A Cloud-based RFID Authentication Protocol with Insecure Communication Channels

Hannan Xiao
School of Computer Science
University of Hertfordshire
Hatfield, United Kingdom
h.xiao@herts.ac.uk

Awatif Ali Alshehri
School of Computer Science
University of Hertfordshire
Hatfield, United Kingdom
aa13ain@herts.ac.uk

Bruce Christianson
School of Computer Science
University of Hertfordshire
Hatfield, United Kingdom
b.christianson@herts.ac.uk

***Abstract*— Radio Frequency Identification (RFID) has become a widespread technology to automatically identify objects and with the development of cloud computing, cloud-based RFID systems attract more research these days. Several cloud-based RFID authentication protocols have been proposed to address privacy and security properties in the environment where the cloud provider is untrusted therefore the tag's data are encrypted and anonymously stored in the cloud database. However, most of the cloud-based RFID authentication protocols assume secure communication channels between the reader and the cloud server. To protect data transmission between the reader and the cloud server without any help from a third party, this paper proposes a cloud-based RFID authentication protocol with insecure communication channels (cloud-RAPIC) between the reader and the cloud server. The cloud-RAPIC protocol preserves tag privacy even when the tag does not update its identification. The cloud-RAPIC protocol has been analyzed using the UPriv model and AVISPA verification tool which have proved that the protocol preserves tag privacy and protects data secrecy.**

Keywords*— *RFID; Cloud-based; Authentication protocol; AVISPA; Privacy; Insecure channel

I. INTRODUCTION

Radio Frequency Identification (RFID) is a wireless technology that uses radio signals to automatically identify objects. An RFID system consists of RFID tags, RFID readers, and a backend server. Tags are attached to objects to identify them uniquely and store the identification information of objects. The reader and the tag exchange data via radio signals. In passive RFID tags, the reader requests the tag to transmit its secret data when it enters the reader zone. The reader then forwards the data to the backend server, which manages the tag's data in its database. The main advantage of using RFID systems is their ability to provide contactless identification of many physical objects at low cost, however, security and privacy concerns have become a major issue. The adversary has the ability to eavesdrop on messages exchanged between the tag and the reader, modify and block messages, and trace and corrupt tags. A broad range of research has thus been conducted to address security problems of RFID systems among which mutual authentication between tag, reader, and server is a challenging issue.

There are two types of architecture used in RFID mutual authentication schemes where both the tag and the reader tell each other their own identities: server-based authentication and server-less authentication [1]. The server-based RFID architecture involves the use of a secure backend server that stores tags' data. In order to identify a tag, the reader communicates with the tag using radio signals, which are assumed to be insecure. Then, the reader sends the tag response through a secure communication channel to the back-end server. The back-end server authenticates the reader and verifies the tag's identity using the stored secret information of tags, and informs the reader if the tag is valid.

In the mutual authentication of a server-less RFID, the readers authenticate tags without the intervention from an online back-end server. Instead, the portable reader initially communicates with a Certificate Authority (CA) via a secure connection to download an Access List (AL). The AL contains the tag identification as index and the corresponding hash digest of the reader identification and tag's secret key as a certificate to prevent forging. The readers can then authenticate tags offline without any help from the back-end server.

With the rapid growth in cloud computing, a new RFID authentication scheme, which takes the advantages of cloud services, is gaining more attention these days. A cloud-based RFID authentication scheme reduces the cost of deploying RFID systems and improves system scalability. However, in this scheme, the cloud server may disclose the secret data of tags to remote attackers [2]. In addition, the reader usually communicates with the cloud server by using a wireless network. Thus, the tags' data should be anonymously stored in the cloud server, and securing the data transmission between the reader and the cloud server is crucial.

In 2012, Kiraz, Bingöl, Kardaş, and Birinci [3] proposed two anonymous RFID mutual authentication protocols for cloud-based RFID architecture without using a trusted third party. Their proposals were designed to prevent a server side attack where the server administrator may disclose information from a user profile. The protocols are based on threshold cryptosystems to preserve tag anonymity even if the server is corrupted. The first protocol is based on $(2, n)$ threshold homomorphic encryption which does not provide tag revocation. Thus, they proposed the other protocol to provide tag revocation up to t tags; which is based on (t, n) threshold homomorphic encryption.

In 2013, Xie, Xie, Zhang, Zhang and Tang [1] proposed a cloud-based RFID authentication protocol was proposed which consists of four parties: RFID tag, RFID reader (referred to as cloud-RAP2013 in this paper), VPN agency, and a cloud server.

It is assumed that the wireless communication channel between the reader and the tag is not secure, whereas the reader communicates with the cloud server through a virtual private network (VPN) with the help of a VPN agency. The tags and the readers' data are stored in the cloud anonymously as an encrypted hash table because the cloud server is untrusted. The VPN agency provides a secure connection between the reader and the cloud server. However, the cost of implementing this protocol is relatively high due to the need of deploying a VPN agency to secure the communication channel between the reader and the cloud server. In the same year Kardas, Celik, Bingol and Levi [4] published another security and privacy RFID mutual authentication protocol was published which uses cloud services to provide system availability and scalability. The authors were concerned whether the confidentiality and privacy properties can be maintained with an untrusted cloud provider. Thus, they defined a new security and privacy model for cloud-based RFID authentication protocols and give the adversary different capabilities. Then, they evaluated their protocol, which achieves the privacy requirement based on their proposed model.

