Session-based security enhancement of RFID systems for emerging open-loop applications

Citation: Wang, Junyu, Christian Floerkemeier, and Sanjay E. Sarma. "Session-Based Security Enhancement of RFID Systems for Emerging Open-Loop Applications." Personal and Ubiquitous Computing 18.8 (2014): 1881–1891.

As Published: http://dx.doi.org/10.1007/s00779-014-0788-x

Publisher: Springer London

Persistent URL: http://hdl.handle.net/1721.1/103797

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Session-Based Security Enhancement of RFID Systems for Emerging Open-Loop Applications

Junyu Wang · Christian Floerkemeier · Sanjay E. Sarma

Abstract Radio Frequency Identification (RFID) is an important technique used for automatic identification and data capture. In recent years, low cost RFID tags have been used in many open-loop applications beyond supply chain management, such as the tagging of the medicine, clothes and belongings after the Point of Sales (POS). At the same time, with the development of semiconductor industry, handheld terminals and even mobile phones are becoming RFID-enabled. Unauthorized mobile RFID readers could be abused by the malicious hackers or curious common people. Even for authorized RFID readers, the ownership of the reader can be transferred and the owners of the authorized mobile reader may not be always reliable. The authorization and authentication of the mobile RFID readers need to take stronger security measures to address the privacy or security issues that may arise in the emerging open-loop applications. In this paper, the security demands of RFID tags in emerging open-loop applications are summarized and two example protocols for authorization, authentication and key establishment based on symmetric cryptography are presented. The proposed protocols adopt a timed-session-based authorization scheme, and all reader-to-tag operations are authorized by a Trusted Third Party (TTP) using a newly defined class of timed sessions. The output of the tags is randomized to prevent unauthorized tracking of the RFID tags. An instance of the protocol A is implemented in 0.13μm CMOS technology and the functions are verified by Field Programmable Gate Array (FPGA). The baseband consumes 44.0μW under 1.08V voltage and 1.92MHz frequency and it has 25,067 gate equivalent (GE). The proposed protocols can successfully resist most security threats towards open-loop RFID systems except physical attacks. The timing and scalability of the two protocols are also discussed in detail.

Keywords: Internet of Things (IoT)·Radio Frequency Identification (RFID)·Authentication·Authorization • Timed Session

J. Wang (✉)
State Key Lab of ASIC & System, Fudan University,
Shanghai 200433 China
e-mail: junyuwang@fudan.edu.cn

C. Floerkemeier
Scandit, Zürich Area, Switzerland
e-mail: floerkem@inf.ethz.ch

S. E. Sarma
Massachusetts Institute of Technology, 77 Massachusetts Ave.,
Cambridge, MA 02139 USA
e-mail: sesarma@mit.edu

1 Introduction

Traditionally, passive Radio Frequency Identification (RFID) are used for closed-loop applications where RFID tags are only used in dedicated domains and the readers are off-line or in a closed internal network. Each tag stores a unique key which is a function of (but not limited to) a master key and the serial number of the tag, also called Unique ID (UID). An RFID reader can recover the key of each tag when it enters the range of the RFID reader, by acquiring the UID of the tag and computing the key in an embedded secure engine, such as a Secure Access Module (SAM). Then, a session key for the transaction afterwards will be generated using a certain key agreement protocol. In recent years, different researchers have discussed the security weakness of the Mifare Classic product [1, 2]. Though Mifare Plus [3] solved issues of the short length of keys and hardware weaknesses, the basic scenario is the same, e.g. the readers are trusted since they are stationary and are often monitored, by some staff or video camera.

The applications of RFID tags in open-loop systems began with the development of the Electronic Product Code (EPC) in 1999 [4-6]. Since then, low-cost RFID tags have been used for the automatic identification of consumer packaged goods (CPG) and logistical units in the supply chain. In an open-loop system, RFID tags communicate with numerous RFID readers which are under the control of various parties, and RFID readers can be connected to the Internet for different applications. The most widely used air interface protocol for open-loop systems is the EPC Gen2 protocol [7], which became ISO/IEC18000-63 in 2013 [8]. The security measures of this protocol focus on meeting the requirements of the target applications in the supply chain such as preventing remote eavesdropping, protecting user memory from unauthorized writing, and disabling the tags after using a *Kill* command.

In recent years, new applications of open-loop RFID systems beyond the supply chain appeared, such as smart drug, smart apparel, personalized healthcare, implantable RFID device etc. With the development of the Internet of Things (IoT) [9-12], many measures are taken to facilitate the communication between a reader and a tag, such as moving target tag localization and tracking in an IoT environment. The emerging open-loop RFID applications require additional security measures to protect, for example, personal information stored on the RFID tags. Security measures used in a closed-loop system are therefore insufficient for such applications for two reasons: first, the tags may move between different organizations, locations and even countries, which means sharing a master key with all the readers which are part of the applications will thus be difficult; secondly, handheld readers with secure engines can possibly be manipulated by attackers or even potentially abused by curious legitimate users. There is thus a motivation for developing new security protocols for emerging open-loop applications with higher security and privacy requirements.

The rest of our paper is organized as follows. Section 2 overviews the security measures of an EPC Gen2 and EPC Gen2v2 protocols; it then summarizes the security demands of RFID systems for emerging open-loop applications; Section 3 introduces the related works; Section 4 presents two session-based RFID authentication protocols that can meet the targeted security demands of open-loop RFID applications; Section 5 presents the implementation of an instance of Protocol A and provides the testing results; Section 6 analyzes the security functionality, timing and scalability of the proposed session-based RFID protocols; finally, Section 7 summarizes our contributions and outlines the future work.

2 Analysis of security demands in emerging open-loop RFID applications

In this section, we investigate the existing security measures in the EPC Gen2 and EPC Gen2v2 protocols before analyzing the security demands of novel emerging applications.

