Journal of Communications and Information Networks, Vol.2, No.3, Sept. 2017
DOI: 10.1007/s41650-017-0013-y

Research paper

Cryptanalysis and improvement of two new RFID protocols based on R-RAPSE

Seyed Salman Sajjadi GhaemMaghami[1*], Afrooz Haghbin[1], Mahtab Mirmohseni[2]

1. Department of Computer Engineering, Science and Research branch, Islamic Azad University, Tehran 1477893855, Iran
2. Department of Electrical Engineering, Sharif University of Technology, Tehran 11365/8639, Iran

* Corresponding author, Email: salman.ghaemmaghami@srbiau.ac.ir

Abstract: RFID (Radio Frequency IDentification) is a pioneer technology which has depicted a new lifestyle for humanity. Nowadays we observe an increase in the number of RFID applications and no one can ignore their numerous usage. An important issue with RFID systems is providing privacy requirements of these systems during authentication. Recently in 2014, Cai et al. proposed two improved RFID authentication protocols based on R-RAPS (RFID Authentication Protocol Security Enhanced Rules). We investigate the privacy of their protocols based on Ouafi and Phan privacy model and show that these protocols cannot provide private authentication for RFID users. Moreover, we show that these protocols are vulnerable to impersonation, DoS and traceability attacks. Moreover, we present two improved efficient and secure authentication protocols to ameliorate the performance of Cai et al.'s schemes. Our analysis illustrates that the existing weaknesses of the discussed protocols are eliminated in our proposed protocols.

Keywords: authentication, RFID protocol, privacy, security, Ouafi Phan privacy model, traceability, impersonation

Citation: S. S. S. GhaemMaghami, A. Haghbin, M. Mirmohseni. Cryptanalysis and improvement of two new RFID protocols based on R-RAPSE [J]. Journal of communications and information networks, 2017, 2(3): 107-122.

1 Introduction

Nowadays, our world is transitioning from an internet of connected individuals to an internet in which everything and everyone is connected, also known as IoT (Internet of Things)[1]. RFID is a technology which provides a contactless identification through magnetic waves. Health-care, livestock and animal tracking, access control, transportation and supply chain can be mentioned as its applications which play an important role in preparing the structures for developing the concept of IoT[2-4]. As it is shown in Fig. 1, RFID systems involve three main parts: back-end server, reader and tag. The tag is a microchip which can be attached to different objects with different purposes in an RFID system that falls in one of the three classes: active, passive and semi-active[5]. A passive tag does not have any battery and obtains sufficient energy to reply the reader from the magnetic field achieved through sending the request by the reader. An active tag contains an inner battery, allows it to start a new connection with the reader over than only be a responder. Although the semi-active tag holds an inner battery, it just responds to

Manuscript received Sept. 10, 2016; accepted Nov. 14, 2016

the received queries from the reader, and performing the inner operations are the only usage of the internal battery[6]. Decreasing the size and cost of RFID tags, have been led to popularity and vast implementation of passive tags in most of novel applications. The back-end server stores all the information of the tags and the readers, and establishes a connection with the tag via tranceiving data from the reader and after investigating the integrity of transferred messages, authenticates the reader and the tag.

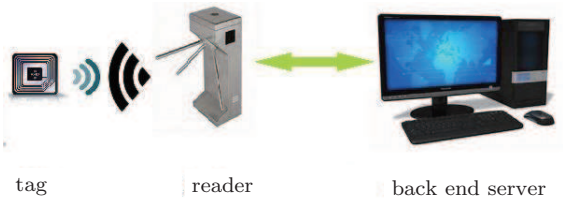

Figure 1 A system model of RFID systems[7]

Although, RFID technology is developing rapidly and providing comfort for users, deficiency of supplying the necessary privacy, will result in irreparable damages such as traceability, DoS and impersonation attacks[8]. Therefore, scholars have proposed protocols to provide security and privacy of users in RFID systems, which generally classify into four classes based on the deployed cryptographic functions[9]. Full-fledged protocols are the first class, which include ordinary cryptographic functions such as public or private key cryptography systems, one-way hash functions and so forth[10]. RNG (Random Number Generators) and one-way hash functions are permitted to use in the second class. The third class is called lightweight, includes RNG functions and CRC (Cyclic Redundancy Code) checksums[2]. Finally, ultra-lightweight is the last classification, limited to the usage of simple bitwise operators such as AND, OR and XOR[11]. Paying attention to the mentioned classification, several protocols have been presented in the last few years[4,12,13]. Yeh et al. proposed an RFID authentication protocol based on EPC Class 1 Generation 2 standard in 2010 which supplies tag privacy[4]. In 2011, Yoon declared that Yeh et al.'s protocol is still vulnerable to data integrity and forward secrecy problems[12]. So, he suggested an improved protocol for places with high level of security. In 2011, Cho et al. proposed a hash based RFID mutual authentication protocol[13]. They believe that their scheme solves the privacy and forgery problems while providing all the security requirements of RFID users[13]. In 2014, Cai et al. investigated Cho et al. and Yoon's protocols[14]. They believe that an authentication procedure must be designed and confirmed theoretically before testing experimentally. So, they defined rules called R-RAPSE, which provide security and privacy for RFID protocols[14]. They showed that Cho et al.'s protocol does not provide data integrity, and it is vulnerable to de-synchronization attack. Therefore, they proposed the IHRMA protocol based on R-RAPSE rules, which is improved version of Cho et al.'s protocol. They also proved that Yoon's protocol cannot preserve the location of tag's owner, which results in weaknesses for providing privacy issues. Thus, they proposed the UISRS protocol as a modification for Yoon protocol[5].

In this paper, we analyze the IHRMA and UISRS protocols and show that they still suffer from some weaknesses. First, we show that similarity between the generated messages and updating procedures enable one to make the tag and reader impersonation attack possible in the IHRMA protocol. Besides, revealing the secret key, in the UISRS protocol, yields to the tag impersonation, traceability and forward traceability attacks after maximum 2^{16} computations and backward traceability attack with 2^{17} runs in the worst case. We use Ouafi and Phan formal privacy model for privacy analysis of the IHRMA and UISRS protocols[14]. Recently, different types of privacy models have been proposed to study the authentication routine in RFID protocols . Among these formal privacy models, Ouafi-Phan privacy model[6,15-20]. Among these formal privacy models, Ouafi-Phan privacy model[17] is one of the well-known which has been proposed in 2008 and due to pertinent queries for different privacy analysis, it has got high attention by researchers[14-16,18,21].

The rest of the paper is organized as follows: in

section 2, we analyze the IHRMA protocol and discuss it. The same analysis for the UISRS protocol is performed in section 3. Our improved RFID authentication protocols are presented in section 4. The proposed protocols are compared with some existing ones in the terms of security and privacy in section 5. Finally, we conclude the paper in section 6.

2 IHRMA protocol

2.1 Analyze of IHRMA protocol

Cai et al. claim that R-RAPSE rules provide sufficient security in their protocol[14]. So, to prevent de-synchronization of secret information in two sides of protocol, they proposed their improved protocol based on R-RAPSE instructions which provide data integrity via hash functions in their method. The structure of the IHRMA protocol is depicted in Tab. 1. The connections between the reader and the back-end server are secure, while the communication links between the reader and the tag are insecure. In this section, we analyze the IHRMA protocol and prove that the protocol is vulnerable to tag and reader impersonation attacks. The notations used in this protocol are as follow:

RID_i : Group ID of random number.

ID_k : ID of the tag k.

S_j: Secret value mutually shared between the server and the tag, used in the jth session.

R_t: Random number generated by the tag.

R_r: Random number generated by the reader.

R_r^{i+j}: Random number generated by the reader in the $(i+j)$th session.

C_1: Data generated by the tag for authentication.

C_l^{i+j}: Data generated by the tag for authentication in the $(i+j)$th session.

C_2: Blind factor.

