ARTICLE IN PRESS

Computer Networks xxx (2010) xxx–xxx

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Vulnerability analysis of RFID protocols for tag ownership transfer

Pedro Peris-Lopez [a,*], Julio C. Hernandez-Castro [b], Juan M.E. Tapiador [b], Tieyan Li [c], Yingjiu Li [d]

[a] Information and Communication Theory Group, Delft University of Technology, Netherlands
[b] Computer Science Department, Carlos III University of Madrid, Spain
[c] Institute for Infocomm Research, A*STAR Singapore, Singapore
[d] Singapore Management University (SMU), Singapore

ARTICLE INFO

Article history:
Received 27 January 2009
Received in revised form 7 September 2009
Accepted 18 November 2009
Available online xxxx
Responsible Editor: L. Salgarelli

Keywords:
Security protocols
Authentication
Ownership transfer
RFID

ABSTRACT

In RFIDSec'08, Song proposed an ownership transfer scheme, which consists of an ownership transfer protocol and a secret update protocol [7]. The ownership transfer protocol is completely based on a mutual authentication protocol proposed in WiSec'08 [8]. In Rizomiliotis et al. (2009) [6], van Deursen and Radomirovic (2008), the first weaknesses to be identified (tag and server impersonation) were addressed and this paper completes the consideration of them all. We find that the mutual authentication protocol, and therefore the ownership transfer protocol, possesses certain weaknesses related to most of the security properties initially required in protocol design: tag information leakage, tag location tracking, and forward traceability. Moreover, the secret update protocol is not immune to de-synchronization attacks.

1. Introduction

Many protocols have been proposed aiming to provide secure contact between RFID reader and tag over the open radio channel. However, due to tag limitations in terms of circuitry (computation power), storage and power consumption, designing an efficient and secure mutual authentication protocol is still a great challenge. In this paper, we analyze a new mutual authentication protocol proposed by Song and Mitchell [8], hereafter referred to as the SM protocol. This protocol is said to resist tag information leakage, tracking, tag and server impersonations, replay attacks, denial of service attacks and forward and backward traceability.

In addition to the mutual authentication problem, some authors are working on how to make an RFID system secure when a tag changes its owner multiple times during its life cycle [2,4,5,7]. This issue is known as the ownership transfer problem. It arises when the server of a new owner takes over tag authorization, and so is given certain private information to interact with the tag in a secure manner. In [7], the usability and security of previous proposals that address the ownership transfer problem are scrutinized before a new ownership transfer scheme is proposed. The new scheme is divided into two sub-protocols, an ownership transfer protocol (wholly based on SM) and a secret update protocol. This paper examines the security of both protocols and identifies their vulnerabilities.

2. Review of RFID protocols for ownership transfer

In this section, we review the newly proposed protocols for secure ownership transfer, including the mutual authentication protocol [8], the ownership transfer protocol, and the secret update protocol [7]. These protocols are viable when tags can generate random strings and support an on-board hash function or MAC.

* Corresponding author.
E-mail addresses: P.PerisLopez@tudelft.nl (P. Peris-Lopez), jcesar@inf.uc3m.es (J.C. Hernandez-Castro), jestevez@inf.uc3m.es (J.M.E. Tapiador), litieyan@i2r.a-star.edu.sg (T. Li), yjli@smu.edu.sg (Y. Li).
URL: http://www.lightweightcryptography.com (P. Peris-Lopez).

1389-1286/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2009.11.007

2.1. Proposed protocols

We use the same notation as in the original paper [7,8] to describe protocols (see Table 1). First, the initialization phase of the tag and server is introduced. An initiator assigns a string s_i of l bits to each tag T_i, computes $t_i = h(s_i)$, and stores t_i in the tag. The parameter l should be great enough so that an exhaustive search aimed at finding the l-bit values of t_i, s_i, and r is computationally infeasible. Additionally (although it is not mentioned in the original paper), length l should be selected to ensure unequivocal identification of tagged items. A legitimate server stores a tuple of the form $[(s_i,t_i)_{new},(s_i,t_i)_{old},D_i]$ for each tag, where $(s_i,t_i)_{new}$ are the newly assigned secrets, $(s_i,t_i)_{old}$ are the old secrets, and D_i is the information associated with the tag. Only $t_{i_{new}}$ is stored on the tag.

2.1.1. Mutual authentication protocol

An outline of the SM mutual authentication protocol is shown below (see Fig. 1).