A. Overview of security measures of EPC Gen2 and EPC Gen2v2 protocols

The first EPC Gen2 protocol has been developed for 10 years. It was introduced mainly for retail supply chain management. Since there is no sensitive information stored on the RFID tags and the tags are not envisioned to be used to identify humans, the security measures in the EPC Gen2 protocol focus on preventing remote eavesdropping and unauthorized access to the tag memory and disabling the tag with a password secured *kill* command. The main security measures of the EPC Gen2 v1.2.0 protocol [7] are listed as follows: (1) After the inventory round, a random number or "handle" generated by the tag is communicated via the "secure" tag-to-reader channel to encrypt the sensitive information transmitted from the reader to the tag, such as passwords and data to be written in the tag; (2) Access passwords are used to prevent unauthorized reading/writing operations to the reserved memory banks and to prevent unauthorized access to other memory banks; (3) A kill password is used to authorize a non-reversible kill operation that disables the tag for privacy reasons.

The security measures of EPC Gen2 protocol are sufficient for its target applications. While in emerging applications where the RFID tags are used for medicine or healthcare for example, the security requirements far exceeds the original security requirements for the Gen2 protocol and the likelihood of attacks against these systems increases. In the existing Gen2 protocol, the EPC code is transmitted through the air as plaintext. Thus, a passive eavesdropper could obtain the ID of an EPC tag by listening to the communication between the reader and tags. Moreover, the Gen2 protocol allows any RFID reader to read the EPC code on the EPC tag. There is thus no mechanism to restrict which readers can identify a particular RFID tag by its EPC code. Since the EPC code is not a random number but a code that represents brand owner and product category, an attacker can thus also detect what kind of products are carried by a particular person. The current Gen2 protocol focuses on preventing remote eavesdropping, but fails in preventing local eavesdropping. The handle in the EPC Gen2 protocol is thus sent in plain text from the tag to the reader, relying on the assumption that the signal backscattered by the tag is too weak to be eavesdropped from far away. The password-based authentication in the Gen2 protocol is also not sufficient to avoid other attacks such as cloning, spoofing, modification, and man-in-the-middle attacks [13-15].

The Gen2 protocol continued to evolve. In 2013, a new version of EPC Gen2 protocol, Gen2v2 [16] was ratified by GS1. In the Gen2v2 protocol, many optional commands are introduced to provide strong security protection, such as *Challenge*, *Authenticate*, *SecureComm*, *AuthComm*, *ReadBuffer*, etc. The commands are intended to be used according to a process specified in a cryptographic suite, which is expected to be finalized by ISO/IEC in 2014. The EPC Gen2v2 protocol is backwards-compatible, with optional security enhancements. In EPC Gen2v2 protocol, with the support of the on-chip Crypto engine, the reader can cryptographically authenticate tags and make them uncloneable. Meanwhile tags can cryptographically authenticate readers (users) and enable secure and privilege-based access of user memories. In addition, Gen2v2 offers enhanced user memory, which is partitioned into files with different access privileges (*read*, *write*, *lock*). EPC Gen2v2 protects consumer privacy through reduction of operating range and/or hidden memory. The Gen2v2 protocol would be incorporated into ISO/IEC 18000-63 soon and make it feasible to have strong security for EPC/RFID system.

B. Security demands of emerging applications to open-loop RFID systems

In emerging open-loop RFID applications, where additional security requirements exist to protect the privacy of the individual or sensitive information on the RFID tag, such as item level tagging of drugs and wearable/implantable devices, RFID tags will be used by a wide range of RFID readers which are under the control of different organizations and parties. The RFID readers can be anything from a static unit in a manufacturing line to a user-owned portable device with RFID capability. There are a number of additional security measures needed to meet the security requirements of these applications.

First, the system should provide support for authentication of multiple parties who are not mutually trusted. In an open-loop application, the ownership of the tag would be transferred from different industry partners to end users with the goods on which they are attached. The tag cannot simply share the same secret with all the readers in different industries; otherwise, the secret of the history of the tag would be revealed unexpectedly. To meet this demand, the communication between a tag and a reader can be authenticated with the support of a trusted third party (TTP), which facilitates the establishment of the mutual authentication between the reader and the tag before transmitting sensitive information.

Secondly, the authentication between a tag and a reader should adopt some mechanism to provide freshness. As the ownership of a tag for emerging open-loop applications may transfer among different parties and even for one party the access control policies for the tag may change at different moments; the freshness of authentication must be provided to ensure that a tag would not reveal its identity to any unauthorized reader or any manipulated authorized reader. The concept of a timed session for open-loop RFID systems is introduced in this paper.

Thirdly, the system should protect the user's privacy and defend it against spoofing and replay attacks. If a tag keeps replying the same identifier, tracking the tag will be possible by querying it at different scenarios. Applications based on unique IDs also have security issues. One solution is that the identifier of a tag responding to RFID readers does not stay unchanged. The tag ID can be simulated using an inexpensive emulator in order to spoof someone's identity. A randomized ID will not only protect the privacy of the holders but also defend these applications against spoofing and replay attacks.

3 Related works

There are numerous server-based authentication protocols for RFID systems in the literature [17-23]. Ben et al [17] propose a protocol in which the tag can authenticate the reader and the server, but do not provide a mechanism for the reader to authenticate the tag. Protocols [18, 19] realize the mutual authentication between tag and server, and the reader mainly acts as a data capture channel with a capability of random number generator and compare time stamp. Mei and Yang [20] propose a protocol in the presence of malicious readers, in which server authenticate both the tag and the reader and the server provide a time control for the authentication, but it is not efficient to read the user memory of the tag. Lee et al [21] propose an authentication protocol supported by physically unclonable function (PUF). In the protocol, both the reader and the tag are authenticated by the reader and the reader can perform (*read/write*) operations after being approved as a legitimate generator, without time limit. Chikouche et al [22] list the vulnerabilities of authentication protocols and Syamsuddin et al [23] summarize low-cost authentication protocols. There are many more previous works on server-based authentication. Most of server-based authentication protocols are key transport or key agreement protocols in which the server, or tag/reader or both tag and reader, have the responsibility for key generation. An important consideration in the design of such protocols is who generates the session keys.