In 2014, Chen, Wu, Sun, and Wang [5] proposed a privacy preserving authentication protocol for RFID systems that utilizes cloud computing was published. The scheme reduces search complexity from a linear search $O(N)$ to a logarithmic search $O(log\, N)$, where N is the number of tags in the system. In addition, the protocol is designed to prevent de-synchronization and tracking attacks. The proposed protocol also avoids weakness in the tree-based structure since the adversary can obtain all the tags' information if he can compromise a small number of tags. Also in 2014, Lin, Hsu, and Cheng [6] published a cloud-based authentication protocol for RFID supply chain systems which consists of four parties: RFID tag, RFID reader, a trust party, and a cloud database. The trust party helps to protect data during the transfer of ownership by updating and re-encrypting the data. Nonetheless, the protocol cannot resist the denial of service attack since the adversary can replay $h(R1), R1 \oplus K_r$ to the tag many times, where K_r is a secret key shared between the reader and the tag, and $R1$ is the reader's pseudorandom number.

In 2015, Abughazalah, Markantonakis, and Mayes [7] have analyzed the security of the cloud-based RFID authentication protocol [1], and pointed out that the cloud-RAP2013 [1] protocol violates data privacy and is vulnerable to location tracking and reader impersonation attacks in which the adversary can be authenticated as a legitimate reader without compromising the tag's secret data. This vulnerability related to the way of calculating the new secret key of the tag. To enhance the cloud-RAP2013 [1] protocol, an improved protocol called "Secure improved cloud-based RFID authentication protocol" was proposed in [7] (referred to as cloud-RAP2015 in this paper). It is assumed that the communication channel between the reader and the cloud server is secure whereas the cloud server may leak information to malicious attackers. However, this assumption is not practical for portable readers since they communicate with the cloud server via a wireless network. In addition, the protocol does not protect tag privacy between any two authentication sessions. Therefore, the attacker can intercept any data transmitted between the reader and the cloud server to perform any possible attack. In addition, the tags' data should be stored anonymously because the cloud provider is untrusted.

However, none of the examined protocols above was designed with the purpose of protecting data transmissions in cloud-based RFID systems with insecure communication channels between the reader and the cloud server.

Motivated by the above observations, the research in the paper is aimed to propose a Cloud-based RFID Authentication Protocol capable to resisting the attacks that may arise in the Insecure Communication channel (Cloud-RAPIC) between the reader and the cloud server and to prevent the location tracking attack even if the tag fails to update its identifier. The contribution of this paper is that we have improved the cloud-RAP2015 protocol [7] to meet the security and privacy requirements for cloud-based RFID systems with mobile readers. We show that our protocol can resist attacks of RFID systems with insecure channels between the reader and the cloud server. Finally, we compare all of the three cloud-based RFID authentication protocols including the cloud-RAP2013 protocol [1], the cloud-RAP2015 protocol [7] and cloud-RAPIC in terms of privacy and security properties satisfied.

The rest of the paper is organized as follows: Section II discusses RFID system privacy and security requirements and the adversary model. In Section III, we describe the proposed cloud-RAPIC protocol. Section IV shows the evaluation of the cloud-RAPIC protocol and a conclusion will be highlighted in Section V.

II. RFID System Security and Adversary Model

A. Privacy and Security Requirement

The requirements of RFID systems can be grouped into two main categories: privacy and security [8[. Tag privacy is one of the most important issues since the tag is attached to a specific product or its owner.

- *Tag data anonymity:* The tag's secret data should not be revealed to any unauthorized entity. Thus, only a legitimate party can access these data.
- *Tag location privacy:* The adversary should not be able to identify the tag or find a link between any two anonymous transactions performed by the same tag.

The following security requirements are important to be satisfied in the design of RFID authentication protocols.

- *Confidentiality:* Ensure the secrecy of all secret data and encrypt them when they are passed through wireless links.
- *Forward Untraceability:* The tag's transmitted data should not be the same for different transactions. The knowledge of a tag's secret data enables the attacker to trace the future transactions of the compromised tag.
- *Backward Untraceability:* The adversary should not be able to establish a link between the current and the past transaction. Thus, preventing the adversary from tracing all of the past transactions using knowledge of a tag's current secret information is important because the

knowledge of a current secret may expose past transactions.

- *Resistance to replay attack*: The adversary should not be able to forward any of the previously exchanged messages to be authenticated as a legitimate party.
- *Resistance to de-synchronization:* A legitimate tag should be able to be authenticated by the reader when needed. Thus, RFID protocols usually keep a copy of the previous tag's data until the next authentication session.
- *Resistance to tag impersonation:* The adversary should not be able to impersonate a legitimate tag to a legitimate reader unless the tag is compromised.
- *Resistance to reader impersonation:* The adversary should not be able to impersonate a legitimate reader even when colluding with a compromised tag.
- *Mutual authentication:* A legitimate reader, tag and the cloud server should be able to safely authenticate each other using a secure mechanism.
- *Data integrity:* Prevent the modification of tag data sent over insecure channel by using a verification technique that ensures data accuracy and consistency.

