Authentication and secret search mechanisms for RFID-aware wireless sensor networks

Shih-I Huang* and Shiuhpyng Shieh

Department of Computer Science and Information Engineering,
National Chiao Tung University,
Hsinchu, Taiwan
E-mail: sihuang@csie.nctu.edu.tw
E-mail: ssp@cs.nctu.edu.tw
*Corresponding author

Abstract: This paper investigates authentication and secure data retrieval issues in Radio-Frequency IDentification (RFID)-aware wireless sensor networks. We propose a network architecture (*ARIES*) consisting of RFIDs and wireless sensor nodes, a mutual authentication protocol (*AMULET*), and a Secret Search Protocol (*SSP*). ARIES utilises RFID-aware sensor nodes to alleviate distance limitation problem in RFID systems. AMULET performs mutual authentication and reduces the cost of re-authentication. SSP solves the privacy problem by offering a search mechanism over encrypted data. SSP does not need to decrypt encrypted data files while searching for specific data and the performance is greatly improved.

Keywords: RFID; wireless sensor networks; authentication; privacy; secret search.

Reference to this paper should be made as follows: Huang, S-I. and Shieh, S. (2010) 'Authentication and secret search mechanisms for RFID-aware wireless sensor networks', *Int. J. Security and Networks*, Vol.

Biographical notes: Shih-I Huang received BS and MS in Applied Mathematics from National Sun-Yat Sen University, and he is working towards his PhD in EECS in National Chiao Tung University. His research interests include network security, information security, wireless sensor network, data protection and data privacy.

Shiuhpyng Shieh received the MS and PhD in Electrical and Computer Engineering from the University of Maryland, College Park, respectively. He is a Professor of the Department of Computer Science, National Chiao Tung University (NCTU), and the Director of Taiwan Information Security Center at NCTU. He currently serves as the Steering Committee Chair of ASIACCS, ACM SIGSAC steering committee member, and the Chair of IEEE Reliability Society Taipei/Tainan Chapter. He also has been the Associate Editors of *ACM TISSEC*, *IEEE TR*, *IEEE TDSC*, *JCS*, *JISE* and Guest Editor of *IEEE Internet Computing*. His research interests include network and system security, wireless security and cryptography.

1 Introduction

Radio-Frequency IDentification (RFID) has been widely used in various applications. An RFID tag is a low-cost device with limited data storage space. An Identification Number (*ID*) is assigned to each tag for identification, and tagging specific targets with RFID tags allows for individualisation and recognition of each target by the attached ID. Through the wireless interface, each tag can report data when queried over radio by an RFID reader. The RFID reader can execute read, write and overwrite commands on each tag over the wireless interface. However, RFID readers can only recognise tags in proximity; a data tag that is out of range cannot be read by a reader. This distance limitation severely restricts RFID deployment. Despite equipping readers and tags with longer-range wireless communication capability, RFID readers still have difficulties in tracking or monitoring tags at a distance. To solve this distance limitation problem, a wireless sensor network can act as a bridge between the tags and the readers when tracking or monitoring remote targets.

A wireless sensor network (Estrin et al., 1999; Frank et al., 1992; Ledlie et al., 2002; Pottie, 1998) consists of groups of sensor nodes connected by wireless links that perform sensing tasks, such as detecting changes in temperature, pressure, etc. These sensors are employed for specialised tasks like surveillance and security, environmental monitoring, location tracking, warfare and healthcare.

Sensor nodes can communicate with RFID tags through the wireless interface. Since sensor nodes are cheap, they can be widely deployed to monitor every target, allowing readers to find targets at a distance. Although the use of sensor nodes solves the distance limitation problem, it introduces additional security challenges.

Examples of such a network composed of sensors and RFID tags include: the management of medical waste disposal, the management of blood storage bag in hospitals, the management of books in libraries, etc. In the aforementioned environment, the collaboration of sensor nodes and tags can form a dynamic, distributed database, where each sensor node contains a tiny database that tracks the data stored in RFIDs. Since sensor nodes are widely deployed, they form a group of distinctive databases. Simply encrypting the database ensures data security; however, it raises the issue of searching secrets.