Using alias IDs instead of real IDs of RFID tags has been mentioned in a few previous papers. Weis et al. [6] introduce the metaID, a hashed key, in the hash lock protocol. It has low key lookup overhead, but the tag can be traced because it responds predictably. In a randomized hash lock protocol from the same paper, the reply of the tag is randomized by hashing with a random number generated on the tag but at the expense of the efficiency of key lookup in the system. Juels [24] propose a time-elapsed pseudonym scheme, but the on-tag pseudonyms are limited by the tag memory, and it is difficult for a passive tag to decide when to change a pseudonym because of the resource limitation. Foley [25] introduce the PID, a pseudo name of the real ID, in the TagFolio system, but the real ID of a tag was transmitted in plain text through an insecure channel. Zhijian et al. [26] propose a security protocol based on secret sharing. It can resist replay attack and forgery attack as the communication is encrypted by a Hash function and the metaID changes after every successful session, but the freshness of the transaction is not ensured. Avoine et al [27] review different privacy-friendly authentication protocols based on symmetric-key cryptography.

Time-stamp is used in some authentication protocols to provide freshness [18, 19, 28, 29]. In [29], time-stamp is used in the reader and/or the server to guarantee the authentication message and prevent against clone attack or replay attack. But to store a tamp-stamp in the tag will cause a *write* operation and thus consume additional power.

In summary, there are many related works which can meet some security requirements of the emerging open-loop applications. However, to the best of our knowledge, there

is still no solution which can meet all the requirements proposed in this paper.

Our motivation is to realize a system that can ensure privacy and access control when RFID tags are prevalently used for new open-loop applications with higher security demands. The goal is to provide an authenticated communication channel between the reader and tag and ensure that only the reader with the correct authorization would be able to access the tag: mutual authentication would be used between the tag and the server, and the trust between the tag and the reader would be set up by a trusted server in terms of timed sessions.

4 Proposed RFID authentication protocols

In this section, we propose two RFID authentication protocols which can meet the security demands discussed in Section II. The protocols proposed in this section are based on a symmetric cipher algorithm, such as AES, so as to maintain the low cost.[1]

There are three principals in the system: the reader (R), the tag (T), and the server (S). The server here refers to a backend database through which we retrieve information about RFID tags, such as keys of the symmetric cipher algorithm, access control policies and product information.

A. Basic assumptions

Our proposed authentication protocols require following basic assumptions:

- The tag, the reader, and the server are able to perform the symmetric cipher algorithm and generate random numbers using a random number generator.
- The tag and the reader have some pre-established keys shared with the server, which is trusted by both the tag and the reader. We assume that there is some reliable mechanism for their establishment. Establishing these two keys is beyond the scope of this paper.
- The server has a built-in database which stores product information related to the tag or the virtual tags, the mapping between the pseudo name and the key index for the tag, PID, and the keys to be used to communicate with the tag and the reader.
- The channel between the reader and the server is assured by current security technologies.

B. Parameters

Here are the parameters and the corresponding definitions in the proposed authentication protocol.

K_{RS}: the pre-established key for communication between the server and the reader.

K_{TS}: the pre-established key for communication between the server and the tag.

K_{RT}: the established session key between the reader and the tag, generated by the server. This key is unique for each session and must be re-created each session where the reader attempts to perform an operation on the tag[2].

PID_n: the pseudo ID of the tag in the current session and the index for the server to search the K_{TS}. It is a random number generated by the server (S). The first PID, PID_0, is pre-established between the tag and the server through a registration procedure.

PID_{n+1}: the pseudo ID of the tag in the next session. It is a random number generated by the server (S).

N_T: a nonce generated by the tag.

N_R: a nonce generated by the reader.

N_S: a nonce generated by the server.

$E_K(M)$: a message M is encrypted by a symmetric cipher algorithm E with a key K.

RID: the ID of the RFID reader.

OP_R: The operations of the reader, such as *read*, *write*, *kill* or *access*.

Vtag: Virtual tag. *Vtag* is a digital tag in the backend server which contains all the digital information of a physical tag and can execute all the operations upon receiving the command from an authorized reader.

MAC: message authentication code, providing integrity and authenticity assurances on the message.

C. Proposed authentication protocols

In this section, we propose two authentication protocols for open-loop RFID systems. Protocol A is for the tags with genuine ID and user memory, while Protocol B is for the tags without genuine ID and user memory.

- **Protocol A**

The authentication protocol A (Fig.1) can be broken down into two major processes. During the first process, the server and the tag authenticate each other according to the three pass mutual authentication protocol of ISO/IEC 9798-2 [30]. The reader serves as a mediator and does not perform any actual operations. During the second process, a session key between the reader and the tag is generated by the server and is distributed to the reader and the tag according to the Otway-Rees Protocol modified by Abadi

[1] Note that, as micro-electronics industry develops, it would be feasible to support a strong public key cryptography in normal tags in the future.

[2] Note that the freshness of a session in the proposed protocols is assured by a nonce and an on-tag counter. In the proposed protocols, all reader-to-tag operations are authorized session by session.

and Needham [31]. The PID is also updated during the second process. The detailed steps are described below. The first process includes step 1 to step 8, and the second process includes step 8 to step 11.

Step 1: This first step starts when the reader initiates communication by sending its identifier, *RID*, and operation code OP_R to the tag.

