RFID Authentication Protocol for Low-cost Tags

Boyeon Song
Information Security Group
Royal Holloway, University of London
Egham, Surrey, TW20 0EX, UK
b.song@rhul.ac.uk

Chris J Mitchell
Information Security Group
Royal Holloway, University of London
Egham, Surrey, TW20 0EX, UK
c.mitchell@rhul.ac.uk

ABSTRACT

In this paper, we investigate the possible privacy and security threats to RFID systems, and consider whether previously proposed RFID protocols address these threats. We then propose a new authentication protocol which provides the identified privacy and security features and is also efficient. The new protocol resists tag information leakage, tag location tracking, replay attacks, denial of service attacks, backward traceability, forward traceability (under an assumption), and server impersonation (also under an assumption). We also show that it requires less tag-side storage and computation than other similarly structured RFID protocols.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Authentication*

General Terms

Design, Security

Keywords

RFID, authentication, security, privacy

1. INTRODUCTION

Automatic Identification and Data Capture (AIDC) covers methods for automatically identifying objects, collecting data about them, and entering that data directly into computer systems [25]. It has become widespread in a broad range of markets, including Point of Sale (POS), warehouse management and logistics, product tracking in a supply chain, passports, transport payments, livestock identification, automated vehicle identification, library book check-in/check-out and patient identification in hospitals [22, 25].

Radio Frequency Identification (RFID) is a wireless AIDC technology that uses radio signals to identify a product, animal or person [8, 22]. The three main components of an RFID system are RFID tags, RFID readers and a back-end server. A tag is an identification device attached to an item, which uses radio frequency (RF) to communicate [9]. The reader is a device that can recognise the presence of RFID tags and read the information supplied by them [9]. The reader queries tags by broadcasting an RF signal, and the tag responds to the reader with a number or other identifying information [22]. The reader forwards the tag response to a back-end server. The server has a database of tags and can retrieve detailed information regarding the tag (or the item attached to the tag) from the tag response.

The main benefits of RFID systems are that they can provide automated and multiple identification capture and system analysis, can read several tags in the field at the same time automatically, and can help to track valuable objects [2]. However, they can threaten the privacy of the owner carrying the tag as a result of automatic identification [3]; more specifically, tag information could be disclosed to unauthorised readers, and multiple readers could cooperate to track the movements of a tag [3, 24]. In addition, many possible security threats arise from the use of wireless communications. Moreover, it is infeasible to use computationally intensive cryptographic algorithms for privacy and security, because memory and processing power in a low-cost tag are limited [3]. Therefore, authentication protocols for RFID systems should not only be designed to address these privacy and security threats, but should also take into account the limited capabilities of RFID tags.

In this paper, we first consider the privacy issues and security threats applying to RFID systems, and also discuss the accompanying performance issues. Section 3 reviews related work and examines weaknesses in existing schemes. Next, we propose a new authentication protocol designed to address the identified privacy and security requirements, whilst also preserving desirable performance characteristics. Section 5 analyses the protocol and compares it to the prior art. The final section summarises the results.

2. REQUIREMENTS FOR RFID PROTOCOLS

2.1 Privacy

One of the main concerns for RFID systems is privacy. RFID systems use a shared radio medium which allows eavesdropping. Unprotected communications between tags and readers over a wireless channel can disclose information about the tags and their positions. We identify the following two major privacy issues.

- **Tag Information Privacy**: In a typical RFID sys-

WiSec'08, March 31–April 2, 2008, Alexandria, Virginia, USA.

tem, when an RFID reader queries an RFID tag, it responds by sending its identifier to the reader; the reader can then request further details by sending this identifier to a server. If unauthorised readers can also get a tag identifier, then they may be able to determine the additional information related to the tag. For example, if the information associated with a tag attached to a passport, ID-card or medical record could be obtained by any reader, then the damage would be very serious. To protect against such information leakage, RFID systems need to be controlled so that only authorised readers are able to access the information associated with a tag.

- **Tag Location Privacy**: If the responses of a tag are linkable to each other or distinguishable from those of other tags, then the location of a tag could be tracked by multiple collaborating tag readers. For example, if the response of a tag to a reader query is a static ID code, then the movements of the tag can be monitored, and the social interactions of an individual carrying a tag may be available to third parties without him or her knowing. If messages from tags are anonymous, then the tag tracking problem can be avoided.

2.2 Security

We classify security threats to RFID protocols into weak and strong attacks.

2.2.1 Weak attacks

These are attacks which are feasible just by observing and manipulating communications between readers and tags.

- **Tag Impersonation**: An eavesdropper could impersonate a target tag without knowing the tag's internal secrets. It could communicate with readers instead of the tag and be authenticated as the tag.

