner of a tag might need to temporarily recover the means of interacting with it. For instance, this happens when the buyer goes back to the seller to have a guaranteed appliance repaired. In this case, the current owner of a tag should be able to transfer its identification rights over a tag to another reader and to recover the exclusive right of identifying this tag at any moment. If the reader to whom the identification rights over a tag were transferred is a previous owner of this tag, the process is called *authorization recovery* [6]), otherwise the process is called *controlled delegation* [17].

1.1 Contribution and plan of this article

Preserving privacy among owners is a challenging task when the ownership of tagged products is transferred between them. Ownership transfer, authorization recovery and controlled delegation schemes should be designed to be secure and private not only against external adversaries, but also against previous owners. In this paper, to the best of our knowledge, we propose the first RFID identification protocol that is secure, private, scalable and able to perform ownership transfer, authorization recovery and controlled delegation.

The rest of this article is organized as follows. In Section 2 we summarize the current literature in the field of ownership transfer and put our contribution in context. In Section 3 we describe our proposal and in Section 4 we analyze its security. Section 5 contains conclusions and suggestions for future work.

2 Related work

A great number of authentication/identification protocols for RFID systems have been proposed in the literature [9]. However, just a few of them have been extended to support ownership transfer, authorization recovery or controlled delegation. When the RFID system supports ownership transfer, the tags can be used by different owners, so they have a longer life cycle. Authorization recovery and controlled delegation are properties especially designed for after-sale and maintenance service.

In identification/authentication protocols, RFID systems typically use a shared key between the tag and the owner's reader. Since the shared key is only known by the owner's reader, this reader alone can complete the tag identification successfully. In ownership transfer protocols there are two players, the current owner and the future owner. Before the protocol, only the current owner can identify the tag; after the protocol is completed, only the new owner can identify the tag successfully. This process is repeated every time the owner of the tag changes. So far, two assumptions have been extensively used in order to achieve the property mentioned above:

- **Centralized scheme:** There exists a Trusted Third Party (TTP) which every entity (owner) trusts. It is assumed that entities can establish a secure communication.
- **Isolated environment:** There exists a secure environment, so that each entity can execute the protocol with the tag without being eavesdropped by an adversary.

Both assumptions make sense depending on the application. A centralized scheme might be used in inventory control or supply-chain management applications, where all tags are identified by readers belonging to the same company. However, when a user buys a product, she does not necessarily trust anyone. Hence, a centralized scheme is not suitable. In this case, it is assumed that the user can go to an isolated environment (*e.g.* the user's home), where he can change the tag key without being eavesdropped by an adversary.

2.1 Ownership transfer model

An ownership transfer protocol allows transfering the rights over a tag T from the current owner to the new owner in a secure and private way. According to [14, 17], there are three different roles (entities) in an ownership transfer protocol:

- *The previous owner.* In the past, this entity had the ability to identify or track T as much as he wanted, but now she cannot do these operations any more.
- *The current owner.* At present, only this entity can identify and track T.
- *The new owner.* She cannot identify or track T at the beginning of the protocol . When the protocol finishes, T can only be identified or tracked by this entity.

When it is required to transfer the ownership of T, the current owner and the new owner run an ownership transfer protocol on T. The secrets and data stored in T are transferred from the current owner to the new owner. Thus, roles change. The current owner of T becomes the previous owner of T, while the new owner of T becomes the current owner of T.

The ownership transfer process motivates the requirements of authorization recovery and controlled delegation mentioned in the introduction. Figure 1 shows the players that can be involved in an ownership transfer protocol.