1. Reader → Tag: The reader generates a random bit-string $r_1 \in_R \{0,1\}^l$ and sends it to T_i.
2. Tag → Reader: The tag T_i generates a random bit-string $r_2 \in_R \{0,1\}^l$, computes $M_1 = t_i \oplus r_2$ and $M_2 = f_{t_i}(r_1 \oplus r_2)$, and sends (M_1, M_2) to the reader.
3. Reader → Server: The reader sends (r_1, M_1, M_2) to the server.
4. Server → Reader: The server chooses t_i from the values $t_{i_{new}}$ or $t_{i_{old}}$ for $1 \leqslant i \leqslant N$. It computes $M_2' = f_{t_i}(r_1 \oplus M_1 \oplus t_i)$. If $M_2' = M_2$, then the server has identified and authenticated the tag as T_i. In this case, let the current secrets be denoted as (s_i,t_i). If no match is found, the server sends ϵ to the reader and stops the session. The server then computes $M_3 = s_i \oplus (r_2 \gg l/2)$ and sends it with D_i to the reader. Finally the server updates $s_{i_{old}}$ and $t_{i_{old}}$ for the tag T_i to s_i and t_i respectively, and sets $s_{i_{new}} = (s_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2$ and $t_{i_{new}} = h(s_{i_{new}})$ in its database.
5. Reader → Tag: The reader forwards M_3 to T_i, which computes $s_i = M_3 \oplus (r_2 \gg l/2)$ and checks if $h(s_i) = t_i$. If the check succeeds, the tag has authenticated the server, and sets $t_i = h((s_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2)$. If the check fails, the tag keeps the current value of t_i unchanged.

2.1.2. Ownership transfer protocol

In [7], Song proposed an ownership transfer protocol wholly based on the SM protocol. For convenience, we denote the current owner of a tag as S_j and the new owner as S_{j+1}. The details of the ownership transfer protocol are shown below (see Fig. 2).

1. $S_{j+1} \rightarrow T_i$: S_{j+1} generates a random string r_1 of l bits and sends it to T_i.
2. $T_i \rightarrow S_{j+1}$: T generates a random string r_2 of l bits, computes $M_1 = t_i \oplus r_2$ and $M_2 = f_{t_i}(r_1 \oplus r_2)$, and then sends (M_1, M_2) to S_{j+1}.
3. $S_{j+1} \rightarrow S_j$: After receiving (M_1, M_2), S_{j+1} sends (r_1, M_1, M_2) to S_j, with a request for ownership of T_i (R_{T_i}).
4. $S_j \rightarrow S_{j+1}$: If the received request R_{T_i} is valid, S_j searches its database for a pair (t_i, s_i) for which the value of t_i satisfies $M_2 = f_{t_i}(r_1 \oplus M_1 \oplus t_i)$. If such a pair (t_i, s_i) is found, S_j sets $r_2 = M_1 \oplus t_i$ and computes $M_3 = s_i \oplus (r_2 \gg l/2)$. Otherwise, the session stops. S_j updates the secrets as $s_{i_{old}} \leftarrow s_i$, $t_{i_{old}} \leftarrow t_i$, $s_{i_{new}} \leftarrow (s_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2$ and $t_{i_{new}} \leftarrow h(s_{i_{new}})$. S_j sends M_3 to S_{j+1}, and transfers the updated secrets $(t_{i_{new}}, s_{i_{new}})$ together with some other necessary information D_i about the tag to S_{j+1} via a secure channel.
5. $S_{j+1} \rightarrow T_i$: When S_{j+1} receives $((t_{i_{new}}, s_{i_{new}}), D_i, M_3)$ from S_j, it stores $(t_{i_{new}}, s_{i_{new}})$, D_i in its database, and forwards M_3 to T_i. T_i then computes $s_i = M_3 \oplus (r_2 \gg l/2)$, and checks if $h(s_i) = t_i$ holds. If $h(s_i) = t_i$ holds, T_i updates its secret as $t_i \leftarrow h((s_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2)$. Otherwise, the session stops.

2.1.3. Secret update protocol

When a new owner S_{j+1} takes ownership of a tag, he can choose a new value for the tag's secrets (t,s), and update the value t on the tag. This updating phase is necessary to provide privacy protection for the new owner of the tag. In other words, its execution will frustrate the identification and traceability of the tag by the old owner S_j. The details of the secret update protocol [7] are shown below (see Fig. 3):

1. $S_{j+1} \rightarrow T_i$: S_{j+1} generates random strings r_1 and s_i' of l bits, and computes $t_i' = h(s_i')$, $M_1 = f_{t_i}(r_1) \oplus t_i'$ and $M_2 = s_i \oplus (t_i' \gg l/2)$. Finally, the new owner sends (r_1, M_1, M_2) to T_i.