- **Replay attack**: In such an attack, an attacker reuses communications from previous sessions to perform a successful authentication between a tag and a server.

- **Denial of Service attack**: An adversary disturbs the interactions between readers and tags by intercepting or blocking messages transmitted. Such an attack could cause a server and a tag to lose synchronisation. For example, the server might update the shared data, while the tag does not; in such a case they would no longer be able to authenticate each other.

2.2.2 Strong attacks

These are threats possible for an attacker which has compromised a target tag. The memory of a low-cost tag is not tamper-resistant, and hence the tag's internal data are liable to be exposed by physical attacks. Thus addressing such attacks is essential for the security of RFID schemes.

- **Backward Traceability**: This occurs if, given all the internal state of a target tag at time t, the attacker is able to identify target tag interactions that occurred at a time $t' < t$ [15]. That is, knowledge of a tag's current internal state could help identify the tag's past interactions, and the past transcripts of a tag may allow tracking of the tag owner's past behaviour [15].

  In some previous papers, *backward untraceability* is referred to as *forward security* [5, 7, 12, 18]. Here we use the terms backward traceability and forward traceability defined as in [15] to clearly distinguish between threats to past and future anonymity.

- **Forward Traceability**: This can similarly be defined as where knowledge of a tag's internal state at time t can help to identify tag interactions that occur at a time $t' > t$ [15]. The only way of maintaining future security once the current tag secrets have been revealed is to detect key compromise as soon as possible, and to replace the exposed key to protect future transactions [15]. This issue is related to tag ownership transfer. This is because, if an RFID scheme does not provide forward untraceability, when the ownership of a tag is transferred, the previous owners might be able to read communications between the new owner and the tag.

- **Server Impersonation**: This means that an adversary with knowledge of the internal state of a tag is able to impersonate the valid server to the tag. This attack does not appear to have been discussed previously, despite its potential importance.
  One reason that this is a genuine threat is because of the following attack. If it is possible to impersonate a server to a tag, an adversary could request a target tag to update its shared secrets. The tag and the real server would then be desynchronised, and incapable of successful communications.

2.3 Performance

RFID schemes cannot use computationally intensive cryptographic algorithms for privacy and security because tight tag cost requirements make tag-side resources (such as processing power and storage) scarce.

- **Capacity minimisation**: The volume of data stored in a tag should be minimised because of the limited size of tag memory.

- **Computation minimisation**: Tag-side computations should be minimised because of the very limited power available to a tag.

- **Communication compression**: The volume of data that each tag can transmit per second is limited by the bandwidth available for RFID tags [3, 18].

- **Scalability**: The server should be able to handle growing amounts of work in a large tag population. It should be able to identify multiple tags using the same radio channel [11]. Performing an exhaustive search to identify individual tags could be difficult when the tag population is large [13].

3. RELATED WORK

Hash-based Access Control (HAC), as defined by Weis et al. [24], is a scheme which involves locking a tag using a one-way hash function. A locked tag uses the hash of a random key as its metaID. When locked, a tag responds to all queries with its metaID. However, the scheme allows a tag to be tracked because the same metaID is used repeatedly [5]. Weis et al. also suggest another scheme, Randomized Access Control (RAC), which employs a random number

generator to prevent the above tracking attack. In each session, a tag generates a new response as a hash function of the tag ID and a random number. However, tag impersonation remains possible because an intercepted response can be replayed. Moreover, it does not provide backward untraceability because the tag ID is fixed.

Ohkubo, Suzki, and Kinoshita (OSK) [18] propose an RFID privacy protection scheme providing indistinguishability (i.e. a tag output is indistinguishable from a truly random value and unlinkable to the ID of the tag) and backward untraceability. This scheme uses a low-cost hash chain mechanism to update tag secret information to provide these two security properties. However, it is subject to replay attacks [5], and hence it permits an adversary to impersonate a tag without knowing the tag secrets.

Henrici and Müller (HM) [10] suggest a scheme relying on one-way hash functions to enhance location privacy. A tag sends two hashed values as its response to a query, and updates its stored values, including its ID, after a successful authentication. Despite this, the scheme still allows a degree of tag tracking, because a tag always replies with the same hashed ID before the next successful authentication [5]. Also, a strong attacker could easily compute the identifiers used in previous sessions by combining the server's random number and the current identifier; that is, it does not provide backward untraceability.