Searching unencrypted data in a conventional remote database is relatively easy, but it leads to a serious problem: these queries may leak private information during transmission. One possible solution to prevent data leakage is to encrypt the original data and place it in a remote database. However, conventional cryptosystems and authentication schemes incur high computation cost, and may not be feasible for a network composed of wireless sensor nodes and RFIDs. Redesigning conventional cryptosystems and authentication schemes is a challenging task.

Owing to the limited resource and computation capability of sensor nodes, it is desirable to search encrypted data without the need to decrypt it. In a typical application, sensor nodes encrypt data to improve security against intrusions. To search data, a sensor node must first decrypt the data, a process that usually causes significant delay. Moreover, computation-limited, low-cost devices, such as sensor nodes and RFID tags, leave the decrypted data vulnerable to disclosure. In such an exposed environment, it is desirable to develop a new secret search method that performs secret search directly on ciphertexts without the need to decrypt them, thereby preserving secrecy and avoiding decryption delay.

For secret search in wireless sensor networks, the following requirements are considered important:

1 *Secrecy*: Storing data in an encrypted form helps retain its confidentiality. Because sensors are vulnerable, computation-limited, and low-cost devices, allowing sensors to decrypt data to perform a search result in unnecessary risk of disclosure. Thus, sensors must execute a secret search directly on ciphertext, rather than plaintext. Furthermore, data transmitted over a wireless interface is susceptible to exposure. Therefore, sensors must only transmit encrypted data. In summary, the data must remain in an encrypted form and should not be decrypted unless necessary to minimise the possibility of disclosure.

2 *Authentication*: Since the network obtains data from a large number of sensors or tags, attackers can easily acquire readers with the same specifications to extract data stored in the tags. Therefore, both the reader and the tag need to verify the authenticity of its communication counterpart before executing read or write operations.

3 *Integrity*: Assuring data integrity prevents attackers from using unauthorised readers to modify or inject data into databases. Readers or tags must verify data integrity upon receipt of data.

4 *Performance*: Requiring a sensor node to decrypt data before searches causes significant and unnecessary delay. Also, the limited computation capabilities of sensor nodes and tags hinder them from performing complex operations, such as encryption and exponential calculations. Therefore, all operations must be redesigned to fit their computation capabilities.

Previous research work focused on authentication. Some papers propose the use of Public Key Infrastructure (PKI) to authenticate two parties through a trusted-third-party (Gu et al., 2006). This solution is inadequate for RFID applications since the PKI requires the reader or tag to save private keys and verify the identity of others with the help of the trusted-third-party. Tags have little storage, and they can only transmit data to devices in proximity. In other words, the trusted-third-party must be located near the tags, which is a difficult requirement to achieve and one that presents other security risks. Moreover, the tag cannot afford additional computational power required to verify others. Therefore, a PKI scheme is not feasible for RFID applications.

A randomised lock protocol (Weis et al., 2004) was proposed for private authentication in a highly constrained computation and storage environment. However, this scheme is neither private nor secure against passive eavesdroppers. As an improvement, a PRF-based private authentication protocol (Molnar and Wagner, 2004) was proposed. Unfortunately, both protocols (Weis et al., 2004; Molnar and Wagner, 2004) require re-authentication of a tag even if another authorised reader previously authenticates the tag. These extra steps are computationally wasteful and unnecessary.

Privacy is a major concern encountered in RFID applications (Gertner et al., 1998, Huang and Shieh, 2005). An RFID tag may store sensitive data associated with a target, which must remain private. Since readers, tags and sensor nodes send messages through a wireless medium, attackers can easily eavesdrop to their communication and extract secret information.

An intuitive way of protecting private data is encryption (Eschenauer and Gligor, 2002). However, tags and sensor nodes have severely limited storage and computation capability; consequently, conventional cryptographic algorithms are not well suited for these devices. As a result, we must redesign security mechanisms to support RFID tags and sensor nodes.