Step 2: The tag then sends its current PID, PID_n, and a nonce N_T to the reader. At the same time, an on-tag counter will be started.

Step 3: The reader dispatches received PID_n and N_T of the tag to the backend server.

Step 4: The backend server takes PID_n and attempts to locate the key K_{TS} for the tag in its database. If the server cannot find a corresponding key, this process terminates and the operation request is denied.

Step 5: If the server finds the key between the server and the tag, it forwards a message encrypted by K_{TS} to the reader. This message consists of a nonce N_S, N_T, and PID_n of the tag. The encrypted message is concatenated with the CBC MAC mode of a selected block cipher (such as DES) to provide the integrity.

Step 6: The reader forwards the message in step 5 from the server to the tag. Then the tag can decrypt this message to verify that the N_T it generated earlier is included. This allows the tag to verify the server's identity. If the server's identity has not been successfully verified, the communication will be terminated by the tag.

Step 7: If the server is successfully verified by the tag, then the tag will prove its own identity to the server by sending the encrypted N_S and N_T. The positions of the two nonces are switched in order to avoid reflection attacks. The PID_n is sent again in this step, since there may be some other tags that communicate with the reader during this session.

Step 8: The reader forwards the message from the tag to the server, and at the same time, the reader sends its *RID* and operation code OP_R to the server.

Step 9: When the server receives the *RID* and the operation code, the server will do several operations: (1) verify if the reader is authorized to perform this operation on the tag according to the access control policies. If the verification is failed, the communication will be terminated; (2) generate a session key, K_{RT}, for this operation if the operation of the reader is authorized. Note that the session key is a random number; and (3) generate a new PID, PID_{n+1}, for the tag. The server will preserve the PID_n and the PID_{n+1} so that the communication can continue next time if the PID of a tag is, for any reason, not updated in time.

Step 10: The server sends back two encrypted messages, one designated for the reader and the other intended for the tag. K_{RT} is included in both messages and the PID_{n+1} is included in the message to the tag. Each message is encrypted using the appropriate key which is only known by the respective recipient.

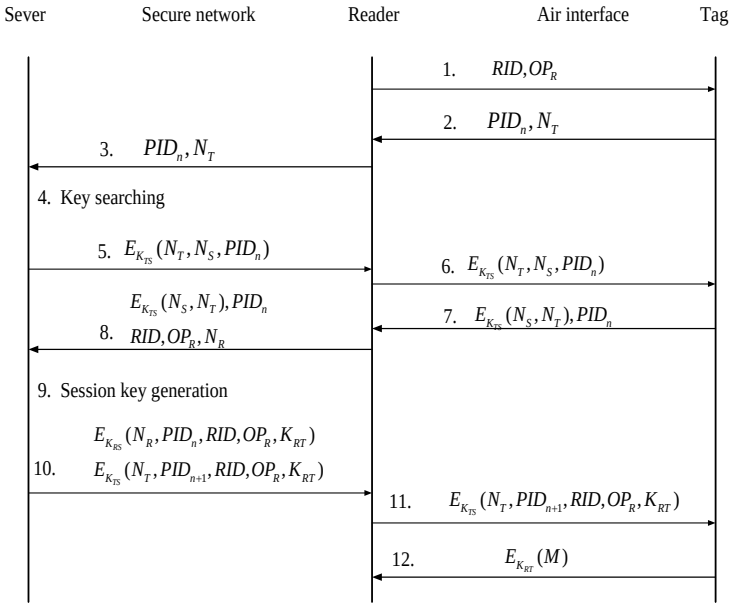

Fig.1. Authentication protocol proposal for tag with real ID and user memory (*Read* operation)

Step 11: When the reader receives the message from the server, it decrypts the message to retrieve the session key and forwards the tag a message encrypted by the key shared by the tag and the server. If the OP_R is a *Write* operation, the reader should encrypt the data to be written into the tag with the session key, K_{RT}, and send it to the tag at the same time.

Step 12: Upon receiving the messages, the tag performs a number of operations: (1) the tag decrypts the message from the server and retrieves the session key, PID_{n+1}, the *RID*, and OP_R; (2) the tag compares the received *RID* and OP_R with those received in step 1. If they are the same, it means that the operation from this reader is authorized by the trusted server. If they are not the same, the tag will not respond to the request of the reader; (3) the tag checks the on-tag counter and compares this with a set time limit. If the time is less than the time limit, the tag will move on to the next step. If the time is more than the time limit, the tag will not respond to the request of the reader; (4) the tag decodes the OP_R. If the OP_R is a *kill* operation, the tag will perform a physical and non-retrievable *kill* operation. If the OP_R is a command other than a *kill* command, the tag will execute the operation and update its PID to the new one specified in the message from the server. If the operation is *read*, the tag will send to the reader the message encrypted by the established session key, K_{RT}. An optional CBC MAC of the message can be generated to ensure data integrity when necessary.

The timed session is controlled by the nonce generated by the tag and an on-tag counter. There is a set time limit for the timed session. The operation must be completed within this time limit. The value of the time limit can be decided by the feasible minimum time and the requirement of applications. With the time limit and a nonce to mark the session, the tag can determine when this session expires. The time is controlled by the tag instead of by the server because the tag could lose power more easily than the server, and it is difficult for the server to keep synchronized with different passive tags.

- **Protocol B**

In the authentication protocol B, as shown in Fig.2, a tag does not have any user memory or even a real ID on it. All information related to the tag, such as the real ID of the tag, its chip state, user memory, and product information, are stored in the backend server. The digital tag in the backend server is named *Vtag* in this protocol. All the operations to a physical tag are mapped into the relevant operations to a *Vtag* except for the *kill* operation. For a *kill* operation, both the physical tag and the *Vtag* should be killed. The following includes a detailed description for Protocol B.