Table 1
The notation.

l	The bit-length of a tag identifier	x_{old}	The previous value of x
N	The number of tags	r	A random string of l bits
f_k	A keyed hash function,	$[x]_{L/R}$	The left/right half part of the string x
	$f_k : \{0,1\}^l \times \{0,1\}^l \rightarrow \{0,1\}^l$ (a MAC algorithm)	ϵ	Error message
h	A hash function, $h : \{0,1\}^l \rightarrow \{0,1\}^l$	$\oplus$	XOR operator
T_i	The ith tag $(1 \leqslant i \leqslant N)$	$\|$	Concatenation operator
D_i	The detailed information on tag T_i	$\leftarrow$	Substitution operator
s_i	A string of l bits assigned to T_i	$x \gg k$	Right circular shift operator, which rotates all bits of x to the right by k bits, as if the left and right ends of x were joined
t_i	T_i's identifier of l bits, which equals $h(s_i)$	$x \ll k$	Left circular shift operator, which rotates all bits of x to the left by k bits, as if the left and right ends of x were joined
x_{new}	The new (refreshed) value of x	$\in_R$	the random choice operator, which randomly selects an element from a finite set using a uniform probability distribution

Please cite this article in press as: P. Peris-Lopez et al., Vulnerability analysis of RFID protocols for tag ownership transfer, Comput. Netw. (2010), doi:10.1016/j.comnet.2009.11.007

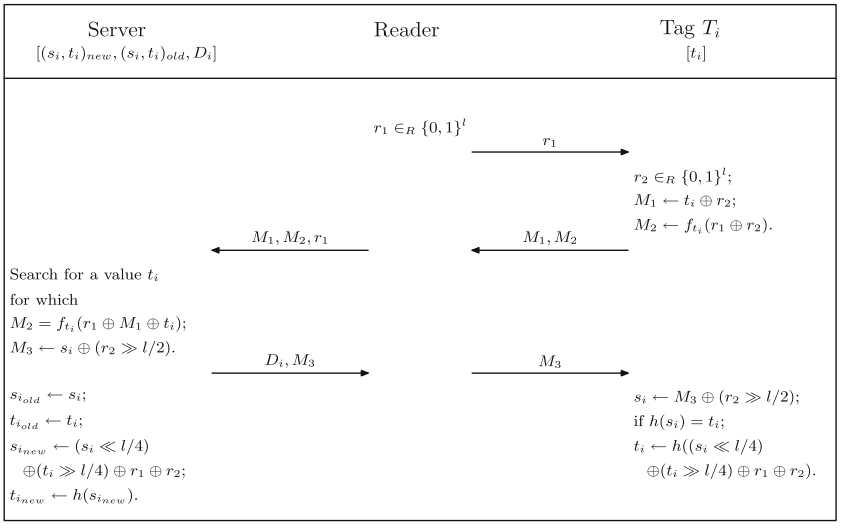

Fig. 1. Mutual authentication protocol.

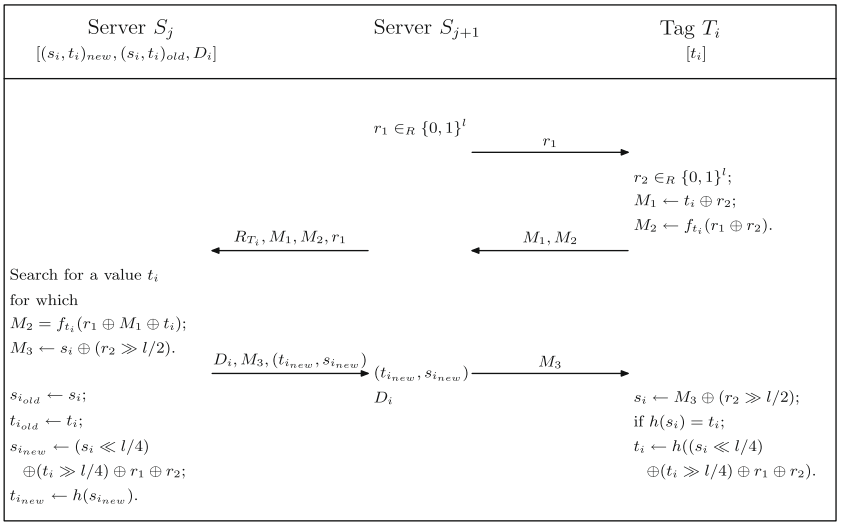

Fig. 2. Ownership transfer protocol.