A new problem arises from encrypting data: RFID readers cannot easily perform queries on data in encrypted form (Chor et al., 1998; Alon et al., 1995). Researchers have investigated secret search over encrypted data in an untrusted file server or external memory environment (Kusilevitz and Ostrovsky, 1997; Dabek et al., 2001; Clarke et al., 2000; Alon et al., 1995; Devanbu and Stubblebine, 2002). A recent method (Song et al., 2000) is proposed

for secret searching on untrusted servers. Unfortunately, their scheme requires complex encryption operations unavailable to both tags and sensor nodes. Another problem of the scheme is that same plaintexts at different places will be encrypted into the same ciphertexts in their proposed scheme III. Hence, malicious attackers could inject meaningful plaintexts into the database and use the corresponding ciphertexts to find their interests without decrypting entire or part of the database.

Other researches tried to solve this searching problem by inserting specific encrypted keywords into the ciphertexts (Gu et al., 2006; Waters et al., 2004; Ballard et al., 2005; Ostrovsky and Skeith, 2005; Chow, 2005). These encrypted keywords can be viewed as indices and could therefore be used in search operations (Bennett et al., 2002; Zheng et al., 1992). However, these keywords are fixed and must be defined beforehand. Therefore, this inconveniency makes them difficult to use. Another solution is to support searching over encrypted data by using multi-party computation and oblivious functions (Sun and Shieh, 1994, 1996; Feldman, 1987). However, this solution requires high computation overhead and therefore is not applicable in a tag or sensor system.

Our contribution is three-fold. First, we propose an architecture consisting of passive RFIDs and RFID-aware sensor networks (*ARIES*). This architecture extends RFID's capabilities through a wireless sensor network by utilising sensor nodes to locate targets at a distance. Second, we design a private mutual authentication protocol (*AMULET*), which is feasible for RFIDs and sensor nodes, and reduces the cost of re-authentication. Third, we present an *SSP* that enables readers to perform searches over encrypted data, allowing data to remain encrypted during transmission or at vulnerable locations. By only using one-way hash functions, pseudorandom number generation functions and XOR operations, SSP accommodates the resource limitations of both tags and sensors. In addition, SSP can solve the problem that same plaintexts at different places will be encrypted into the same ciphertexts.

The rest of this paper is organised as follows. Section 2 introduces the proposed *ARIES* architecture for RFID and sensor networks, while Section 3 presents our *AMULET* mutual authentication protocol for readers and tags. The SSP is presented to query encrypted data in Section 4, and more advanced properties are discussed in Section 5. Finally, Section 6 provides security proof of the proposed schemes, and Section 7 concludes our work.

2 ARIES

In this section, we introduce our system architecture, participating roles and their set-ups when deploying such a network. The notations we used are listed here:

ID_j: The identity of RFID tag j

$S_{i,j}$: A secret key shared by reader i and tag j

EK_i: A symmetric encryption key used by reader i

R: Nonce

f: $\{0,1\}^* \rightarrow \{0,1\}^\delta$ A pseudorandom number generating function

H: $\{0,1\}^* \rightarrow \{0,1\}^\delta$: A one-way hash function.

Motivated by the distance limitation problem of RFID readers, we propose an **AR**chitecture of RF**I**Ds and RFID-aware s**E**nsor network**S** (*ARIES*). Three roles are involved in our proposed system: RFID reader (abbr. as reader in what follows), RFID tag (abbr. as tag in what follows), and RFID-aware Sensor node (abbr. as sensor node in what follows). Since tags (tags on moveable targets) may be quite far away from readers, sensor nodes in our architecture are used as the gap between readers and tags by transmitting commands from reader to tag or sending tag data to readers, allowing readers to trace any tag located far away.

Although an *RFID reader* is called a reader by convention, it also has writing capability. Thus, a reader can perform read, write and overwrite operations on RFID tags through the wireless interface. In our system, readers have access to a shared database storing all authorised *ID*s. To construct a secure channel between readers and tags, the readers share a unique secret key s with each tag. While readers save all tag pairs (s, ID) in the shared database, each tag stores its individual secret key s locally. Additionally, each reader possesses a unique encryption key EK_i to encrypt data, which it saves locally and remotely (on the shared database). EK_i can be used to verify the ownership of encrypted data.