2.2 Tag impersonation attack

Now, we show that the IHRMA protocol cannot prevent an adversary $\mathcal{A}$ from performing an impersonation attack. Cai et al. have used mod operator in definition of RID_i for better performance of their improved protocol[14]. RID_i is described as follows:

$$RID_i = (R_t - R_t \bmod S_{j+1})(0:47) \\ \| \ (R_t + S_j - R_t \bmod S_j)(48:95). \quad (1)$$

Remark 1 Note the following property of the mod operator. If we have a and b as two integers where a, b, then a mod b will results in a.

Using Remark 1, in the IHRMA protocol[14], there will be a situation where $R_t < S_j$, so the above equation will change as below,

$$\begin{aligned} &RID_i \\ &= (R_t - R_t + 1)(0:47) \ \| \ (R_t + S_j - R_t)(48:95) \\ &= (1)(0:47) \ \| \ S_j(48:95). \end{aligned} \quad (2)$$

Learning phase: The attacker $\mathcal{A}$ sends an execute query (R, T_0, i), and obtains $(R_r, C_1, C_2 \oplus R_t, C_3, C_4, C_5)$. By changing the value of R_t, which makes new messages as C_4 and C_5, prevents the back-end server authentication in the tag side. Therefore, the secret value is not updated in the tag.

Attack phase: In session $(i+1)$, the attacker impersonates him/herself as a trusted tag. To this aim, the adversary performs as follows:

1) After receiving R'_r from the reader in the $(i+1)$ session, as it is shown in 2.1 and 2.2 steps of Tab. 1, the attacker $\mathcal{A}$ calculates $C_{1,\text{Fic}}$ and $C_{2,\text{Fic}}$ with the values stored in the last session, and sends them to the reader.

$$C_{1,\text{Fic}} = C_1, \quad (3)$$

$$D_{1,\text{Fic}} = C_2 \oplus R_t \oplus R_r \oplus R_r^{i+1}. \quad (4)$$

2) Reader sends R_r^{i+1}, $C_{1,\text{Fic}}$ and $D_{1,\text{Fic}}$ to the back-end server.

3) The back-end server calculates C_2^* and obtains R_t^{i+1} as

$$R_t^{i+1} = D_{1,\text{Fic}} \oplus C_2^*. \quad (5)$$

4) By computing R_t^{i+1}, the back-end server calculates RID_{i+1}. As we stated, if $R_t < S_j$, the obtained RID_{i+1} equals to Eq. (1).

5) The back-end server, computes $C_1^{*(i+1)}$ and authenticates the tag, if $C_1^{*(i+1)} = C_{1,\text{Fic}}$. Noting

Table 1 IHRMA protocol[14]

back-end server $(ID_k, S_j, S_{j-1}, DATA)$	reader	tag (ID_k, S_j)
	1.1 request: R_r	
	1.2 $\xrightarrow{R_r}$	
		2.1 $C_1 = H(ID_k \oplus R_t \oplus R_r \oplus RID_i)$
		2.2 $C_2 = S_j(0:47)\|\|ID_k(48:95)$
	3.1 $\xleftarrow{C_1, C_2 \oplus R_t, R_r}$	2.3 $\xleftarrow{C_1, C_2 \oplus R_t}$
4.1 for each tuple $(ID_k; S_j)$		
calculate $C_1^* = S_j(0:47)\|\|ID_k(48:95)$		
calculate $R_t^* = C_2 \oplus C_2^*$		
compute RID_i^*		
calculate $C_1^* = H(ID_k \oplus R_t^* \oplus R_r \oplus RID_i)$		
verify $C_1^* \stackrel{?}{=} C_1$		
if it does not match		
repeat with tuple $(ID_k; S_{j-1})$		
if it does not match, reveal the protocol		
4.2 calculate $C_3 = H(C_2 \oplus RID_i)$		
4.3 calculate $C_4 = R_t \oplus S_{j+1}$		
4.4 calculate $C_5 = R_t \oplus H(S_{j+1})$		
4.5 $\xrightarrow{DATA\|\|C_3\|\|C_4\|\|C_5}$	5.1 $\xrightarrow{C_3\|\|C_4\|\|C_5}$	6.1 calculate $C_3^* = H(C_2 \oplus RID_i)$
4.6 updating		if $C_3^* \neq C_3$
$S_j - 1 \leftarrow S_j$ $S_j \leftarrow S_j + 1$		reveal the protocol
		else
		6.2 calculate $S_{j+1} = R_t \oplus C_4$
		6.4 calculate $C_5^* = R_t \oplus H(S_{j+1})$
		if $C_5^* \neq C_5$
		reveal the protocol
		else
		server is authenticated
		6.2 updating $S_j \leftarrow S_j + 1$

$R_t < S_j$ and $R_t^{i+1} < S_j$, we have $RID_{i+1} = RID_i$; and the back-end server verifies the tag as a legal one, updates its secret values and sends C_3^{i+1}, C_4^{i+1} and C_5^{i+1} to the reader. Ultimately, the back-end server endorses the falsified tag as an allowable one.

Proof

$$\begin{aligned} & C_1^{*(i+1)} \\ &= H(ID_k \oplus R_t^{i+1} \oplus R_r^{i+1} \oplus RID_{i+1}) \\ &= H(ID_k \oplus R_t^i \oplus R_r^i \oplus R_r^{i+1} \oplus R_r^{i+1} \oplus RID_{i+1}) \\ &= H(ID_k \oplus R_t^i \oplus R_r^i \oplus RID_{i+1}) \\ &= H(ID_k \oplus R_t^i \oplus R_r^i \oplus RID_i) \\ &= C_{1,\mathrm{Fic}}. \end{aligned} \tag{6}$$

Equality of $C_1^{*(i+1)}$ with $C_{1,\mathrm{Fic}}$ will result in victory of an adversary. For random selection of R_t and R_r, the success probability of each assumption is 1/2. The total probability of the above attack is 1/4, while its complexity is two runs of the protocol.

2.3 Reader impersonation attack

As discussed in subsection 2.2, the structure of generating RID_i in the IHRMA protocol in Eq. (1), allows the attacker to perform the tag impersonation attack. Now, we express that this definition of RID_i, also results in the reader impersonation attack, with the success probability of 1/4. Its implementation is described as follows:

Learning phase: The attacker eavesdrops the ith session of the protocol by sending an execute query (R, T_0, i) and obtaining $(R_r, C_1, C_2 \oplus R_t, C_3, C_4, C_5)$. Now, to prevent updating secret values, the attacker blocks the 5.1 step of the protocol.

Table 2 UISRS protocol[14]

back-end server $(K_0, P_0, \alpha_0, K_n, P_n, \alpha_n, ID_r, EPC_s, DATA)$	reader (ID_r)	tag $(K_i, P_i, \alpha_i, EPC_s)$
	1.1 generate R_r	
	1.2 $\xrightarrow{R_r}$	2.1 generate R_t
		2.2 $M_1 = P(EPC_s \oplus R_r \oplus R_t) \oplus K_i$
		2.3 $\beta = R_t \oplus K_i$
		2.4 $\alpha_i = \alpha_i \oplus R_t$
	3.1 $V = H(ID_r \oplus R_r)$	2.5 $\gamma = R_t \oplus P(\alpha_i \oplus K_i)$
	3.2 $\xleftarrow{M_1,\beta,\gamma,\alpha_i,R_r,V}$	2.6 $\xleftarrow{M_1,\beta,\gamma,\alpha_i}$
4.1 for each ID_r in server		
verify $H(ID_r \oplus R_r) \stackrel{?}{=} V$		
for each tuple EPC_s, K_o, K_n		
4.2 calculate $I_O = M_1 \oplus K_o$ and $I_N = M_1 \oplus K_n$		
verify $I_O \stackrel{?}{=} P(EPC_s \oplus R_r \oplus \beta \oplus K_o)$		
verify $I_N \stackrel{?}{=} P(EPC_s \oplus R_r \oplus \beta \oplus K_n)$		
$x = O/N$		
4.3 verify $\beta \oplus K_x \oplus P(\alpha_i \oplus K_x) \stackrel{?}{=} \gamma$		
4.4 calculate $R_t^* = \beta \oplus K_x$		
verify $\alpha_x \stackrel{?}{=} \alpha_i \oplus R_t^*$		
4.5 calculate $M_2 = P(EPC_s \oplus R_t^*) \oplus P_x$		
4.6 calculate $Info = DATA \oplus ID_r$		
4.7 $MAC = H(DATA \oplus R_r)$	5.1 $DATA = Info \oplus ID_r$	
4.8 $\xrightarrow{M_2,Info,MAC}$	5.2 verify	
4.9 updating	$H(DATA \oplus R_r) \stackrel{?}{=} MAC$	
if $x = N$	5.3 $\xrightarrow{M_2}$	6.1 verify $M_2 \oplus P_i \stackrel{?}{=} P(EPC_s \oplus R_t)$
$K_o \leftarrow K_n \leftarrow P(K_n)$		updating
$P_o \leftarrow P_n \leftarrow P(P_n)$		$K_{i+1} \leftarrow P(K_i)$
$\alpha_o \leftarrow \alpha_n \leftarrow P(R_t \oplus R_r)$		$P_{i+1} \leftarrow P(P_i)$
else		$\alpha_{i+1} \leftarrow P(R_t \oplus R_r)$
$\alpha_n \leftarrow P(R_t \oplus R_r)$		
end if		