2. $T_i \rightarrow S_{j+1}$: T_i receives (r_1, M_1, M_2) from S_{j+1}, and computes $t'_i = M_1 \oplus f_{t_i}(r_1)$ and $s_i = M_2 \oplus (t'_i \gg l/2)$. If $h(s_i) = t_i$, T_i has authenticated S_{j+1} as an authorized server. Otherwise, the session stops. Then T_i updates its secret as $t_i \leftarrow t'_i$, generates a random string r_2 of l bits and computes $M_3 = f_{t_i}(r_1 \oplus r_2)$. Finally, it sends (r_2, M_3) to S_{j+1}. Upon receiving (r_2, M_3), S_{j+1} checks whether M_3 is equal to $f_{t'_i}(r_1 \oplus r_2)$. If the validation succeeds, it indicates that T_i possesses the new secret t'_i. In this case, S_{j+1} updates its secrets $s_{i_{new}}$ and $t_{i_{new}}$ to s'_i and t'_i, respectively; otherwise, S_{j+1} starts a new session.

3. Vulnerability analysis

We use the standard Dolev–Yao intruder model [1] in which the adversary has full control over the "network". In this model, the adversary can eavesdrop, block, modify and inject messages in any communication between (in our case) a reader and a tag. However, as is commonly assumed, we assume that communications between servers and readers are secure, which can be guaranteed by using fully-fledged cryptographic technologies.

3.1. Server impersonation and denial-of-service attack

It is claimed that the mutual authentication protocol [7,8] is secure against the server impersonation attack because an attacker cannot compute a valid M_3 message without knowing s_i. However, an active attacker who does not know (s_i, t_i) can use the eavesdropped messages in a valid session to compute a correct M_3, thus impersonating a legitimate server. Such an attack on a tag causes loss of synchronization (a denial-of-service attack) between the tag and the impersonated server. The reader is referred

Please cite this article in press as: P. Peris-Lopez et al., Vulnerability analysis of RFID protocols for tag ownership transfer, Comput. Netw. (2010), doi:10.1016/j.comnet.2009.11.007

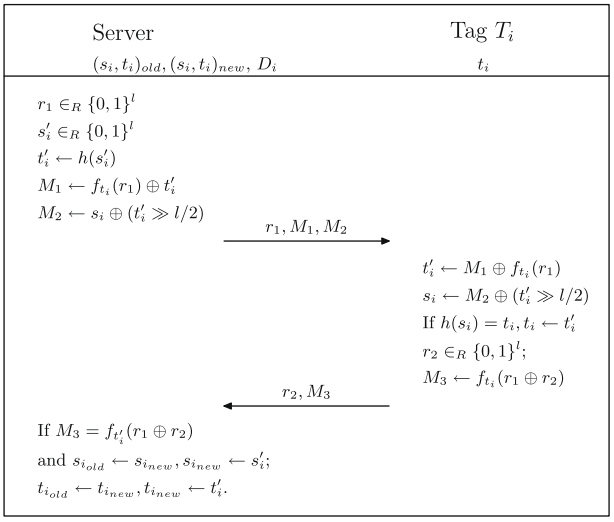

Fig. 3. Secret update protocol.

to [6] for a complete description of the attack. Although we have described the attack in terms of the protocol [8], it applies directly to the protocol [7] due to the great similarity.

3.2. Tag impersonation attack

Song et al. argue that the mutual authentication protocol is secure against the tag impersonation attack, since it is difficult for an attacker to compute a valid response (M_1, M_2) to a reader's query without knowing t_i. However, Deursen et al. [9] show how an adversary can easily forge a valid message (M_1, M_2) by computing a XOR operation with messages/nonces previously intercepted.

3.3. Tracking or compromise of location privacy

It is claimed that the mutual authentication protocol guarantees location privacy for tag owners, since the responses M_1 and M_2 provided by the tags are anonymous due to the inclusion of fresh random numbers in these messages. However, the attacker is able to listen in on both the backward and forward channels. In other words, the messages sent by the reader can be eavesdropped too. Our analysis shows that an attacker, by simply listening in on the channel, blocking or altering (e.g. flipping a bit) message M_3 sent from the reader to a target tag, is able to discriminate this tag within a population of N tags (this privacy concept is compatible with the Juels–Weis untraceability model [3]). Details of the attack are given below:

1. The adversary eavesdrops a valid session between the target tag and the reader. A legitimate reader generates a random string $r_1 \in_R \{0,1\}^l$ and sends r_1 to the tag. After receiving r_1, the tag chooses a random string $r_2 \in_R \{0,1\}^l$, and computes messages $M_1 = t_i \oplus r_2$ and $M_2 = f_{t_i}(r_1 \oplus r_2)$. The tag sends (M_1, M_2) to the reader. The valid reader then queries the server and gets the updating message $M_3 = s_i \oplus (r_2 \gg l/2)$. The adversary acquires values M_1 and M_3 and prevents updating of the tag's internal value (t_i) by blocking or altering message M_3.
2. The target tag is introduced in a population of N tags $-N$ being an arbitrary value. The adversary listens in on an authentication session between one of these legitimate tags and the reader. As in the above case, the adversary can eavesdrop messages M'_1 and M'_3, and frustrate the correct reception of message M'_3.

$$M'_1 = t_j \oplus r'_2, \tag{1}$$
$$M'_3 = s_j \oplus (r'_2 \gg l/2), \tag{2}$$

where j can take one of the following values $j = 1, \ldots, N$, and N is the size of the population in which the target tag was introduced. The attacker can guess if the answers provided originate from the target tag by means of the following computation:

$$M_1 \oplus (M_3 \ll l/2) \stackrel{?}{=} M'_1 \oplus (M'_3 \ll l/2). \tag{3}$$

Proof. We now prove that $M''_1 \oplus (M''_3 \ll l/2)$ is a constant value which does not depend on the fresh random numbers generated in each session. The attack is completely feasible because once the tag is introduced into the population of N-tags, the attacker only has to listen in to the channel and prevent the correct reception of message M''_3.

The equations M''_1 and M''_3 are expanded in the following way; we use $[x]_L$ to denote the left half of the string x, and $[x]_R$ to denote the right half

$$M''_1 = t_j \oplus r''_2 = [t_j]_L \oplus [r''_2]_L || [t_j]_R \oplus [r''_2]_R, \tag{4}$$
$$M''_3 = s_j \oplus (r''_2 \gg l/2) = [s_j]_L \oplus [r''_2]_R || [s_j]_R \oplus [r''_2]_L. \tag{5}$$

Then, M''_3 is rotated to the left by l/2 bits:

$$(M''_3 \ll l/2) = [s_j]_R \oplus [r''_2]_L || [s_j]_L \oplus [r''_2]_R. \tag{6}$$

Please cite this article in press as: P. Peris-Lopez et al., Vulnerability analysis of RFID protocols for tag ownership transfer, Comput. Netw. (2010), doi:10.1016/j.comnet.2009.11.007

Finally,

$$M_1'' \oplus (M_3'' \ll l/2) = [t_j]_L \oplus [s_j]_R || [t_j]_R \oplus [s_j]_L. \quad \square \qquad (7)$$

We prove that $M_1'' \oplus (M_3'' \ll l/2)$ only depends on t_j and s_j, which are different values in each tag (unequivocal identification). Therefore, the attacker can identify a tag from a population of tags under the assumption that the updating is precluded in those tags. This is not a rare condition; in fact this situation is considered in the definition of the mutual authentication protocol as a countermeasure for desynchronization attacks. In fact, the protocol definition does not say how many times a tag can be identified using the same identifier t_i. In conclusion, the target tag can be indefinitely identified from a population of tags under the assumption of not updating.

3.4. Information leakage – revealing back-end database content

Prior to using the mutual authentication protocol, an initialization phase should be completed by an initiator. The initiator assigns a string s_i of l bits to each tag T_i. However, with the aim of offering a superior security level, the result of computing a l-bit hash function, that is $t_i = h(s_i)$, is stored on the tag instead of s_i. The length of the values should be great enough so that an exhaustive search of the l-bit values of t_i and s_i is computationally infeasible. We will show that although the above idea is well founded, an attacker can acquire all the pairs $\{s_i, t_i\}$ without computing an exhaustive search. In fact, the attacker only needs to interact with a legitimate reader/back-end database once to acquire the information linked to a tag. Details of the attack are given below. We first describe the process for acquiring the private information linked to an specific tag and then present the generalization process which reveals the complete content of the database.

The attacker selects an l-bit t_j and wishes to acquire the private information associated with it, s_j. Then, he starts an authentication session with a legitimate reader:

1. A legitimate reader generates a random string $r_1 \in_R \{0,1\}^l$ and sends r_1 to the adversary.
2. The adversary chooses a random string $r_2 \in_R \{0,1\}^l$ and computes messages $M_1 = t_j \oplus r_2$ and $M_2 = f_{t_j}(r_1 \oplus r_2)$.
3. The adversary sends messages (M_1, M_2) to the reader.
4. The reader forwards values r_1, M_1, M_2 to the server. The server checks the values, and generates an updating message $M_3 = s_j \oplus (r2 \gg l/2)$, which is sent to the reader.
5. The reader forwards message M_3 to the adversary.
6. The adversary can obtain the private information s_j by simply computing an XOR operation, $s_j = M_3 \oplus (r2 \gg l/2)$.