B. Adversary Model

We assume that an adversary A has full control of the wireless communication channel between the reader and the tag. Thus, A is able to perform a passive attack by eavesdropping and analyzing messages over legitimate sessions of the protocol. Additionally, A can perform an active attack such as store, edit, corrupt, replay, block, and inject messages. We also assume that A always knows the cipher text, how the protocol works and the functions executed at each party. Therefore, A can potentially impersonate a legitimate tag or reader by following the steps specified under the protocol. We define two types of the adversary model according to his capabilities.

- *Weak adversary model:* The adversary has all of the capabilities mentioned above but he is not able to access the data in the cloud server.
- *Strong adversary model:* The adversary has all of the capabilities mentioned above. In addition, he also has the capability to access the data in the cloud server and analyse the data stored in the database to find any relationship between them.

III. THE CLOUD-RAPIC PROTOCOL

A. Design Principle and Notations

The goal of the proposed protocol is to protect tag's data in an untrusted cloud server and achieve the privacy and security properties for RFID systems with mobile readers. The proposed cloud-RAPIC protocol is based on the strengths of cloud-RAP2013 protocol [1] and the robustness of the cloud-RAP2015 protocol [7]. In addition, we introduce ideas in order to prevent attacks that are found in both protocols and secure data between the reader and the cloud server.

TABLE I. NOTATIONS USED IN THE CLOUD-RAPIC PROTOCOL

Notation	Description
Tid_i	A unique identifier of the i^{th} tag
Rid	The reader identifier
K_i	The secret key of the i^{th} tag
K_{rs}	A symmetric key shared between the reader and the cloud server
$R1$	The reader's pseudorandom number
$R2$	The tag's pseudorandom number
$E_{mk}(.)$	Encrypted data using a master key shared between all system readers
$D_{mk}(E_{mk}(.))$	Decryption of the encrypted data
J	The transaction number
$h(Rid\|\|Tid_{new}^{J+1})$	A one way hash function of Rid and the new tag identifier Tid
$E_{mk}(Tid_{new}^{J+1}\|\|K_{new}^{J+1}\|\|x)$	Encrypted tag's new data for the next transaction
$h(Rid\|\|Tid_{old}^{J})$	A one way hash function of Rid and the old tag identifier Tid
$E_{mk}(Tid_{old}^{J}\|\|K_{old}^{J}\|\|x)$	Encrypted tag's old data of the previous transaction
x	A value specifies whether the tag keeps new/old values of Tid and K
$h(.)_L$	The left half part of the hash value
$h(.)_R$	The right half part of the hash value

The cloud-RAPIC protocol uses the notations listed in Table II, extending from those used in cloud-RAP2013 [1] and cloud-RAP2015 [7].

B. The Encryted Hash Table in Cloud-RAPIC

In cloud-based RFID authentication protocols, the data of the rags and readers are stored in the cloud anonymously as an encrypted hash table (EHT). The proposed Cloud-RAPIC protocol uses $h(Rid\|Tid_i)$ as index field in the EHT and $h(Tid_i\|K_i\|x)$ in the data field as shown in Table II, i.e. Tid_i and K_i are updated after each successful protocol execution. In the cloud-RAPIC protocol, we assume that the communication channel between the reader and the cloud server is not secure.

C. Protocol Description

The proposed cloud-RAPIC protocol has two phases: the initialization phase and the authentication phase.

Initialization phase: The system operator should do the following in a closed environment before using the protocol.

1) *In the* ith *tag:* the system operator assigns the tag Tid_I, K_I and Rid.
2) *In the reader:* the system operator assigns the Rid, the reader's master key mk and the shared secret key K_{rs} with the cloud server.

TABLE II. THE ENCRYPTED HASH TABLE FORMAT OF CLOUD-RAPIC

Old_index_field	***Old_data_field***	***New_index_field***	***New_data_field***
$h(Rid\|\|Tid_{old}^{J})$	$E_{mk}(Tid_{old}^{J}\|\|K_{old}^{J}\|\|x)$	$h(Rid\|\|Tid_{new}^{J+1})$	$E_{mk}(Tid_{new}^{J+1}\|\|K_{new}^{J+1}\|\|x)$
⋮	⋮	⋮	⋮
$h(Rid\|\|Tid_{old}^{J})$	$E_{mk}(Tid_{old}^{J}\|\|K_{old}^{J}\|\|x)$	$h(Rid\|\|Tid_{new}^{J+1})$	$E_{mk}(Tid_{new}^{J+1}\|\|K_{new}^{J+1}\|\|x)$