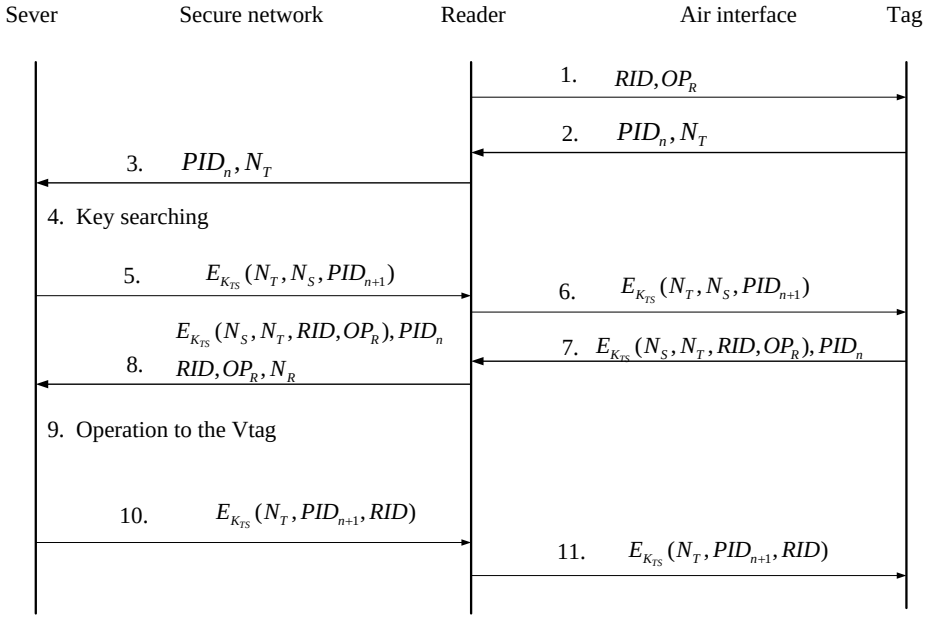

Fig. 2. Authentication protocol proposal for RFID tag without real ID and user memory (*Kill* operation)

Step 1: This first step starts when the reader initiates communication by sending its identifier, *RID*, and operation code OP_R to the tag.

Step 2: The tag then sends its current PID, PID_n, and a nonce N_T to the reader. At the same time, an on-tag counter will be started.

Step 3: The reader dispatches the received PID_n and N_T of the tag to the backend serve.

Step 4: The backend server takes the PID_n and attempts to locate the key K_{TS} for the tag in its database. If the server cannot find a corresponding key, this process terminates and the operation request is denied.

Step 5: If the server finds the key between the server and the tag, it generates a new PID for the tag, PID_{n+1}, and forwards a message encrypted by K_{TS} to the reader. This message consists of a nonce N_S, N_T, and PID_{n+1} of the tag.

Step 6: The reader forwards the message in step 5 from the server to the tag. Then the tag decrypts the message to verify that the N_T it generated earlier is included. This allows the tag to verify the server's identity. If the server's identity has not been successfully verified, the communication will be terminated by the tag.

Step 7: The tag proves its own identity to the server by sending to the server the encrypted N_S and N_T, as well as *RID*, and OP_R received in the first step. PID_n is sent along with the message because there may be other tags that communicate with the server during this session. If the operation is not a *kill* operation, the tag will update its PID from PID_n to PID_{n+1} at the end of step 7.

Step 8: The reader forwards the message from the tag to the server, and, at the same time, the reader sends its *RID* and operation code, OP_R, to the server.

Step 9: When the server receives the message from the reader, the server verifies if the reader is authorized to perform this operation on the tag according to the access control strategy. Suppose in our case that the reader is authorized; the operations other than *kill* and *read* would then be operated to the corresponding *Vtag* in the backend server.

Step 10: If the operation is *read*, the backend server will send the message to the reader in a secure way via the network. If the operation is a *kill* operation, the server sends back a message encrypted by K_{TS} to the reader. This message consists of the nonce N_T, *RID* of the reader, and PID_{n+1} of the tag. The reader will then forward this message to the tag. The server will perform a non-retrievable *kill* operation to the virtual tag in the database at the end of step 10.

Step 11: When the reader receives the message from the server, it forwards the message to the tag. Upon receiving the message forwarded by the reader, the tag performs a number of operations: (1) the tag decrypts the message from the server and retrieves N_T, PID_{n+1}, and *RID*; (2) the tag compares the received *RID* with the one received in the step 1. If they are the same, it means that the *Kill* operation from this reader is authorized by the trusted server. If they are not the same, the tag will not respond to the request of the reader; (3) the tag checks the on-tag counter, and compares with a set time limit. If the time is less than the time limit, the tag will perform a physical and non-retrievable *kill* operation. If the time is more than the time limit, the tag will not respond to the request of the reader.

The timed session in this protocol is controlled by the nonce generated by the tag and the on-tag counter so that the tag can determine when this session expires. The operation must be completed within this time limit. A time limit can be added for the operations to the virtual tags in the server too. The time limit for the operations to the server can be decided according to the application requirements and can be different from that for the operations to the tag.