Molnar and Wagner (MW) [17] propose a private authentication protocol for library RFID which uses a shared secret and a pseudorandom number function to protect the messages communicated between tag and reader. This scheme cannot provide backward untraceability. Once a tag is compromised, the attacker can trace past communications from this tag [5], because a tag's identifier and secret key are static. They also build a new tree-based protocol to provide scalable private authentication, with reader work $O(\log N)$, $O(\log N)$ rounds of interaction, and $O(\log N)$ tag storage, where N denotes the number of tags, and the N tags are considered as leaves in a balanced binary tree [17]. However, this approach requires that each tag stores the $\lceil \log N \rceil$ secrets corresponding to the path from the root to the tag, and privacy is weakened when an adversary is able to tamper with at least one tag [4, 19]. Also, the more tags an adversary tampers with, the more privacy is exposed [19].

Dimitriou's scheme (D) [6] is an RFID authentication protocol that enforces user privacy and protects against tag cloning. This scheme uses a challenge-response approach, where a tag uses a hash of its identifier as a response to a reader query to maintain scalability at the server, and the back-end server sends a message using the updated identifier to the tag after receiving the tag response, to authenticate the server to the tag. However, between valid sessions, the tag identifier remains the same, thereby making the scheme vulnerable to tracking and tag impersonation through reuse of the hashed tag identifier. Additionally, the scheme is prone to DoS attacks.

The identification scheme of Karthikeyan and Nesterenko (KN) [11] uses simple matrix multiplication, and does not require computationally expensive cryptographic mechanisms. Security is based on the difficulty of recovering the multiplicand or multiplier from the product of two matrices. An intruder could launch a DoS attack and could also attempt to mount a brute-force matrix or key guessing attack as discussed in [11]. In addition, the scheme cannot resist replay and tracking attacks [5].

Duc et al. (DPLK) [7] present a synchronisation-based communication protocol for the EPCGlobal Class 1 Gen-2 RFID tag. It uses a pseudo-random number generator and a cyclic redundancy code. It cannot prevent replay attacks before the next successful authentication. Most seriously, a DoS attack could permanently desynchronise a server and a tag [5]. It also does not provide backward untraceability if the fixed EPC code and the access key PIN are compromised [5].

A mutual authentication protocol for RFID conforming to the EPC Class 1 Generation 2 standards was introduced by Chien and Chen (CC) [5]. A challenge-response protocol is used to prevent replay attacks. The server database maintains copies of both old and new tag keys to resist DoS attacks. Both the authentication key and the access key are updated after a successful session in order to give backward untraceability. However, the scheme still permits backward and forward traceability, because a strong attacker that compromises a tag can identify a tag's past interactions from the previous communications and the fixed EPC of the tag, and can also read the tag's future transactions. Moreover, an adversary can successfully masquerade as an authorised server to a tag if it has the tag secrets.

Lim and Kwon (LK) [15] describe an RFID authentication scheme satisfying both forward and backward untraceability and enabling perfect ownership transfer. They define *update* as deterministic evolution (of stored secrets) and *refresh* as probabilistic evolution, where the refresh process is introduced to help provide forward untraceability. A tag and a server both refresh their secrets using exchanged random numbers if an authentication procedure completes successfully. If an authentication procedure fails, the tag updates its secrets (i.e. using a deterministic process). The protocol provides forward untraceability from the moment that an adversary misses just one successful authentication session after it has compromised the tag secret. It uses two hash key chains: a forward key chain for tag secret evolution, and a backward key chain, used in reverse order, for server validation. These techniques make the scheme partially secure against server impersonation. The database keeps old and new key chains for relevant secrets in order to solve the desynchronisation problem arising from DoS attacks. However, it does have potentially scalability issues, because the server needs to perform significant computations to update tag secrets, and a large database is required to manage two key chains for each tag.

In this paper, we propose a scheme that significantly reduces the necessary storage and computation in a tag by comparison with the DPLK, CC and LK schemes, as well as preventing the attacks mentioned above.

4. A NOVEL AUTHENTICATION PROTOCOL

4.1 Design Principle

The goals of our new protocol are to meet the privacy and security requirements given in section 2, and also to maximise performance efficiency.

Our protocol involves three flows, and uses a challenge-response approach. It uses random numbers to give anonymity

for each tag response, as in the CC and LK protocols. The server database stores both the most recent and the current data for each tag to protect against desynchronisation between the server and tags, like the HM, CC and LK protocols. If the authentication procedure is successful, then the server and tag refresh their shared secrets probabilistically using exchanged random numbers, thereby providing untraceability, as in the LK protocol.

One of the main features of the protocol is that a random number generated by a tag serves as a temporary secret for the tag. Another feature is that a tag only needs to store an identifier, where this identifier is the cryptographic hash of a bit-string assigned to the tag. A server database stores tag bit-strings as well as tag identifiers. The bit-string is used for server validation. The scheme is designed to minimise the use of complex cryptographic functions, and instead wherever possible uses combinations of simple functions such as right and left shifts and bit-wise exclusive-or operations to combine data strings.