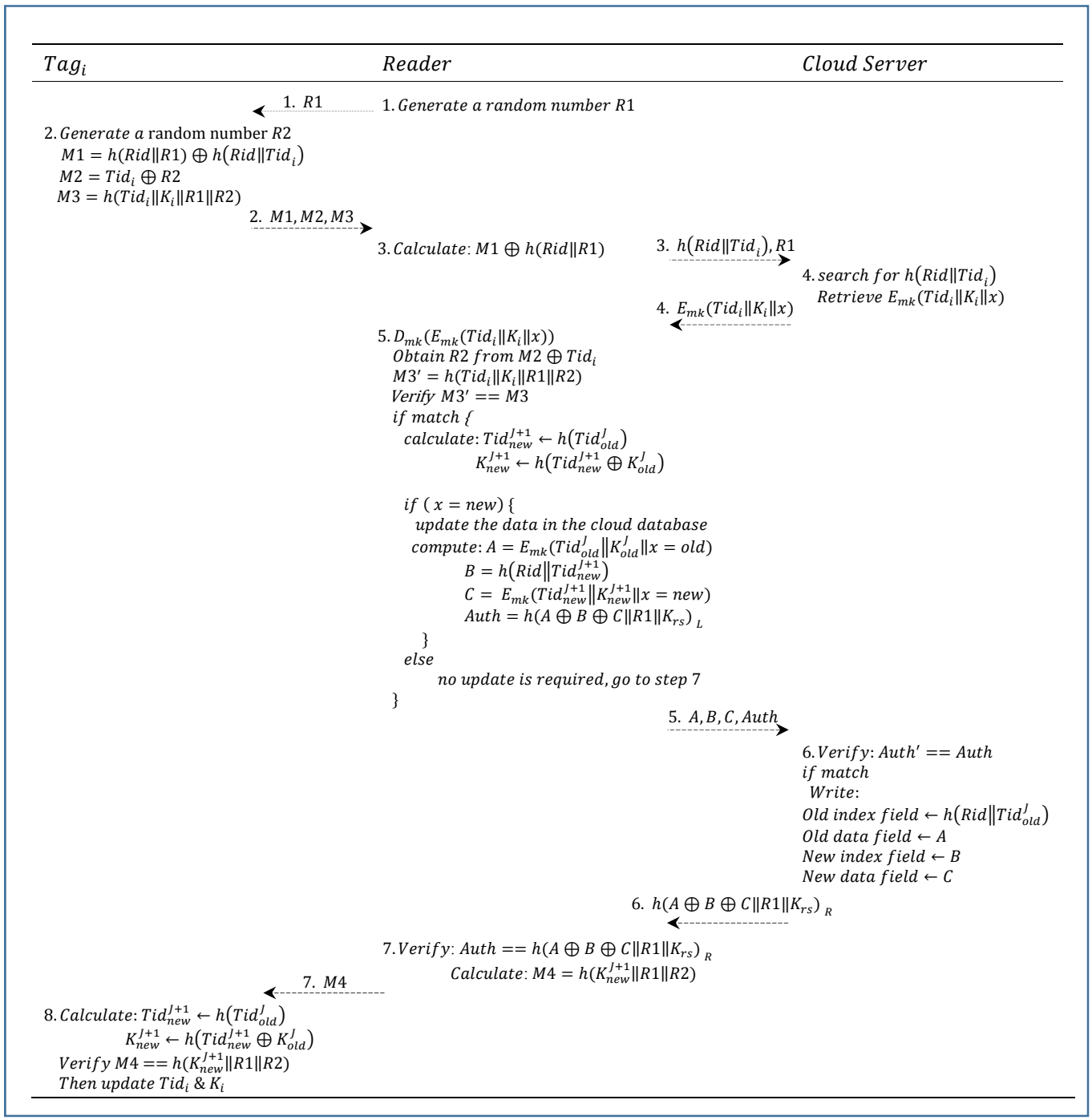

Fig. 1. The authentication phase of the cloud-RAPIC protocol

3) *In the cloud server:* the system operator assigns the key K_{rs} shared between the reader and the cloud server, a unique $h(Rid\|Tid_i)$ in the New_index_field and the corresponding encrypted data $E_{mk}(Tid_i\|K_i\|new)$ in the New_data_field. The data in the Old_index_field and Old_data_field are set to null initially.

Authentication phase: This process involves the following sequence of steps as illustrated in Fig. 1.

1) $\boldsymbol{Reader \rightarrow Tag_i: R1}$

The reader generates a random number $R1$ and sends it as a challenge to the tag.

2) $\boldsymbol{Tag_i \rightarrow Reader: M1, M2, M3}$

The tag generates $R2$ and calculates the following for authentication purpose:

$M1 = h(Rid\|R1) \oplus h(Rid\|Tid_i)$, $h(Rid\|Tid_i)$ is used as an index field in the cloud server to retrieve the tag_i data.

$M2 = Tid_i \oplus R2$

$M3 = h(Tid_i\|K_i\|R1\|R2)$, $M3$ is used to authenticate the i^{th} tag.

3) $\boldsymbol{Reader \rightarrow Server: h(Rid\|Tid_i), R1}$

The reader calculates $M1 \oplus h(Rid\|R1)$ and sends $h(Rid\|Tid_i), R1$ to the cloud server in order to retrive the data related to the i^{th} tag.

4) $\boldsymbol{Server \rightarrow Reader: E_{mk}(Tid_i\|K_i\|x)}$:

The server searches among all values stored in the $Old\ index\ field$ and New_index_field (see Table II) for the matching $h(Rid\|Tid_i)$. If there is a match, it retrieves the associated $E_{mk}(Tid_i\|K_i\|x)$ and sends it to the reader where x specifies whether the tag stores the new or old data of Tid_i and K_i in its memory.

5) $\boldsymbol{Reader \rightarrow Server: A, B, C, Auth}$

The reader obtains the tag Tid_i, K_i and x by decrypting the data using its master key mk. Then, it extracts $R2$ from $M2 \oplus Tid_i$ to verify the correctness of the received $M3$. The reader calculates $M3' = h(Tid_i\|K_i\|R1\|R2)$ to authenticate the tag. Moreover, the reader authenticates the cloud server if the data in the cloud are identical to those received from the tag.