5 Implementation and analysis

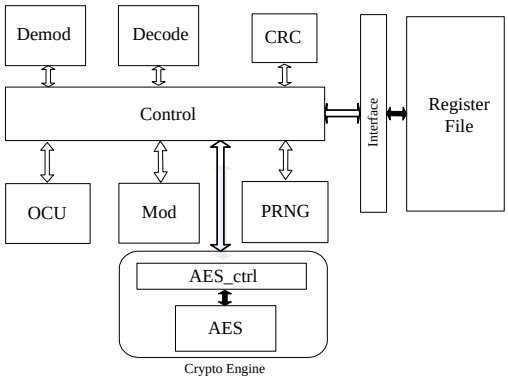

Fig.3. Architecture of the UHF tag baseband

An Ultra High Frequency (UHF) tag baseband (Fig.3) conforming to the proposed Protocol A is implemented using 0.13μm CMOS technology. In the implemented instance protocol, we use 128-bit K_{TS}/K_{RS}, 64-bit $RID/PID_n/PID_{n+1}$, 32-bit $N_T/N_S/N_R$, 32-bit OP_R/K_{RT}, and 128-bit *M*. A 128-bit Advanced Encryption Standard (AES) crypto engine is used for encryption and decryption. A 20-bit counter in the Control Module is used for freshness control. In Fig.3, Demod stands for demodulation module; PRNG stands for pseudo random number generator; OCU stands for output control unit. CRC module provides 16-bit cyclic redundancy check (CRC) to ensure the message integrity. The symmetric key shared by server and tag (KTS) is stored in the register file (RF). Since the frequency is 1.92 MHz, the authentication needs to be finished within 0.5 second.

TABLE I. AUTHEN COMMAND

	Cmd	Step	Reserved	CSI	Message	RN	CRC
bits	8	2	2	8	Variable	16	16
Content	1101 1010	Step	00	--	Authen Messages	Handle	CRC

A customized command "AUTHEN" (Table I) is introduced based on the EPC Gen2v2 protocol. All the authentication messages related to the proposed protocol A are encapsulated into this command. In TABLE I, Step stands for the authentication step according to the proposed security protocol. RN stands for handle specified in the EPC Gen2v2 protocol. We did not use the field of CSI (crypto suite indicator) in the EPC Gen2v2 protocol.

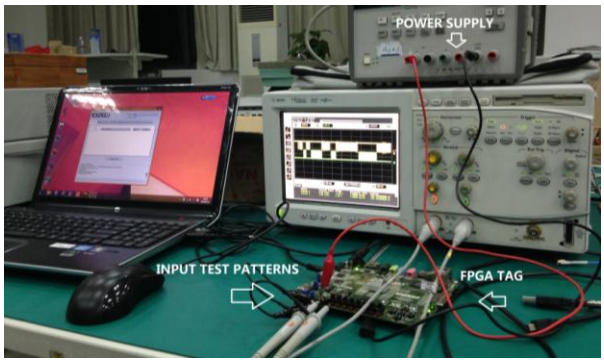

Fig.4a Test setup

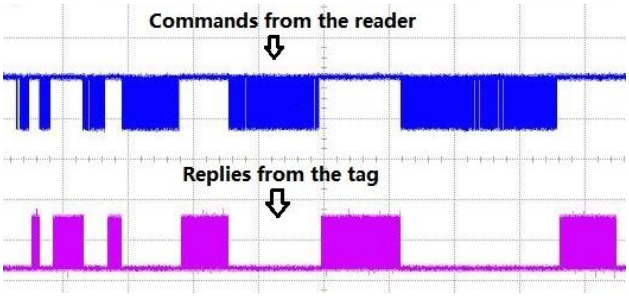

Fig.4b Test patterns

Fig.4. FPGA veryfication of Protocl A

The functionality of baseband is verified using FPGA. The testing environment (Fig.4a) includes an FPGA, a PC, An oscilloscope, and a DC Power supply. The testbench from the reader is loaded to the tag baseband from the FPGA. When the reader sends QUERY, ACK, REQRN (these commands are defined in EPC Gen2v2 protocol) and AUTHEN commands in order, the tag responds correctly according to the authentication protocol (Fig.4b).

According to the simulation results, the whole baseband consumes 44.0μW under 1.08V voltage and 1.92MHz clock frequency. The tag baseband has about 25,067 gate equivalent (GE). The layout area is about 400 × 400 μm^2.

The read range of the secure tag can be deduced from the Friis Formula [32]. Assuming that the transmit power of the reader is 1W; the power dissipation of the tag frontend is 5 μW; and the tag antenna gain is 15%~20%, the read range is about 0.5m~1m when none-ideal factors are taken into account. It is acceptable for UHF RFID with high security requirement.

The cost of tags in Protocol B is lower than those in Protocol A because there are no on-chip user memories in Protocol B and the control unit of tags in Protocol B is simpler. However, the backend server in Protocol B is more complicated than that in Protocol A, since the *Vtag*s needs to be maintained by the server.

6 Analysis

6.1 Security analysis

The proposed protocols have many useful features for resolving the privacy and security issues that we have discussed earlier.

1) Privacy: the privacy issue can be resolved by both of the two proposed protocols, since the tag's PID changes once the information on the tag is being accessed. This prevents the tag from revealing its real ID and thus avoids tracking. Also, the changeable PID is not associated with the real ID of the tag in any way. Therefore, reading from the tag does not directly reveal a unique ID of the tag which identifies the product with the tag.

2) Replay attack: For our protocol, the tag is identified by the PID, and the PID changes every session after the tag's information is accessed by a reader. This prevents replay attacks with the PID, since each PID becomes obsolete at the end of every reader operation. The nonces N_S and N_T used in the proposed protocols can provide the freshness of the message transmitted between the reader and tag and can prevent replay attacks with these messages.

3) Eavesdropping: For the proposed protocol A, when the tag actually communicates its on-chip information to a reader, the information is sent through a message encrypted by a session key. This ensures that an eavesdropper cannot easily obtain the information in plain text. Thus, as long as the encryption algorithm and the established keys are secure, eavesdropping is no longer a threat for our protocol. For the proposed protocol B, the tag does not have any user information on the tag chip, and the tag only works as a token to trigger the operation in the backend server; eavesdropping is then no longer a threat either.

4) Impersonation: Impersonation is not possible as long as the PID is changed every time and the key between the server and tag is secure, since the cloned tag with the obsolete PID cannot be recognized by the server in the next session.