4.2 Preliminaries

4.2.1 Definitions

The protocol uses a hash function, a keyed hash function (a message authentication code algorithm) and a pseudorandom number generator.

Definition 1 A *hash function* h is an efficiently computable function which maps an arbitrary length input to a fixed length output; i.e. $h : \{0,1\}^* \rightarrow \{0,1\}^n$ [16, 25]. The basic requirements for a cryptographic hash function are [16, 25]:

- preimage resistance — for any output y, it is computationally infeasible to find an input x such that $h(x) = y$, given no corresponding input is known.
- 2nd-preimage resistance — given x, it is computationally infeasible to find $x' \neq x$ such that $h(x) = h(x')$.
- collision resistance — it is computationally infeasible to find any pair of distinct inputs x and x' such that $h(x) = h(x')$.

Definition 2 A *message authentication code* (MAC) algorithm is a family of functions f_k parameterised by a secret key k, with the following properties [16]:

- ease of computation — given a value k and an input x, the MAC $f_k(x)$ is easy to compute.
- compression — f_k maps an input x of arbitrary finite bit-length to an output $f_k(x)$ of fixed bit-length n.
- computation-resistance — given a sequence of text-MAC pairs $(x_i, f_k(x_i))$ for a fixed key k, it is computationally infeasible to compute a text-MAC pair $(x, f_k(x))$ for any $x \neq x_i$.

Definition 3 A *pseudorandom bit generator* (PRBG) is a deterministic algorithm which, given a truly random binary sequence of length k, outputs a binary sequence of length $l > k$ which "appears" to be random. The input to the PRBG is called the *seed*, while the output of the PRBG is called a *pseudorandom bit sequence* [16].

4.2.2 Assumptions

Our protocol works under the following assumptions.

- Each tag has a unique *initiator*, which creates and maintains security parameters for the tag.
- Each initiator maintains a secure database containing a set of values for each tag that it manages.
- Each tag has a rewritable memory which may be susceptible to compromise.
- Every tag reader has a PRBG.
- The tags can generate random numbers and perform a hash function and a keyed hash function.
- The channel between the server and the reader is secure.
- The reader and the tags communicate over an insecure channel and their communications are subject to eavesdropping or modification.

Standardised cryptographic hash functions such as SHA-1 are too expensive for use in today's low-cost RFID tags [14, 25]. Instead, the following methods could be adopted to serve as hash functions. Weis [25] discusses using nonlinear feedback shift registers (NLFSRs) to build a low-cost hash function, as NLFSRs require very few gates to implement (besides the register) [14]. Yüksel [26] presents implementations of low-cost hash functions for a block size of 64 bits [14]. A further potential alternative is the Whirlpool hash function, which has been standardised by ISO/IEC and evaluated by the New European Schemes for Signatures, Integrity and Encryption (NESSIE) project [21]. Pramstaller et al. [20] present a compact hardware implementation of Whirlpool, which uses an innovative state representation that makes it possible to significantly reduce the required hardware resources.

A one-way function such as a cryptographic hash function, or a block cipher can be used to generate pseudorandom bit sequences [16]. In practice, an iterated keyed hash function which takes a cheap and weak pseudorandom source (for instance circuitry noise) and an internal key as its inputs could be used as a PRBG [19, 23].

4.2.3 Notation

We use the following notation in the protocol description.

h	A hash function, $h : \{0,1\}^l \rightarrow \{0,1\}^l$
f_k	A keyed hash function, $f_k : \{0,1\}^l \times \{0,1\}^l \rightarrow \{0,1\}^l$ (a MAC algorithm)
N	The number of tags
l	The bit-length of a tag identifier
T_i	The i-th tag ($1 \leq i \leq N$)
D_i	The detailed information associated with tag T_i
u_i	A string of l bits assigned to T_i
t_i	T_i's identifier of l bits, which equals $h(u_i)$
x_{new}	The new (refreshed) value of x
x_{old}	The most recent value of x
r	A random string of l bits
ϵ	Error message
$\oplus$	XOR operator
$\|$	Concatenation operator
$\leftarrow$	Substitution operator

$x \gg k$ Right circular shift operator, which rotates all bits of x to the right by k bits, as if the left and right ends of x were joined.

$x \ll k$ Left circular shift operator, which rotates all bits of x to the left by k bits, as if the left and right ends of x were joined.

$\in_R$ The random choice operator, which randomly selects an element from a finite set using a uniform probability distribution

4.3 Protocol Description

The protocol is summarised in Figure 1.

4.3.1 Initialisation

The following steps must be performed by an initiator prior to using the protocol.