Attack phase: The attacker $\mathcal{A}$ acts as a reader and starts $(i+1)$ session by generating R_r^{i+1} randomly and sending to the tag T_0, that leads to the reader impersonation attack, as discussed below:

1) The tag did not update its secret values in the last session. Moreover, by considering Eq. (2), the tag generates a new R_t^{i+1} calculates C_1^{i+1} and sends it with D^{i+1} to the counterfeit reader which can be written as,

$$C_1^{i+1} = H(ID_k \oplus R_t^{i+1} \oplus R_r^{i+1} \oplus RID_{i+1}), \quad (7)$$

$$D^{i+1} = C_2^{i+1} \oplus R_t^{i+1}. \quad (8)$$

2) As the secret values of the tag did not update during the last session, $C_2^{i+1} = C_2^i$. Therefore, the attacker calculates C_4^{i+1} and C_5^{i+1} through the received messages C_1^{i+1}, D^{i+1} and the stored values of last session, as follows:

$$\begin{aligned} C_4^{i+1} &= C_2^{i+1} \oplus R_t^{i+1} \oplus C_2^i \oplus R_t^i \oplus R_t^i \oplus S_{j+1} \\ &= R_t^{i+1} \oplus S_{j+1}, \end{aligned} \quad (9)$$

$$\begin{aligned} C_5^{i+1} &= R_t^i \oplus H(S_{j+1}) \oplus C_2^{i+1} \oplus R_t^{i+1} \oplus C_2^i \oplus R_t^i \\ &= R_t^{i+1} \oplus H(S_{j+1}), \end{aligned} \quad (10)$$

and sends $DATA \parallel C_3^{i+1} \parallel C_4^{i+1} \parallel C_5^{i+1}$ to the tag.

3) If $R_t < S_j$ occurs for two subsequent runs of the protocol with the probability of $1/2$ in each time, the tag authenticates the attacker $\mathcal{A}$ as a legal one and the attacker performs reader impersonation attack with the probability of $1/4$.

3 UISRS protocol

Cai et al. proposed the UISRS protocol based on R-RAPSE rules and believed that their protocol would provide all privacy necessity for end-users[14]. Their protocol is depicted in Tab. 2 and the channels between all parts of this system are insecure. The elements of the UISRS protocol are listed below:

EPC_s: Each EPC block with 96 bits, divided to six blocks with 16 bits length. Xoring these six blocks generates it.

K_i: Confirmation key stored in the back-end server.

P_i: Access key stored in the tag, to authenticate the back-end server.

α_i: The back-end server index, stored in the tag to find the corresponding information of the tag in the server.

3.1 Analysis of UISRS protocol

Cai et al. proposed the UISRS protocol to provide privacy of end-user[14]. They believed randomness of α_i prepares the sufficient properties to prevent an adversary $\mathcal{A}$ from performing any attack on the protocol. But we find that there are still major weaknesses with this protocol.

3.2 Secret parameter reveal

An important issue to be considered is preventing an adversary to access secret values of end-users. In this section, we show that the UISRS protocol is vulnerable to the secret parameter reveal which is described below:

Learning phase: An attacker eavesdrops the ith session of the protocol and blocks the step 5.3. Therefore, it obtains $\{M_1^i, \beta^i, \gamma^i, \alpha^i, R_r^i, v, M_2^i\}$ while the tag does not update its secret values.

Attack phase: The attacker uses stored values of the ith session and noting their similarity, finds the secret value K_i, with the probability of 1. The attacker computes Z as below:

$$\begin{aligned} Z &= \beta^i \oplus \gamma^i \oplus \alpha_i \\ &= R_t \oplus K_i \oplus \alpha_i \oplus R_t \oplus P(\alpha_i \oplus K_i) \\ &= K_i \oplus \alpha_i \oplus P(\alpha_i \oplus K_i). \end{aligned} \tag{11}$$

K_i is a 16 bit string, so it is one of the element of $\{\Im_1, \Im_2, \cdots, \Im_{2^{16}}\}$. The attacker knows the value of α^i through the last session, so he/she is able to obtain K_i after maximum 2^{16} runs as below:

$$\begin{aligned} &\text{For } 2 < i < 2^{16} \\ &\text{choose } K_i \in \{\Im_1, \Im_2, \cdots, \Im_{2^16}\} \\ &\text{Let } W = K_i \oplus \alpha_i \oplus P(\alpha_i \oplus K_i) \\ &\text{If } Z = w \\ &\text{Return } K_i \text{ as a correct secret key} \\ &\text{End} \end{aligned} \tag{12}$$

Therefore, similarity in generating messages in UISRS protocol makes it vulnerable to secret parameters reveal. Moreover, the lack of precision in updating procedure allows the attacker $\mathcal{A}$ to calculate K , in the $(i+x)$th session by performing $PRNG$ for x times on the K_i.

3.3 Tag impersonation attack

Revealing the secret parameter in the UISRS protocol lets an adversary $\mathcal{A}$ impersonate a legitimate tag after maximum 2^{16} runs, which is proved in subsection 3.2. The method of applying this attack is as follows:

Learning phase: An adversary $\mathcal{A}$ sends an execute query (R, T_0, i) and obtains $\{M_1^i, \beta^i, \gamma^i, \alpha_i, R_r^i\}$. He/she is able to guess K_i with the probability of 1, as discussed in subsection 3.2, so obtains R_t,

$$R_t^i = \beta^i \oplus K_i. \tag{13}$$

Attack phase: In session $(i+1)$, after sending a request and R_r^{i+1} to the tag by the reader, the attacker $\mathcal{A}$ introduces him/herself as a legal tag and the following events take place:

1. The attacker $\mathcal{A}$ generates $\{M_1^{i+1}, \beta^{i+1}, \gamma^{i+1}, \alpha_{i+1}\}$ as below and sends them to the reader,

$$M_1^{i+1} = M_1^i,$$

$$\beta^{i+1} = R_t^i \oplus R_r^i \oplus R_r^{i+1} \oplus K_i,$$

$$\alpha_{i+1} = R_t^i \oplus R_r^i \oplus R_r^{i+1} \oplus \alpha_i',$$
$$\gamma^{i+1} = R_t^i \oplus R_r^i \oplus R_r^{i+1} \oplus P(\alpha_{i+1} \oplus K_i). \quad (14)$$

As the value of EPC_S is not known to the attacker, it is not possible to compute M_1^{i+1}. Therefore the attacker uses the stored value of M_1^i for M_1^{i+1}. Thus, the attacker calculates α_i as $P(R_t \oplus R_i)$ and replaces R_t^{i+1} with $R_t^i \oplus R_r^i \oplus R_r^{i+1}$, to form the new messages.