If $M3' == M3$, the reader updates the tag's data as following:

$Tid_{new}^{J+1} \leftarrow h(Tid_{old}^{J})$

$K_{new}^{J+1} \leftarrow h(Tid_{new}^{J+1} \oplus K_{old}^{J})$

Then, the reader checks the synchronization of tag's data with the data in the cloud server based on the value of x.

- If $x = new$, the tag is synchronized and the data should be updated to be used in the next transaction:

$A = E_{mk}(Tid_{old}^{J}\|K_{old}^{J}\|x = old)$

$B = h(Rid\|Tid_{new}^{J+1})$

$C = E_{mk}(Tid_{new}^{J+1}\|K_{new}^{J+1}\|x = new)$

The reader computes and sends the left half of the authentication message for integrity and authentication purpose, *i.e.,* $Auth = h(A \oplus B \oplus C\|R1\|K_{rs})_L$.

- If $x = old$, no update is required and go to step 7).

6) $\boldsymbol{Server \rightarrow Reader: h(A \oplus B \oplus C\|R1\|K_{rs})_R}$

The server checks the integrity of the left half of the received $Auth == h(A \oplus B \oplus C\|R1\|K_{rs})_L$. If a match is found, the server updates the database and replies with the right half $Auth$ of as an acknowledgement to the reader:

$Old\_index\_field \leftarrow h(Rid\|Tid_{old}^{J})$

$Old\_data\_field \leftarrow E_{mk}(Tid_{old}^{J}\|K_{old}^{J}\|old)$

$New\_index\_field \leftarrow h(Rid\|Tid_{new}^{J+1})$

$New\_data\_field \leftarrow E_{mk}(Tid_{new}^{J+1}\|K_{new}^{J+1}\|new)$

7) $\boldsymbol{Reader \rightarrow Tag_i: M4}$

The reader verifies the right half part of the received authentication message $h(A \oplus B \oplus C\|R1\|K_{rs})_R$ only if step 3) is performed by the protocol. Then, the reader computes $M4 = h(K_{new}^{J+1}\|R1\|R2)$ in order to be authenticated by the tag and inform the tag to update its data.

8) $\boldsymbol{Tag_i}$: The tag calculates the following:

$Tid_{new}^{J+1} \leftarrow h(Tid_{old}^{J})$

$K_{new}^{J+1} \leftarrow h(Tid_{new}^{J+1} \oplus K_{old}^{J})$, J is the transaction number.

Then, the tag verifies the correctness of the received $M4 == h(K_{new}^{J+1}\|R1\|R2)$; if it compares successfully, the reader will be authenticated and the tag's data will be updated. Otherwise, the reader will not be authenticated and the tag will keep its old data of Tid_{old}^{J} and K_{old}^{J}.

IV. Security Analysis of Cloud-RAPIC

This section analyses the cloud-RAPIC protocol and formally verifies the privacy and security requirements discussed in section II.A.

A. Analysis of The Cloud-RAPIC Protocol

- *Tag data anonymity:* The tag's secret data are encrypted before being transmitted in the insecure channel. Moreover, the index field in the cloud database, which is used to extract the encrypted tag's data is stored as a hash value of the reader Rid concatenated with tag Tid_i as shown in Table II. Thus the adversary cannot establish any link between the Old_index_field and the New_index_field.

- *Tag location privacy (Untraceability):* Although the tag updates its secret data $\mathrm{Tid_i}$ and $\mathrm{K_i}$ in the cloud_RAP2015 protocol [7] after each successful session, the tag is still vulnerable to location tracking attack until it updates its $\mathrm{Tid_i}$ since the tag response is static for $\mathrm{h(Tid_i)}$ in M1. Thus, in the cloud_RAPIC protocol we modify the tag response in M1 using Rid which is known previously to the tag during the initialization phase of the protocol. Therefore, whenever the tag is queried, its response will appear random due to the use of a fresh random number R1 within the hash function.
- *Mutual Authentication:* the cloud_RAP2015 protocol [7] efficiently satisfies the mutual authentication between the reader and the tag but not between the reader and the cloud server. Thus, in the cloud_RAPIC protocol we introduce the use of the secret key $\mathrm{K_{rs}}$ to provide an authentication procedure between the reader and the cloud server, which is used to calculate the message $\mathrm{Auth} = \mathrm{h(A \oplus B \oplus C \| R1 \| K_{rs})}_{\mathrm{L}}$ in order to authenticate the reader by the cloud server.
- *Resistant to replay attack:* The cloud_RAPIC protocol resists replay attacks between the reader and the cloud server using the reader's random number R1 in the message $\mathrm{Auth} = \mathrm{h(A \oplus B \oplus C \| R1 \| K_{rs})}_{\mathrm{L}}$. Thus, the adversary cannot reuse this message in later sessions of the protocol.
- *Resistant to de-synchronisation attack:* The value that specifies the synchronisation of tag's data (x=*old/new*) is encrypted by the reader's master key in $\mathrm{E_{mk}(Tid_i \| K_i \| x)}$. Thus, the adversary cannot change this value since the reader is the only entity that can extract this value.
- *Tag data integrity:* The authentication message $\mathrm{h(A \oplus B \oplus C \| R1 \| K_{rs})}_{\mathrm{L}}$ is also used to ensure the integrity of the $\mathrm{A, B}$ and C messeges sent to the cloud server. When the cloud server receives $\mathrm{A, B}$ and C it re-calculates the Auth message using the received R1 and the secret key $\mathrm{K_{rs}}$ share with the reader.