- An initiator (e.g. the tag manufacturer) assigns a string u_i of l bits to each tag T_i, computes $t_i = h(u_i)$, and stores t_i in the tag, where l should be large enough so that an exhaustive search to find the l-bit values t_i and u_i is computationally infeasible.
- The initiator stores the entries $[(u_i, t_i)_{\text{new}}, (u_i, t_i)_{\text{old}}, D_i]$ for every tag that it manages. The first pair is for the newly assigned values of u_i and t_i, the second pair is for the previously assigned values, and D_i is for the tag information (e.g., price, date, etc.). Initially $(u_i, t_i)_{\text{new}}$ is assigned the initial values of u_i and t_i, and $(u_i, t_i)_{\text{old}}$ is set to null.

4.3.2 Authentication Process

1. **Reader:** A reader generates a random bit-string $r_1 \in_R \{0,1\}^l$ and sends it to T_i.
2. **Tag:** The tag T_i generates a random bit-string $r_2 \in_R \{0,1\}^l$ as a temporary secret for the session, and computes $M_1 = t_i \oplus r_2$ and $M_2 = f_{t_i}(r_1 \oplus r_2)$. T_i then sends M_1 and M_2 to the reader.
3. **Reader:** The reader transmits M_1, M_2 and r_1 to the server.
4. **Server:**
   (a) The server searches its database using M_1, M_2 and r_1 as follows.
      i. It chooses t_i from amongst the values $t_{i(new)}$ or $t_{i(old)}$ stored in the database.
      ii. It puts $r_2 \leftarrow M_1 \oplus t_i$, and computes $M_2' = f_{t_i}(r_1 \oplus r_2)$.
      iii. If $M_2' = M_2$, then it has identified and authenticated T_i. It then goes to step (c). Otherwise, it returns to step i.
   (b) If no match is found, the server sends ϵ to the reader and stops the session.
   (c) The server computes $M_3 = u_i \oplus (r_2 \gg l/2)$ and sends it with D_i to the reader.
   (d) The server updates $u_{i(old)}$ and $t_{i(old)}$ for the tag T_i to u_i and t_i respectively, and sets $u_{i(new)} \leftarrow (u_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2$ and $t_{i(new)} \leftarrow h(u_{i(new)})$.
5. **Reader:** The reader forwards M_3 to T_i.
6. **Tag:** T_i computes $u_i \leftarrow M_3 \oplus (r_2 \gg l/2)$ and checks that $h(u_i) = t_i$. If the check succeeds, the tag has authenticated the server, and sets $t_i \leftarrow h((u_i \ll l/4) \oplus (t_i \gg l/4) \oplus r_1 \oplus r_2)$. If the check fails, the tag keeps the current value of t_i unchanged.

5. ANALYSIS

5.1 Privacy and Security

The protocol has the following privacy and security properties.

- **Tag Information Privacy**: The detailed information D_i of tag T_i is stored in the database of the server, which is assumed to be secure. A server and a reader communicate via a secure channel. Only a legitimate server can extract a tag identifier t_i from the pair (M_1, M_2) sent by T_i. The server will send D_i to a reader immediately after successful authentication of T_i. Thus, only an authorised server and reader are able to access the information associated with a tag.
- **Tag Location Privacy**: The responses of the tag T_i are anonymous, since the tag only ever sends pairs (M_1, M_2), which cannot be linked to any particular tag. In other words, the eavesdropper can neither link tag responses to previous responses from the same tag, nor distinguish one tag's responses from another's. It is thus difficult to track the location of a tag.
- **Tag Impersonation attack**: For a weak attacker to impersonate T_i, it must be able to compute a valid response (M_1, M_2) to a reader query. However, it is hard to compute such a valid pair without knowledge of t_i. Of course, a strong attacker is able to clone a tag.
- **Replay attack**: The scheme is a challenge-response authentication protocol using random numbers to resist replay attacks. The messages M_1, M_2 and M_3 are functions of freshly generated nonces r_1 and r_2, and thus these messages cannot be reused in other sessions.
- **Denial of Service attack**: If an attacker prevents the third flow from reaching the tag, the shared secrets of the server and T_i might be out of synchronisation, since the server will refresh u_i and t_i but the tag T_i will keep the current t_i. However, the server maintains both the old and new values of u_i and t_i for each tag T_i in its database, so that the server can resynchronise with the tag in such a situation.
- **Backward Traceability**: A strong attacker cannot identify the past interactions of T_i, even if it knows T_i's present internal state. That is, an attacker is unable to discover the previous identifiers of T_i because they are the cryptographic hashes of values not available to the tag.
- **Forward Traceability**: The scheme provides forward untraceability if a strong attacker misses M_3 just once in a single successful authentication session after compromising T_i's secret. That is, if the attacker cannot

