RFID in eHealth: How to Combat Medication Errors and Strengthen Patient Safety

Pedro Peris-Lopez[1,*] Masoumeh Safkhani[2] Nasour Bagheri[3] Majid Naderi[2]

[1]Computer Security Lab, Carlos III University of Madrid, 28911 Leganés, Madrid, Spain
[2]Electrical Engineering Department, Iran University of Science and Technology, Tehran 1684613114, Iran
[3]Electrical Engineering Department, Shahid Rajaee Teacher Training University, Tehran 1678815811, Iran

Received 3 Sep 2012; Accepted 14 Dec 2012; doi: 10.5405/jmbe.1276

Abstract

A medication error is an adverse event or even a miss in the treatment process that may harm a patient. As a consequence of this, patients' diseases may recrudesce and the mortality rate could rise. Therefore, medication errors have consequences in human terms and result in higher medical costs. Advanced inpatient medication safety systems can help reduce such errors in hospitals. Radio-frequency identification (RFID)-based systems are a promising solution for such applications. In this context, RFID grouping-proofs have been proposed to generate evidence that a collection of tags were read at the same time. These proofs are useful for automating the five rights method (right patient, right drug, right dose, right route, and right time). Wu *et al*. recently proposed an RFID grouping-proof. Unfortunately, the security level offered by this protocol is too low, as demonstrated here. This study shows how a passive attacker can conduct a full-disclosure attack. The cost of the proposed attack is only $O(2^{32})$, which is inappropriate for medical applications, and thus the attack requires only a few minutes on a personal computer. In addition, a de-synchronization attack can be conducted with only two runs of the protocol. Finally, as the idea of using grouping-proofs seems useful to enhance patient safety, a protocol called EKATE is proposed and a security analysis is performed.

Keywords: eHealth, Radio-frequency identification (RFID), Medication errors, Patient safety

1. Introduction

The use of radio-frequency identification (RFID) was increased due to its advantages over other identification systems. Some hospitals and medical institutes have started to establish RFID testing projects for applications such as drug administration, access to medical records, equipment/patient identification, and patient/staff tracking to improve patient safety, reduce medical errors, and optimize business processes [1]. The increasing number of academic articles related to RFID applications and issues in eHealth clearly shows the high applicability and wide interest in using this technology in the healthcare sector [2,3].

The typical RFID infrastructure that hospitals may hold [4] is shown in Fig. 1. RFID systems consist of tags, readers, and a back-end database which is often integrated with the hospital information system (HIS) [5]. RFID tages can be applied to staff members, patients, medical equipment, and drug packages. Staff members are equipped with hand-held readers (e.g., Mobile Clinical Assistant Reader (MCAR)) that allow access to the hospital applications through the wireless network. Moreover, fixed readers are deployed to cover the infrastructure of the hospital, including gates, rooms, operating theatres, corridors, and medical equipment. Figure 2 shows the patient life cycle.

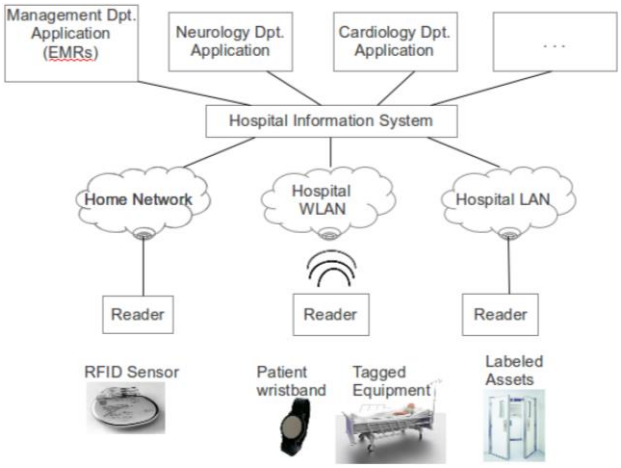

Figure 1. RFID in eHealth systems.

After arrival (admission), a patient is labeled with an RFID wristband that contains an identification number (ID) and some extra information. Next, the patient is examined (examination

* Corresponding author: Pedro Peris-Lopez
Tel: +34-91-6248877; Fax: +34-91-6249129
E-mail: pperis@inf.uc3m.es

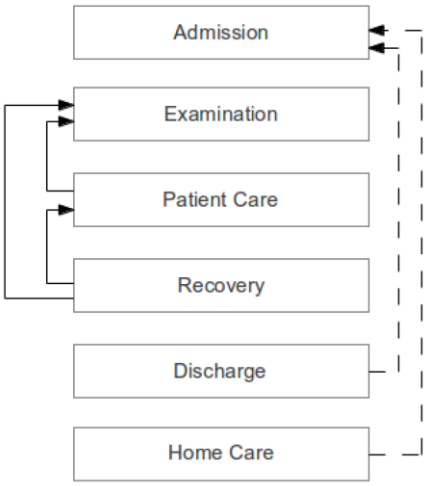

Figure 2. Patient life cycle.

phase) and some actions are taken to treat her illness. During this phase, actors (i.e., patients, nurses, and doctors) are identified and their actions are registered in the electronic medical record (EMR) of the patient. During the patient care the staff identification and the log of their actions are similarly performed. The vital signs (e.g., blood pressure and temperature) of the patient are added to the EMR and her medicines and specimen collections are identified. The assets (e.g., beds, serum holders) can be identified and health workers can easily access the history. During the recovery phase, RFID technology provides a positioning system for tracking patients, which is essential for Alzheimer and psychiatric patients. Finally, when the doctor discharges the patient, the cleaning staff receives a message. An intelligent process is triggered and information about the room items (e.g., last cleaning of the mattress) can be read through the tags. Moreover, automatic and accurately inventory of the room can be performed, with the information sent to the stock management system. The patient can benefit from the use of RFID technology at home if her home network is connected to the HIS. If so, nurses, doctors, and home caregivers can be identified at the patient's house. Moreover, drugs can be identified and an alarm can be raised in case of error (e.g., incorrect drug, dose, route, or time).

1.1 Medication errors and patient safety

Patient safety is a crucial issue in all hospitals. As reported in [6], the rate of potentially life-threatening medication errors ranges from 3% to 21% while that of clinically significant errors ranges from 3.3% to 31%. The problem is that hospitals cannot quickly and precisely identify actors (staff and patients) and check the proper administration of medicines [7]. Medication errors can be produced during different phases: prescription, manufacturing, or dispensing of the formulation, administering the treatment, and monitoring therapy. Information technology (IT) could help prevent medication errors. For example, the use of IT tools can easily prevent the misinterpretation of a hand-written prescription. Drug and patient identification systems can automate certain processes to guarantee that the appropriate prescription is given to each patient.

International studies have shown that medications errors occur predominately with medication orders (prescription and dispensing) or the administration of medication [8,9]. Table 1 summarizes the kinds of error and their ratios (average values obtained from [23-25,27] and [27] respectively). Medication errors can be reduced using the five rights method: right patient, right drug, right dose, right route, and right time [10]. The five rights should be accepted by nurses as the goal of the medication process. Nevertheless, nurses work under a lot of pressure and the use of intelligent systems could reduce their workload and decrease their slips and lapses [11,3].

Table 1. Types and ratios of medication errors.