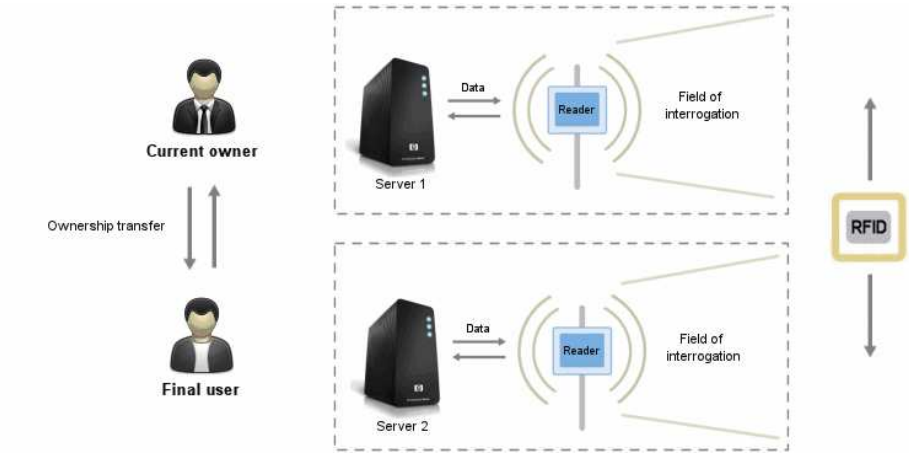

Fig. 1. A generic scenario for ownership transfer

2.2 Previous proposals

The first solution to the ownership transfer problem based on a centralized scheme was proposed by Saito *et al.* in [12]. In this protocol, tags receive encrypted messages using the TTP key (K_{TTP}). Since all tags know K_{TTP}, they

are able to correctly decrypt the messages coming from the trusted third party. However, tags are not tamper-resistant and, hence, tampering just one tag is enough to break the security of the whole system by finding K_{TTP}. Saito *et al.* [12] also consider the case where a centralized scheme cannot be used. In this case, they assume that the backward channel (from tag to reader) is unreadable by adversaries. Therefore, a secure communication between tags and readers is achieved by the reader encrypting messages using a nonce sent by the reader to the tag that is refreshed at each session. However, although the range of the backward channel is shorter than the range of the forward channel from reader to tag, in general the backward channel cannot be considered as unreadable by adversaries.

Another centralized scheme for ownership transfer and controlled delegation is proposed by Molnar *et al.* in [10]. They use a centralized trusted center named TC to manage tag keys in a tree structure. The delegation property is achieved when the trusted center gives a subtree of pseudonyms to a reader. Using this subtree of pseudonyms, a reader is able to identify a tag q times where q is the number of leaves of this subtree. The keys shared by several tags are a protocol weakness that reduces security. Therefore, the privacy of the whole system decreases quickly when more tags are compromised [2].

Unlike Molnar *et al.*'s protocol [10], other proposals use a counter into the non-volatile memory of tags in order to control how many identifications can be done after the execution of a controlled delegation protocol. The basic idea is that tags give different responses depending on whether the counter c is less than some threshold c_{max} or equals c_{max}. As shown by Fouladgar and Afifi [6], while $c < c_{max}$ the tag responds using a key known just by the trusted center and by the readers (the trusted center gave rights to the readers previously). Once the counter reaches its maximum value, the tag encrypts its responses using a key known just by the trusted center and hence, the tag identification is only possible through the trusted server. However, using a counter on the tag side raises two main issues: i) the counter must be in the non-volatile memory of the tag; ii) giving rights to more than one reader causes that tag keys to be shared by several readers. Notice that a reader that knows the identification key of some tag is able to know where and when this tag is being identified by any other reader.

Schemes based on a centralized and trusted third party require that readers be online at each execution of the ownership transfer protocol. Further, some of the parties involved in the ownership transfer process may not actually trust the TTP. There exist some proposals without a TTP. To the best of our knowledge, the first decentralized protocol relying on the assumption that owners are able to change the tag key in an isolated environment was proposed by Yoon and Yoo [16]. However, this protocol has security vulnerabilities well described in [8]. Under the same assumption, Chen *et al.* [3] proposed an improvement of Osaka *et al.*'s scheme [11] through authentication of the reader by the tag before changing the tag key. Although this protocol guarantees a successful ownership transfer

process, a previous owner can neither check whether a tag was in its possession nor can temporarily recover the means of interacting with a tag.