2. The reader computes $V = H(ID_r \oplus R_r^{i+1})$ and sends $\{M_1^{i+1}, \beta^{i+1}, \gamma^{i+1}, \alpha_{i+1}, R_r^{i+1}, V^{i+1}\}$ to the back-end server.

3. As the reader is legal, the back-end server authenticates it and to confirm the tag, for each tuple $(EPC_s; K_o, K_n)$, calculates $I_x = M_1 \oplus K_x$. Given that the tag did not update its secret values in the last session, x refers to the old parameters. Then, the back-end server checks if it is equal with $(EPC_s \oplus R_r \oplus \beta \oplus K_x)$, where

$$\begin{aligned} M_1 \oplus K_i &= P(EPC_S \oplus R_r^{i+1} \oplus R_t^{i+1}) \oplus K_i \oplus K_i \\ &= P(EPC_S \oplus R_r^{i+1} \oplus R_t^i \oplus R_r^i \oplus R_t^{i+1}) \\ &= P(EPC_S \oplus R_t^i \oplus R_r^i) \\ &= P(EPC_S \oplus R_r^{i+1} \oplus \beta^{i+1} \oplus K_i). \end{aligned} \quad (15)$$

4. The back-end server checks the correctness $\beta^{i+1} \oplus K_i \oplus P(\alpha_{i+1} \oplus K_i) \stackrel{?}{=} \gamma^{i+1}$ and obtains

$$\begin{aligned} &\beta^{i+1} \oplus K_i \oplus P(\alpha_{i+1} \oplus K_i) \\ &= R_t^i \oplus R_r^i \oplus R_r^{i+1} \oplus K_i \oplus K_i \oplus P(\alpha_{i+1} \oplus K_i) \\ &= R_t^i \oplus R_r^i \oplus R_r^{i+1} \oplus P(\alpha_{i+1} \oplus K_i) = \gamma^{i+1}. \end{aligned} \quad (16)$$

5. Finally, the back-end server authenticates the attacker as a legal tag and calculates R_t^{i+1}.

3.4 Tag traceability attack

Now we show that the UISRS protocol does not provide enough privacy and it is vulnerable to traceability attack which can be applied by an adversary $\mathcal{A}$, as described below:

Learning phase: The attacker sends an execute query (R, T_0, i) and stores the obtained values $\{M_1^i, \beta^i, \gamma^i, \alpha_i, R_r^i, V, M_2^i\}$ in session i and blocks the 5.3 step of the protocol to prevent the tag's updating procedure.

Challenge phase: An attacker $\mathcal{A}$ chooses two fresh tags T_0 and T_1 and sends a Test query $(T_0, T_1, i+1)$. After choosing $b \in \{0,1\}$ randomly, the attacker takes the tag T_b. Then, he/she starts a new session by generating R_r^{i+1} randomly and sending an execute query $R, T_b, i+1$ to the tag T_b, which results in obtaining $\{M_1^{i+1}, \beta^{i+1}, \gamma^{i+1}, \alpha_{i+1}\}$.

Guess phase: The attacker $\mathcal{A}$ stops this procedure and announces b' as his/her guess of b as

$$b' = \begin{cases} 0, & \text{if } \alpha_i \oplus \beta_i = \alpha_{i+1} \oplus \beta_{i+1}, \\ 1, & \text{otherwise.} \end{cases} \quad (17)$$

As a result, we get

$$\begin{aligned} ADV_{\mathcal{A}}^{\text{uprive}}(k) &= \left| \text{pr}(b' = b) - \frac{1}{2} \right| \\ &= \left| 1 - \frac{1}{2} \right| = \frac{1}{2} \gg \varepsilon. \end{aligned} \quad (18)$$

Proof From the structure of protocol in Tab. 2, we observe that preventing the tag updates its secret values during the ith session and using R_t in both α and β messages allow an adversary to trace the tag after a successful eavesdropping.

3.5 Tag forward traceability attack

Providing forward untraceability means, preventing an adversary from tracking the specific tag. Even, the owner of the tag must not be able to trace his/her tag after giving it over. The UISRS protocol does not provide forward untraceability and the attacker can trace the tag T_0, after N runs of the protocol $(\forall N)$. This attack is as follows:

Learning phase: In the ith session of the protocol, an adversary $\mathcal{A}$ sends a Corrupt query (T_0, K') and obtains $\{K_i, P_i, \alpha_i, EPC_s\}$. Weaknesses of updating process in UISRS protocol let the attacker calculate K_{i+2} by performing two times PRNG on K_i.

Challenge phase: An attacker $\mathcal{A}$ chooses two fresh tags T_0 and T_1 and sends a Test query (T_0, T_1, i). After choosing $b \in \{0,1\}$ randomly, the attacker takes the tag T_b. Now, the attacker starts a new session by sending an execute query $(R, T_0, i+2)$ and obtaining $M_1^{i+2}, \beta^{i+2}, \gamma^{i+2}, \alpha_{i+2}, R_r^{i+2}$.

Guess phase: The attacker $\mathcal{A}$ stops the game G and after calculating f, g, h, announces b' as his/her guess of b as follow:

$$f = P(P(K_i^b)), \tag{19}$$

$$g = \beta^{i+2} \oplus f, \tag{20}$$

$$h = P(EPC_s \oplus R_r^{i+2} \oplus g) \oplus f, \tag{21}$$

$$b' = \begin{cases} 0, & \text{if } h = M_1^{i+2}, \\ 1, & \text{otherwise.} \end{cases} \tag{22}$$

As a result, we have:

$$ADV_{\mathcal{A}}^{\text{uprive}}(k) = \left| pr(b' = b) - \frac{1}{2} \right| = \frac{1}{2} \gg \varepsilon. \tag{23}$$

Proof As stated before, the UISRS protocol suffers from weaknesses in updating technique and transmitting constant messages. The attacker calculates K_{i+2}, as in learning phase, and obtains R_t^{i+2} through $\beta^{i+2} \oplus K_{i+2}$. Moreover, since EPC_s is constant in all sessions, the attacker performs as

$$\begin{aligned} h =& P(EPC_s \oplus R_r^{i+2} \oplus g) \oplus f \\ =& P(EPC_s \oplus R_r^{i+2} \oplus \beta^{i+2} \oplus f) \oplus f \\ =& P(EPC_s \oplus R_r^{i+2} \oplus \beta^{i+2} \oplus P(P(K_i^b))) \\ & \oplus P(P(K_i^b)) \\ =& P(EPC_s \oplus R_r^{i+2} \oplus R_t^{i+2}) \oplus K_{i+2}^b \\ =& M_1^{i+2}. \end{aligned} \tag{24}$$

3.6 Tag backward traceability attack

Backward untraceability assures the owner of the tag that no one is able to know what he/she had done before. Another vulnerability to the UISRS protocol in privacy issues is lack of the backward untraceability. Reasons which result in this vulnerability are using the same value for EPC_s in all sessions and predictability of K_{i+1} in each session. An adversary $\mathcal{A}$ performs this attack as follows:

Learning phase: An adversary $\mathcal{A}$ sends a Corrupt query (T_0, K') and obtains $\{K_i, P_i, \alpha_i, EPC_s\}$. As discussed in subsection 3.2, K_i is a 16 bit string which is one of the elements of $\{K_1, K_2, , K_{2^{16}})\}$. Therefore, by knowing the updating process of the tag's secret key, the attacker can guess K_{i-1} at maximum runs of 2^{16}, which is calculated as

For $2 < i < 2^{16}$
choose $K_{i-1} \in \{\Im_1, \Im_2, \cdots, \Im_{2^16}\}$
If $K_i = P(K_{i-1})$ (25)
Return K_{i-1} as a correct secret key
End

P_{i-1} can be found similarly, which is the result of the same inaccuracy in updating procedure. Therefore, the attacker can guess P_{i-1} at maximum runs of 2^{16} as calculated:

For $2 < i < 2^{16}$
choose $P_{i-1} \in \{p_1, p_2, \cdots, p_{2^16}\}$
If $p_i = P(p_{i-1})$ (26)
Return p_{i-1} as a correct access key
End

Challenge phase: An attacker $\mathcal{A}$ chooses two fresh tags T_0 and T_1 and sends a Test query (T_0, T_1, i). After choosing $b \in \{0, 1\}$ randomly, the attacker takes the tag T_b, sends an execute query $(R, T_0, i-1)$ and obtains $\{M_1^{i-1}, \beta^{i-1}, \gamma^{i-1}, \alpha_{i-1}, R_r^{i-1}\}$.