B. Formal Analysis

In this section, we analyze the cloud-RAPIC protocol to verify the privacy of tag's holders and the security of tag's data.

1) Verifying the tag's privacy using UPriv:

We evaluate the privacy requirements using the UPriv model (UPriv) proposed by Ouafi and Phan [9] which was proposed to evaluate the privacy requirement in RFID protocols. The model can be described as follows:

The protocol parties are $T \in Tags$ and $R \in Readers$ that interact with each other in protocol sessions. The communication channel between the tag and reader is assumed to be fully controlled by the adversary A who communicates with them passively or actively. The communication channel between the reader and the cloud server is assumed to be fully controlled by the adversary A too. The Adversary A has the ability to issue the following queries:

- *Execute (R, T, i) query*: This query allows *A* to perform a passive attack by eavesdropping a session *i* between *R* and *T* during an honest execution of the protocol. Therefore, *A* can read all messages exchanged between *R* and *T*.
- *Send (U, V, m, i) query*: This query allows the adversary *A* to perform an active attack by impersonating a legitimate reader *U∈Readers* in a session *i* of the protocol and sending a message *m* to a tag *V∈Tags*. In addition, the adversary is allowed to modify or block a message *m* during its transmission between *R* and *T*.
- *Corrupt (T, K) query*: This query returns the secret key *K_i* of a tag *T∈Tags* if the adversary can perform a physical attack and get access to the tag's memory.
- *Test (T, i) query*: This query allows the adversary A to invoke a Test query for session i. Depending on a randomly selected bit *b∈{0,1}*, *A* is given *T_b∈{T_0,T_1}* randomly, where *T_0* represents the targeted tag and *T_1* is any other tag in the system. If *A* can guess the bit *b* correctly, then he will succeed to identify his target.

Untraceable privacy (UPriv) [9] is defined by using the game g played by an adversary *A* and a collection of session instances between the reader and the tag. The game g involves three phases: learning, challenge, and guess phase.

- *Phase 1 (Learning):* The adversary *A* eavesdrops on a legitimate session $(\mathrm{i}+1)$ using *Execute (*$\mathrm{R, T_0, i+1}$*) query* during the execution of the protocol between a tag $\mathrm{T_0}$ and the reader R. Thus, A obtains the following:

  $$\mathrm{M1} = \mathrm{h(Rid \| R1)} \oplus \mathrm{h\left(Rid \| Tid_{T0}\right)}$$

  $$\mathrm{M2} = \mathrm{Tid_{T0}} \oplus \mathrm{R2}$$

  $$\mathrm{M3} = \mathrm{h(Tid_{T0} \| K_{T0} \| R1 \| R2)}$$

- *Phase 2 (Challenge):* A is given a tag $\mathrm{T_b} \in \{\mathrm{T_0}, \mathrm{T_1}\}$, A starts a new session with $\mathrm{T_b}$ and generates his random number Ri ,then A sends Ri using a *Send (*$\mathrm{R, T_b, Ri, i+1}$*) query* to the tag $\mathrm{T_b}$ and obtains the following:

  $$\mathrm{M1} = \mathrm{h(Rid \| Ri)} \oplus \mathrm{h\left(Rid \| Tid_{Tb}\right)}$$

  $$\mathrm{M2} = \mathrm{Tid_{Tb}} \oplus \mathrm{R2}$$

  $$\mathrm{M3} = \mathrm{h(Tid_{Tb} \| K_{Tb} \| Ri \| R2)}$$

- *Phase 3 (Guess):* The adversary *A* cannot find any relationship between the tag T_0 response in the learning phase and the tag T_b response in the challenge phase since the response is changed every time the tag is queried.

This is due to the use of fresh random numbers $R1$ and $R2$ in all messages $M1, M2,$ and $M3$. Moreover, the tag response looks random whenever it queried since the reader random number $R1$ is encrypted within the hash value in $M1$.

2) Verifying data secrecy using AVISPA:

A number of successful verification tools have been proposed by developers to analyze and validate security protocols, such as Casper/FDR developed in [10], the extended ProVerif [11],

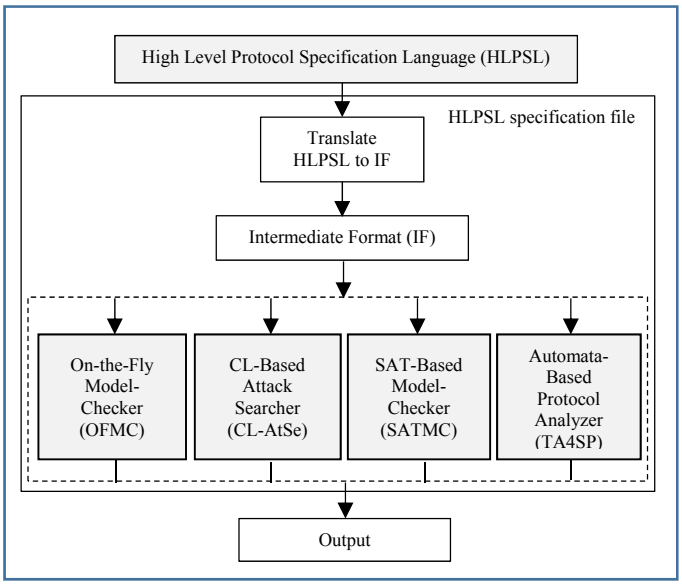

Fig. 2. The architecture of the AVIPA tool

and Automated Validation of Internet Security Protocols and Applications (AVISPA) developed by Armando *et al.* [12]. These tools use different models and verification techniques.