The property of authorization recovery [14] allows a previous owner of a tag to identify it again. This property is considered a particular case of delegation and could be achieved simply restoring the key of the previous owner into the tag. For instance, Dimitriou [4] proposes a simple *tag release* command that restores the manufacture key of the tag. However, this solution gives authorization recovery rights just to the tag manufacturer. Song *et al.* [14] solve this drawback in their proposal. Each tag owner remembers the key of the previous tag owner. Hence, updating the tag key to the key used by the previous owner is enough for authorization recovery. Although both protocols [14, 4] do not support controlled delegation, they have the advantage of not needing a trusted third party. The recent work in ownership transfer [17] shows that ownership transfer and controlled delegation are possible in a decentralized scheme. The authors propose a scheme that achieves all the desired properties defined so far in ownership transfer. However, their scheme can be used only in those identification/authentication protocols where: i) keys are not updated; or ii) keys are updated using Song *et al.*'s protocol [14]. Such a constraint has severe effects on the scalability of the identification process. Note that, although most identification protocols with key updating are not private against active adversaries, they can scale better than protocols without key updating.

In this paper, a new decentralized identification protocol with ownership transfer and controlled delegation is proposed. On the reader's side, tag identification is scalable in the number of tags. Once the tag has been identified, its identification key is updated. This property is used in the ownership transfer process in order to update the owner's key. Similarly to the proposals [10] and [14], the protocol has the advantage of supporting controlled delegation without needing a counter on the tag's side. Also, although it is based on synchronization unlike previous proposals [17, 14, 4], our protocol is still secure and private against active adversaries.

3 Our protocol

As discussed in Section 2, there exist several secure and private RFID identification protocols with ownership transfer [3, 4, 6, 10–12, 14, 16, 17]. However, just a few of them can deal with controlled delegation [4, 6, 14, 17]. Besides, none of these protocols leverages the updating phase performed during the ownership transfer process in order to efficiently identify the tags.

We propose an RFID identification protocol that is efficient both at the tag and at the server sides. Our protocol is based on the synchronization between the server and the tag, but even when these are desynchronized, it still can be scalable and resistant against denial-of-service attacks. The protocol has been designed to guarantee that an attacker cannot distinguish whether a tag is synchronized or not. This fact adds one more degree of difficulty for an attacker who wants to trace a tag or mount a denial-of-service attack. Together with

the above mentioned features, our proposal supports ownership transfer and controlled delegation in a decentralized scheme. The result is a secure, private and scalable RFID identification protocol that supports ownership transfer and controlled delegation.

Our proposal is partially based on the Fernández-Mir *et al.* protocol [5] and consists of five phases: initialization, synchronized identification, desynchronized identification, update and ownership transfer. The update phase is executed after each successful identification. If the tag cannot be updated and the system is desynchronized, it can still be identified by a desynchronized identification phase that preserves all the security and privacy properties. Table 1 summarizes the notation used to describe the new proposal.

Table 1. Notation used in the protocol

R	Reader
T	Tag
id	Tag identifier
ik	Identification key
uk	Update key
r_i	Random number
$h()$	One-way hash function
$h_k()$	Keyed hash function (HMAC)
$PRNG$	Pseudo-random number generator
$SYNC$	Synchronization state
C_i	Pseudo-random bit sequence
S	Pseudo-random bit sequence
m_i	Update message
k_δ	Hash chain
$\|\|$	Concatenation operator
$\oplus$	XOR operator

3.1 System environment

The proposed protocol requires a server that hosts its own database and readers that transmit information from tags to the server. For each tag, the server stores a record containing the following data (see Table 2):

- id: The tag identity.
- $Info_{id}$: The tag information.
- uk: An update key uk that only changes after the execution of a successful ownership transfer protocol. This key is only known by the current tag owner.
- ik *and* $h_{id}(ik)$: A tag identification key ik and its associated hash value $h_{id}(ik)$ using the tag identifier as key. As described, the hash value is used to identify a tag in a lookup table.

