Journal of Computational and Applied Mathematics 259 (2014) 571–577

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

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

Cryptanalysis of the Cho et al. protocol: A hash-based RFID tag mutual authentication protocol

Masoumeh Safkhani[a], Pedro Peris-Lopez[b], Julio Cesar Hernandez-Castro[c], Nasour Bagheri[d,*]

[a] *Electrical Engineering Department, Iran University of Science and Technology, Tehran, Iran*
[b] *Computer Science Department, University Carlos III of Madrid, Spain*
[c] *School of Computing, University of Kent, Cornwallis South Building, Canterbury Kent CT2 7NF, UK*
[d] *Electrical Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran*

ARTICLE INFO

Article history:
Received 15 February 2013
Received in revised form 6 June 2013

Keywords:
RFID
Privacy
Authentication
De-synchronization attack
Tag impersonation attack
Reader impersonation attack

ABSTRACT

Radio frequency identification systems need secure protocols to provide confidentiality, privacy protection, mutual authentication, etc. These protocols should resist active and passive attacks such as forgery, traceability, replay and de-synchronization attacks. Cho et al. recently proposed a hash-based mutual authentication protocol (Cho et al., 2012) and claimed that their scheme addresses all privacy (Juels, 2006) and forgery concerns (Dimitriou, 2005; Yang et al., 2005) linked to RFID technology. However, we show in the following that the protocol fails to bear out many of the authors' security claims, which renders the protocol useless. More precisely, we present the following attacks on this protocol:

1. *De-synchronization attack*: the success probability of the attack is 1 while the attack complexity is one run of the protocol.
2. *Tag impersonation attack*: the success probability of the attack is $\frac{1}{4}$ for two runs of the protocol.
3. *Reader impersonation attack*: the success probability of the attack is $\frac{1}{8}$ for two runs of the protocol.

We also show an additional and more general attack, which is still possible when the conditions needed for the ones above do not hold, and that highlights the poor design of the group *ID* (RID_i^t). Additionally we show how all the above mentioned attacks are applicable against another protocol, highly reminiscent of that of Cho et al. (2012) and designed in Cho et al. (2011), and also against an enhanced version of the Cho et al. protocol proposed by Kim (2012). Finally we end up by showing how slight modifications in the original protocol can prevent the aforementioned security faults.

1. Introduction

Radio frequency identification (RFID) technology is a relatively new wireless technology[1] that may have a great capability to influence many aspects of life in the near future as a promising pervasive computing solution. It has already been used

* Corresponding author. Tel.: +98 912 6231332.
E-mail addresses: M_Safkhani@iust.ac.ir (M. Safkhani), pperis@inf.uc3m.es (P. Peris-Lopez), J.C.Hernandez-Castro@kent.ac.uk (J.C. Hernandez-Castro), nbagheri@srttu.edu, na.bagheri@gmail.com (N. Bagheri).

1 Some authors claim that it has antecedents as far back as the Friend or Foe (FoF) systems used during WWII.

0377-0427/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cam.2013.09.073

in libraries, e-passports, manufacturing, inventory control, supply chain management, e-health and many other areas [1–3]. Tags, readers and back-end databases are generally considered to be the three basic components of most commonly deployed RFID systems. Tags are generally attached to the objects that they identify through the exchange of radio signals with the reader. The back-end database (DB) mainly aids the reader, with its extra storage and further computational capabilities. The main problem currently affecting RFID systems is data security, particularly in the reader to tag channel (backward and forward channels), as the reader to DB channel can afford to employ classical cryptographic solutions such as SSH or SSL, but the tag, in general, cannot. These associated security problems may waive away all the benefits of the RFID technology and seriously limit its wide deployment and overall success. For instance, an RFID system may be subject to privacy problems, for the object which is supposed to be identified through the tag can be tracked, together with its owner, by simply repeatedly communicating with the tag in a completely unassisted and transparent way. Furthermore, it is quite challenging to find solutions in these very constrained environments that can ensure that, on top of privacy being achieved, the customer's data will remain secure. Additionally, instead of stealing private information (data and location), an adversary may aim to estimate the cardinality of tags (counting attacks [4]). And these are only some simple examples of the many different and sometimes contradictory security requirements that we might need in these constraint environments where we cannot apply classical and well known security solutions. Hence the need for new security proposals, including new protocols such as the several RFID mutual authentication protocols proposed lately [5–10]. Unfortunately, the security of many of these has been seriously questioned [11–16].