Type	Ratio
Wrong Administration rates	13.4%
Omission of dose	20.94%
Drug compatibility	8%
Wrong dose	18.02%
Wrong drug	5.7%
Wrong patient	1.95%
Wrong time	12.22%
Allergic related error	6.73%
Additional/ unauthorised dose	9.25%

Hospitals have started to use various solutions to guard medication activities [12]. Computerized physician order entry (CPOE) prevents hand-writing errors and can be combined with clinical decision support tools to avoid prescription errors (e.g., drug-and-drug iteration). An automatic dispensing system (EDS) efficiently manages, stores, and dispenses medications. Barcode systems improve the identification processes of drugs and biological products. Finally, RFID-based solutions empower the aforementioned identification and traceability processes due to its advantages over barcode solutions. For instance, RFID tags do not need line of sight to be read and can be read from a distance of several meters at a rate of hundreds of reads per second.

1.2 Research problem

In the healthcare sector, grouping-proof protocols can be used for patient, staff, drug, or asset management [2]. Several RFID grouping-proof protocols have been recently proposed in the eHealth context [13,14,15]. Nevertheless, later analysis has shown that these solutions do not provide the claimed security level and the schemes have serious security faults [16,17,18]. Wu *et al*. [19] proposed a lightweight grouping-proof protocol that overcomes the security weaknesses of its predecessors. The present study verifies the security and efficiency of this protocol. The following research questions are addressed:

1. Is it possible for an adversary to clone a legitimate tag or disclose private information stored on a tag's memory? If yes, what is the complexity of this attack? Sections 3.1 and 3.2 show how an adversary can disclose the whole content of a tag's memory, compromising privacy.
2. Is the protocol secure against known attacks in the RFID context, and particularly resistant against de-synchronization attacks? Section 3.3 shows how the server and the tag can be

misled to update their common secret values to different values.

3. Is the protocol efficient in terms of the consumed resources and the offered security level? Section 3.4 presents a performance analysis.
4. If the protocol is insecure, is it possible to design a new (or improved) protocol that offers an adequate security level? Section 4 proposes the EKATE protocol and presents its analysis.

1.3 Our contribution

This article scrutinizes the security level guaranteed by a proposal for medication systems based on RFID technology proposed by Wu *et al*. [19] and shows how this protocol offers low security. More precisely, the security level of this scheme is upper bounded by $O(2^{\frac{L}{2}})$ (far from the $O(2^{L})$ optimal value), where L is the length of the secret parameter. The authors suggest fix L to 64, which provides inadequate security protection for medical environments. That is, from the point of view of an adversary, the attack has a complexity of about 2^{32} invocations of pseudo-random number generator (PRNG) functions that can be executed on a PC in a fraction of a minute. Furthermore an adversary can make the tag and the sever reach different internal states (i.e., the entities will not share a key). This will make the tag and server unsynchronized, making the protocol useless. The efficiency of the protocol was not examined in depth. It is shown that the scheme can be slightly changed to increase efficiency by up to 25% while retaining the security level. However, neither the original protocol nor the modified version is recommended for critical applications due to their low security. Finally, a protocol called EKATE that offers an appropriate security level is proposed for preventing medical errors.

The rest of this paper is organized as follows. Section 2 reviews the related literature and describes Wu *et al.*'s grouping-proof protocol. A security analysis and performance evaluation are presented in Section 3. Section 4 introduces an enhanced and generalized grouping-proof protocol, called EKATE, and presents its security and performance analyses. Finally, conclusions are given in Section 5.

2. Literature review

RFID was originally designed for identifying labeled items. Researchers have focused on designing secure authentication protocols [20]. Recently, RFID technology with grouping-proof has been proposed to improve patient safety. RFID-based solutions are baked up by the wide usage of RFID technology and its recommendation by regulation authorities (e.g., in the USA, pharmaceutical manufactures should attach RFID tags to products as recommended by the Food and Drug Administration (FDA)).

The concept of grouping-proof (also known as yoking-proof, existence-proof, clumping-proof, binding-proof, and coexistence-proof) for an RFID system was first introduced by Juels [21]. Grouping-proof schemes allow multiple RFID tags to be scanned at once such that their co-existence is guaranteed. In RFID authentication protocols, the reader is permanently connected to the verifier (back-end database), which can be on-line (similar to authentication protocols) or off-line. Readers interested in the design and analysis of grouping-proofs can refer to [16].

In the healthcare sector, grouping-proof protocols can be used for patient, staff, drug, or asset management [2]. For instance, in a typical application of a grouping-proof protocol, the reader (e.g., MCAR held by a nurse) scans the identification of the nurse, the wristband of a patient, and the unit dose of drugs needed for her treatment (see Fig. 3). Therefore, the patient is identified and her medication is automatically checked.

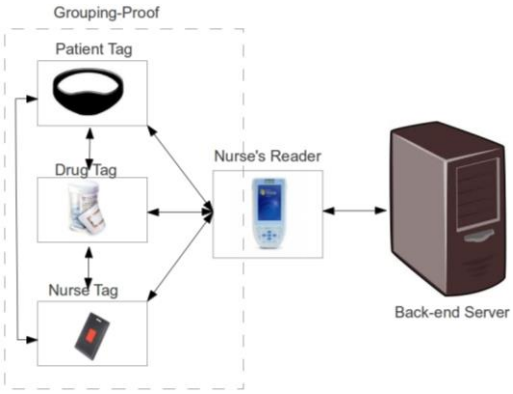

Figure 3. Diagram of grouping-proof in eHealth.

Several RFID grouping-proof protocols have been proposed in the eHealth context. In [14], Huang and Ku proposed a grouping-proof protocol for medication safety of inpatients. The authors distinguish the pallet tag from other tags (items). In the medical sector, the pallet corresponds with the patient's bracelet and the tagged items refer to the other labels that interact with the patient (e.g., drugs, medical equipment, etc.). This protocol uses a PRNG function to encrypt messages exchanged between the entities, i.e., pallet, items, and readers. Unfortunately, later analyses demonstrated important security pitfalls: possible denial of service attacks [13], traceability and impersonation attacks [17], and forgery attacks [16].

To overcome the above weaknesses, Chien *et al.* [13] proposed on-line and off-line protocols. However, their protocols suffer from traceability, impersonation, forgery, and replay attacks [17,3]. In [3], Peris-Lopez *et al.* proposed an interesting solution to enhance inpatient medication safety, called the Inpatient Safety RFID System (IS-RFID). Nevertheless, Yen *et al.* [18] showed that IS-RFID does not detect denial of proof attacks and that the generated medication evidence cannot defend against counterfeit evidence generated from the hospital. Finally, they designed two lightweight RFID-based solutions for secure inpatient medication administration, one with an on-line verifier and the other with an off-line verifier. RFID tags are very constrained devices with severe computational and storage limitations. To optimize the footprint supported on-board of RFID tags, Yu *et al.* [15] proposed a lightweight grouping-proof protocol, which is exclusively based on bitwise operations (e.g., XOR and AND).

Although the protocol seems quite attractive, the scheme is vulnerable against impersonation attacks, as showed in [19].

In [19], Wu *et al*. proposed a lightweight grouping-proof protocol to overcome the security weaknesses of another protocol [15]. Table 2 summarizes the notation and Fig. 4 shows the protocol. The protocol uses a 16-bit PRNG function and bitwise exclusive OR operations (i.e., XOR), which are supported on-chip in electronic product code (EPC) Class-1 Genetation-2 tags [22]. Since the channel between the reader and the back-end database is secure, the two entities can be condensed into one and the two names can be used indistinctly. The core function of Wu *et al*.'s protocol is a permutation function, denoted by F, which is a random permutation function mapping within the range $[1, 2^L]$, where L is the security parameter. The implementation of the F function is shown in Fig. 5, where $M = \{m_0, m_1, m_2, m_3\}$, $C = \{c_0, c_1, c_2, c_3\}$, and $C = F(M)$.