5) Cloning: An adversary cannot clone the tag since the key and other secret information is used with the random number when it is transmitted in the insecure channel.

6) Man-in-the-middle attack: Theoretically, we cannot prevent an adversary from intercepting the communication between the reader and tag and trying to obtain useful information. However, in this case, we do not believe that a man-in-the-middle attack is a serious security threat for our system since all the sensitive information being communicated is already protected by the authentication process.

7) Ownership transfer: The proposed protocols can solve the security issue when the ownership of the tags transfers between different industry partners and end users because that the PID will change at the end of each session, and the reader does not share any secret with the tag. When a tag transfers from one owner to another, there is no need to worry about the credibility of the previous owner or the

new one, since both the forward security and the backward security have been realized in the proposed protocols.

8) Unauthorized massive tag killing: In both of the two proposed protocols, the *kill* operation is authorized by the server session by session, and it is not possible for the reader to kill a large number of tags without authorization.

9) De-synchronization attack: The backend server stores both the current and previous pair of PIDs so that the protocol can still work even when the tag fails to update its PID by a de-synchronization attack between the tag and the reader. Admittedly, if the adversary can keep the tag from updating its PID, the tag will reply with the previous PID. But we don't think this attack is practical, since the adversary has to be close enough in a certain period of time and take risks being noticed by the users.

6.2 Scalability analysis

RFID systems for emerging open-loop applications should have high scalability to accommodate massive operations of RFID tags in the future.

For the proposed protocol A, the session key is generated by the server and transported to the reader and tag each time. So the scalability is mainly about key searching. Since all the PID and K_{TS} K_{RS} pairs are stored in the backend server, it is not applicable for a single server/service to perform authentication for all possible PIDs, so the backend server has to be a distributed system. Since the PID is not a structured number, the system based on Distributed Hash Table (DHT) may have higher efficiency than the system based on Domain Name Service (DNS) when searching the keys. In step 4, the server computes the Hash value according to PID_n of the tag. This value indicates a list in the Hash table. Then the server iterates through the list to find a node which has the same value as PID_n and thus acquire the private key. In step 9, the server can insert a new node to save the value of PID_{n+1} and the private key in the list and may delete a node in step 12. For the proposed protocol B, not only the PID and K_{TS} pairs, but also the *Vtag*s, are stored in the backend server. Besides the operations specified in Protocol A, managing the *Vtag* in a safe and efficient way is another area for open research.

We did not provide any specific structure for backend servers since this is not the focus of this paper; however, according to the progress of distributed systems in computer science, it is feasible to combine the proposed protocols with the distributed network system to provide high scalability for future applications. There are some related works on overlay network systems for RFID applications [33, 34].

6.3 Timing analysis

The complete protocol requires two requests between the reader and the backend server and three requests and responses between the reader and the tag. The response time of the whole system to an operation of a reader is mainly decided by the latency of network, since the time for the communication between the reader and the tag is negligible. We consider a round time trip of 200ms which is typical of communicating to a server, though these numbers vary due to traffic or load on the server itself [35]. The backend system needs some time to perform key searching in the distributed system. For instance, in a DHT system, it takes at most *O(log n)* steps to find a node corresponding to a random input key, where *n* is the number of nodes in the system. Once the server receives a query, it must run AES decryption, which has been tested to be fairly fast [36]. The database query itself is also quite fast due to the many optimization techniques [37]. Therefore, the total latency should be less than 0.5 second. As the target applications of the proposed protocols are in the B2C domain and multiple object identification is not required, the latency is acceptable.

7 Conclusions and future work

We analyze the security demands of emerging open-loop applications to RFID tags and find that the reader should not be naturally trusted but be authorized in a timed session in such applications, especially when portable readers or RFID-enabled mobile phones are to be commonly used in the near future.

Two sample protocols based on timed sessions are proposed in this paper. For the proposed protocol for RFID systems with real ID and user memory, any operation of the reader to a tag is executed with a session key generated by the server. For the proposed protocol for RFID systems without real ID and user memory, the tag has only a security engine, acting as a token to trigger the operation of the server to the corresponding *Vtag* in the networked database, except for the *kill* operation. The two proposed protocols can resist most of the reported security threats to RFID systems for open-loop applications and have good scalabilities. An instance of protocol A was implemented and verified by FPGA.

The proposed protocols can realize authorization and authentication, ensure freshness and protect privacy with acceptable cost, power and timing. Optimization of the key management will be our future work.

Acknowledgments This work is supported by National Natural Science Foundation of China (61211140046, 61076022) and the National High Technology Research and Development Program ("863"Program) of China (2011AA100701), and the Shanghai Pujiang Program. Thanks to Mi Shao, Ye Yao, Linghao Zhu and Linyin Wu for their help with this paper.