Recently Cho et al. proposed a hash-based mutual authentication protocol [17] and claimed that their protocol addresses all privacy [18] and forgery concerns [19,20] present in RFID systems. However, we show in the following that their protocol falls quite short of the authors' security claims. More precisely, we present tag and reader impersonation and de-synchronization attacks. All attacks have in common both a high success probability and negligible complexity, thus making a strong case against the further use and deployment of the Cho et al. protocol.

The rest of the paper is organized as follows: In Section 2 we describe the notation and some preliminaries used through the paper. We briefly review the Cho et al. protocol in Section 3. Our de-synchronization, tag impersonation and reader impersonation attacks are presented in Section 4. In Section 5 we analyze the security of two other related protocols and show how they are as insecure as the Cho et al. scheme. Concluding remarks are presented in Section 7.

2. Preliminaries

The notation used throughout this paper is as follows:

• ID_k:	Identifier of the *k*th tag.
• $h(.)$:	One-way hash function.
• $\|$:	A concatenation operation.
• $\oplus$:	Exclusive-or operation.
• s^t and s^r:	Secret values shared between the tag and back-end database.
• s_i^t and s_i^r:	Secret values used in the *i*th session.
• $DATA^k$:	*k*th tag's related information.
• $F_y^x(z)$, RID:	$F_y^x(z)$ is a function that has z as an argument and performs an operation based on x; y is the communication session information. RID is the resulting value.
• R^r:	Random number generated by the reader.
• R^t:	Random number generated by the tag.
• s_j:	The secret value used in the *j*th session.
• RID_i:	A 96-bit group ID random number.
• α:	Message generated by a tag for authentication.
• $X_{(a:b)}$:	A substring of X including all consecutive bits from the *a*th to the *b*th.
• X^i:	Parameter X related to the *i*th tag.

Adversary model: In the proposed attacks of this paper, the attacker is an active adversary who is able to intercept, modify and block the ongoing reader–tag communication without being detected. Note that the protocol's designers considered such an adversary in their analysis [17]. Moreover any secure protocol should withstand both passive and active attackers.

3. The Cho et al. hash-based RFID mutual authentication protocol

Cho et al. [17] recently proposed a mutual authentication protocol for RFID systems. Their protocol uses a one-way hash function, and is claimed to provide enough security against various of the most common security attacks linked to RFID systems. To protect privacy, each session of mutual authentication is randomized by employing two nonces R^r and R^t, generated by the reader and the tag respectively, and using two fresh values denoted by RID_i^t and RID_i^r which are derived from

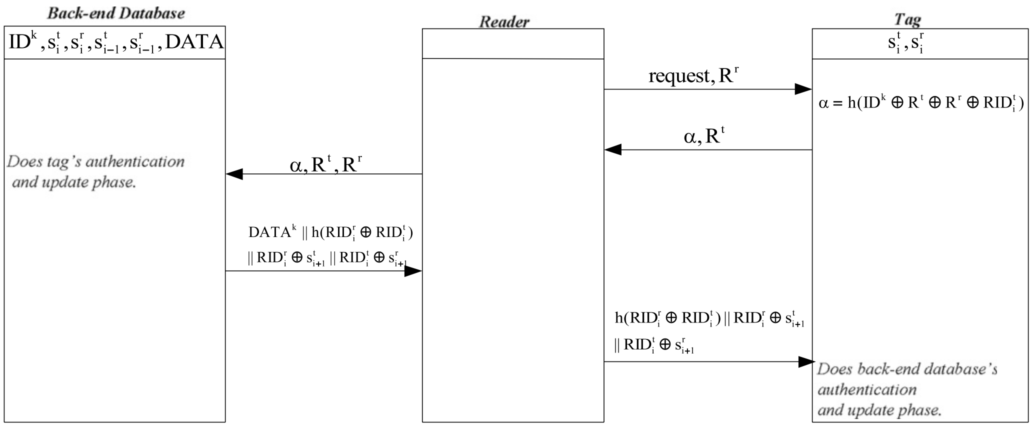
Back-end Database
ID^k, s_i^t, s_i^r, s_{i−1}^t, s_{i−1}^r, DATA
Reader
Tag
s_i^t, s_i^r
request, R^r
α = h(ID^k ⊕ R^t ⊕ R^r ⊕ RID_i^t)
α, R^t
Does tag's authentication and update phase.
α, R^t, R^r
DATA^k || h(RID_i^r ⊕ RID_i^t) || RID_i^r ⊕ s_{i+1}^t || RID_i^t ⊕ s_{i+1}^r
h(RID_i^r ⊕ RID_i^t) || RID_i^r ⊕ s_{i+1}^t || RID_i^t ⊕ s_{i+1}^r
Does back-end database's authentication and update phase.