Repeatedly executing the above attack for all the possible values of t_j ($t_{j+1} = t_j + 1$ for $0 < j < 2^l$ and $t_1 = 0$), the adversary is able to acquire all the information stored in the back-end database:

$s_1 = M_3 \oplus (r_2 \gg l/2)$	t_1
$s_2 = M_3' \oplus (r_2' \gg l/2)$	t_2
$s_3 = M_3'' \oplus (r_2'' \gg l/2)$	t_3
...	
$s_N = M_3''' \oplus (r_2''' \gg l/2)$	t_N

where the maximum size of the database is determined by l variable ($N = 2^l$).

We prove that an exhaustive search is not necessary to acquire s_j. Its acquisition means that information privacy, which should be one of the principal security objectives of the system, is not guaranteed. In fact, this attack is very harmful because after N authentication sessions, the attacker possesses the private information $\{s_j, t_j\}_{j=1}^N$ linked to each tag and stored on the back-end database. The security flaw described above arises mainly because the answer provided by the server depends only on the XOR between s_j and the random number r_2 selected by the tag/attacker. Finally, the reader should note that the detailed information D_j cannot be compromised as this kind of information is transmitted over a secure channel (e.g. SSL connection between the server and reader).

3.5. Tracking of future transactions

It is claimed that the mutual authentication protocol is secure against forward traceability even when the tag is compromised and the attacker knows t_i. In order to guarantee this security property, it is assumed that the attacker cannot prevent T_i from receiving the last message M_3, or that he does not have access to all the values r_1, r_2 and s_i that are needed to refresh t_i [7]. We will show that even with these restrictions, it is possible to compute the new identifier t_i. Details are given below:

1. The target tag and a legitimate reader execute a valid session. The reader generates a random string $r_1 \in_R \{0,1\}^l$ and sends r_1 to the tag. After receiving r_1, the tag chooses a random string $r_2 \in_R \{0,1\}^l$, and computes messages $M_1 = t_i \oplus r_2$ and $M_2 = f_{t_i}(r_1 \oplus r_2)$. The tag sends messages (M_1, M_2) to the reader. The valid reader then queries the server and gets the updating message M_3, which is forwarded to the target tag.
2. The tag is compromised, revealing the value r_2. Additionally we assume that the attacker knows messages M_1 and M_3 and the random number r_1 that were sent over the channel uncoded. The reader should note that all the above assumptions accord with Song et al.'s restrictions. At this point, the attacker can compute the new identifier in the following way:

$$t_i = h((M_1 \gg l/4) \oplus (M_3 \ll l/4) \oplus r_1 \oplus r_2). \qquad (8)$$

Proof. The verification of Eq. (8) is equivalent to verifying that $((s_i \ll l/4) \oplus (t_i \gg l/4))$ is equal to $(M_1 \gg l/4) \oplus (M_3 \ll l/4)$. Note that $t_i = h((s_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2)$ in the protocol definition.

We can start expanding the equations for M_1 and M_2. We use $[x]_{L/R-MSB/LSB}$ to symbolize the most and least significant bits of the left/right half of the string x.

$$M_1 = t_i \oplus r_2, \quad (9)$$
$$M_1 = [t_i]_{L-MSB} \oplus [r_2]_{L-MSB} || [t_i]_{L-LSB} \oplus [r_2]_{L-LSB} || [t_i]_{R-MSB} \oplus [r_2]_{R-MSB} || [t_i]_{R-LSB} \oplus [r_2]_{R-LSB},$$
$$M_3 = s_i \oplus (r_2 \ll l/2), \quad (10)$$
$$M_3 = [s_i]_{L-MSB} \oplus [r_2]_{R-MSB} || [s_i]_{L-LSB} \oplus [r_2]_{R-LSB} || [s_i]_{R-MSB} \oplus [r_2]_{L-MSB} || [s_i]_{R-LSB} \oplus [r_2]_{L-LSB}.$$