Table 2. Notation in Wu *et al*.'s protocol.

R_i	RFID reader *i*
T_i	RFID tag *i*
IDS_i	Index pseudonym of tag *i*
ID_i	Unique identifier of tag *i*
X_i, Y_i	Two private keys of tag *i*
r_x	Random number generated by party *x*, where *x* is a tag (Tag_x) or the server
T	Timestamp
PRNG	16-bit PRNG function
F	Random permutation function (16 invocations of PRNG function)
$\oplus$	Exclusive OR operation
$\|$	Concatenation operation

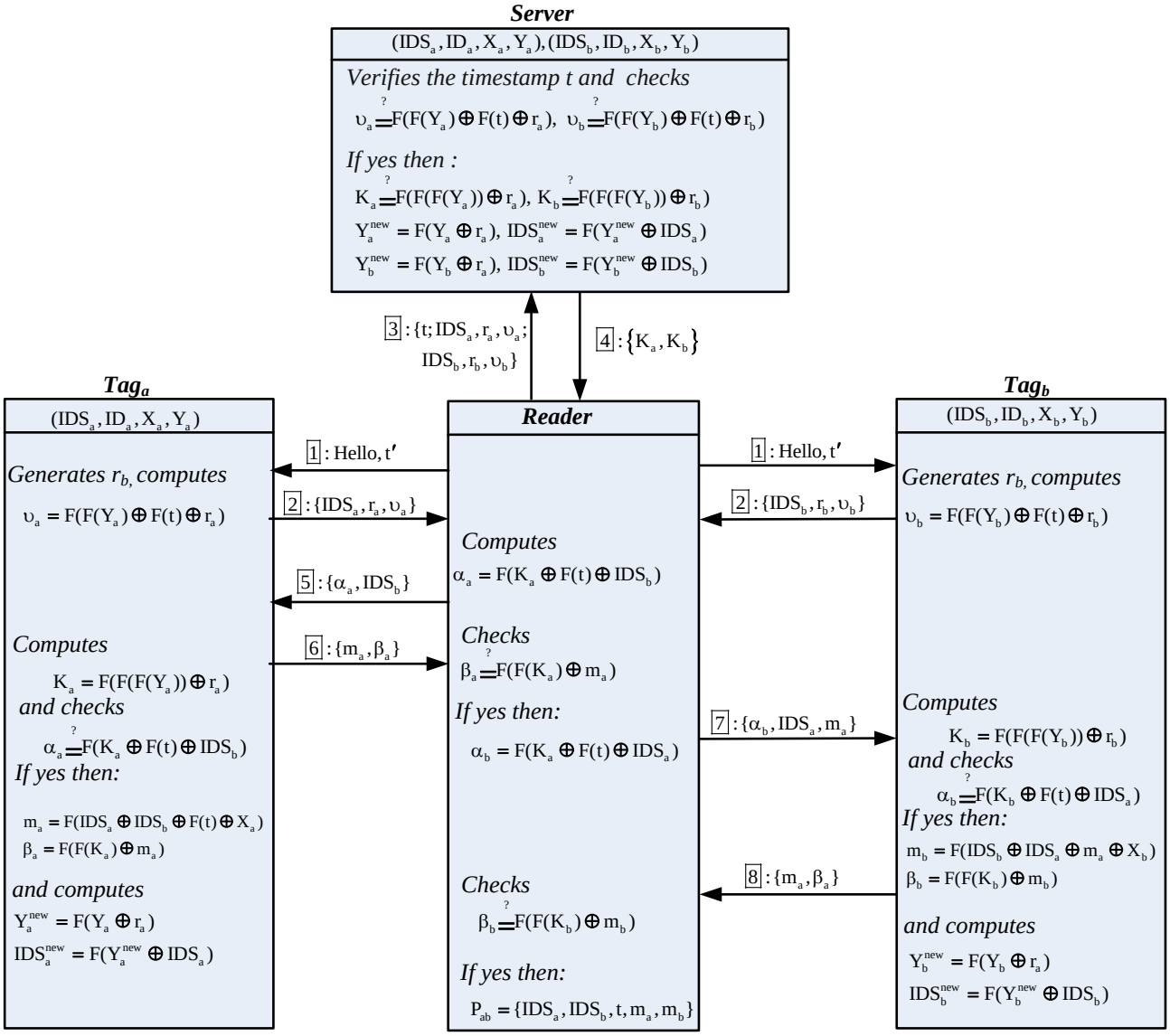

Figure 4. Wu *et al*.'s grouping-proof protocol [19].

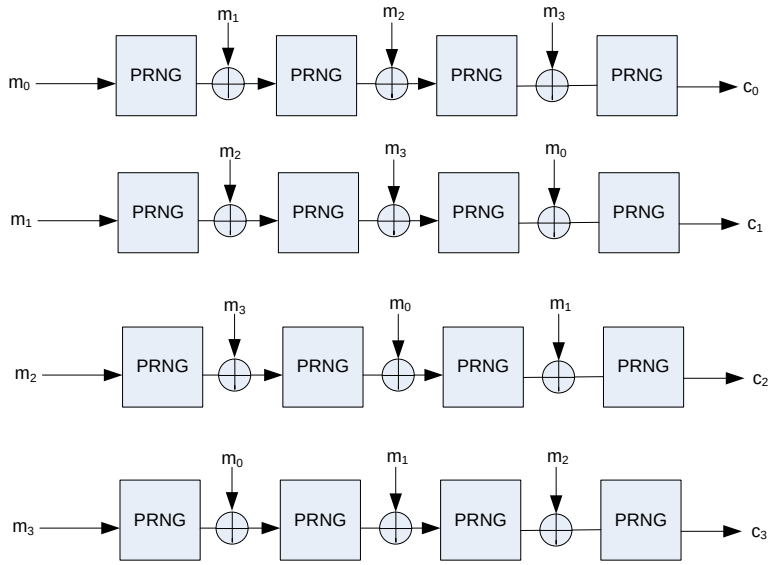

Figure 5. Random permutation function mapping $C = F(M)$ proposed in [18].

The messages exchanged between the reader (R) and the two tags (T_a and T_b) involved in the proof are described below:

1. R broadcasts "Hello" with timestamp t to the tags in its working range.
2. Once T_a and T_b receive this message, they reply with $\{IDS_a, r_a, \vartheta_a = F(F(Y_a) \oplus F(t) \oplus r_a)\}$ and $\{IDS_b, r_b, \vartheta_b = F(F(Y_b) \oplus F(t) \oplus r_b)\}$, respectively.
3. The reader forwards them along with timestamp t to the server.
4. The server verifies that the received t is under a reasonable threshold. If the verification succeeds, it:
   - checks $\vartheta_a \stackrel{?}{=} F(F(Y_a) \oplus F(t) \oplus r_a)$
   - checks $\vartheta_b \stackrel{?}{=} F(F(Y_b) \oplus F(t) \oplus r_b)$

If either of the two authentication tokens is incorrect, the server will inform the reader that the protocol has failed. Otherwise it proceeds as follows:

- computes two keys $K_a = F(F(F(Y_a)) \oplus r_a)$ and $K_b = F(F(F(Y_b)) \oplus r_b)$
- sends K_a and K_b to the reader
- updates Y_a, IDS_a, Y_b, and IDS_b to $Y'_a = F(Y_a \oplus r_a)$, $F(Y'_a \oplus IDS_a)$, $Y'_b = F(Y_b \oplus r_b)$, and $F(Y'_b \oplus IDS_b)$, respectively.