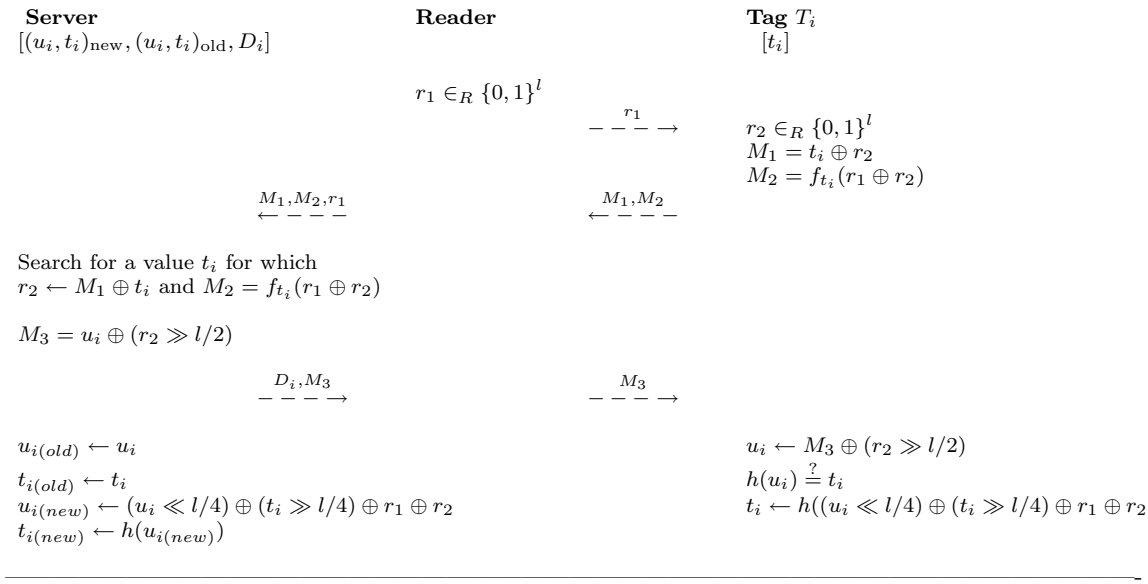

Figure 1: The Protocol

prevent T_i from receiving the last message M_3, or does not have access to all the values r_1, r_2 and u_i that are needed to refresh t_i, then it cannot compute the new identifier and track future transactions.

- **Server Impersonation attack**: A legitimate server responds with a message M_3 to a tag in order to enable the tag to authenticate the server. A strong attacker cannot create a valid M_3 without knowing u_i, and thus is unable to impersonate a legitimate server to T_i just by compromising t_i. However, if an attacker has access to all the exchanged messages and knows the value of t_i used in a single authentication session, it can compute the refreshed u_i for the following session. Hence, our protocol only resists such an attack on the assumption that an adversary does not have access to at least one of the values r_1, M_1 and M_3 in an authentication session that is performed between an authorised server and a tag, for which the adversary knows the tag's secret t_i.

In Table 1, we compare our protocol with the protocols introduced in section 3 with respect to the privacy and security properties discussed in sections 2.1 and 2.2. It is clear from Table 1 that the LK scheme and the protocol proposed in section 4 satisfy the greatest number of the privacy and security properties.

5.2 Performance

Like the protocol described in section 4, the DPLK, CC and LK schemes use pseudorandom number generators and checksum algorithms to protect the integrity of messages. We compare the storage and computation requirements of both a server and a tag for these three protocols and the protocol given in section 4.

Table 2 gives the storage needed in the server database and the memory required in a tag for each of these protocols. In the case of the database, we restrict our attention to the storage required to support tag authentication. Table 2 indicates that the protocol given in section 4 requires the least tag-side memory.

In order to compare the computational requirements of the four protocols, we take into account the checksum algorithms and secret updating functions that require the most computation in each protocol. The DPLK and CC protocols for the EPCglobal Gen-2 RFID tag make use of a cyclic redundancy code (CRC) as a checksum algorithm, which is efficient but less cryptographically strong, whereas the LK scheme uses pseudorandom functions (PRF), and the protocol given in section 4 uses hash functions for checksum computation and secret updating, which provide enhanced security. Table 3 compares the number of times each such function is computed in these protocols, under the assumption that they utilise the same algorithm.

From Table 3 it follows that the protocol given in section 4 requires fewer complex function invocations for the tag than the other three protocols. Thus our protocol has practical performance advantages over the DPLK, CC and LK schemes, whilst also providing the identified privacy and security properties.