We formally evaluated the security properties of the protocol using the AVISPA verification tool [12] since it employs four back-ends with different validation techniques: On-the-Fly Model-Checker (OFMC), Constraint Logic-Based Attack Searcher (CL-AtSe), SAT-Based Model-Checker (SATMC), and Tree Automata-Based Protocol Analyzer (TA4SP). These model checker tools are all based on the same specification language, called High Level Protocol Specification Language (HLPSL). The AVISPA platform uses a translator called HLPSL2IF, which transforms a HLPSL specification of security protocols written by the user into a low level specification called Intermediate Format which can be analyzed by the four back-end tools as shown in Fig.2.

Six HLPSL sections were declared: the reader, the tag, the server, the session, the environment roles, and the security goal section. Moreover, we defined the adversary's capabilities and the goals for each security property that the proposed protocol should satisfy.

Secrecy is modelled by means of the goal predicate *secret.* For instance, the secrecy of K_{rs} shared between the reader and the cloud server is modelled by means of the goal predicate *secret(Krs,rs_Acc_Pass,{R,S})* which implies that the K_{rs} is a secret shared between agent *R* and agent *S* and *rs_Acc_Pass* is the secret term that should be declared as a constant in the goal section. When the adversary learns this secret value, the security property will be violated.

Authentication property is modelled by means of witness and request events. For instance, the authentication property between the reader and the cloud server is modelled by means of witness and request as following: *R* and the *S* should certainly agree on the exchanged message $Auth$. Thus, we have modelled a strong authentication of *R* by *S* using the goal predicate

```
% OFMC
% Version of 2006/02/13
SUMMARY
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
C:\progra~1\SPAN\testsuite\results\enhanced_protocol_.if
GOAL
as_specified
BACKEND
OFMC
COMMENTS
STATISTICS
parseTime: 0.00s
searchTime: 0.14s
visitedNodes: 40 nodes
depth: 5 plies
```

Fig. 3. Evaluation result of the cloud-RAPIC protocol

witness(R, S, s_auth_r, Auth') which declares that agent reader witness for the value $Auth'$, while *request(S, R, s_auth_r, Auth')* is used for strong authentication property of R by S which declares that agent server requests a check of the value $Auth'$. This goal is declared by the constant *s_auth_r* in the goal section.

The evaluation results of the AVISPA tool shows that the protocol is safe and satisfies all of the security and authentication properties defined in the goal section as presented in Fig. 3.

C. Security and Privacy Comparison

In this section we compare the proposed cloud-RAPIC protocol with the cloud-RAP2013 protocol [1] and the cloud-RAP2015 protocol [7].

In the cloud-RAP2013 protocol [1], when the tag enters the reader zone, it generates $M1 = h(Rid\|Tid_i\|K_i)$ as an authentication request to the reader. The reader then passes the index message $M1$ to the cloud server to retrieve the tag's data $E(Rid\|Tid_i\|K_i)$ and verifies Rid by checking the integrity of Rid, obtains Tid_i and K_i; then it generates a random number $R1$ as a challenge to the tag.

The tag responses by calculating $M2 = h(Rid\|Tid_i\|R1)$ and generating a random number $R2$ as its challenge to the reader. The reader checks the tag's response, and if it compares successfully, the reader sends queries to the cloud server $h(Rid\|Tid_i\|(K_i + 1))$ and verifies the answers until it finds the last valid record $E(Rid, Tid_i, M)$ where M is the last number of a session between a reader and a tag. If $K_i = M$, this means that the tag is synchronized with the cloud server. Otherwise, the tag has been desynchronised.

Then, the reader updates the data and sends to the cloud server $h(Rid\|Tid_i\|M')$ and $E(Rid\|Tid_i\|M')$, where $M' = M + 1$. The cloud server writes the new record into the database, calculates $h(Rid\|Tid_i\|M') \oplus h(E(Rid\|Tid_i\|M'))$, and sends it to the reader. After the cloud- server confirms the updating process, then the reader calculates $h(Rid\|Tid_i\|R2) \oplus M'$ and $h(Rid\|Tid_i\|M')$ and sends them to the tag to update its data. The tag computes $h(Rid\|Tid_i\|R2)$ XORed with $h(Rid\|Tid_i\|R2) \oplus M'$ in order to obtain M', then verifies the correctness of $h(Rid\|Tid_i\|M')$ and updates $K_i = M'$.

However, the authors in the the cloud-RAP2015 [7] protocol pointed out that the cloud-RAP2013 [1] protocol violates data privacy and is vulnerable to location tracking, and reader impersonation attacks in which the adversary can be authenticated as a legitimate reader without compromising the tag's secret data. This vulnerability related to the way of calculating the new secret key of the tag $M' = M + 1$.