5. Upon receiving K_a and K_b, the reader computes $\alpha_a = F(K_a \oplus F(t) \oplus IDS_b)$ and sends $\{\alpha_a, IDS_b, t\}$ to T_a.
6. On receiving $\{\alpha_a, IDS_b, t\}$, T_a computes $K_a = F(F(F(Y_a)) \oplus r_a)$ and checks $\alpha_a \stackrel{?}{=} F(K_a \oplus F(t) \oplus IDS_b)$. If so, T_a does the following:
   - computes $m_a = F(IDS_a \oplus IDS_b \oplus F(t) \oplus X_a)$ and $\beta_a = F(F(K_a) \oplus m_a)$
   - sends m_a and β_a to the reader
   - and updates Y_a and IDS_a to $Y'_a = F(Y_a \oplus r_a)$ and $IDS'_a = F(Y'_a \oplus IDS_a)$, respectively.
7. Upon receiving $\{m_a, \beta_a\}$, the reader first verifies whether $\beta_a \stackrel{?}{=} F(F(K_a) \oplus m_a)$. If it holds, the reader computes $\alpha_b = F(K_b \oplus F(t) \oplus IDS_a)$ and sends $\{\alpha_b, IDS_a, m_a\}$ to T_b.
8. On receiving $\{\alpha_b, IDS_a, m_a\}$, T_b uses its current secret key Y_b to compute $K_b = F(F(F(Y_b)) \oplus r_b)$ and check whether $\alpha_b \stackrel{?}{=} F(K_b \oplus F(t) \oplus IDS_a)$. If so, T_b computes $m_b = F(IDS_b \oplus IDS_a \oplus m_a \oplus X_b)$ and $\beta_b = F(F(K_b) \oplus m_a)$ and sends them to the reader. Then, T_b computes $Y'_b = F(Y_b \oplus r_b))$ and $IDS'_b = F(Y'_b \oplus IDS_b)$ and updates Y_b and IDS_b to Y'_b and IDS'_b, respectively.
9. Upon receiving $\{m_b, \beta_b\}$, the reader first checks whether $\beta_b \stackrel{?}{=} F(F(K_b) \oplus m_a)$. If it holds, the reader confirms that T_a and T_b exist in the field simultaneously. At this moment, the reader holds IDS_a, IDS_b, t, m_a, and m_b and can collect them to form the proof $P_{ab} = \{IDS_a, IDS_b, t, m_a, m_b\}$, whose validity can be checked by the server.

3. Security analysis of wu *et al.*'s protocol

The main novelty of Wu *et al.*'s grouping-proof protocol is that the $F(M)$ function is composed using several invocations of a PRNG and some XOR computations. The security of the proposed scheme lies in the strength of $F(M)$.

This section analyzes the complexity of computing $F^{-1}(M)$. Then, it proves that the security of Wu *et al.*'s scheme is reduced to the security of the F function.

3.1 Inversion of F function

To determine the complexity of computing $F^{-1}(M)$, the complexity of determining M when $F(M)$ is given is deduced. Let $L = 64$, $M = (m_0, m_1, m_2, m_3)$, and $C = F(M)$, where $C = (c_0, c_1, c_2, c_3)$ and $\{c_i\}_{i=0}^{3}$ are computed as follows:

$$\begin{aligned} c_0 &= PRNG(PRNG(PRNG(PRNG(m_0) \oplus m_1) \oplus m_2) \oplus m_3) \\ c_1 &= PRNG(PRNG(PRNG(PRNG(m_1) \oplus m_2) \oplus m_3) \oplus m_0) \\ c_2 &= PRNG(PRNG(PRNG(PRNG(m_2) \oplus m_3) \oplus m_0) \oplus m_1) \\ c_3 &= PRNG(PRNG(PRNG(PRNG(m_3) \oplus m_0) \oplus m_1) \oplus m_2) \end{aligned} \quad (1)$$

Wu *et al.* used the F function instead of a simple PRNG function to increase the security level of their protocol

compared to that of EPC-friendly proposals. Then, the security level of the F function was determined. Given the output of the F function, i.e. $C = F(M)$, we present an approach to find its input, i.e., M. Our proposal to disclose the M value is also useful for determining the security level of F. The procedure is described below:

1. Generate a table called T_{PRNG}. For $i = 0$ to $2^{16}-1$, store PRNG(i) in the i-th row of the table.
2. Generate a table called T^{-1}_{PRNG}. For $i = 0$ to $2^{16}-1$, store i in the PRNG(i)-th row of the table (if for $i \neq j$: PRNG(i) = PRNG(j), then store both i and j in the same row).
3. Given c_0, c_1, c_2, c_3, and T^{-1}_{PRNG}, invert the last layer of the F function to obtain the following values, where $PRNG^{-1}$ denotes the inverse of the PRNG function:

$$\begin{aligned} c'_0 &= PRNG^{-1}(c_0) = PRNG(PRNG(PRNG(m_0)\oplus m_1)\oplus m_2)\oplus m_3 \\ c'_1 &= PRNG^{-1}(c_1) = PRNG(PRNG(PRNG(m_1)\oplus m_2)\oplus m_3)\oplus m_0 \\ c'_2 &= PRNG^{-1}(c_2) = PRNG(PRNG(PRNG(m_2)\oplus m_3)\oplus m_0)\oplus m_1 \\ c'_3 &= PRNG^{-1}(c_3) = PRNG(PRNG(PRNG(m_3)\oplus m_0)\oplus m_1)\oplus m_2 \end{aligned} \quad (2)$$

4. Given the above equations, perform a meet-in-the-middle attack to retrieve (m_0, m_1, m_2, m_3) as follows:
   (a) Generate a table called T_{2PRNG} of size 2^{32} rows and 4 columns, where columns are denoted by m_0, m_1, m_2, and m_3, respectively, and values on row x are denoted by m_0^x, m_1^x, m_2^x, and m_3^x, respectively.
   (b) For $i = 0$ to $2^{16}-1$ and for $j = 0$ to $2^{16}-1$
   - store i and j on m_0^x and m_1^x, where $x = PRNG(PRNG(i)\oplus j)$
   (c) For $i = 0$ to $2^{16}-1$ and for $j = 0$ to $2^{16}-1$
   - store i and j on m_2^x and m_3^x, where $x = PRNG^{-1}(PRNG(j\oplus c'_0)\oplus i$
   (d) For $x = 0$ to $2^{32}-1$ verify whether the following conditions are satisfied:

$$\begin{aligned} c'_1 &= PRNG^{-1}(c_1) = PRNG(PRNG(PRNG(m_1^x)\oplus m_2^x)\oplus m_3^x)\oplus m_0^x \\ c'_2 &= PRNG^{-1}(c_2) = PRNG(PRNG(PRNG(m_2^x)\oplus m_3^x)\oplus m_0^x)\oplus m_1^x \\ c'_3 &= PRNG^{-1}(c_3) = PRNG(PRNG(PRNG(m_3^x)\oplus m_0^x)\oplus m_1^x)\oplus m_2^x \end{aligned} \quad (3)$$

5. Any tuple $\{m_0^x, m_1^x, m_2^x, m_3^x\}$ that satisfies all of the above-mentioned conditions is the target answer, i.e., $M = \{m_0^x, m_1^x, m_2^x, m_3^x\}$.

Following the proposed attack, given $C = (c_0, c_1, c_2, c_3)$, $M = \{m_0, m_1, m_2, m_3\}$ can be found with a computational cost of $O(2^{32})$ invocations of the PRNG function and an $O(2^{32})$ storage complexity. Therefore, the security level of the F function is upper bounded by $O(2^{32})$ calls to the PRNG function. In general, that bound is $O(2^{\frac{L}{2}})$, where L is the length of the security parameter.