An *RFID tag* is a small, thin, readable and writeable device that can store limited data. Embedded with a transceiver, each tag can communicate via wireless channels with other devices, such as readers or sensor nodes. Because tags have limited computation capability, intensive operations, such as encryption, are impractical for tags. Therefore, we will introduce new methods supporting lightweight authentication in Section 3.

An *RFID-aware sensor node* is a tiny device capable of detecting RFID tags. It is also outfitted with a transceiver to communicate with readers and tags through a wireless interface. Like tags, sensor nodes are cheap and widely dispersible.

As mentioned earlier, sensor nodes can compensate for the distance limitation of RFID readers. To reach readers, we assume that the sensor network allows for multi-hop communication. Furthermore, readers, tags and sensor nodes can maintain secure communications. However, we do not introduce a security scheme between readers and sensors, tags and sensors, or readers and tags. Instead, we merely indicate that secure channels exist through shared secret keys or pre-distributed verifiable key pairs.

To prevent replay attacks, we assume that each reader, tag and sensor node has a synchronised timer, allowing them to verify that an authentication process has not expired. Though it is unpractical to put a timer into a tag, the tag yet can have a timer virtually by neighbouring sensor nodes periodically sending their timer readings. Our system merely requires loose time synchronisation because of infrequent authentication. Because past

researchers have investigated time synchronisation (van Greunen and Rabaey, 2003; Generiwal et al., 2003), we do not address this issue here. Another consideration is tags can be compromised and send bogus timestamps. Several existing protocols using majority vote can successfully solve this. We do not intend to discuss this as it is beyond our scope.

In our architecture, readers can request data from faraway tags via sensor nodes. Figure 1 depicts the RFID readers, RFID tags and RFID-aware wireless sensor nodes that make up the *ARIES* architecture. The sensor node collects data from tags in its vicinity and stores it in a local tiny database, where each attribute represents characteristics of the target. Table 1 represents a sample distributed tiny database. The (Target ID, Sensor ID) pair indicates the Target ID that is detected by Sensor ID. These pairs roughly reveal the geographical information about all targets. The ($Attr_1$, $Attr_2$ …, $Attr_n$)-tuple manifests the data stored in the target. This distributed database can be used not only to search specific event with some user-interesting values, but also to track the location of every sensor node and target.

Figure 1 *ARIES* architecture (see online version for colours)

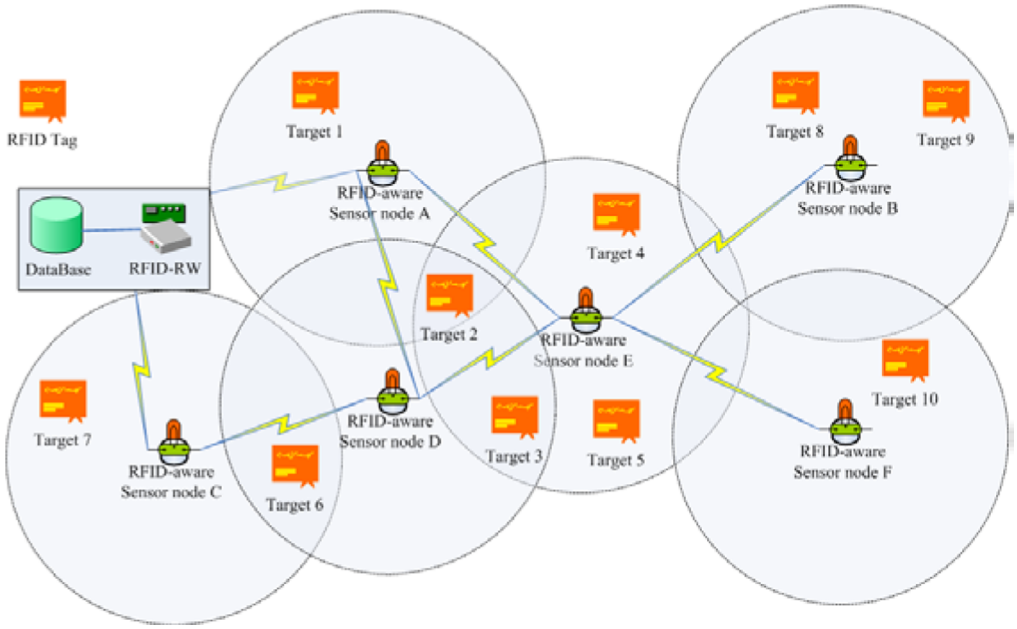


Table 1 Distributed tiny database (see online version for colours)