Fig. 1. The Cho et al. hash-based RFID tag mutual authentication protocol.

R^t and R^r respectively. Since the secret value of the tag s_j^t gets updated at each successful protocol run, the back-end database keeps a record of the two latest secret values used, denoted by s_{i-1}^t, s_{i-1}^r, s_i^t and s_i^r respectively, to combat de-synchronization attacks. The protocol—see Fig. 1—works as follows:

1. The reader generates a random number R^r and sends *request* along with R^r to the tag.
2. Upon its reception, the tag generates another random number R^t and acts as follows:
   (a) It computes $RID_i^t = F_i^{s^t}(R^t) = (R^t - R^t \bmod s^t + 1)_{(0:47)} \| (R^t + s^t - R^t \bmod s^t)_{(48:95)}$ and $\alpha = h(ID^k \oplus R^t \oplus R^r \oplus RID_i^t)$.
   (b) It sends α and R^t to the reader.
3. Once the reader receives α and R^t, it passes them with R^r to the back-end database.
4. To authenticate the tag and update the secret values s^t and s^r, the back-end database acts as follows:
   (a) For any record i in its database (the ith record includes $(ID_i^k, s_{i-1}^t, s_{i-1}^r, s_i^t, s_i^r, DATA_i^k)$ for a tag), it computes the group ID $(RID_i^{t'})$ and $\alpha' = h(ID^k \oplus R^t \oplus R^r \oplus RID_i^{t'})$ for each tuple $(ID_i^k, s_{i-1}^t, s_{i-1}^r)$ and (ID_i^k, s_i^t, s_i^r).
   (b) If it finds a match between the received α and the retrieved α', it authenticates the tag and updates its record. Assuming that (ID_i^k, s_i^t, s_i^r) is a tuple for which tag i has been authenticated, the back-end database updates the record of the authenticated tag as follows:
      - it assigns s_i^t to s_{i-1}^t and s_i^r to s_{i-1}^r,
      - it generates new secret values s_{i+1}^t and s_{i+1}^r, and assigns them to s_i^t and s_i^r respectively.
   (c) The back-end database generates $DATA^k \| h(RID_i^r \oplus RID_i^t) \| RID_i^r \oplus s_{i+1}^t \| RID_i^t \oplus s_{i+1}^r$ and sends it to the reader.
5. The reader acquires the information of the tagged object from the message received and passes $h(RID_i^r \oplus RID_i^t) \| RID_i^r \oplus s_{i+1}^t \| RID_i^t \oplus s_{i+1}^r$ to the tag.
6. Once the message has been received, the tag generates RID_i^r as $RID_i^r = F_i^{s^r}(R^r) = (R^r - R^r \bmod s^r + 1)_{(0:47)} \| (R^t + s^r - R^r \bmod s^r)_{(48:95)}$. Then it verifies the correctness of the received $h(RID_i^r \oplus RID_i^t)$ in order to authenticate the back-end database.
7. If the tag has authenticated the back-end database, it extracts s_{i+1}^t and s_{i+1}^r from $RID_i^r \oplus s_{i+1}^t$ and $RID_i^t \oplus s_{i+1}^r$ respectively and updates its secret value s_i^t to s_{i+1}^t and s_i^r to s_{i+1}^r.

The authors claimed several security properties for the protocol [17, Section 5] including but not limited to the following properties: (1) resistance against de-synchronization attacks; (2) resistance against reader impersonation attacks, in which the adversary sends carefully crafted or random requests with different $h(RID_i^r \oplus RID_i^t)$ values, attempting to be authenticated by the tag; (3) resistance against tag impersonation attacks, in which the adversary tries to generate a valid α in order to be authenticated by the reader as a valid tag. However, in the following sections we present several attacks on this protocol that contradict the authors' security claims.

4. Cryptanalysis of the Cho et al. protocol

4.1. De-synchronization attack

Cho et al. [17] claim that their protocol is resistant against de-synchronization attacks via keeping a record of the *old* secret value of s to avoid from getting de-synchronized when a tag does not receive correctly the last message of a protocol run. However, we observed a flaw on the protocol that can be used to de-synchronize both the tag T_i and the reader R quite easily.[2] More precisely, the adversary can follow the steps described below:

[2] An early draft of these results is available on [21]. Later, similar results were reported in [22].