3.2 Full disclosure attack

Section 3.1 showed that given $F(M)$, it is possible to retrieve M at a cost of 2^{32} evaluations of the PRNG function and 2^{32} storage spaces. In this section, this procedure is used to retrieve any secret parameter stored in the memory of the target tag. F^{-1} represents the inversion of F and has a cost to the adversary of 2^{32} evaluations per call. To disclose the secret parameters of tag T_a that takes part in the proof with R_i, an adversary (A) can follow the steps described below:

Phase 1(Learning): A eavesdrops one successful execution of the protocol and stores the exchanged messages between R_i and T_a, which includes t, IDS_a, r_a, and $\vartheta_a = F(F(Y_a)\oplus F(t)\oplus r_a$.

Phase 2 (Secret Disclosure): To retrieve the secret parameters of tag T_a, given $\{t, IDS_a, r_a, \vartheta_a\}$, the adversary (A) can run the following steps in off-line mode:

1. $Z \leftarrow F^{-1}(\vartheta_a)$
2. $F(Y_a) \leftarrow Z \oplus F(t) \oplus r_a$
3. $Y_a \leftarrow F^{-1}(Y_a)$
4. $IDS_a^{new} \leftarrow F\ (Y_a \oplus IDS_a)$
5. return Y_a and IDS_a^{new}

Following the given attack, the adversary discloses the tag secret key Y_a and its next IDS value IDS_a^{new} with 100% success and a complexity of two calls to F^{-1} (i.e., 2^{33} executions of the PRNG function). This attack ruins all the security properties offered by the protocol. In fact, once the secret parameters of the tag are known, it is straightforward for an adversary to run many other 100% successful attacks: 1) traceability attacks; 2) tag impersonation attacks; 3) reader impersonation attacks; 4) de-synchronization attacks.

Remark 1 It should be noted that an adversary can disclose the secret parameters of the other participant in the proof (tag T_b) if the related values $(IDS_b, r_b, \vartheta_b)$ are also eavesdropped. Although the proposed attack is a passive attack, an active adversary could send a random timestamp t and "Hello" messages to the target tags, store their replies, and then jump to phase 2 of the given attack. In this case, the adversary does not need to wait for a session initiated by a legitimate reader.

3.3 De-synchronization attack

Although the above attack ruins all the security properties of Wu *et al.*'s [19] protocol, there are other weak points in the protocol's design. This section presents an efficient and practical de-synchronization attack. In this attack, the adversary misleads the tag and the server to update their common secret values to different values. As consequence of this in the healthcare environment is that the patient tag will not be authenticated by the server in future transactions and the patient will not receive the expected treatment.

Remark 2 It should be noted that a naive approach to de-synchronize tag T_i and the server is to block α_i message that is sent from the reader to the tag. At this stage, the server will update the tag records but no update will be performed on the tag side. To fix this problem, the server has to keep the old and new secret values. In the rest of this section, the most advanced scenario, in which the server keeps two records (old and new) for each tag, is assumed.

The proposed attack is based on the observation that the random number generated by the tag T_i, i.e., r_i, is directly XORed with $F(t)$. Therefore, given t, the adversary can calculate $F(t)$ and determine r_i such that $F(t)\oplus r_i$ meets a desired condition. The proposed attack has two phases. In

phase 1, the adversary eavesdrops one session of the protocol through the insecure wireless channel between the reader and the patient tag. In phase 2, the adversary uses the information captured in phase 1 to impersonate the tag and forces the server to update its secret parameters such that in future sessions it will not recognize the tag with a high probability. More precisely, in phase 1, the adversary A eavesdrops one run of the protocol between the reader R_i and tag T_i and stores the exchanged values. The values captured are as follows:

$$\{t, IDS_i, r_i, \vartheta_i = F(F(Y_i) \oplus F(t) \oplus r_i\} \quad (4)$$

At the end of this phase, both the server and the tag update Y_i^{new} and IDS_i^{new} to $F(Y_i \oplus r_i)$ and $F(Y_i^{new} \oplus IDS_i)$, respectively. In addition, the server keeps the old records, i.e., Y_i and IDS_i.

Next, in phase 2, A waits until a legitimate reader initiates a new session of the protocol and A impersonates T_i as described below:

1. R broadcasts a "Hello" message with its new timestamp (i.e., $t' \neq t$) to all the tags in its working range.
2. Once A receives this query, she uses the eavesdropped values, i.e., $\{t, IDS_i, r_i, \vartheta_i = F(F(Y_i) \oplus F(t) \oplus r_i\}$ to compute the tuple $\{IDS'_i = IDS_i, r'_i = r_i \oplus F(t) \oplus F(t'), \vartheta'_i = \vartheta_i\}$ and finally sends them to the reader.
3. The reader forwards $\{IDS'_i, r'_i, \vartheta'_i\}$ values with timestamp t' to the server.
4. The server verifies whether t' is under a reasonable threshold. If the verification succeeds, it:
   - checks $\vartheta'_i \stackrel{?}{=} F(F(Y_i) \oplus F(t') \oplus r'_i)$ where:

$$F(F(Y_i) \oplus F(t') \oplus r'_i) = F(F(Y_i) \oplus F(t') \oplus r_i \oplus F(t) \oplus F(t')) = \\ = F(F(Y_i) \oplus F(t) \oplus r_i) = \vartheta_i = \vartheta'_i \quad (5)$$

   - computes key $K_i = F(F(F(Y_i)) \oplus r'_i)$ and sends it to the reader
   - updates Y_i^{new} and IDS_i^{new} to $F(Y_i \oplus r'_i)$ and $F(Y_i^{new} \oplus IDS_i)$, respectively. The old values are not updated due to the fact that an old IDS ($IDS' = IDS$) was used. That is, the server was deceived by the attacker into thinking that the previous session failed.

Therefore, given that $t \neq t'$, at the end of the attack the server keeps two records (old and new values) for the tag, where neither of them match the stored values in the tag (see Table 3 for details). The complexity of the proposed attack is the execution of two protocol sessions. The adversary success probability is $1 - 2^{-2k}$, where k is the bit-length of the random numbers used in the protocol.

Table 3. De-synchronization attack: internal secret values

Server	Tag
$Y^{old} = Y$	
$IDS^{old} = IDS$	
$Y_s^{new} = F(Y \oplus r \oplus F(t) \oplus F(t'))$	$Y_T^{new} = F(Y \oplus r)$
$IDS_s^{new} = F(Y_s^{new} \oplus IDS)$	$IDS_T^{new} = F(Y_T^{new} \oplus IDS)$

Remark 3 It should be noted that a tag backscatters indiscriminately its IDS when it is queried by a reader. The IDS does not change if the tag has not taken part in a successful run of the protocol. Therefore, once the tag is in a de-synchronized state, it will never update its IDS. An adversary could exploit this property to track the tag holder.