6. CONCLUSION

We have reviewed the privacy, security, and performance requirements for RFID protocols. We have classified privacy issues into tag information and tag location privacy, and security threats into weak and strong attacks. We have also introduced the concept of server impersonation as a novel type of strong attack. We have presented a new RFID authentication protocol designed to meet the identified requirements. It has been compared with existing protocols with respect to both its privacy and security properties and its storage and computational requirements. The comparisons have shown that the protocol described here is both more secure than previously proposed schemes and has practical advantages over them, because it provides the greatest num-

Table 1: Privacy and security properties

Property	HM	MW	D	KN	DPLK	CC	LK	Sec. 4
Information Privacy	○	○	○	○	○	○	○	○
Location Privacy	–	○	–	–	○	○	○	○
Tag Impersonation	–	○	–	○	–	○	○	○
Replay attack	–	○	–	–	–	○	○	○
DoS attack	○	○	–	–	–	○	○	○
Backward Traceability	–	–	○	–	–	–	○	○
Forward Traceability	–	–	–	–	–	–	△	△
Server Impersonation	–	–	–	–	–	–	△	△

○ : provided
△ : provided under an assumption
– : not provided

Table 2: Storage requirements

Storage	DPLK	CC	LK	Sec. 4
Server	$(l + l_1 + l_2) \cdot N$	$(l + 2l_1 + 2l_2) \cdot N$	$2(l + 3l_3 + ml_4) \cdot N$	$4l \cdot N$
Tag	$l + l_1 + l_2$	$l + l_1 + l_2$	$l + l_3$	l

N : The number of tags
m : The maximum number of authentication failures in the LK protocol (e.g., $m = 64$ [15])
l : The bit-length of a tag identifier (e.g., $l = 64, 96$ or 128 [1, 5, 15])
l_1 : The bit-length of a PIN in the DPLK and CC protocols (e.g., $l_1 = 32$ [5])
l_2 : The bit-length of a session key in the DPLK and CC protocols (e.g., $l_2 = 16$ [5])
l_3 : The bit-length of a server validator in the LK protocol (e.g., $l_3 = 32$ [15])
l_4 : The bit-length of a tag secret transmitted in the LK protocol (e.g., $l_4 = 32$ [15])

Table 3: Computation requirements

Computation		DPLK	CC	LK	Sec. 4
Server	On receiving the 2nd flow	$(k+1)F$	kF	$(k_1+1)F$	kF
	On sending the 3rd flow	$1F$	$1F$	$1F$	–
	On updating or refreshing	$1F$	$2F$	$(k_1+k_2+m)F$	$1F$
	Total	$(k+3)F$	$(k+3)F$	$(2k_1+k_2+m+2)F$	$(k+1)F$
Tag	On sending the 2nd flow	$2F$	$1F$	$1F$	$1F$
	On receiving the 3rd flow	$1F$	$1F$	$2F$	$1F$
	On updating or refreshing	$1F$	$2F$	$1F$	$1F$
	Total	$4F$	$4F$	$4F$	$3F$

F : A computationally complex function (such as a CRC, PRF or hash function)
N : The number of tags
m : The maximum number of authentication failures in the LK protocol (e.g., $m = 64$ [15])
n : The length of the backward key chain in the LK protocol (e.g., $n = 2^{20}$ [15])
k : An integer satisfying $1 \leq k \leq 2N$
k_1 : An integer satisfying $0 \leq k_1 \leq m - 1$
k_2 : An integer satisfying $0 \leq k_2 \leq n - 2$

ber of identified privacy and security features and requires the least storage and computation in a tag.

7. REFERENCES

[1] EPCglobal class 1 gen 2 RFID specification. White paper, Alien Technology-RFID products & technology for new supply chain efficiency, visibility & security, 2005. www.alientechnology.com/docs/AT_wp_EPCGlobal_WEB.pdf.

[2] ActiveWAVE. *Advantages of RFID.* www.activewaveinc.com/technology_rfid_advantage.html.

[3] G. Avoine. *Cryptography in Radio Frequency Identification and Fair Exchange Protocols.* PhD thesis, Ecole Polytechnique Federale de Lausanne (EPFL), Lausanne, Switzerland, December 2005.

[4] G. Avoine, E. Dysli, and P. Oechslin. Reducing time complexity in RFID systems. In B. Preneel and S. Tavares, editors, *Selected Areas in Cryptography — SAC 2005*, volume 3897 of *Lecture Notes in Computer Science*, pages 291–306, Kingston, Canada, August 2005. Springer-Verlag.

[5] H. Chien and C. Chen. Mutual authentication protocol for RFID conforming to EPC class 1 generation 2 standards. *Computer Standards & Interfaces*, 29(2):254–259, February 2007.

[6] T. Dimitriou. A lightweight RFID protocol to protect against traceability and cloning attacks. In *Conference on Security and Privacy for Emerging Areas in Communication Networks — SecureComm*, pages 59–66, Athens, Greece, September 2005. IEEE.