1. Eavesdrops on one session of protocol.
2. Changes the last message that was sent by R to T_i from $h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r$ to $h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t \oplus \Delta\|RID_i^t \oplus s_{i+1}^r \oplus \Delta'$, for $\Delta \neq 0$ and $\Delta' \neq 0$.
3. The tag authenticates the reader on the basis of the received $h(RID_i^r \oplus RID_i^t)$ and assigns $s_i^t \oplus \Delta$ to s_{i+1}^t and $s_i^r \oplus \Delta'$ to s_{i+1}^r.

Following the proposed attack, the secret values contained in T_i are set to $s_{i+1}^t \oplus \Delta$ and $s_{i+1}^r \oplus \Delta'$ while the stored values on the back-end database are s_i^t, s_i^r, s_{i+1}^t and s_{i+1}^r and the reader has no record of $s_{i+1}^t \oplus \Delta$ and $s_{i+1}^r \oplus \Delta'$. Hence, the back-end database will never authenticate T_i in future sessions of the protocol. The success probability of our de-synchronization attack is 1 and the complexity of the attack is only one run of the protocol.

4.2. Tag impersonation attack

Cho et al. [17] claim that it would not be possible for an adversary to generate a valid message (α, R^t) such that the back-end database will authenticate it as a valid tag. More precisely, the authors state that to generate a valid α and R^t and impersonate the tag, the adversary at least has to find the secret values s_i^t and ID^k, which are protected by $h(.)$. However, we present a rather simple attack in which an adversary can impersonate a legitimate tag without any knowledge of the secret values s_i^t and ID^k. Our attack is based on the fact that for $a < b$ we can state that

$$a \bmod b \equiv a.$$

Given this simple fact and assuming that $R < s$ we have

$$F_i^s(R) = RID_i = (R - R \bmod s + 1)_{(0:47)}\|(R + s - R \bmod s)_{(48:95)} = (1)_{(0:47)}\|(s)_{(48:95)}$$

which is independent from R. Now, we can use this observation to mount a tag impersonation attack, as described below:

1. Adversary eavesdrops on one protocol session and obtains R^r, α, R^t and $h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r$. Assuming that $R^t < s_j^t$ then we will have that $RID_i^t = (1)_{(0:47)}\|(s_j^t)_{(48:95)}$.
2. On the next protocol session, when the reader sends *request* along with R'^r, the adversary impersonates the tag by replying with the tuple $\alpha, R^t \oplus R^r \oplus R'^r$.
3. The back-end database uses the values (s_j^t, R'^t) of the tag to generate $RID_i'^t$.
4. The back-end database uses ID^k, R'^t, R'^r and $RID_i'^t$ to verify whether $\alpha \overset{?}{=} h(ID^k \oplus R'^t \oplus R'^r \oplus RID_i'^t)$.
5. If $R^t < s_j^t$ then $RID_i'^t = (1)_{(0:47)}\|(s^t)_{(48:95)} = RID_i^t$ and we have
$$h(ID^k \oplus R'^t \oplus R'^r \oplus RID_i'^t) = h(ID^k \oplus R^t \oplus R^r \oplus R'^r \oplus R'^r \oplus RID_i^t) = h(ID^k \oplus R^t \oplus R^r \oplus RID_i^t) = \alpha.$$
6. Since $\alpha = h(ID^k \oplus R'^t \oplus R'^r \oplus RID_i'^t)$, the back-end database authenticates the adversary as a legitimate tag, and the impersonation attack is successful.

For the success of this attack, the adversary needs that both of the aforementioned conditions hold. For a random selection of R^t and R^r, the probability of each assumption is $\frac{1}{2}$. Hence the success probability of the proposed tag impersonation attack is $\frac{1}{4}$, and the complexity of the attack is only two protocol runs, with negligible memory and computational requirements.

Remark 1. The described attack works as long as the tag has not updated its secret value s^t. However, when the adversary carries out the eavesdropping phase described in Step 1 of the above attack, if it additionally blocks the last protocol message in which the reader sends $h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r$ to the tag, then the attack can be used even after one update of the secret value s. The reason is derived from the protocol mechanism against de-synchronization attacks, in which it keeps a record of s_{old} in the back-end database.