3.4 Performance of wu et al.'s protocol

The security level of Wu et al.'s scheme is conditioned by the design of the random permutation function (F). As shown previously, the security of F is upper bounded by $O(2^{\frac{L}{2}})$, where L is the security parameter. An interesting observation regarding the attack presented in Section 3.1 is that the last layer of the F function (i.e., the last invocation of the PRNG function in the computation of $\{c_i\}_{i=0}^{3}$) has a complexity of $O(2^{\frac{L}{4}})$ calls to PRNG (i.e. $O(2^{16})$ when $L = 64$). Therefore, dropping the last invocation of PRNG in the computations of $\{c_i\}_{i=0}^{3}$ does not decrease the attack complexity but improves the performance of the F function by 25%. That is, in the original F function, 16 invocations of PRNG are needed to compute $C = F(M)$ whereas 12 are sufficient in the new function. The new F function, called F_{imp}, is depicted in Fig. 6. Given the output C of F_{imp} (i.e. $C = F_{imp}(M)$) and following the procedure described in Section 3.1, it is possible to calculate the input M with a cost of $O(2^{32})$ invocations of the PRNG function and consuming $O(2^{32})$ spaces of storage.

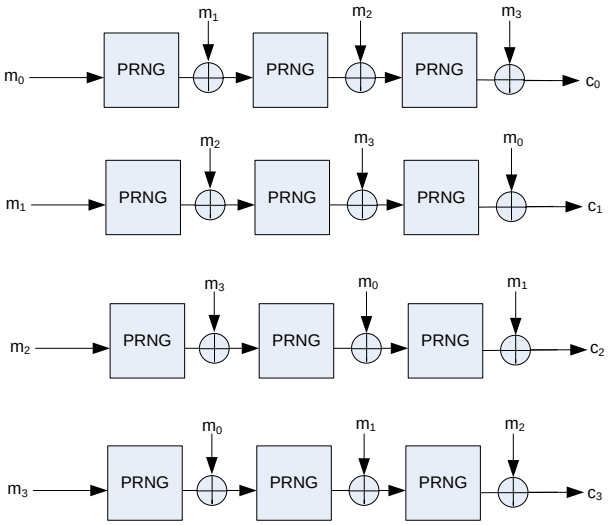

Figure 6. Proposed random permutation function mapping $C = F_{imp}(M)$.

All the calculations in Wu et al.'s scheme are calls to the F function or XOR computations. Therefore, any optimization in the F function decreases the protocol complexity roughly at the same rate. Since the security level of F_{imp} is equivalent to that offered by the F function, while the latter has more invocations of the PRNG, the usage of F_{imp} seems more appropriate. Nevertheless, neither of these functions is recommended in critical applications such as medication administration due to the low security level provided.

4. EKATE-improved grouping-proof

To fix the weaknesses of Wu et al.'s protocol and generalize it to any number of tags, an encryption function is

used instead of the random permutation F. This encryption function can be any secure lightweight block cipher such as PRESENT-128 [23] for which the block length is 64 bits and the key length is 128 bits. The design guidelines presented in [16] are followed and the classical assumption that the channel between the reader and the back-end database is secure is made. Nevertheless, considering the possible existence of dishonest readers, the server also contributes to the protocol randomness by generating a random number r_s, which is stored in the reader memory for the next session. In addition, to support the cases where several readers are interacting with the server, an identity is assigned to each reader, denoted by ID_r. The improved grouping-proof, called EKATE and depicted in Fig. 7, runs as described below:

1. R broadcasts "Hello" with $t' = ID_r \oplus r_r \oplus t$ to all the tags in its working range, T_j for $1 \le j < i$, where t is its timestamp.
2. Once T_j, for $1 \le j < i$, receives the message, it replies with $\{IDS_j, r_j, \vartheta_j = E_{(Y_j \oplus t') \| r_j}(t')\}$.
3. The reader forwards them with its t' and ID_r to the server.
4. The server extracts the timestamp t as $t' \oplus ID_r \oplus r_r$ and verifies that the received t is under a reasonable threshold.
   - If the verification succeeds, it verifies whether $\vartheta_j \stackrel{?}{=} E_{(Y_j \oplus t') \| r_j}(t')$ for $1 \le j < i$.

If either of the above conditions is not satisfied, the server will inform the reader that the run has failed. Otherwise, it does as follows:

- computes a key for each tag T_j, $K_j = E_{(Y_j \oplus t') \| r_j}(ID_j)$ where $1 \le j < i$
- sends $K_1, ..., K_i$ and a new nonce r_s to the reader and stores r_s in the record linked to the current reader
- updates Y_j and IDS_j, for $1 \le j < i$, to $Y_j^{new} = E_{(Y_j \oplus t') \| r_j}(Y_j)$ and $IDS_j^{new} = E_{(Y_j \oplus t') \| r_j}(IDS_j)$, respectively, and also keeps the old values.

5. Upon receiving $K_1, ..., K_i$ and a new nonce r_s, the reader updates r_r using the received r_s from the server and

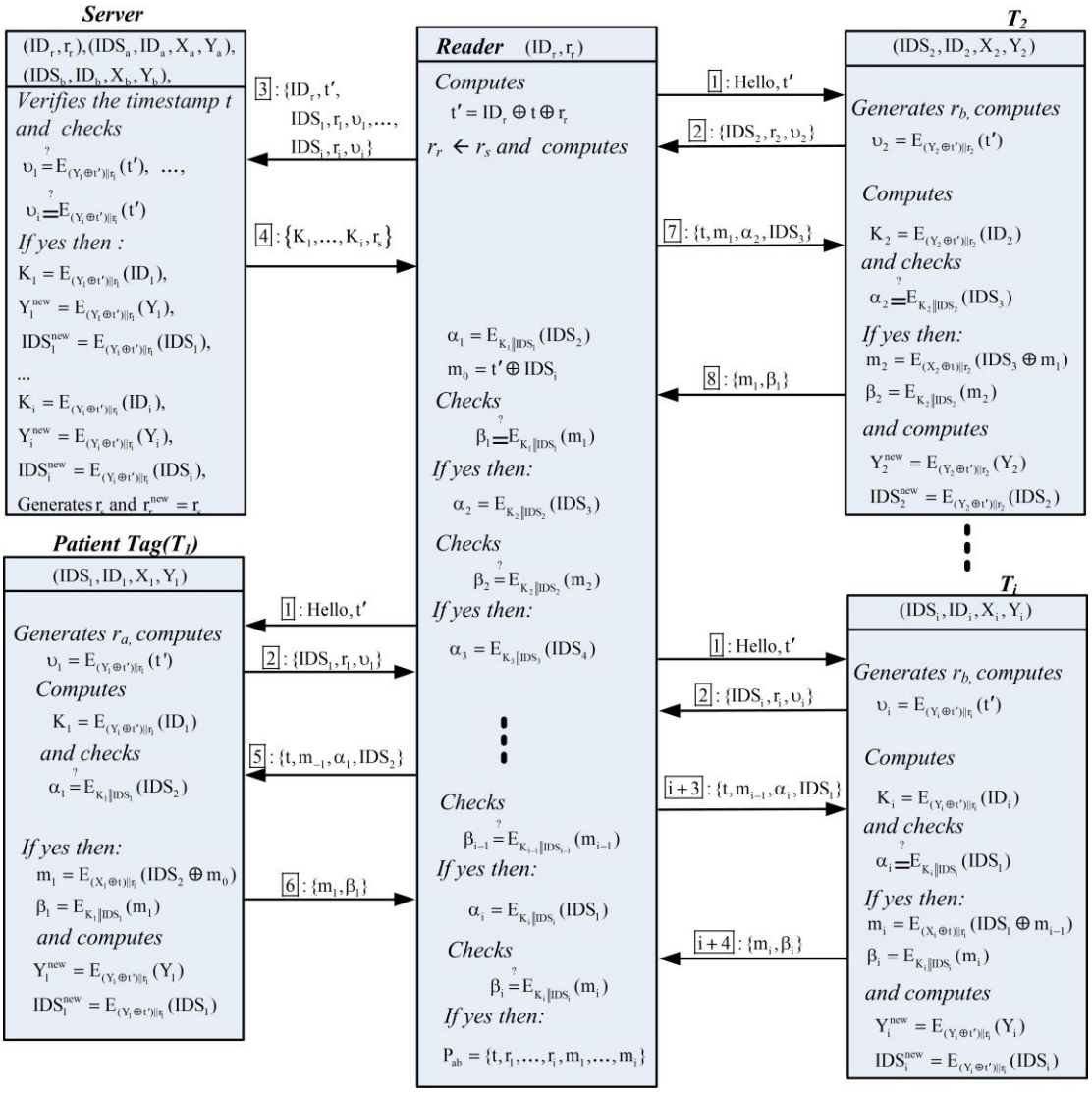

Figure 7. EKATE grouping-proof.

computes $\alpha_1 = E_{K_1||IDS_i}(IDS_2)$ and $m_0 = t' \oplus IDS_i$, and finally sends m_0, α_1, and IDS_2 to T_1.