– ik_{old} *and* $h_{id}(ik_{old})$: A previous tag identification key ik_{old} and its associated hash value $h_{id}(ik_{old})$ are kept in order to prevent denial-of-service attacks.
– $Table\ k$: A table used to identify a tag in case of desynchronization between the tag and the server. Row 1 and row 2 in the table store hash chains of size MAX starting from ik_{old} and ik, respectively.

Table 2. DB values for each tag

Tag data		$Table\ k$	
id		k_i^{old}	k_i
$Info_{id}$	0	$h_{id}(ik_{old})$	$h_{id}(ik)$
uk	1	$h_{uk}(k_0^{old}\|\|id)$	$h_{uk}(k_0\|\|id)$
ik	2	$h_{uk}(k_1^{old}\|\|id)$	$h_{uk}(k_1\|\|id)$
$h_{id}(ik)$	$\vdots$	$\vdots$	$\vdots$
ik_{old}			
$h_{id}(ik_{old})$	MAX	$h_{uk}(k_{MAX-1}^{old}\|\|id)$	$h_{uk}(k_{MAX-1}\|\|id)$

3.2 Protocol phases

In this section we describe the five different phases of our protocol. Table 4 in Appendix A depicts the four phases after initialization.

Initialization. Two unique keys, ik and uk, are generated for each tag. Then, ik, uk and id are written on the non-volatile memory of the tag via a secure channel. This information will be used later by the tag to perform identification and ownership transfer. After tag initialization, the server stores the tag data in its database (see Table 2).

Synchronized identification phase. When the tags are synchronized, they execute this phase in order to be identified by a legitimate reader.

1. First, the reader broadcasts a nonce r_0.
2. Then, the synchronized tag answers with the following information: $h_{id}(ik)$, $C_0 = PRNG(ik||r_0||r_1)$[1] and r_1. After sending all data, the tag switches to a desynchronized state ($SYNC = 0$) until the update phase ends.
3. Upon reception of these data, the reader forwards them to the server.
4. Then, the server searches inside a lookup table the value $h_{id}(ik)$ and obtains the tag's data ik and id. Using the identification key ik and the nonces r_0 and r_1, the server checks C_0 and decides whether to send id to the reader. Note that by checking C_0 the server avoids phishing or replay attacks. Finally, the server saves r_0 and r_1. These values are used in the update phase.

[1] The pseudo-random number generator PRNG is supposed to be secure and unpredictable.

Update phase. All the features of our protocol are mainly based on this phase.

1. The server composes m by concatenating a new identification key $m_L = ik_{new}$ and its hash value $m_R = h_{id}(ik_{new})$. Finally, the server computes $C_1 = m \oplus S$ where $S = PRNG(h_{uk}(ik)||id||r_0||r_1)$ is an unpredictable pseudorandom sequence. The server sends C_1 to the reader.
2. The reader forwards C_1 to the tag.
3. Upon reception of C_1, the tag generates its own pseudorandom sequence $S' = PRNG(h_{uk}(ik)||id||r_0||r_1)$, and computes $m' = C_1 \oplus S'$. By splitting m', the tag obtains m'_R and m'_L and checks whether $m'_R = h_{uk}(m'_L)$. If so, the tag can be sure that m' is, indeed, m; otherwise the tag rejects the reader's response. After the reader authentication, the tag splits m and updates its data: $ik = m'_L$ and $SYNC = 1$.

Desynchronized identification phase. This phase is executed when reader and tag are desynchronized, *e.g.* the message C_1 corresponding to the update phase was incorrect or it had not been received. The steps are as follows:

1. The reader sends r_0 to the tag, like in the synchronized identification phase.
2. The tag generates a new nonce r_1 and computes:
   $C_0 = PRNG(h_{uk}(k_{delta})||id||r_0||r_1)$, $\delta = \delta + 1$, and $k_\delta = h_{ik}(k_{\delta-1})$ where $k_0 = h_{id}(ik)$. Finally, the tag sends k_δ and C_0 to the reader.
3. Upon reception of the tag's response, the reader forwards the data to the server, who will search the value k_δ in the database using a lookup table generated with all the $Table\,k$ in the database (see Table 2). If the k_δ value is not found, the identification process fails. Otherwise, if k_δ is found in one or more records of the database, the server obtains the correct identifier through search of the id matching the C_0 value.
4. After a correct identification of the tag, the reader starts the Update phase.