Guess phase: The attacker finishes the game G and announces b' as his/her guess of b as follows:

$$b' = \begin{cases} 0, & \text{if } M_2^{i+2} = P(EPC_S \oplus R_t^*) \oplus P_x, \\ 1, & \text{otherwise.} \end{cases} \tag{27}$$

As a result, we have

$$ADV_{\mathcal{A}}^{\text{uprive}}(k) = \left| pr(b' = b) - \frac{1}{2} \right| = \frac{1}{2} \gg \varepsilon. \tag{28}$$

Proof Choosing a correct value for K_{i-1}, let the attacker calculates R_t^* in the $(i-1)$th session by computing $\beta^{i-1} \oplus K_{i-1}$. Then, M_2^{i+2} can be computed, if he/she knows the precise value for the access key. As a result, the attacker is able to perform backward traceability attack at maximum runs $2 \times 2^{16} = 2^{17}$ computations.

4 Improvements of Cai et al.'s protocols

In this section, we propose an improvement on the IHRMA protocol to overcome its related weaknesses

Table 3 The improved IHRMA protocol

back-end server $(ID_k, S_j, S_{j-1}, DATA)$	reader	tag (ID_k, S_j)
	1.1 request: R_r	
	1.2 $\xrightarrow{R_r}$	2.1 $RID_i = H(S_j \oplus R_t)$
		2.2 $C_1 = H(ID_k \oplus R_r \oplus RID_i)$
	3.1 $\xleftarrow{C_1, C_2 \oplus R_t, R_r}$	2.3 $C_2 = H(S_j(0:47) \| \| ID_k(48:95))$
		2.4 $\xleftarrow{C_1, C_2 \oplus R_t}$
4.1 for each tuple $(ID_k; S_j)$		
calculate $C_2^* = H(S_j(0:47) \| \| ID_k(48:95))$		
calculate $R_t^* = C_2 \oplus C_2^*$		
compute RID_i^*		
calculate $C_1^* = H(ID_k \oplus R_t^* \oplus R_r \oplus RID_i^*)$		
verify $C_1^* \stackrel{?}{=} C_1$		
if it does not match		
repeat with tuple $(ID_k; S_{j-1})$		
if it does not match, reveal the protocol		
4.5 calculate $C_3 = H(RID_i \oplus C_2)$		
4.6 $\xrightarrow{DATA \| \| C_3}$	5.1 $\xrightarrow{C_3}$	6.1 calculate $C_3^* = H(RID_i \oplus C_2)$
4.7 updating		if $C_3^* \neq C_3$
$S_j - 1 \leftarrow S_j \;\; S_j \leftarrow S_j + 1$		reveal the protocol
		else
		server is authenticated
		6.2 updating $S_j \leftarrow S_j + 1$

which is depicted in Tab. 3. The purpose of the improved protocol is keeping the merits of analyzed protocol besides preventing from increasing the amount of complexity and computation.

4.1 Improvements of the IHRMA protocol

Structure of generating RID_i can be highlighted as the greatest weak point in the IHRMA protocol. By changing the RID_i to $H(S_j \oplus R_t)$, the attacker is prevented to access this message. One of the other threats in the IHRMA protocol is the manner of producing and transmitting messages C_3, C_4 and C_5, which lets the attacker leave the protocol unfinished via varying R_t. By omitting C_4 and C_5 messages in our improved protocol and generating C_3 as $H(C_2 \parallel RID_i)$, the adversary is not able to perform DoS attack.

Phase 1: Reader's request

The reader generates a random number R_r and transmit it as a request to the tag.

Phase 2: Tag response

The tag randomly generates a number R_t.

The tag computes $RID_i = H(S_j \oplus R_t)$.

The tag computes $C_1 = H(ID_k \oplus R_r \oplus RID_i)$.

The tag computes $C_2 = H(S_j(0:47) \parallel ID_k(48:95)$ and transmits $\{C_1, C_2 \oplus R_t\}$ oracles to the reader.

Phase 3: Reader response

The reader puts R_r beside the received oracles and transfers $\{R_r, C_1, C_2 \oplus R_t\}$ to the back-end server.

Phase 4: Back-end server authentication and updating

The back-end server stores two last values as the secret values S_j and for each tuple $(ID_k; S_j)$, Computes $C_2^* = H(S_j(0:47) \parallel ID_k(48:95))$, which C_2^* is a guess of C_2.

An estimation for R_t is obtained as $R_t^* = C_2 \oplus R_t \oplus C_2^*$.

The back-end server computes $C_1^* = H(ID_k \oplus R_r \oplus H(S_j \oplus R_t^*))$ and checks its equality with the received message C_1. If they are equal, the back-end server will authenticate the tag as a legal one. The back-end server will compute the new C_2^*, R_t^* and C_1^* for S_{j-1} if the above equality is not provided. The tag's confirmation will occur if C_1^* is equivalent with C_1, else it will reveal the protocol.

The back-end server computes $C_3 = H(RID_i \parallel$

C_2) and transmits $DATA \parallel C_3$ to the reader and updates original S_{j-1} and S_j with S_j and S_{j+1}.

Phase 5: Reader response

The reader checks the $DATA$ and transfers the message C_3 to the tag.

Phase 6: Tag authentication and updating

The tag computes $C_3^* = H(RID_i \parallel C_2)$ and checks $C_3^* \stackrel{?}{=} C_3$. If not, the session is terminated, else the back-end server authentication is completed, then updates S_j by S_{j+1}.

4.1.1 Tag impersonation resistance

Presence of the mod operator in the IHRMA protocol causes the occurrence of Remark 1. In our improved protocol, we change the creature of RID_i through using hash function which prevents the attacker omitting R_t in RID_i. Moreover, as our new messages are preserved by hash functions, the attacker will not be able to substitute R_t^{i+1} with $(R_t^i \oplus R_r^i \oplus R_r^{i+1})$.

4.1.2 Reader impersonation resistance

In this attack, an adversary purposes to impersonate the valid reader. To reach this goal, he/she eavesdrops one session of the protocol and implements the stored values $\{C_2^i, C_5^i\}$. On the other side, the *mod* operator lets the attacker omit the random value of R_t. In our improved protocol, the new definition for RID_i which is based on hash function makes it impossible for an adversary to guess the correct value for $H(S_j \oplus R_t)$. Moreover, the similarity between the transferred messages in the IHRMA protocol resulted in the reader impersonation attack. Therefore, the proposed protocol omits the $\{C_3 \parallel C_4 \parallel C_5\}$ messages and replaces it with C_3. Due to this fact, an adversary cannot use $\{C_4, C_5, RID_i\}$ messages to impose impersonation attacks.

4.1.3 DoS attack resistance

Utilizing R_t in C_4 and C_5 messages in the IHRMA protocol, lets the attacker XOR a random number with them, which prevents the tag from authenticating the back-end server and the reader as a legal parts of an RFID system. In our improved protocol we generate C_3 as $H(RID_i \parallel C_2)$ and by omitting C_4 and C_5 messages, C_3 is the only oracle sent to the tag for authentication. Therefore, an adversary is not able to alter messages as he/she desires, which provides a secure and reliable authentication procedure.