Rotating M_1 by $l/4$ bits to the right, we have

$$(M_1 \gg l/4) = (t_i \oplus r_2) \gg l/4, \quad (11)$$
$$(M_1 \gg l/4) = [t_i]_{R-LSB} \oplus [r_2]_{R-LSB} || [t_i]_{L-MSB} \oplus [r_2]_{L-MSB} || [t_i]_{L-LSB} \oplus [r_2]_{L-LSB} || [t_i]_{R-MSB} \oplus [r_2]_{R-MSB}.$$

Rotating M_3 by $l/4$ bits to the left, we have

$$(M_3 \ll l/4) = (s_i \oplus (r_2 \ll l/2)) \ll l/4, \quad (12)$$
$$(M_3 \ll l/4) = [s_i]_{L-LSB} \oplus [r_2]_{R-LSB} || [s_i]_{R-MSB} \oplus [r_2]_{L-MSB} || [s_i]_{R-LSB} \oplus [r_2]_{L-LSB} || [s_i]_{L-MSB} \oplus [r_2]_{R-MSB}.$$

Computing an XOR operation between Eqs. (11) and (12), we have

$$(M_1 \gg l/4) \oplus (M_3 \ll l/4) = [t_i]_{R-LSB} \oplus [s_i]_{L-LSB} || [t_i]_{L-MSB} \oplus [s_i]_{R-MSB} [t_i]_{L-LSB} \oplus [s_i]_{R-LSB} || [t_i]_{R-MSB} \oplus [s_i]_{L-MSB} = (s_i \ll l/4) \oplus (t_i \gg l/4). \quad \square$$

We prove that if a target tag is compromised, we are able to generate future tag identifiers under the assumptions made by Song et al.'s model. Therefore, the adversary can trace future transactions of the compromised tag. Additionally, the secret information s_i associated with the tag can be obtained simply by applying the attack described in Section 3.4.

3.6. De-synchronization attack on the secret update protocol

In Song's secret update protocol, the new owner/server updates a tag with message (r_1, M_1, M_2) and gets confirmation (r_2, M_3) from the tag if the update is successful. It is claimed that the server can be re-synchronized with the tag even if the confirmation message M_3 is blocked or incorrectly received. For that, the server maintains the new and old values of s_i and t_i. However, we will show that an active attacker can block the first message (r_1, M_1, M_2) from reaching the tag, and then send a second message (r_1, M_1', M_2'). This last message will be accepted by the tag, resulting in de-synchronization between the tag and the server. In fact, this attack is based on the same principles as the attack presented in [6] because of the similarities between the messages exchanged in the authentication protocol $\{M_1 = t_i \oplus r_2, M_3 = s_i \oplus (r_2) \gg l/2\}$ and the messages passed in the secret update protocol $\{M_1 = f_{t_i}(r_1) \oplus t_i', M_3 = s_i \oplus (t_i' \gg l/2)\}$. We briefly present the attack below and omit its proof to avoid repetition.

1. A legitimate server starts the secret update protocol by sending (r_1, M_1, M_2) to the tag, where $r_1 \in_R \{0,1\}^l$, $M_1 = f_{t_i}(r_1) \oplus t_i'$ and $M_2 = s_i \oplus (t_i' \gg l/2)$. Note that s_i and $t_i = h(s_i)$ are the tag's current secrets and that $s_i' \in_R \{0,1\}^l$ and $t_i' = h(s_i')$ are the new secrets to be updated.
2. The adversary blocks the message (r_1, M_1, M_2) from reaching the target tag. Then, the adversary sends (r_1, M_1', M_2'), where $M_1' \in_R \{0,1\}^l$ and $M_2' = M_2 \oplus ((M_1 \oplus M_1') \gg l/2)$.
3. The tag will accept (r_1, M_1', M_2') and update its secret $t_i = M_1' \oplus f_{t_i}(r_1)$ to a new value, which is absolutely unknown to the server.

So, the attacker can de-synchronize the tag and the new owner due to the high linearity of the messages M_1 and M_3.

4. Conclusion

In this paper, we have analyzed a series of protocols related to RFID ownership transfer, including a mutual authentication protocol, an ownership transfer protocol, and a secret update protocol. We emphasize that although many of our attacks are described with reference to the authentication protocol, they are also directly applicable to the ownership transfer protocol because the messages exchanged in these two protocols are identical. Our research shows that the mutual authentication protocol and the ownership transfer protocol do not guarantee most of the security properties required in their design (e.g. information and location privacy). Moreover, we have shown that the update protocol is vulnerable to a de-synchronization attack.