4.3. Reader impersonation attack

The authors of the proposal under study [17] claim that their protocol security is in part due to an adversary having no control at all of the values R^t and RID_i^t that in addition are generated and changed on every session, even in cases where the secret values s^t and s^r remain non-updated. However, this reasoning is flawed. We present in the following an attack in which an attacker can impersonate a legitimate reader, without requiring any knowledge of the secret values s^t, s^r and ID^k, or any control over the generated R^t. Our attack is based on the given observation that for $R^t < s_j^t$ one can state that

$$RID_i^t = (1)_{(0:47)}\|(s_j^t)_{(48:95)}$$

which is independent from R^t. The same argument is valid for RID_i^r, where for $R^r < s_j^r$ one can state that

$$RID_i^r = (1)_{(0:47)}\|(s_j^r)_{(48:95)}$$

which is independent from R^r. The proposed reader impersonation attack is described below:

1. The adversary eavesdrops on one session of protocol and obtains R^r, α, R^t and $h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r$, where for $R^t < s^t$ and $R^r < s^r$ one can state that
$$RID_i^t = (1)_{(0:47)}\|(s^t)_{(48:95)},$$
$$RID_i^r = (1)_{(0:47)}\|(s_j^r)_{(48:95)}.$$
2. She then blocks the last message from the reader to the tag, $h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r$. Hence, the tag does not update its secret values s_i^t and s_i^r.
3. The adversary supplants a legitimate reader and sends *request* with the stored R^r to the tag and receives the tag's response, α' and R'^t.
4. For $R'^t < s^t$ and $R^r < s^r$ we can state that
$$RID_i'^t = (1)_{(0:47)}\|(s^t)_{(48:95)} = RID_i^t,$$
$$RID_i'^r = (1)_{(0:47)}\|(s^r)_{(48:95)} = RID_i^r.$$
5. The adversary replies to the tag with $h(RID_i^r \oplus RID_i^t)\|\Delta\|\Delta'$, where Δ and Δ' can be any random values.
6. For the (four) given assumptions, $h(RID_i^r \oplus RID_i^t) = h(RID_i'^r \oplus RID_i'^t)$ and the tag authenticates the adversary as a legitimate reader.

Like in the attack presented in Section 4.2, the adversary will succeed in her attack if the assumptions are correct, i.e., $R^t < s^t$, $R^r < s^r$, $R'^t < s^t$ (and $R^r < s^r$). For a random selection of R^t, R^r and R'^t, the success probability of each assumption is $\frac{1}{2}$. Hence, the total probability of the above reader impersonation attack is $\frac{1}{8}$ and the complexity of the attack is eavesdropping on one execution of the protocol and supplanting a subsequent session.

Remark 2. The given attack de-synchronizes the tag from the reader, because after its execution the tag updates its secret values s^t and s^r to $s_j^t = RID_i'^r \oplus \Delta$ and $s_j^r = RID_i'^t \oplus \Delta'$ respectively, but the legitimate reader has no knowledge of it.

4.4. Other weaknesses

The attacks presented in the previous sections heavily rely on the poor design of the group *ID* (RID_i^t) and the fact that under a relatively common set of conditions it is to a large extent independent from R^t, a fact that is quite contrary to the first intuition when we see how it is constructed:
$$RID_i^t = (R^t - R^t \bmod s_i^t + 1)_{(0:47)}\|(R^t + s_i^t - R^t \bmod s_i^t)_{(48:95)}$$
and that no doubt was missed by the designers. But this is not all: Further attacks are possible even if the starting condition for our impersonation attacks, that is $R^t < s^t$, does not hold. In this case, RID_i^t is still independent from R^t and we will have that $RID_{low}^t = M * s_j^t + 1$ and $RID_{high}^t = K * s_j^t$ with M, K small integers, typically 1 or 2.

As a consequence of this and the uninspired mix of RID_i^t and RID_i^r in the hash function, the value of the XOR between the two has, in many cases, a very low entropy with probability around $\frac{1}{4}$. Values such as $0x11de01f7181ad000000000001$, $0xd762106bdb5e000000000003$, $0xefa4ae6d8b11000000000007$ and $0x15153139fadac00000000000f$ are quite characteristic. This, apart from being a quite serious design error, could allow for a guessing attack on the hash function that can be performed in about 2^{48} hash operations (finding the correct input value) instead of the intended 2^{96} — considering variables of 96 bits. Although probably not very practical, this is undeniably another bad security property that, in our view, completely undermines an already quite faulty protocol.

5. Related works

Cho et al. in [23] proposed another RFID hash-based mutual authentication protocol. Unfortunately, the protocol is highly inspired by the Cho et al. scheme of [17], inheriting identical design criteria and using the same weak group ID (RID_i). The analysis is straightforward and all the above mentioned attacks are also applicable on this new version. To avoid repetition, we omit its description.