4.1.4 Traceability attack resistance

In traceability attack, the attacker attempts to trace the target tag or reader to obtain information about their location and transferred data. Likeness between transferred messages during one or consecutive sessions will result in privacy attacks. Messages produced in our proposed protocol is completely randomized based on usage of random value R_t in its structure. Moreover, our messages are secured via using a hash function. On the other hand, our secret values are updated at the end of each sessions which prevents an attacker from reusing the stored values during last sessions. Putting these facts together, the proposed protocol provides sufficient privacy to avoid an adversary tracing an element of an RFID system.

4.2 Improved version of UISRS protocol

As described in section 3, the UISRS protocol is vulnerable to different attacks, so in this subsection, we propose an improved version of Cai et al.'s protocol to overcome its weaknesses, which is shown in Tab. 4. Also, the security and privacy analysis of our proposed protocol is provided.Using the $R_t \oplus R_r$ for producing M_1 is one of the main weaknesses in UISRS protocol, which helps the attacker to calculate γ and β. Moreover, updating procedure is the other weakness which results in traceability attacks. Therefore, we improve the UISRS protocol by substituting M_1, α_i and γ_i messages as

$$M_1 = P(EPC_s \oplus R_r) \oplus P(R_t) \oplus K_i, \tag{29}$$

$$\alpha_i = \alpha_i \oplus R_{\text{new}}, \tag{30}$$

$$\gamma = R_t \oplus P(\alpha_i \oplus K_i) \oplus P_i. \tag{31}$$

We define R_{new} as a new random number generated by the tag. Moreover, the updating procedure is

Table 4 The improved version of UISRS protocol

back-end server $(K_0, P_0, \alpha_0, K_n, P_n, \alpha_n, ID_r, EPC_s, DATA)$	reader (ID_r)	tag $(K_i, P_i, \alpha_i, EPC_s)$
	1.1 generate R_r 1.2 $\xrightarrow{R_r}$	2.1 generate R_t and R_{new} 2.2 $M_1 = P(EPC_s \oplus R_r) \oplus P(R_t) \oplus K_i$ 2.3 $\beta = R_t \oplus K_i$ 2.4 $\alpha_i = \alpha_i \oplus R_{\text{new}}$ 2.5 $\gamma = R_t \oplus P(\alpha_i \oplus K_i) \oplus P_i$

4.1 for each ID_r in server
verify $H(ID_r \oplus R_r) \stackrel{?}{=} V$
for each tuple (EPC_s, K_o, K_n)
4.2 calculate
$I_O = M_1 \oplus K_o$ and $I_N = M_1 \oplus K_n$
verify
$I_O \stackrel{?}{=} P(EPC_s \oplus R_r) \oplus P(\beta \oplus K_o) \oplus K_o$
$I_N \stackrel{?}{=} P(EPC_s \oplus R_r) \oplus P(\beta \oplus K_n) \oplus K_n$
$x = O/N$
4.3 verify $\beta \oplus K_x \oplus P(\alpha_i \oplus K_x) \oplus P_x \stackrel{?}{=} \gamma$
4.4 calculate $R_t^* = \beta \oplus K_x$
compute $R_{\text{new}} = \alpha_i \oplus \alpha_x$
4.5 calculate $M_2 = P(EPC_s \oplus R_t^*) \oplus P_x$
4.6 calculate $Info = DATA \oplus ID_r$
4.7 $MAC = H(DATA \oplus R_r)$
4.8 $\xrightarrow{M_2, Info, MAC}$
4.9 updating
if $x = N$
$K_o \leftarrow K_n \leftarrow P(K_x \oplus R_{\text{new}})$
$P_o \leftarrow P_n \leftarrow P(P_x)$
$\alpha_o \leftarrow \alpha_n \leftarrow P(R_t \oplus R_r \oplus P_x)$
else
$\alpha_n \leftarrow P(R_t \oplus R_r \oplus P_x)$
end if
	3.1 $V = H(ID_r \oplus R_r)$
	3.2 $\xleftarrow{M_1, \beta, \gamma, \alpha_i, R_r, V}$
	5.1 $DATA = Info \oplus ID_r$
	5.2 verify
	$H(DATA \oplus R_r) \stackrel{?}{=} MAC$
	5.3 $\xrightarrow{M_2}$
		2.6 $\xleftarrow{M_1, \beta, \gamma, \alpha_i}$
		6.1 verify $M_2 \oplus P_i \stackrel{?}{=} P(EPC_s \oplus R_t)$
		6.2 updating
		$K_{i+1} \leftarrow P(K_i \oplus R_{\text{new}})$
		$P_{i+1} \leftarrow P(P_i)$
		$\alpha_{i+1} \leftarrow P(R_t \oplus R_r \oplus P_i)$

changed to,

$$K_{i+1} \longleftarrow P(K_i \oplus R_{\text{new}}), \tag{32}$$

$$\alpha_{i+1} \longleftarrow P(R_t \oplus R_r \oplus P_i). \tag{33}$$

The improved protocol is depicted in Tab. 4. Now, we describe how the applied modifications remove the mentioned weaknesses of UISRS protocol.

Phase 1: Reader request

The reader generates a random number R_r and sends it the tag.

Phase 2: Tag response

The tag generates two random number R_t and R_{new}.

The tag computes $M_1 = P(EPC_s \oplus R_r) \oplus P(R_t) \oplus K_i$.

The tag computes $\beta = R_t \oplus K_i$.

The tag computes $\alpha_i = \alpha_i \oplus R_{\text{new}}$.

The tag computes $\gamma = R_t \oplus P(_i \oplus K_i) \oplus P_i$ and transmits $\{M_1, \alpha_i, \gamma, \beta\}$ messages to the reader.

Phase 3: Reader response

The reader computes $V = H(ID_r \oplus R_r)$ and transmits $\{M_1, \alpha_i, \gamma, \beta, R_r, V\}$ to the back-end server.

Phase 4: Back-end server evaluations

The back-end server authenticates the reader through computing $(ID_r \oplus R_r)$ for each ID_r and compares it with the received V.

For each two stored values K_0, K_n and EPC_s, the back-end server computes $I_x = P(EPC_s \oplus R_r) \oplus P(\beta \oplus K_x) \oplus K_x$ to investigate whether the value of

the received key is related to the old or new one.

The back-end server computes $\beta \oplus K_x \oplus P(\alpha_i \oplus K_x) \oplus P_x$ and compares it with the received γ. If the above steps are correct, the back-end server authenticates tag as a legal part of an RFID system.

The back-end server computes R_t^* and R_{new} through $\beta \oplus K_x$ and $\alpha_i \oplus \alpha_x$ respectively.

The back-end server computes $M_2 = P(EPC_s \oplus R_t^*) \oplus P_x$, $Info = DATA \oplus ID_r$ and $H(DATA \oplus R_r)$.

The back-end server transmits the $\{M_2, Info, MAC\}$ messages to the reader and updates K_n, P_n and α_n values.

Phase 5: Reader evaluations

The reader computes $DATA = Info \oplus ID_r$.

The reader computes $H(DATA \oplus R_r)$ and compares it with the received MAC. If they are equal, the back-end server is authenticated and the reader transmits M_2 to the tag.

Phase 6: Tag evaluations

The tag calculates $M_2 \oplus P_i$ and compares it with $P(EPC_s \oplus R_t)$. If they are equal, the tag will authenticate the back-end server and the reader.

The tag updates original K_{i+1}, P_{i+1} and α_{i+1} with $P(K_i \oplus R_{\text{new}})$, $P(P_i)$ and $P(R_t \oplus R_r \oplus P_i)$ respectively.

4.2.1 Secret parameter reveal resistance

Revealing the secret parameters, provides imposing different types of attack by an adversary. As we show in section 3, the weaknesses in definition of α_i and its updating method, will result in achieving the correct K_i by the attacker. The improved UISRS protocol prevents revealing random number generated by the tag. Moreover, the new definition of α_i as $P(R_t \oplus R_r \oplus P_i)$, will not permit an attacker to obtain the secret parameter K_i.