6. Once it receives the message, T_1 computes $K_1 = E_{(Y_1 \oplus t')||r_1}(ID_1)$ and verifies whether $\alpha_1 \stackrel{?}{=} E_{K_1||IDS_1}(IDS_2)$. If it holds, T_1 does as follows:
   - computes $m_1 = E_{(X_1 \oplus t)||r_1}(IDS_2 \oplus m_0)$ and $\beta_1 = E_{K_1||IDS_1}(m_1)$ and sends them to the reader
   - updates Y_1 and IDS_1 to $Y_1^{new} = E_{(Y_1 \oplus t')||r_1}(Y_1)$ and $IDS_1^{new} = E_{(Y_1 \oplus t')||r_1}(IDS_1)$, respectively.
7. For $1 < j < i$:
   - Upon receiving the message $\{m_{j-1}, \beta_{j-1}\}$, the reader first verifies whether $\beta_{j-1} = E_{K_{j-1}||IDS_{j-1}}(m_{j-1})$. If so, the reader computes $\alpha_j = E_{K_j||IDS_j}(IDS_{j+1})$ and sends $\{m_{j-1}, \alpha_j, IDS_{j+1}\}$ to T_j.
   - On receiving the message $\{m_{j-1}, \alpha_j, IDS_{j+1}\}$, T_j computes $K_j \stackrel{?}{=} E_{(Y_j \oplus t')||r_j}(ID_j)$ and verifies whether $\alpha_j \stackrel{?}{=} E_{K_j||IDS_j}(IDS_{j+1})$. If it holds, T_j does as follows:
   - computes $m_j = E_{(X_j \oplus t)||r_j}(IDS_{j+1} \oplus m_{j-1})$ and $\beta_j = E_{K_j||IDS_j}(m_j)$ and sends them to the reader
   - updates Y_j and IDS_j to $Y_j^{new} = E_{(Y_j \oplus t')||r_j}(Y_j)$ and $IDS_j^{new} = E_{(Y_j \oplus t')||r_j}(IDS_j)$, respectively.
8. Upon receiving the message $\{m_{i-1}, \beta_{i-1}\}$, the reader first verifies whether $\beta_{i-1} \stackrel{?}{=} E_{K_{i-1}||IDS_{i-1}}(m_{i-1})$. If so, the reader computes $\alpha_i = E_{K_i||IDS_i}(IDS_1)$ and sends $\{m_{i-1}, \alpha_i, IDS_1\}$ to T_i.
9. On receiving the message $\{m_{i-1}, \alpha_i, IDS_1\}$, T_i computes $K_i = E_{(Y_i \oplus t')||r_i}(ID_i)$ and verifies whether $\alpha_i \stackrel{?}{=} E_{K_i||IDS_i}(IDS_1)$. If it holds, T_i does as follows:
   - computes $m_i = E_{(X_i \oplus t)||r_i}(IDS_1 \oplus m_{i-1})$ and $\beta_i = E_{K_i||IDS_i}(m_i)$ and sends them to the reader
   - updates Y_i and IDS_i to $Y_i^{new} = E_{(Y_i \oplus t')||r_i}(Y_i)$ and $IDS_i^{new} = E_{(Y_i \oplus t')||r_i}(IDS_i)$, respectively.
10. Upon receiving $\{m_i, \beta_i\}$, the reader first checks $\beta_i \stackrel{?}{=} E_{K_i||IDS_i}(m_i)$. If it holds, the reader confirms that $T_1, ..., T_i$ exist in the field simultaneously. At this moment, the reader has $IDS_1, \ldots, IDS_i$, t, r_r, $r_1, \ldots, r_i$, $m_1, \ldots$ and m_i and can collect them to form the proof $P_{1,...,i} = (t, r_r, r_1, ..., r_i, m_1, ..., m_i)$, whose correctness can be checked by the server when necessary.

In the EKATE proof, similar to Wu et al. in [19], the server has to perform O(i.N) computations to verify $P_{1,...,i}$, where N is the total number of tags covered by the current reader and i is the number of tags involved in the proof.

4.1 Security of EKATE protocol

In the improved protocol, any exchanged messages are, explicitly or implicitly, randomized either by t', which is contributed by the reader and the server, or r, which is contributed by the tag. Therefore, the adversary cannot use messages previously eavesdropped in a later session. More precisely, the improved protocol can resist certain possible attacks as follows:

Replay Attack: In the proposed protocol, after each authentication, the server updates the nonce r_r. Hence, it is not possible to use the previous eavesdropped message at a later time. The only exception is the case where the reader stores a message at time t_1, based on the random number r_r, and tries to use it at time t_2. Nevertheless, the server updates the nonce and $t_2 = t_1 \oplus r_r \oplus r'_r$ should be satisfied, where r'_r is the new record of the random number generated by the server. That is, a dishonest reader has no control over the time in order to successfully perform the above attack.

Impersonation Attack: It is not viable to impersonate the reader to tag T_j (reader impersonation) because an adversary, who is not connected to the server, cannot calculate a valid K_j and therefore a valid α_j. Similarly an adversary cannot impersonate T_j to the reader (tag impersonation) because she has no knowledge of Y_j and ID_j to generate K_j, which is randomized by t', contributed by the reader. Therefore, the protocol resists reader and tag impersonation attacks.

ID-theft Attack by a Dishonest Reader: The tag ID is always transmitted encrypted with a key that is only shared between the tag and the server, and the dishonest reader cannot decrypt it. Hence, any dishonest reader which is involved in a proof cannot disclose any extra information related to the tag.

De-synchronization Attack: The server updates the tag secret values after its successful authentication. If the protocol continues such that the tag also authenticates the server, then they share the same secret values and are in a synchronized state. However, if the protocol is aborted before the tag updates its secrets values, then the tag can be identified by means of the old values kept in the server. It should be noted when the server authenticates the tag, any eavesdropped information up to this point would be useless (with a high probability) because the server refreshes the reader random number. Hence, it is infeasible to deceive the server to use eavesdropped information linked to old sessions while the tag has updated its records (i.e., the attack presented in Section 3.2 does not work against the proposed protocol).

Creating Wrong Proof: To create a wrong proof, the reader/adversary should impersonate the tags, which is not viable. In addition, any proof generated at time t cannot be used for time $t'' \neq t$ because t is part of the encryption key for generating m_j.