References

1. Nohl S. K., Evans D., and Pl¨otz H (2008) Reverse-Engineering a Cryptographic RFID Tag." USENIX Security Symposium
2. Garcia F. D., Gans G.K., Muijrers R., Rossum P.V., Verdult R., Schreur R. W. and Jacobs B. (2008) Dismantling MIFARE Classic. Proceedings of the 13th European Symposium on Research in Computer Security, pp. 97-114
3. http://www.cn.nxp.com/products/identification_and_security/smart_card_ics/mifare_smart_card_ics/mifare_plus/
4. Sarma S.E., Brock D., Ashton K (1999) The Networked Physical World, Proposals for Engineering the Next Generation of Computing, Commerce & Automatic-Identification, Whitepaper,http://autoid.mit.edu/whitepapers/MIT-AUTOID-WH-001.PDF
5. Sarma S.E., Brock D., Engels D.(2001) Radio frequency identification and the electronic product code, IEEE Micro, Volume: 21, Issue: 6, pp.50-54
6. Weis S., Sarma S.E., Rivest R. L., Engels D.(2004) Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems, Security in Pervasive computing, Vol. 2802, pp. 201-212
7. Class 1 Generation 2 UHF Air Interface Protocol Standard Version 1.2.0 (2008), http://www.gs1.org/gsmp/kc/epcglobal/uhfc1g2/uhfc1g2_1_2_0-standard-20080511.pdf
8. ISO/IEC 18000-63:2013 Information technology -- Radio frequency identification for item management -- Part 63: Parameters for air interface communications at 860 MHz to 960 MHz Type C http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=59643
9. Ashton K (2009), That 'internet of things' thing, - RFiD Journal, 2009 - itrco.jp.
10. Atzoria L, Ierab A, Morabito G (2010) The Internet of Things: A survey, Computer Networks, Volume 54, Issue 15, pp 2787–2805
11. Sun Y, Yan H, Lu C, Bie R, Zhou ZB (2013) Constructing the web of events from raw data in the Web of Things, Journal Mobile Information Systems. DOI 10.3233/MIS-130173.
12. Guo J, Zhang H, Sun Y, Bie R (2013) Square-root unscented Kalman filtering-based localization and tracking in the Internet of Things, Personal and Ubiquitous Computing, DOI:10.1007/s00779-013-0713-8
13. Juels A (2006) RFID Security and Privacy: A Research survey. IEEE Journal on Selected Areas in Communications, vol. 24, issue 2, pp 381-394.
14. Koscher K., Juels A., Brajkovic V., and Kohno T. (2009) EPC RFID Tag Security weaknesses and defenses: passport cards, enhanced drivers licenses, and beyond. In proceedings of the 16th ACM conference on Computer and communications security, pp 33-42.
15. Engels D.W., Kang Y.S., Wang J (2013) On security with the new Gen2 RFID security framework. In proceedings of IEEE international conference on RFID.
16. EPC™ Radio-Frequency Identity Protocols Generation-2 UHF RFID specification for RFID Air Interface Protocol for Communications at 860 MHz – 960 MHz Version 2.0.0 (2013), http://www.gs1.org/sites/default/files/docs/uhfc1g2/uhfc1g2_2_0_0_standard_20131101.pdf
17. Niu, B., Zhu X, Li H (2013) An ultra-lightweight and privacy-preserving authentication protocol for mobile RFID systems IEEE Wireless Communications and Networking Conference (WCNC), pp 1864 – 1869
18. Kaul S.D., Awasthi A.K. (2013) RFID Authentication Protocol to Enhance Patient Medication Safety, Journal of medical systems. Vol. 37, Issue 6, pp 1-6.
19. Wu Z.Y, Lin S.C, Chen T.L., Wang C.A (2013) Secure RFID Authentication Scheme for Medicine Applications. Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), pp 175 – 181
20. Mei S., Yang X.(2012) An efficient authentication protocol for low-cost RFID system in the Presence of malicious readers. In proceedings of the 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), pp 2111 – 2114
21. Lee Y.S., Kim T.Y., Lee H.J. (2012) Mutual Authentication Protocol for Enhanced RFID Security and Anti-counterfeiting. 26th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp 558 – 563
22. Chikouche N., Cherif F., Benmohammed M.(2012) Vulnerabilities of two recently RFID authentication protocols International Conference on Complex Systems (ICCS), pp 1 – 6
23. Syamsuddin, I., Han S., Dillon T.A (2012) survey on low-cost RFID authentication protocols, International Conference on Advanced Computer Science and Information Systems (ICACSIS), pp 77 – 82.
24. Juels A.(2004) Minimalist cryptography for low-cost tags, Security in Communication Networks. Revised selected papers, Volume 3352 of LNCS, pp.149-164.
25. Foley J. T. (2007) Security approaches for Radio Frequency Identification systems, MIT Ph. D thesis
26. Gao Z, Jiang Y, Lin Z (2012) An Effective RFID Security Protocol Based on Secret Sharing. Proceedings of the Second International Conference on Instrumentation & Measurement, Computer, Communication and Control.
27. Avoine, G., Bingol M.A., Carpent X., Yalcin, S.B.O. (2013) Privacy-Friendly Authentication in RFID Systems: On Sublinear Protocols Based on Symmetric-Key Cryptography. IEEE Transactions on Mobile Computing, Volume: 12 , Issue: 10, pp 2037 – 2049.
28. Tsudik, G. (2006) YA-TRAP: yet another trivial RFID authentication protocol. Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops). pp 640– 643.
29. Wu X, Zhang M, Yang X (2013) Time-stamp based mutual authentication protocol for mobile RFID system. 22nd Wireless and Optical Communication Conference. (WOCC). pp 702 – 706.
30. ISO. Information Technology – Security Techniques – Entity Authentication – Part 2: Mechanisms Using Symmetric Encipherment Algorithms ISO/IEC 9798-2, 2nd edition, International Standard (1999)
31. Abadi M., Needham R. (1994), "Prudent engineering practice for cryptographic protocols," In IEEE symposium on research in Security and Privacy, pp. 122 – 136. IEEE computer Society Press
32. Shaw J A. Radiometry and the Friis transmission equation[J]. American Journal of Physics, 2012, 81(1): 33-37.
33. Fabian, Gunther O.(2007) Distributed ONS and its Impact on Privacy, IEEE International Conference on Communications, pp. 1223 – 1228
34. Doi Y., Wakayama S., Ozaki S. A (2008) Design for Distributed Backup and Migration of Distributed Hash Tables, International Symposium on Applications and the Interne (SAINT 2008), pp 213 – 216.
35. Aikat J., Kaur J., Smith F.D., Jeffay K.(2003) Variability in tcp round-trip times
36. Encryption performance, http://www.cpktec.com/performance.html.
37. Mysqlqueryperformance,http://dev.mysql.com/tech-resources/articles/mysql-54.html