It should be remarked that, the desynchronized identification phase can be executed consecutively just MAX times. If the number of consecutive identifications by a desynchronized identification phase is greater than MAX, the server will not be able to identify the tag any more (denial of service). The value of the parameter MAX is extensively discussed in [5].

Controlled delegation phase. This phase is run when the current owner needs to delegate identification rights to a new reader.

1. First, the current owner runs a successful synchronized identification phase but skips the update phase. At this stage, the tag is desynchronized and therefore responds with k_δ values when queried.
2. Then, the current owner just needs to give to the new reader the following infomation: id and n pairs $(k_\delta, h_{uk||ik}(k_\delta))$.
3. Later, the current owner is able to recover full control over the tag using one of the two following strategies: i) run a successful synchronized identification phase together with an update phase or ii) query the tag n times where n is the number of values given to the new reader.

When the new reader identifies a tag n times by controlled delegation, loose the right to identify this tag and only the current owner is able to identify the tag again. If the new reader needs to identify this tag again, it must request authorization to the current owner one more time

This procedure is also described in Table 3 in Appendix 5.

Owner transfer phase. This phase is used to transfer ownership of the tag from the current owner to the new owner. The basic idea is to use a temporary key, as shown in Figure 2.

1. First, the current owner updates the key uk with uk_{tmp}. This prevents the new owner from backward tracking the tag.
2. Next, the current owner gives to the new owner the key uk_{tmp}.
3. Finally, the new owner updates uk_{tmp} with uk_{new} to prevent the old owner from forward tracking the tag.

After the previous protocol, the current owner plays the role of the previous owner while the new owner becomes the current owner. It should be remarked that a tag can know that uk is being updated by computing the left part of C_2. On the other hand, the new owner should update uk in an isolated environment in order to prevent the current owner from eavesdropping the messages and computing uk_{new}.

$$uk \longrightarrow uk_{tmp} \longrightarrow uk_{new}$$

Fig. 2. Life-cycle of the update key during the ownership transfer process

Authorization recovery. In our scheme, an authorization recovery process can be performed as a controlled delegation process. However, we must assume that the current owner is unwilling to give the tag's identifier to another reader. The previous owner must search in its data base an identifier that matches C_0 using one of the provided pairs $(k_\delta, h_{uk||ik}(k_\delta))$. Note that this checking process can be only performed by a previous owner of a tag and, hence, the tag can be identified only by a legitimate previous owner.

3.3 Protocol states

In this protocol, a tag can be in one of the following states: initialized, synchronized, desynchronized and owner transfer. The tag changes its state by means of the following operations:

– (a) Initializing a tag. Once a tag has been initialized, it goes to the synchronized state.

– (b) Identifying a tag when it is synchronized. Once the tag is synchronized, if an identification is requested then its state changes to desynchronized.
– (c) Updating a tag. After a tag has been identified, the reader sends an update message. If the message is verified properly by the tag, the tag goes to the synchronized state.
– (d) Identifying a tag when it is desynchronized does not change the tag's state. Thus, the tag will remain desynchronized.
– (e) Running an owner transfer protocol. When the current owner runs the owner transfer phase, the tag changes its state to owner transfer. When the operation is verified successfully, the tags state is set to synchronized.
– (f) Disabling a tag. If the desynchronized identification phase is run more than MAX consecutive times, the tag is disabled (denial of service).

Figure 3 shows the different states and operations that are possible in the protocol.