[7] D. N. Duc, J. Park, H. Lee, and K. Kim. Enhancing security of EPCglobal gen-2 RFID tag against traceability and cloning. In *Symposium on Cryptography and Information Security — SCIS 2006*, Hiroshima, Japan, January 2006. The Institute of Electronics, Information and Communication Engineers.

[8] K. Finkenzeller. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification.* Wiley, 2nd edition, 2003.

[9] B. Glover and H. Bhatt. *RFID Essentials.* O'Reilly, 2006.

[10] A. Henrici and P. Müller. Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers. In R. Sandhu and R. Thomas, editors, *International Workshop on Pervasive Computing and Communication Security — PerSec 2004*, pages 149–153, Orlando, Florida, USA, March 2004. IEEE Computer Society.

[11] S. Karthikeyan and N. Nesterenko. RFID security without extensive cryptography. In *Workshop on Security of Ad Hoc and Sensor Networks — SASN '05*, pages 63–67, Alexandria, Virginia, USA, November 2005. ACM Press.

[12] T. Le, M. Burmester, and B. Medeiros. Forward-secure RFID authentication and key exchange. Cryptology ePrint Archive Report 2007/051, IACR, 2007.

[13] H. Lee, J. Yang, and K. Kim. Enhanced mutual authentication protocol for low-cost RFID. White Paper WP-HARDWARE-031, Auto-ID Labs, 2006.

[14] M. Lehtonen, T. Staake, F. Michahelles, and E. Fleisch. From identification to authentication — a review of RFID product authentication techniques. In *Printed handout of Workshop on RFID Security — RFIDSec 2006*, 2006.

[15] C. Lim and T. Kwon. Strong and robust RFID authentication enabling perfect ownership transfer. In P. Ning, S. Qing, and N. Li, editors, *Conference on Information and Communications Security — ICICS '06*, volume 4307 of *Lecture Notes in Computer Science*, pages 1–20, Raleigh, North Carolina, USA, December 2006. Springer-Verlag.

[16] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*, volume 6 of *Discrete Mathematics and Its Applications.* CRC Press, 1996.

[17] D. Molnar and D. Wagner. Privacy and security in library RFID: Issues, practices, and architectures. In B. Pfitzmann and P. Liu, editors, *Conference on Computer and Communications Security — ACM CCS*, pages 210–219, Washington, DC, USA, October 2004. ACM Press.

[18] M. Ohkubo, K. Suzki, and S. Kinoshita. Cryptographic approach to "privacy-friendly" tags. In *RFID Privacy Workshop*, MIT, MA, USA, November 2003. http://www.rfidprivacy.us/2003/agenda.php.

[19] R. D. Pietro and R. Molva. Information confinement, privacy, and security in RFID systems. In J. Biskup and J. Lopez, editors, *European Symposium Research Computer Security — ESORICS 2007*, volume 4734 of *Lecture Notes in Computer Science*, pages 187–202, Dresden, German, September 2007. Springer-Verlag, Berlin.

[20] N. Pramstaller, C. Rechberger, and V. Rijmen. A compact FPGA implementation of the hash function whirlpool. In *ACM/SIGDA 14th international symposium on Field Programmable Gate Arrays — FPGA'06*, ACM Press, pages 159–166, New York, 2006.

[21] B. Preneel et al. Final report of European project IST-1999-12324: New European schemes for signatures, integrity, and encryption. Available at: www.cosic.esat.kuleuven.be/nessie/, April 2004.

[22] F. Thornton, B. Haines, A. M. Das, H. Bhargava, A. Campbell, and J. Kleinschmidt. *RFID Security.* Syngress, Massachusetts, USA, 2006.

[23] G. Tsudik. YA-TRAP: Yet another trivial RFID authentication protocol. In *Fourth IEEE Annual Conference on Pervasive Computing and Communications — PerCom 2006*, pages 640–643, Pisa, Italy, March 2006. IEEE Computer Society Press.

[24] S. Weis, S. Sarma, R. Rivest, and D. Engels. Security and privacy aspects of low-cost radio frequency identification systems. In D. Hutter, G. Müller, W. Stephan, and M. Ullmann, editors, *International Conference on Security in Pervasive Computing — SPC 2003*, volume 2802 of *Lecture Notes in Computer Science*, pages 454–469, Boppard, Germany, March 2003. Springer-Verlag.

[25] Stephen Weis. Security and privacy in radio-frequency identification devices. Master's thesis, Massachusetts Institute of Technology (MIT), Massachusetts, USA, May 2003.

[26] K. Yüksel. Universal hashing for ultra-low-power cryptographic hardware applications. Master's thesis, Dept. of Electronical Engineering, Worcester Polytechnic Institute, Worcester, MA, USA, 2004.