The cloud-RAP2015 [7] protocol assumes a secure communication channel between the reader and the cloud server. The data in the cloud server is stored in a form as encrypted hash table. That is, the reader uses a hash function to encrypt tag's Tid_i as an index field $h(Tid_i)$ and encrypts the tag's data $E_{mk}(Tid_i \| K_i)$ in the corresponding data field using its master key. However, this assumption is not suitable for RFID systems with portable readers and the tag is vulnerable to the location tracking attack until it updates its identifier. We evaluate the cloud-RAP2015 [7] protocol using the UPriv model [9] and the AVISPA verification tool [10] and find that without the secure channel assumption the adversary can perform a man-in-the middle attack in which the adversary can de-synchronize the tag and the cloud database in one protocol session.

Security comparisons between all of the three cloud-based authentication protocols are shown in Table III. The properties on cloud-RAP2013 protocol [1] come from the analysis made in [7] and the properties of cloud-RAP2015 protocol are conducted by us. The notation × denotes that a protocol does not satisfy the given requirement while the notation √ means that the requirement is satisfied in a scheme. The notation Δ means that the mutual authentication between the reader and the cloud server is partially satisfied where the reader authenticates the cloud server based on the correctness of the received tag's data but there is no mechanism to authenticate the reader by cloud server.

TABLE III. PRIVACY AND SECURITY COMPARISONS

Properties	**Cloud-RAP2013**	**Cloud-RAP2015**	**Cloud-RAPIC**
Tag data anonymity	×	×	√
Resistance to location tracking	×	×	√
Resistance to forward Untraceability	√	×	√
Resistance to replay attack (Reader-Tag)	√	√	√
Resistance to de-synchronization (Reader-Tag)	√	√	√
Resistance to tag impersonation	√	√	√
Resistance to reader impersonation	×	√	√
Mutual authentication (Reader-Tag)	×	√	√
Mutual authentication (Reader-Server)	Δ	Δ	√
Data integrity (Reader-Server)	×	×	√

× : not satisfied; √ : satisfied; Δ : partially satisfied.

V. CONCLUSION

In this paper, we proposed cloud-RAPIC, a cloud-based RFID authentication protocol with insure communication channels between the reader and the cloud server. The cloud-RAPIC protocol protects data transmitted between all protocol parties without any help from a third party. Moreover, the cloud-RAPIC protocol resists the location tracking attack between any two authentication sessions even if the tag does not update its identifier. We proved that cloud-RAPIC works well for RFID systems with mobile readers and satisfies a high level of security and privacy requirements. In addition, it does not require more computational capabilities at the reader and the tag sides compared with the original cloud-RAP2013 [1] and the improved cloud-RAP2015 [7] RFID authentication protocols.

REFERENCES

[1] W. Xie, L. Xie, C. Zhang, Q. Zhang, and C. Tang, "Cloud-based RFID authentication," 2013 IEEE International Conference on RFID, April 2013, pp. 168-175.

[2] D. Zissis, and D. Lekkas, "Addressing cloud computing security issues," Future Generation computer systems, vol. 28, pp. 583-592, March 2012.

[3] M. S. Kiraz, M. A. Bingöl, S. Kardaş, and F. Birinc, "Anonymous RFID authentication for cloud services," International Journal of Information Security Science. vol. 1, pp. 32-42, June 2012.

[4] S. Kardas, S. Celik, M. Bingol, and A. Levi, "A new security and privacy framework for RFID in cloud computing," 5th IEEE International Conference on Cloud Computing Technology and Science. Bristol, December 2013, pp. 171-176.

[5] S. M. Chen, M. E. Wu, H. M. Sun, and K. H. Wang, "CRFID: An RFID system with a cloud database as a back-end server," Future Generation Computer Systems. 30, January 2014, pp. 155-161.

[6] I. C. Lin, H. H. Hsu, and C. Y. Cheng, "A Cloud-Based Authentication Protocol for RFID Supply Chain Systems," Journal of Network and Systems Management. vol. 23, pp. 978-997, September 2014.

[7] S. Abughazalah, K. Markantonakis, and K. Mayes, "Secure Improved Cloud-Based RFID Authentication Protocol," Springer International Publishing. Switzerland, vol. 8872, pp. 147-164, March 2015 [7th International Workshop, SETOP].

[8] B. Song, and C. J. Mitchell, "RFID authentication protocol for low-cost tags," Proceedings of the first ACM conference on Wireless network security. Alexandria, April 2008, pp. 140-147.

[9] K. Ouafi, and R. Phan, "Privacy of recent RFID authentication protocols," In Proceedings of the 4th International Conference of Information Security Practice and Experience. Sydney, vol. 4991, pp. 263-277, April 2008.

[10] G. Lowe. Casper: A compiler for the analysis of security protocols. In Proceedings of the 10th IEEE Computer Security Foundations Workshop, Rockport, MA, June 1997. pp. 18-30.

[11] R. Küsters and T. Truderung, "Reducing protocol analysis with XOR to the XOR-free case in the horn theory based approach," In Proceedings of the 15th ACM conference on Computer and communications security CCS '08, New York, October 2008. pp.129-138.

[12] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuéllar, and P. Drielsma, "The AVISPA tool for the automated validation of internet security protocols and applications," In Proceedings of the 17th International Conference on Computer Aided Verification, Springer Berlin Heidelberg. vol. 3576, July 2005, pp. 281–285.