4.2 Performance analysis of EKATE protocol

Table 4 shows a performance comparison between Wu et al.'s protocol and the EKATE scheme. The table shows that the proposed grouping-proof does not significantly increase the computational cost while offering a higher security level. In this table, *l* denotes the bit-length of parameters, with a value of 64 suggested in the original protocol [19]. The first three columns evaluate the number of calls to the cryptographic primitives (PRNG or encryption functions) and the usage of XOR operations. The computations needed for the updating of the internal values are included in these calculations. In addition, for the reader and the server, the average cost per tag is computed. The last two columns evaluate the memory requirements (stored bits (SB)) and the usage of the channel (number of transferred bits (TB)). Wu *et al.*'s proof makes extensive use of the PRNG whereas EKATE makes few calls to

the encryption function. The memory requirements and the number of messages exchanged are similar.

Table 4. Performance comparison.

	# PRNG-*16*	*#E*	*#⊕*	*# TB*	*# SB*
Server of Wu *et al*.	*128*	*0*	*30l*	*l*	*4l*
Server of EKATE	*1*	*4*	*3l*	*2l*	*5l*
Reader of Wu *et al*.	*64*	*0*	*15l*	*9l*	*0*
Reader of EKATE	*1*	*2*	*2l*	*9l*	*2l*
Tag of Wu *et al*.	*208*	*0*	*50l*	*5l*	*4l*
Tag of EKATE	*1*	*7*	*4l*	*5l*	*4l*

5. Conclusion

Security and performance analyses of Wu *et al*.'s scheme were conducted. The security level offered by Wu *et al*.'s proof is $2^{\frac{L}{2}}$, which is far from the optimum security margin of 2^L claimed by Wu *et al*., where L the bit-length of the security parameter. Regarding performance, the F function was not carefully designed. A slight change in its design leads to a 25% improvement in performance. The grouping-proof EKATE, which offers an appropriate security level and performance, was proposed. The proposed proof is a feasible tool for preventing medical errors.

References

[1] Y. Y. Chen, D. C. Huang, M. L. Tsai and J. K. Jan, "A design of tamper resistant prescription RFID access control system," *J. Med. Syst.*, 36: 2795-2801, 2012.

[2] S. Fosso Wamba, "RFID-enabled healthcare applications, issues and benefits: An archival analysis (1997-2011)," *J. Med. Syst.*, 36: 3393-3398, 2012.

[3] P. Peris-Lopez, A. Orifla, A. Mitrokotsa and J. A. C. van der Lubbe, "A comprehensive RFID solution to enhance inpatient medication safety," *Int. J. Med. Inform.*, 80: 13-24, 2011.

[4] M. Bouet and G. Pujolle, "RFID in ehealth systems: applications, challenges, and perspectives," *Ann. Telecommun.-Ann Telecommun.*, 65: 497-503, 2010.

[5] W. Yao, C. H. Chu and Z. Li, "The adoption and implementation of RFID technologies in healthcare: A literature review," *J. Med. Syst.*, 36: 3507-3525, 2012.

[6] H. S. Lorraine, N. Smith, J. Burke and A. Willman, "Report: A report on patient safety in Europe: medication errors and hospital-acquired infection," *J. Res. Nurs.*, 13: 451-454, 2008.

[7] P. Sun, B. Wang and F. Wu, "A new method to guard inpatient medication safety by the implementation of RFID," *J. Med. Syst.*, 32: 327-332, 2008.

[8] E. O. Ronda and G. Hughes, "Medication errors: what they are, how they happen, and how to avoid them," *QJM-An Int. J. Med.*, 102: 513-521, 2009.

[9] V. Jylha and K. Saranto, "Electronic documentation in medication reconciliation- a challenge for health care professionals," *Appl. Nurs. Res.*, 21: 237-239, 2008.

[10] M. Daniels and R. Daniels, "Nursing Fundamentals: Caring & Clinical Decision Making," New York: Thomson Delmar Learning, 2004.

[11] J. W. Cooper, "Adverse drug reaction-related hospitalizations of nursing facility patients: a 4-year study," *South. Med. J.*, 92: 485-490, 1999.

[12] Y. C. Yu, T. W. Hou and T.-C. Chiang, "Low cost RFID real lightweight binding proof protocol for medication errors and patient safety," *J. Med. Syst.*, 36: 823-828, 2012.

[13] H. Y. Chien, C. C. Yang, T. C. Wu and C. F. Lee, "Two RFID-based solutions to enhance inpatient medication safety," *J. Med. Syst.*, 35: 369-375, 2011.

[14] H. H. Huang and C. Y. Ku, "A RFID grouping proof protocol for medication safety of inpatient," *J. Med. Syst.*, 33: 467-474, 2009.

[15] Y. C. Yu, H. T. W. and T. C. Chiang, "Low cost RFID real lightweight binding proof protocol for medication errors and patient safety," *J. Med. Syst.*, 36: 823-828, 2012.

[16] P. Peris-Lopez, A. Orifla, J. C. Hernandez-Castro and J. A. C. van der Lubbe, "Flaws on RFID grouping-proofs. Guidelines for future sound protocols," *J. Netw. Comput. Appl.*, 34: 833-845, 2011.

[17] A. K. Wickboldt and S. Piramuthu, "Patient safety through RFID: vulnerabilities in recently proposed grouping protocols," *J. Med. Syst.*, 36: 431-435, 2012.

[18] Y. C. Yen, N. W. Lo and T. C. Wu, "Two rfid-based solutionsfor secure inpatient medication administration," *J. Med. Syst.*, 36: 2769-2778, 2012.

[19] S. Wu, K. Chen and Y. Zhu, "A secure lightweight RFID binding proof protocol for medication errors and patient safety," *J. Med. Syst.*, 36: 2743-2749, 2012.

[20] D. Henrici, "RFID Security and Privacy-Concepts, Protocols and Architectures," Springer; 1 edition, 2008.

[21] A. Juels, "Yoking-proofs for RFID tags," *Proc. IEEE Pervasive Comput. and Communi. Workshops*, 2: 138-143, 2004.

[22] E. Inc, "Class-1 Generation-2 UHF RFID protocol for communications at 860 MHzC960 MHz (version 1.1.0)," Available: http://www.epcglobalinc.org/standards/uhfc1g2/uhfc1g2-1-1-0-standard-20071017.pdf.

[23] A. Bogdanov, L. R. Knudsen, G. Le, C. Paar, A. Poschmann, M. J. B.Robshaw, Y. Seurin and C. Vikkelsoe, "Present: An ultra-lightweight blockcipher," *Proc. of Workshop on Cryptography Hardware and Embedded Syst.*, 4727: 450-466, 2007.

[24] E. B. Fortescue, R. Kaushal, C. P. Landrigan, K. J. McKenna, M. D. Clapp, F. Federico, D. A. Goldmann and D. W. Bates, "Prioritizing strategies for preventing medication errors and adverse drug events in pediatric inpatients," *Pediatrics*, 111: 722-729, 2003.

[25] C. Headford, S. McGowan and R. Clifford, "Analysis of medication incidents and development of a medication incident rate clinical indicator," *Collegian*, 8: 26-31, 2001.

[26] W. Hicks, S. C. Becker, D. Krenzischek and S. C. Beyea, "Medication errors in the pacu: a secondary analysis of medmarx endings," *J. Perianesth. Nurs.*, 19: 18-28, 2004.

[27] E. Tissot, C. Cornette, S. Limat, J. L. Mourand, M. Becker, J. P. Etievent, J. L. Dupond, M. Jacquet and M. C. W. Lemsi, "Observational study of potential risk factors of medication administration errors," *Pharm. World Sci.*, 25: 264-268, 2003.

[28] V. Wirtz, N. Barber and K. taxis, "An observational study of intravenous medication errors in the united kingdom and in Germany," *Pharm. World Sci.*, 25: 104-111, 2003.