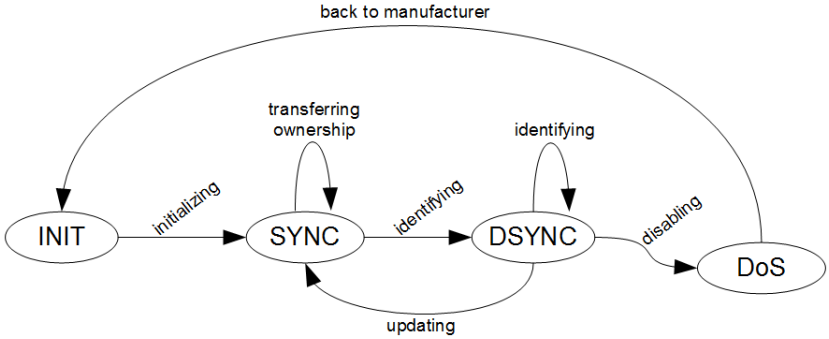

Fig. 3. The state diagram contains the states and transitions of an RFID tag in our scheme. The states are the following: i) tag initialized (INIT), ii) tag synchronized (SYNC), iii) tag desynchronized (DSYNC), and iv) tag disabled (DoS).

4 Analysis

The protocol has the following security and privacy properties.

4.1 Privacy

We consider the following privacy property:

Monitoring and location. Location privacy is guaranteed because, in each identification, the data sent are always different, thanks to the use of nonces r_0 and r_1. Indeed, values ik, $h_{id}(ik)$, C_0 and r_1 are updated in each identification.

If the reader and the tag are not synchronized, then the updated elements are: k_δ, C_0 and r_1. If both devices are synchronized, $h_{id}(ik)$ can appear twice with a probability of $1/2^\ell$ where ℓ is the size of the hash function.

The probability for k_δ is the same as $h_{id}(ik)$ in consecutive secondary identifications.

4.2 Security

Below are the most common attacks which can be launched on RFID schemes:

Denial of service. An attacker can query the tag or interrupt the update phase $MAX + 1$ times, so that after that, a legitimate reader will not find k_δ in the database and will not be able to identify the tag. Since MAX is a security parameter of our system, we can set this value in order to increase the resistance of the proposed protocol against this attack. If the table is large enough, the time needed to consume all the values will be high enough. For instance, for $MAX = 100,000$ the required time to obtain all values of $Table\ k$ by an attacker, taking into account that each identification needs an interval of 200 ms, is 5.5 hours. By contrast, the disk space needed in a system with only 100,000 tags is 24 Tb.

To prevent massive attacks, the system can identify the tags regurarly. In this case, if an attacker is trying to knock out a tag, once the system updates that tag, the attacker must restart the DoS attack from zero. Furthermore, the value of MAX can be parameterized to resist DoS attacks. The fact that a high value of MAX behave a high space in disk is assumed due to the low cost of storage units today, yet this is one aspect to improve in a future versions of this protocol.

Impersonation of devices.

- *Impersonating a certain tag.* An attacker does not know values ik, uk and id. These elements are sent in the initialization phase using a secure communication channel. An attacker trying to impersonate a certain tag without the knowledge of these values will be detected by the reader. If an attacker is able to obtain the tag values, this information does not compromise other tags. Hence, impersonating a tag requires physically tampering with it and, in that case, it allows impersonating only that tag.
- *Impersonating the server.* This case is similar to the previous property. The server can only be impersonated by an attacker who knows the entire database. We assume that the database is hosted in a secure environment.
- *Replay attacks.* This type of attack is not possible in this protocol because, in each identification, each of the sides (tag and server) provides a new value which is computed at random. In this way, a replay attack at the server side will be successful only if the message which has been captured previously contains the expected r_0 value. The same happens at the tag side but with

the value r_1. Random nonces r_0 and r_1 are generated in each device and they are unlikely to be repeated over a short period of time. In this way, value $h_{id}(ik)$ is different in each synchronized identification phase. In the desynchronized identification phase case, k_δ is always different.

4.3 Ownership transfer

Our proposal satisfies the following properties.

New owner privacy. Our protocol is designed to guarantee that the transactions between the current owner and the tag cannot be traced by the previous owner. When the ownership of the tag is transferred, uk and ik are randomly updated, thereby ensuring that previous owners cannot identify the tag anymore unless the current owner allows controlled delegation.