4.2.2 Replay attack resistance

An adversary $\mathcal{A}$ tries to verify, deny or omit the transferred massages through impersonating a legal tag or reader, in replay attacks. In our improved protocol, usage of a new random number, R_{new}, and the new structure of produced messages make it impossible for an adversary to re-use the eavesdropped ones through last sessions.

4.2.3 Impersonation attack resistance

In UISRS protocol, it is possible to employ the revealed secret parameter from a latter session which leads to impersonate a tag. An adversary $\mathcal{A}$ has to know the values of K_i, R_t and EPC_s to compute M_1, β and γ messages. By defining a new format for M_1 and α_i, an adversary is not able to access the secret value of the improved UISRS protocol. Besides, it is not feasible to use the last stored M_1 after the current successful run of protocol.

4.2.4 Traceability attack resistance

Our proposed protocol is resistant to eavesdropping and tracing attacks. As discussed in subsection 3.4, the UISRS protocol is suffering from likeness between β and α_i messages, which results in traceability and DoS attacks. In our proposed protocol an attacker is not able to trace the target tag, because of implementing two new random number R_t and R_{new} and defining β and α_i as $R_t \oplus K_i$ and $\alpha_i \oplus R_{\text{new}}$, respectively. On the other hand, as stated in subsections 3.5 and 3.6, the UISRS protocol is vulnerable to forward and backward traceability attacks which are because of the simplicity of updating procedure for confirmation key and access key. In our proposed protocol we employ the random values in the structure of updating procedure as $K_{i+1} = P(K_i \oplus R_{\text{new}})$ and $\alpha_{i+1} = P(R_t \oplus R_r \oplus P_i)$. Therefore, updating the secret parameters with random values make it impossible for an attacker to use the stored values from the last sessions.

5 Results and comparisons

5.1 Performance analysis of our proposed protocol

In this section, we present the performance analysis of our proposed authentication protocols and compare them with some similar protocols in terms of

resistance against different attacks. As our improved protocols are based on the framework of the existing ones, there are not so much difference in structure between the IHRMA and UISRS, and proposed protocols. Considering section 4, Tab. 5 compares the improved IHRMA protocol with the Cai et al.[14] and Cho et al.'s[13] protocols. Although Cho et al.[13] believed that their proposed protocol was not vulnerable to attacks, Cai et al.[14] investigated their protocol and found weaknesses with the Cho et al.'s protocol. They proved that not only Cho et al.'s protocol suffers from lack of data integrity, but also de-synchronization between the server and the tag is its major weaknesses which is stated with more detail in Ref. [14]. As we discussed in section 2, the IHRMA protocol is vulnerable to tag impersonation attack with the probability of 1/4. Also we showed that an adversary can perform reader impersonation attack with the probability of 1/4. In section 4, it is shown that our proposed protocol provides sufficient protection from tag and reader impersonation attacks. Moreover, the improved IHRMA protocol assures that it will not permit an attacker to trace the target tag or imposing denial of service.

Table 5 Comparison of improved IHRMA protocol with similar ones

protocol\feature	v_1	v_2	v_3	v_4
Cho et al.[13]	no	no	no	no
IHRMA[14]	no	no	no	no
our protocol	yes	yes	yes	yes

v_1: Protection from tag impersonation
v_2: Protection from reader impersonation
v_3: Prevention from DoS attack
v_4: Prevention from traceability attack

Yeh et al.[4] claimed that their proposed protocol could be implemented for application with high security performance, while Yoon demonstrated in Ref. [12] that Yeh et al.'s protocol suffered from lack of providing data integrity and traceability issues. Cai et al.[14] described that Yoon protocol could not assure the privacy of the tag owner, so they proposed UISRS protocol. In section 3, we proved that the secret parameter would reveal during an attack. We showed in subsection 3.3 that revealing the secret parameter will lead to tag impersonation attack after maximum 2^{16} runs. Moreover, it is shown in subsection 3.4 that preventing tag from updating its stored values beside implementing the same amount of R_t for generation of messages α and β, makes the UISRS protocol vulnerable to traceability attack. We proved in subsection 3.5 that weaknesses in updating techniques and using constant value for EPC_s in all sessions led to forward traceability attack. We showed that predictability of K_i in the UISRS protocol is vulnerable to backward traceability with $O(2^{17})$ attack complexity in subsection 3.6. Tab. 6 compares our proposed protocol with the above mentioned ones. Our proposed protocol prevents revealing the values of secret parameters which brings protection against replay and DoS attacks. Moreover, the improved protocol provides immunity against traceability, backward and forward traceability and impersonation attacks. We analyzed our proposed protocols in section 4 and proved that they assure privacy and security versus the mentioned vulnerabilities.

Table 6 Comparison of improved IHRMA protocol with similar ones.

protocol\feature	w_1	w_2	w_3	w_4	w_5	w_6	w_7
Yeh et al.[4]	yes	no	yes	yes	no	no	yes
Yoon[11]	no	no	yes	yes	no	no	no
UISRS[13]	no	no	no	no	no	no	no
our protocol	yes	yes	yes	yes	yes	yes	yes

w_1: Secret parameter reveal resistance
w_2: Protection from replay attack
w_3: Protection from impersonation attack
w_4: Prevention from DoS attack
w_5: Prevention from traceability attack
w_6: Prevention from forward traceability attack
w_7: Prevention from backward traceability attack

5.2 Computational performance comparison

In this subsection, we compare our proposed protocols with the studied ones in the matter of stored val-

Table 7 Comparison of improved UISRS protocol with similar ones

protocol	tag	reader	back-end server	storage of tag	storage of server	rounds of communication
Cho et al.[13]	2H	-	4H	2	4	5
IHRMA[13,14]	3H	-	4H	2	4	5
improved IHRMA	4H	-	5H	2	4	5
Yoon[12]	6H	1H	2H+7P	4	9	5
UISRS[14]	6H	1H	2H+7P	4	9	5
improved UISRS	8H	1H	2H+8P	4	9	5

H: Hash function P: PRNG function

ues, complexity and rounds of communication. Our results of comparison are depicted in Tab. 7.

Stored values. Co et al.[13] and IHRMA protocols[14] stored two parameters (ID_k, S_j) in the tag. As we discussed before, developing the RFID system is directly related to the cost of RFID tags. In order to diminish the price and size of tags, there should be the least dependency to the amount of computations and memories of tags in the proposed RFID protocols. As there are not serious worries about the size of the back-end server, it implements stronger processors and higher storage memories. Co et al.[13] and IHRMA protocol[14] stored $(ID_k, S_j, S_{j-1}, DATA)$ parameters in back-end server memories. Our proposed improved IHRMA protocol uses the same parameters in both tag and the back-end server memories.

Our improved UISRS protocol stored four parameters $(K_i, P_i, \alpha_i, EPC_s)$ in the tag, where nine values for the $(K_0, P_0, \alpha_0, K_n, P_n, \alpha_n, ID_r, EPC_s, DATA)$ parameters are stored in the back-end server. As it is shown in Tab. 7, the storage memories for the Yoon[12] and the UISRS[14] protocols are the same with our proposed ones. Therefore, not only our proposed protocols used the same storage memories for authentication process, but also they are not vulnerable to the weaknesses mentioned in sections 2 and 3 anymore.

Complexity: Here, the efficiency of our proposed protocols are compared with the analyzed protocols, by comparing their computational cost. As it is shown in Tab. 7, the improved IHRMA protocol is consisted of four hash functions in the tag side, where the IHRMA protocol contains three hash functions. On the other hand, the proposed protocol holds five hash functions in the back-end server that is one more than the IHRMA protocol. Increasing the complexity of protocols in the back-end server can be handled via using powerful processors. It should be considered that the extent of usage of RFID systems are related to reducing the cost of RFID tags, which brings decreasing the complexity of tag's protocols. But, we must mention that a utilizable authentication protocol should provide adequate security and privacy, beside containing low complexity. Although our improved IHRMA protocol contains more complexity, but it assures a secure and private authentication procedure.