In [24] Kim also analyzed the Cho et al. scheme in relation to de-synchronization attacks and proposed an improved version of this scheme. In fact he only modified the message generated in the step 4.(c) of the original scheme. In detail, the back-end database generates the following message and sends it to the reader:
$$DATA^k\|h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r\|h(s_{i+1}^t \oplus s_{i+1}^r).$$

The first four sub-messages are equal to the original ones and the only $h(s_{i+1}^t \oplus s_{i+1}^r)$ sub-message is added. It aims to guarantee the integrity of the generated s_{i+1}^t and s_{i+1}^r and offer protection against the attack introduced in Section 4.1 (equivalently also in [24]). Despite this modification, this enhanced version is as insecure as its predecessor. First of all, Kim et al. did not take any precaution to avoid our tag impersonation and reader impersonation attacks presented in Sections 4.2 and 4.3. Therefore, the above attacks can be successfully applied against this scheme (with some minor modifications in the case of the reader impersonation attack) and are very harmful from the security point of view. Secondly and despite Kim's claims, the scheme suffers from de-synchronization attacks. The attack can be built on the basis of the procedure exposed in Section 4.1 and the fact that $\forall x, x \oplus x = 0$. Specifically the adversary acts as follows:

1. Eavesdrops on one session of protocol.
2. She alters the last message that was sent from R to T_i from $h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r\|h(s_{i+1}^t \oplus s_{i+1}^r)$ to $h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t \oplus \Delta\|RID_i^t \oplus s_{i+1}^r \oplus \Delta\|h(s_{i+1}^t \oplus s_{i+1}^r)$, for $\Delta \neq 0$.
3. The tag authenticates the reader on the basis of the received $h(RID_i^r \oplus RID_i^t)$, extracts $s_{i+1}'^t = s_{i+1}^t \oplus \Delta$ and $s_{i+1}'^r = s_{i+1}^r \oplus \Delta$, and verifies whether $h(s_{i+1}'^t \oplus s_{i+1}'^r)$ is equal to the received $h(s_{i+1}^t \oplus s_{i+1}^r)$. It holds because $h(s_{i+1}'^t \oplus s_{i+1}'^r) = h(s_{i+1}^t \oplus \Delta \oplus s_{i+1}^r \oplus \Delta) = h(s_{i+1}^t \oplus s_{i+1}^r)$. Therefore the tag authenticates the back-end database and accepts the received secret values $s_{i+1}'^t$ and $s_{i+1}'^r$.

Once the attack is executed, the secret values kept in the T_i memory are set to $\{s_{i+1}^t \oplus \Delta, s_{i+1}^r \oplus \Delta\}$ while the values stored on the back-end database are $\{s_i^t, s_i^r, s_{i+1}^t, s_{i+1}^r\}$, and the reader has no record of $s_{i+1}^t \oplus \Delta$ and $s_{i+1}^r \oplus \Delta$. Hence the back-end database will never authenticate T_i in future sessions of the protocol. The de-synchronization attack succeeds with probability 1 and the complexity of the attack is only one run of the protocol. From all the above, we can conclude that the two new versions ([23,24]) of the Cho et al. protocol are as insecure as the original one.

6. Countermeasures

In this section we show how the design of the Cho et al. protocol can be slightly modified in such a way that it offers protection against our attacks. The proposed attacks in this article exploit the poor design of the group ID (RID_i^t and RID_i^r). Furthermore the scheme abuses bitwise and arithmetic operations such as XOR, subtraction and module. As consequence of this, some exchanged messages are independent of the nonces linked to each section (e.g., RID_i^t is independent of R_i^t).

To overcome these weaknesses, the design of RID_i^t and RID_i^r has to be revised in such a way that any modification of the random numbers involved in the session is spread over all resulting bits. This can be achieved by using the hash function: $RID_i^t = h(R_i^t \oplus s_i^t)$ and $RID_i^r = h(R_i^r \oplus s_i^r)$. If these two values are used, the last sub-message in step 4(c) of the protocol has also to be modified. A possible alternative can be the message $DATA^k\|h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r\|h(s_{i+1}^t\|s_{i+1}^r)$. Hence, the improved protocol works as follows:

1. The reader generates a random number R^r and sends *request* along with R^r to the tag.
2. Upon its reception, the tag generates another random number R^t and acts as follows:
   (a) It computes $RID_i^t = h(R_i^t \oplus s_i^t)$ and $\alpha = h(ID^k \oplus R^t \oplus R^r \oplus RID_i^t)$.
   (b) It sends α and R^t to the reader.
3. Once the reader receives α and R^t, it passes them with R^r to the back-end database.
4. To authenticate the tag and update the secret values s^t and s^r, the back-end database acts as follows:
   (a) For any record i in its database (the ith record includes $(ID_i^k, s_{i-1}^t, s_{i-1}^r, s_i^t, s_i^r, DATA_i^k)$ for a tag), it computes the group ID, $RID_i^{t'}$ and $\alpha' = h(ID^k \oplus R^t \oplus R^r \oplus RID_i^{t'})$ for each tuple $(ID_i^k, s_{i-1}^t, s_{i-1}^r)$ and (ID_i^k, s_i^t, s_i^r).
   (b) If it finds a match between the received α and the retrieved α', it authenticates the tag and updates its record. Assuming that (ID_i^k, s_i^t, s_i^r) is a tuple for which tag i has been authenticated, the back-end database updates the record of the authenticated tag as follows:
      - it assigns s_i^t to s_{i-1}^t and s_i^r to s_{i-1}^r,
      - it generates new secret values s_{i+1}^t and s_{i+1}^r, and assigns them to s_i^t and s_i^r respectively.
   (c) The back-end database generates $DATA^k\|h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r\|h(s_{i+1}^t\|s_{i+1}^r)$ and sends it to the reader.
5. The reader acquires the information of the tagged object from the message received and passes $h(RID_i^r \oplus RID_i^t)\|RID_i^r \oplus s_{i+1}^t\|RID_i^t \oplus s_{i+1}^r\|h(s_{i+1}^t\|s_{i+1}^r)$ to the tag.
6. Once the message has been received, the tag generates RID_i^r as $RID_i^r = h(R_i^r \oplus s_i^r)$. Then it verifies the correctness of the received $h(RID_i^r \oplus RID_i^t)$ in order to authenticate the back-end database.
7. If the tag has authenticated the back-end database, it extracts s_{i+1}^t and s_{i+1}^r from $RID_i^r \oplus s_{i+1}^t$ and $RID_i^t \oplus s_{i+1}^r$ respectively, and verifies the integrity through the received $h(s_{i+1}^t\|s_{i+1}^r)$ to update its secret values s_i^t to s_{i+1}^t and s_i^r to s_{i+1}^r.

After the given modification, the scheme offers protection against common attacks. This is because if the attacker alters any bit of R_t or R_r, the output of RID_i^t or RID_i^r will be different and unpredictable — this is guaranteed by the use of the cryptographic hash function. Therefore, this modification renders our tag/reader impersonation attacks ineffective. Additionally if an adversary attempts to impersonate a reader after eavesdropping on a protocol session (de-synchronization attack), she will not know the nonce RID_i^r generated by the reader and will not be able to control it since this random number is contributed by the tag and protected by using the hash function. As a result the adversary fails in her attempt to impersonate the reader and would be detected if she alters the message sent in step 4(c) of the protocol. That is, the de-synchronization attack is useless against the improved protocol. Summarizing the protocol offers a higher security level with the penalty of needing two extra calls to the hash function $h(.)$ in comparison to the original protocol.

7. Conclusion

In this paper we present a security analysis of the Cho et al. hash-based mutual authentication protocol [17]. We show de-synchronization, tag impersonation and reader impersonation attacks with a high success probability (i.e., 1, $\frac{1}{4}$ and $\frac{1}{8}$, respectively), needing to eavesdrop on two or one protocol sessions and with associated negligible memory and computational costs. This clearly shows serious flaws in the proposal, which contradicts the security claims of the authors and its feasibility as a secure protocol. Moreover we show how two new protocols [23,24] highly reminiscent of that from [17] suffer from the same security faults, which reveals each of them to be an insecure authentication mechanism. Finally we show how slight modifications in the original protocol increase the security level provided by the scheme and destroy the workability of the above mentioned attacks.

References

[1] Wen-Tsai Sung, Ming-Han Tsai, Data fusion of multi-sensor for IOT precise measurement based on improved PSO algorithms, Computers & Mathematics with Applications 64 (5) (2012) 1450–1461.

[2] Xiaowei Zhu, Samar K. Mukhopadhyay, Hisashi Kurata, A review of RFID technology and its managerial applications in different industries, Journal of Engineering and Technology Management 29 (1) (2012) 152–167.

[3] Xiaoli Xu, Tao Chen, Mamoru Minami, Intelligent fault prediction system based on internet of things, Computers & Mathematics with Applications 64 (5) (2012) 833–839.

[4] Zhuzhong Qian, Ce Chen, Ilsun You, Sanglu Lu, ACSP: a novel security protocol against counting attack for UHF RFID systems, Computers & Mathematics with Applications 63 (2) (2012) 492–500.