Previous owner privacy. Since the update key is changed twice ($uk \rightarrow uk_{tmp} \rightarrow ik$), the previous owner's privacy is also guaranteed. As a result, the current owner cannot trace previous transactions between the previous owner and the tag.

Authorization recovery. The protocol satisfies this property because the current owner can delegate the reading of the tag in a controlled fashion.

5 Conclusions

The novelty of our proposal consists of leveraging the update phase, that is used in the RFID identification protocol, in order to implement the ownership transfer protocol. Moreover, the protocol allows controlled delegation without the need of a counter in the non-volatile memory of tags. This feature is especially important considering that: i) tags are resource-constrained devices; ii) readers should not share the identification key of a tag. Finally, the protocol does not need a TTP, so that the users can perform ownership transfers any time and anywhere.

Disclaimer and acknowledgments

The authors are solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organization. This work was partly supported by the Spanish Ministry of Education through projects TSI2007-65406-C03-01 "E-AEGIS" and CONSOLIDER CSD2007-00004 "ARES", by the Spanish Ministry of Industry, Commerce and Tourism through project "eVerification" TSI-020100-2009-720 and by the Government of Catalonia under grant 2009 SGR 1135. The last author is partly supported as an ICREA Acadmia Researcher by the Government of Catalonia.

References

1. B. Alomair and R. Poovendran. Privacy versus scalability in radio frequency identification systems. *Computer Communications*, 33(18):2155 – 2163, 2010.
2. G. Avoine, E. Dysli, and P. Oechslin. Reducing Time Complexity in RFID Systems. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography – SAC 2005*, volume 3897 of *Lecture Notes in Computer Science*, pages 291–306, Kingston, Canada, August 2005. Springer.
3. H.B. Chen, W.B. Lee, Y.H. Zhao, and Y.L. Chen. Enhancement of the RFID security method with ownership transfer. In *Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication*, pages 251–254. ACM, 2009.
4. T. Dimitriou. rfidDOT: RFID delegation and ownership transfer made simple. In *Proceedings of the 4th international conference on Security and privacy in communication netowrks*, pages 1–8. ACM, 2008.
5. A. Fernández-Mir, J. Castellà-Roca, and A. Viejo. Secure and Scalable RFID Authentication Protocol. In J. Garcia-Alfaro et al., editor, *DPM 2010 and SETOP 2010*, volume 6514 of *Lecture Notes in Computer Science*, pages 231–243, Athens, Greece, September 2010. Springer.
6. S. Fouladgar and H. Afifi. An efficient delegation and transfer of ownership protocol for RFID tags. In *First International EURASIP Workshop on RFID Technology, Vienna, Austria*, 2007.
7. C. Goebel, C. Tribowski, O. Gunther, R. Troeger, and R. Nickerl. RFID in the Supply Chain: How to Obtain a Positive ROI-The Case of Gerry Weber. In *11th International Conference on Enterprise Information Systems (ICEIS 2009), Milan, Italy, May 6*, volume 10, pages 95–102, 2009.
8. G. Kapoor and S. Piramuthu. Vulnerabilities in some recently proposed RFID ownership transfer protocols. In *2009 First International Conference on Networks & Communications*, pages 354–357. IEEE, 2009.
9. M. Langheinrich. A survey of RFID privacy approaches. *Personal and Ubiquitous Computing*, 13(6):413–421, 2009.
10. D. Molnar, A. Soppera, and D. Wagner. A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags. In *Selected Areas in Cryptography*, pages 276–290. Springer, 2006.
11. K. Osaka, T. Takagi, K. Yamazaki, and O. Takahashi. An Efficient and Secure RFID Security Method with Ownership Transfer. In *Computational Intelligence and Security, 2006 International Conference on*, volume 2, pages 1090–1095. IEEE, 2007.
12. J. Saito, K. Imamoto, and K. Sakurai. Reassignment scheme of an RFID tags key for owner transfer. *Embedded and Ubiquitous Computing*, pages 1303–1312, 2005.
13. S.E. Sarma. Towards the five-cent tag. Technical report, 2001.
14. B. Song and C.J. Mitchell. Scalable RFID security protocols supporting tag ownership transfer. *Computer Communications*, 2010.
15. European Union. Commission recommendation of 12 may 2009 on the implementation of privacy and data protection principles in applications supported by radiofrequency identification. Technical Report Official Journal of the European Union, May 2009.
16. E.J. Yoon and K.Y. Yoo. Two security problems of RFID security method with ownership transfer. In *Network and Parallel Computing, 2008. NPC 2008. IFIP International Conference on*, pages 68–73. IEEE, 2008.