References

[1] D. Dolev, A.C. Yao, On the security of public key protocols, Technical report, Stanford, CA, USA, 1981.
[2] S. Fouladgar, H. Afifi, A simple privacy protecting scheme enabling delegation and ownership transfer for RFID tags, Journal of Communications 2 (6) (2007) 6–13.
[3] A. Juels, S. Weis, Defining strong privacy for RFID, in: Proceedings of IEEE PerCom07, 2007, pp. 342–347. Full version available at IACR ePrint Archive, <http://eprint.iacr.org/2006/137>.
[4] D. Molnar, A. Soppera, D. Wagner, A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags, in: Proceedings of Selected Areas in Cryptography SAC'05, August 2005, LNCS, vol. 3897, Springer-Verlag, 2005, pp. 276–290.
[5] K. Osaka, T. Takagi, K. Yamazaki, O. Takahashi, An efficient and secure RFID security method with ownership transfer, in: Proceedings of Computational Intelligence and Security CIS'06, September 2006, LNCS, vol. 4456, Springer-Verlag, 2006, pp. 778–787.
[6] P. Rizomiliotis, E. Rekleitis, S. Gritzalis, Security analysis of the Song–Mitchell authentication protocol for low-cost RFID tags, Communications Letters, IEEE 13 (4) (2009) 274–276.
[7] B. Song, RFID tag ownership transfer, in: Proceedings of Workshop on RFID Security, Budapest, Hungary, July 2008.
[8] B. Song, C.J. Mitchell, RFID authentication protocol for low-cost tags, in: Proceedings of the ACM Conference on Wireless Network Security (WiSec'08), April 2008, ACM Press, Alexandria, Virginia, USA, 2008, pp. 140–147.
[9] T. van Deursen, S. Radomirovic, Attacks on RFID protocols, Cryptology ePrint Archive, Report 2008/310, 2008. <http://eprint.iacr.org/>.

Pedro Peris-Lopez has an M.Sc. in Telecommunications Engineering and a Ph.D. in Computer Science. His research interests include protocol and primitive design, lightweight cryptography and cryptanalysis. He is currently focusing on Radio Frequency Identification (RFID) systems and doing a PosDoc on RFID security at the Information and Communication Theory Group of Delft University of Technology. For additional information visit: http://www.lightweight-cryptography.com.

Please cite this article in press as: P. Peris-Lopez et al., Vulnerability analysis of RFID protocols for tag ownership transfer, Comput. Netw. (2010), doi:10.1016/j.comnet.2009.11.007

Julio C. Hernandez-Castro has a degree in Maths and a MSc in Coding Theory and Network Security. He has worked heavily in the past on the applications of artificial intelligence techniques (notably evolutionary computation) to Cryptography and Cryptanalysis. He got his Ph.D. in Computer Science from Carlos III University in Madrid in 2003 and was there till February 2009 when he was appointed Senior Lecturer in the School of Computing of Portsmouth University in the UK. He has published more than 30 papers in international journals and more than 50 in international conferences. He is a keen chess player.

Juan M. E. Tapiador is Research Associate in the Department of Computer Science Department at the University of York, UK. He holds a M.Sc. in Computer Science from the University of Granada (2000), where he obtained the Best Student Academic Award, and a Ph.D. in Computer Science (2004) from the same university. His research interests include cryptography, steganography and computer security.

Tieyan Li is a senior researcher at Institute for Infocomm Research (I^2R, Singapore) from Oct. 2001. He obtained his Ph.D. degree in 2003 at School of Computing, National University of Singapore. Dr. Li is experienced in practical system developments such as networking, system integration and software programming. He is also active in academic security research fields with tens of journal and conference publications and several patents. Currently his areas of research are in applied cryptography and network security, as well as security issues in RFID, sensor, multimedia and tamper resistant hardware/software, etc. Dr. Li has served as the PC member and reviewer for a number of security conferences and journals.

Yingjiu Li is a tenure-track assistant professor working in the area of information security. He has published more than 60 research papers in journals and conferences including IEEE Transactions on Dependable and Secure Computing, Journal of Computer Security, IEEE Symposium on Security and Privacy, ACM Conference on Computer and Communications Security, USENIX Security Symposium, ACM Conference on Information, Computer and Communications Security, and European Symposium on Research in Computer Security. He has served in more than twenty conference program committees over the past three years including the program committees for the IEEE Symposium on Security and Privacy in 2007 and 2008. Yingjiu Li is a senior member of the ACM. The URL for his web page is http://www.mysmu.edu/faculty/yjli/.

Please cite this article in press as: P. Peris-Lopez et al., Vulnerability analysis of RFID protocols for tag ownership transfer, Comput. Netw. (2010), doi:10.1016/j.comnet.2009.11.007