[5] Chien Hung-Yu, SASI: a new ultralightweight RFID authentication protocol providing strong authentication and strong integrity, IEEE Transactions on Dependable and Secure Computing 4 (4) (2007) 337–340.

[6] Lars Kulseng, Zhen Yu, Yawen Wei, Yong Guan, Lightweight mutual authentication and ownership transfer for RFID systems, in: Proceedings of IEEE INFOCOM 2010, pp. 1–5, 2010.

[7] Yu-Yi Chen, Meng-Lin Tsai, Jinn-Ke Jan, The design of RFID access control protocol using the strategy of indefinite-index and challenge–response, Computer Communications 34 (3) (2011) 250–256.

[8] Tzu-Chang Yeh, Chien-Hung Wu, Yuh-Min Tseng, Improvement of the RFID authentication scheme based on quadratic residues, Computer Communications 34 (3) (2011) 337–341.

[9] Dang Nguyen Duc, Kwangjo Kim, Defending RFID authentication protocols against DoS attacks, Computer Communications 34 (3) (2011) 384–390.

[10] Chiu Chiang Tan, Bo Sheng, Qun Li, Secure and serverless RFID authentication and search protocols, IEEE Transactions on Wireless Communications 7 (4) (2008) 1400–1407.

[11] Mihály Bárász, Balázs Boros, Péter Ligeti, Krisztina Lója, Dániel Nagy, Passive attack against the M2AP mutual authentication protocol for RFID tags, in: Proceedings of First International EURASIP Workshop on RFID Technology, 2007.

[12] Tianjie Cao, Elisa Bertino, Hong Lei, Security analysis of the SASI protocol, IEEE Transactions on Dependable and Secure Computing 6 (1) (2009) 73–77.

[13] Julio C. Hernandez-Castro, Juan M.E. Tapiador, Pedro Peris-Lopez, Jean-Jacques Quisquater, Cryptanalysis of the SASI ultralightweight RFID authentication protocol with modular rotations, Technical report.

[14] Raphael C.-W. Phan, Cryptanalysis of a new ultralightweight RFID authentication protocol—SASI, IEEE Transactions on Dependable and Secure Computing 6 (4) (2009) 316–320.

[15] Tieyan Li, Robert H. Deng, Vulnerability analysis of EMAP—an efficient RFID Mutual Authentication Protocol, in: Second International Conference on Availability, Reliability and Security – AReS 2007, Vienna, Austria, April 2007.

[16] Mihály Bárász, Balázs Boros, Péter Ligeti, Krisztina Lója, Dániel Nagy, Passive Attack Against the M2AP Mutual Authentication Protocol for RFID Tags, in: First International EURASIP Workshop on RFID Technology, Vienna, Austria, September 2007.

[17] Jung-Sik Cho, Young-Sik Jeong, Sang Oh Park, Consideration on the Brute-force Attack Cost and Retrieval Cost: a Hash-based radio-frequency identification (RFID) Tag Mutual Authentication Protocol, Computers & Mathematics with Applications (2012).

[18] Ari Juels, RFID security and privacy: a research survey, Selected Areas in Communications 24 (2) (2006) 381–394.

[19] Tassos Dimitriou, A Lightweight RFID Protocol to Protect against Traceability and Cloning Attacks, in: Proceedings of SecureComm'05, pp. 59–66, 2005.

[20] Jeongkyu Yang, Jaemin Park, Hyunrok Lee, Kui Ren, Kwanjo Kim, Mutual Authentication Protocol for Low-cost RFID, in: Proceedings of the Workshop on RFID and Lightweight Cryptography, pp. 17–24, 2005.

[21] Masoumeh Safkhani, Pedro Peris-Lopez, Julio César Hernández Castro, Nasour Bagheri, Majid Naderi, Cryptanalysis of Cho et al.'s protocol, A hash-based mutual authentication protocol for RFID systems, IACR Cryptology ePrint Archive 2011 (2011) 1–7.

[22] Hyunsung Kim, Desynchronization attack on hash-based RFID mutual authentication protocol, Journal of Security Engineering 9 (4) (2012) 357–366.

[23] Jung-Sik Cho, Sang-Soo Yeo, Sung Kwon Kim, Securing against Brute-force attack: a hash-based RFID mutual authentication protocol using a secret value, Computer Communications 34 (3) (2011) 391–397.

[24] Hyunsung Kim, Enhanced hash-based RFID mutual authentication protocol, in: Computer Applications for Security, Control and System Engineering, in: Communications in Computer and Information Science, vol. 339, 2012, pp. 70–77.