It is also shown that the improved UISRS protocol consisted of eight PRNG functions, where the UISRS protocol includes six ones in the tag side. Moreover, the back-end server consisted of two hash functions and eight PRNG operators in the improved UISRS protocol, where Yoon and Cai et al.'s protocol involved two hash functions and seven PRNG operators. Therefore, privacy analysis shows without increasing too much computational cost, our improved UISRS protocol removes all privacy concerns and provides secure and confidential communications for RFID users.

Rounds of communication: Our proposed improved IHRMA protocol included five rounds of communication. Three of these rounds are related to connection between the reader and the tag, while two others are associated to connection between the reader and the back-end server. There are the

same rounds of communications in the Co et al.[13] and IHRMA[14] protocols. Reducing the complexity and storage memory in tag were our goals for designing improved authentication protocols. Therefore, three rounds of communications are used between the tag and the reader, include transmission of $(C_1, C_2 \oplus R_t, R_r, C_3)$ messages, while authentication process is hold on the back-end server through calculating (C_2, R_t, RID_i) messages to generate C_3.

Yoon[12], UISRS[14] and our proposed improved UISRS protocol are consisted five rounds of communications. Three of them are associated between the reader and the tag, while 2 rounds related to the communication between the reader and the back-end server. So, our proposed protocols assure a secure and private performance without increasing rounds of communication in an RFID authentication system.

6 Conclusion

RFID applications are developing in different areas, which provide comfortability, rapidity and accuracy, but an important issue that must be considered is the assurance of a secure and private connection during the communication procedure. This paper investigates the performance and vulnerabilities of two recent authentication protocols proposed based on R-RAPSE rules. Based on Ouafi-Phan formal privacy model, it is shown that both IHRMA and UISRS protocols cannot provide private authentication for RFID users. To enhance the proficiency of these two protocol, we proposed two improvements in this paper. In addition, we proved that our proposed protocols provide required privacy and security against various types of attacks and can solve the drawbacks of the discussed works.

References

[1] T. Yu, V. Sekar, S. Seshan, et al. Handling a trillion (unfixable) flaws on a billion devices: rethinking network security for the Internet-of-Things [C]//The 14th ACM Workshop on Hot Topics in Networks, HotNets-XIV, Philadelphia, USA, 2015: 5.

[2] K. Baghery, B. Abdolmaleki, M. Emadi. Game-based cryptanalysis of a lightweight crc-based authentication protocol for EPC tags [J]. Amirkabir international journal of electrical & electronics engineering, 2014, 46(1): 27-36.

[3] E. Pagnin, C. Dimitrakakis, A. Abidin, et al. On the leakage of information in biometric authentication [C]//The 15th International Conference on Cryptology in India, New Delhi, India, 2014: 265-280.

[4] T. C. Yeh, Y. J. Wang, T. C. Kuo, et al. Securing RFID systems conforming to EPC Class 1 Generation 2 standard [J]. Expert systems with applications, 2010, 37(12): 7678-7683.

[5] S. S. S. Ghaemmaghami, M. Mirmohseni, A. Haghbin. A privacy preserving improvement for SRTA in telecare systems [J]. arXiv: 1510.04197.

[6] G. Avoine. Cryptography in radio frequency identification and fair exchange protocols [D]. Institut De Systemes De Communication Section Des Systemes De Communication ÉCole Polytechnique FÉDÉRale De Lausanne Pour LObtention Du Grade De Docteures Sciences Par Gildas Avoine Dea DIntelligence Artificielle Et Algorithmique, Université de Caen Basse- Normandie, France, 2005.

[7] S. S. Ghaemmaghami, A. Haghbin, M. Mirmohseni. Traceability improvements of a new RFID protocol based on EPC C1 G2 [J]. The ISC international journal of information security, 2016, 8(2): 99-109.

[8] E. Pagnin, A. J. Yang, G. Hancke, et al. HB+DB, mitigating man-in-the-middle attacks against HB+ with distance bounding [C]//The 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks, New York, USA, 2015: 3.

[9] H. Y. Chien. Sasi: a new ultralightweight RFID authentication protocol providing strong authentication and strong integrity [J]. IEEE transactions on dependable and secure computing, 2007, 4(4): 337-340.

[10] L. Batina, J. Guajardo, B. Preneel, et al. Public-key cryptography for RFID tags and applications [C]//RFID Security, Springer, 2008: 317-348.

[11] K. Fan, N. Ge, Y. Gong, et al. Ultras: ultra-lightweight RFID authentication scheme for mobile device [C]//International Conference on Wireless Algorithms, Systems, and Applications, Qufu, China, 2015: 114-122.

[12] E. J. Yoon. Improvement of the securing RFID systems conforming to EPC Class 1 Generation 2 standard [J]. Expert systems with applications, 2012, 39(1): 1589-1594.

[13] J. S. Cho, S. S. Yeo, S. K. Kim. Securing against brute-force attack: a hash-based RFID mutual authentication protocol using a secret value [J]. Computer communications, 2011, 34(3): 391-397.

[14] Q. Cai, Y. Zhan, J. Yang. The improvement of RFID authentication protocols based on R-RAPSE [J]. Jour-

nal of networks, 2014, 9(1): 28-35.
[15] C. H. Lim, T. Kwon. Strong and robust RFID authentication enabling perfect ownership transfer [C]//The 8th International Conference on Information and Communications Security, Raleigh, USA, 2006, 1-20.
[16] S. Vaudenay. On privacy models for RFID [C]//The 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Malaysia, 2007: 68-87.
[17] K. Ouafi, R. C. W. Phan. Privacy of recent RFID authentication protocols [C]//Information Security Practice and Experience, Springer, 2008: 263-277.
[18] A. Juels, S. A. Weis. Defining strong privacy for RFID [J]. ACM transactions on information and system security, 2009, 13(1): 7.
[19] G. Avoine, I. Coisel, T. Martin. Untraceability model for RFID [J]. IEEE transactions on mobile computing, 2014, 13(10): 2397-2405.
[20] D. Moriyama, S. Matsuo, M. Ohkubo. Relations among notions of privacy for RFID authentication protocols [J]. IEICE transactions on fundamentals of electronics, communications and computer sciences, 2014, 97(1): 225-235.
[21] G. Avoine. Adversarial model for radio frequency identification [Z]. IACR Cryptology ePrint Archive, 2005: 49.

About the authors

Seyed Salman Sajjadi Ghaem-Maghami [corresponding author] obtained his M.S. degree in electrical engineering communications from Science and Research Branch Islamic Azad University, Tehran, Iran in 2015 and B.S. degree in electrical engineering electronic from Karaj Islamic Azad University, Karaj, Iran, in 2010. His research interests include lightweight cryptography, RFID security and privacy, Internet of Things, and wireless communications. (Email: salman.ghaemmaghami@srbiau.ac.ir)

Afrooz Haghbin obtained her B.S. degree in electrical engineering from Sharif University of Technology, Tehran, Iran, in 2001. She obtained her M.S. degree from Tehran University and her Ph.D. degree from Tarbiat Modares University, Tehran, Iran, all in electrical engineering in 2004 and 2009, respectively. She is currently with the electrical and Computer Department of Science and Research Branch in Azad University, Tehran, Iran, as assistant professor. Her research interests include MIMO wireless communications, channel coding, precoding, multicarrier modulation and estimation theory. (Email: haghbin@srbiau.ac.ir)

Mahtab Mirmohseni is an assistant professor at Department of Electrical Engineering, Sharif University of Technology (SUT), since 2014. She is also affiliated with the Information Systems and Security Laboratory (ISSL), Sharif University of Technology, Tehran, Iran. She received the B.S., M.S. and Ph.D. degrees from department of electrical engineering, Sharif University of Technology, Tehran, Iran in the field of communication systems in 2005, 2007 and 2012, respectively. She was a postdoctoral researcher at Royal Institute of Technology (KTH), Stockholm, Sweden, in the School of Electrical Engineering till February 2014. Her current research interests include different aspects of information theory, mostly focusing on molecular communication, secure communication and energy-constrained networks. (Email: mirmohseni@sharif.edu)