17. C. Yu Ng, W. Susilo, Y. Mu, and R. Safavi-Naini. Practical RFID Ownership Transfer Scheme. *Journal of Computer Security - Special Issue on RFID System Security*, 2010.

A Tables

Table 3. Controlled delegation and owner transfer phases

Current User		**Tag**		**Final User**
Controlled delegation phase				
	Synchronized identification phase			
	$\longrightarrow$			
		$SYNC = 0$		
	id and n pairs $(k_\delta, h_{uk\|\|ik}(k_\delta))$ (with secure channel)			
	$\longrightarrow$			
			Desynchronized identification phase	
			$\longleftarrow$	
		$SYNC = 0$		
Owner transfer phase				
$m_L = uk_{new}$, $m_R = h_{uk}(uk_{new})$ $m = m_L\|\|m_R$ $S = PRNG(h_{uk}(id)\|\|r_0\|\|r_1)$ $C_2 = m \oplus S$				
	C_2			
	$\longrightarrow$			
		$S^{'} = PRNG(h_{uk}(id)\|\|r_0\|\|r_1)$ $m^{'} = C_2 \oplus S^{'}$ $m_R^{'} \stackrel{?}{=} h_{uk}(m_L^{'})$ $uk = m_L^{'}$		
	all data of the tag (with secure channel)			
	$\longrightarrow$			
			Owner transfer phase	
			$\longleftarrow$	

Table 4. Our protocol

Server	**Reader**		**Tag**
Initialization phase (with secure channel)			
		ik, uk, id ⟶	
Synchronized identification phase			
	$r_0 \in R\{0,1\}^*$	r_0 ⟶	
			$r_1 \in R\{0,1\}^*$ $C_0 = PRNG(ik\|\|r_0\|\|r_1)$ $\delta = 0$ Computes $h_{id}(ik)$ $SYNC = 0$
		$h_{id}(ik), C_0, r_1$ ⟵	
	$h_{id}(ik), C_0, r_1, r_0$ ⟵		
Searches $h_{id}(ik)$ Checks C_0			
Update phase			
$m_L = ik_{new}$, $m_R = h_{uk}(ik_{new})$ $m = m_L\|\|m_R$ $S = PRNG(h_{uk}(id)\|\|r_0\|\|r_1)$ $C_1 = m \oplus S$			
		C_1 ⟶	
	C_1 ⟶		
			$S^{'} = PRNG(h_{uk}(id)\|\|r_0\|\|r_1)$ $m^{'} = C_1 \oplus S^{'}$ $m_R^{'} \stackrel{?}{=} h_{uk}(m_L^{'})$ $ik = m_L^{'}$ $SYNC = 1$
Desynchronized identification phase			
	$r_0 \in R\{0,1\}^*$	r_0 ⟶	
			$r_1 \in R\{0,1\}^*$ $k_0 = h_{id}(ik)$ $\delta = \delta + 1$ $k_\delta = h_{ik}(k_{\delta-1}\|\|id)$ $C_0 =$ $PRNG(h_{uk}(k_{delta})\|\|id\|\|r_0\|\|r_1)$
		k_δ, C_0, r_1 ⟵	
	k_δ, C_0, r_1 ⟵		
Searches k_δ on $Table\,k$. Checks C_0 Start update phase			