Target ID	Sensor ID	Attr 1	Attr 2	----------	Attr N
ID_1	*Sensor A*	*Attr*(*A*1)	*Attr*(*A*2)	----------	*Attr*(*An*)
ID_2	*Sensor A*	*Attr*(*A*1)	*Attr*(*A*2)	----------	*Attr*(*An*)
ID_2	*Sensor B*	*Attr*(*B*1)	*Attr*(*B*2)	----------	*Attr*(*Bn*)
⋮					
ID_7	*Sensor K*	*Attr*(*K*1)	*Attr*(*K*2)	----------	*Attr*(*Kn*)

This architecture is workable for passive RFID tags, needs only RF signals to charge and becomes active. No extra power waste will be needed, and thus each sensor node can reduce unnecessary power consumption in reading or writing RFID tags.

2.1 AMULET

In this section, we are going to introduce a lightweight authentication mechanism between readers and tags. Since all query actions are initiated by readers, the sensor nodes are merely viewed as generic routers and used only to forward these queries to tags. Therefore, our scheme only focuses on building low-computation authentication between readers and tags.

AMULET involves two phases namely the setting phase and authentication phase. The setting phase initialises necessary components, such as IDs and keys, which will be used for authentication. The authentication phase performs mutual authentication for sensor nodes and tags.

2.1.1 Setting phase

In AMULET, we need to set up two components: the tag and the reader. For each tag, it is assigned with an unique identification, ID_i, and a unique secret, s_i. All pairs of (ID_i, s_i) are stored in the reader's database that will be used in authentication phase. These settings are performed in factor or library before deploying them into real work.

Since the passive tag has limited computation capabilities, it cannot afford complicated operations. It is reasonable to assume that in our paradigm the tag can afford lightweight operations including XOR and a pseudorandom number generating function f (Molnar and Wagner, 2004). The pseudorandom number generating function f can also be stored in both the tag and the reader.

2.1.2 Authentication phase

Authentication is the first step in building a trust relationship between readers and tags. Since readers and tags rely on wireless communication, attackers may eavesdrop on transmitted data and extract passwords. Previous research characterises RFID communication as asymmetrical in signal strength. That is, attackers have an easier time listening in on signals from reader to tag than on data from tag to reader. Additionally, attackers can easily purchase readers and tags to perform malevolent operations. Therefore, we propose **A MU**tua**L** auth**E**ntication pro**T**ocol (*AMULET*) for readers and tags to prevent attackers from impersonating authorised entities.

Wagner (2004) propose a PRF-based private authentication protocol in Molnar and Wagner (2004), which extends Weis's randomised hash lock protocol. Their authentication scheme comprises a triple of probabilistic polynomial time algorithms (G, R, T) (for Generator, Reader and Tag). Also, each tag possesses a unique secret s_i and identification ID_i, and the reader contains a database storing all pairs of (s_i, ID_i). In their protocol, each reader needs to authenticate every target, even if another reader previously validates the tag. This redundant authentication imposes unnecessary overhead on low-computation power devices.

In our scheme, we assign each tag a unique secret s_i and identification ID_i and store all the tag pairs (s_i, ID_i) in a database. According to the protocol outlined in Figure 2, *AMULET* involves the following steps:

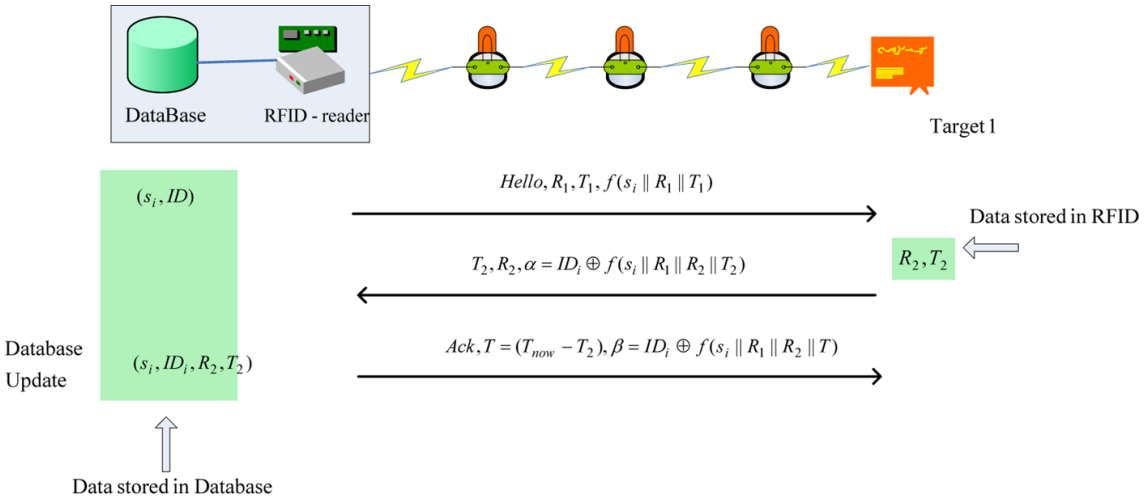
1 To begin the authentication process, the reader chooses a random number $R_1 \in \{0, 1\}^n$, checks the current time T_1, and calculates $f(s_i \| R_1 \| T_1)$, where $\|$ indicates string concatenation. For a reader to authenticate a tag with ID_i, the reader then sends a *Hello* packet to the tag that includes R_1, T_1 and $f(s_i \| R_1 \| T_1)$.

2 When the tag receives a *Hello* packet, it chooses a random number $R_2 \in \{0, 1\}^n$, checks the current time T_2, and calculates $\alpha = ID_i \oplus f(R_1 \| R_2 \| T_2)$. The tag sends a packet containing R_2, T_2 and α back to the reader and also saves a copy of R_2 and T_2. It is quite reasonable that tags have enough memory space to store these two parameters.

3 Upon receiving R_2, T_2 and α, the reader verifies that $\alpha = ID_i \oplus f(s_i \| R_1 \| R_2 \| T_2)$ and $T_2 > T_1$. It then checks for the current time T_3, computes the time difference $T = T_3 - T_2$, calculates $\beta = ID_i \oplus f(s_i \| R_1 \| R_2 \| T)$, and returns an *Ack* (acknowledgement) packet to the tag that includes T and β. In addition, the reader updates the original tag pair (s_i, ID_i) to (s_i, ID_i, R_2, T_2).

4 Finally, the tag validates the *Ack* packet by checking $ID_i = \beta \oplus f(s_i \| R_1 \| R_2 \| T)$.

Figure 2 AMULET architecture (see online version for colours)

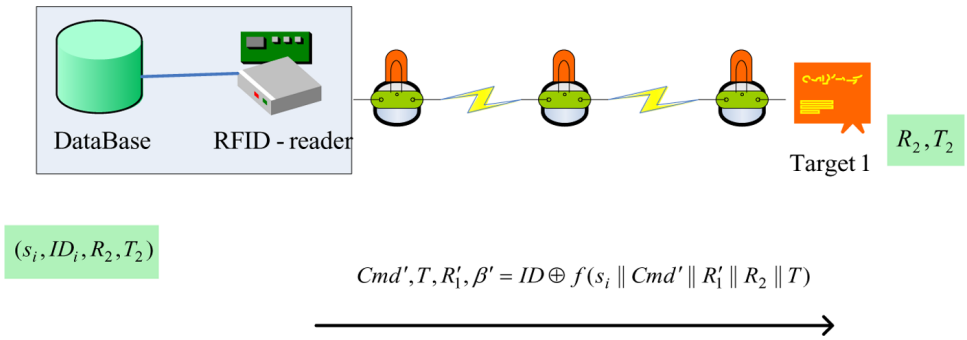


AMULET can reduce the re-authentication cost when a reader wishes to send commands to an authenticated tag. The reader need not re-authenticate the tag because the database stores the tag's information (s_i, ID_i, R_2, T_2). As depicted in Figure 3, the tag can verify future commands by the following two steps:

1 If a new reader queries the database and obtains (s_i, ID_i, R_2, T_2) instead of (s_i, ID_i), then it recognises that another reader already authenticated the tag with this ID_i. As a result, it chooses a random number $R_1' \in \{0, 1\}^n$, checks for the current time T_3, computes the difference in time $T = T_3 - T_2$, and calculates $\beta' = ID_i \oplus f(s_i \| Cmd' \| R_1' \| R_2 \| T)$. The reader then sends its command Cmd', along with R_1', T, and β, to the tag.

2 Upon receipt of the Cmd' packet, the tag verifies that $T = T_3 - T_2$ and $ID_i = \beta' \oplus f(s_i \| Cmd' \| R_1' \| R_2 \| T)$ before executing Cmd'. Otherwise, the tag drops the command.

Figure 3 Commands verification without re-authentication process (see online version for colours)



As previously mentioned, it is harder to eavesdrop on the channel from tag to reader than from the reader to tag; accordingly, *AMULET* provides security against passive eavesdropping on the reader-to-tag link. A common attack to authentication protocols is man-in-the-middle attack, which *AMULET* naturally resists. Although an attacker can gather R_1 and T_1 from the reader and R_2, T_2, and $\alpha = ID_i \oplus f(s_i \| R_1 \| R_2 \| T_2)$ from the tag, it does not possess the secret key s_i, and thus cannot modify or inject its own α. Consequently, man-in-the-middle attacks will not succeed against our protocol, and we will formally prove this property in Section 4. Furthermore, *AMULET* can defeat replay attacks when tags check that T has not expired and β or β' is valid for a first-time authentication or re-authentication procedure, respectively.

3 SSP

To preserve data privacy, simply encrypting data prevents attackers from discerning the contents. However, traditional cryptography is not feasible in tags and sensor nodes because of their limited computation capability. Moreover, it is difficult to search encrypted data. To solve this problem, we propose **an** *SSP*, which maintains data in an encrypted form but allows authorised readers to perform searches without disclosing data during transmissions or queries.

SSP involves two phases: data encryption phase and data search phase. The data encryption phase encrypts data and stores corresponding ciphertext to tags. The data search phase describes how to achieve private search on ciphertexts.

3.1 Data encryption phase

In SSP, tags store each characteristic of their associated target as an attribute of the target. We can formally describe a target as $B = (Attr1, A\,ttr2, \ldots\ AttrN)$, where N is the number of attributes. For example, a tag attached to a book may store the book's ID, title, authors, check-in and check-out time, borrower's ID, etc. Personal attributes like borrower's ID must not be exposed to unauthorised readers or attackers. As shown in Figure 4, SSP involves the following steps:

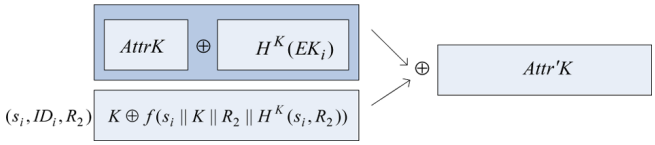
1 For an attribute *AttrK*, the reader first generates $H^K(s_i, R_2)$ by iteratively hashing (s_i, R_2) K times, where K indicates the number of the sequential order of *AttrK*.

2 Next, the reader generates $H^K(EK_i)$ by iteratively hashing EK_i K times.

3 After calculating $f(s_i \| K \| R_2 \| H^K(s, R_2))$, the reader XOR it with K to form $\lambda = K \oplus f(s_i \| K \| R_2 \| H^K(s_i, R_2))$.

4 Finally, the reader computes $Attr'K = AttrK \oplus H^K(EK_i) \oplus \lambda$ and overwrites *AttrK* with $Attr'K$.

Figure 4 SSP operations for attribute K (see online version for colours)



Once every attribute is overwritten, attackers will learn nothing from the encrypted data. Since K is different for all attributes, each attribute generates a different encrypted attribute value even if some attribute values happen to be the same. This will keep attributes relatively private. Figure 5 illustrates SSP's operations.

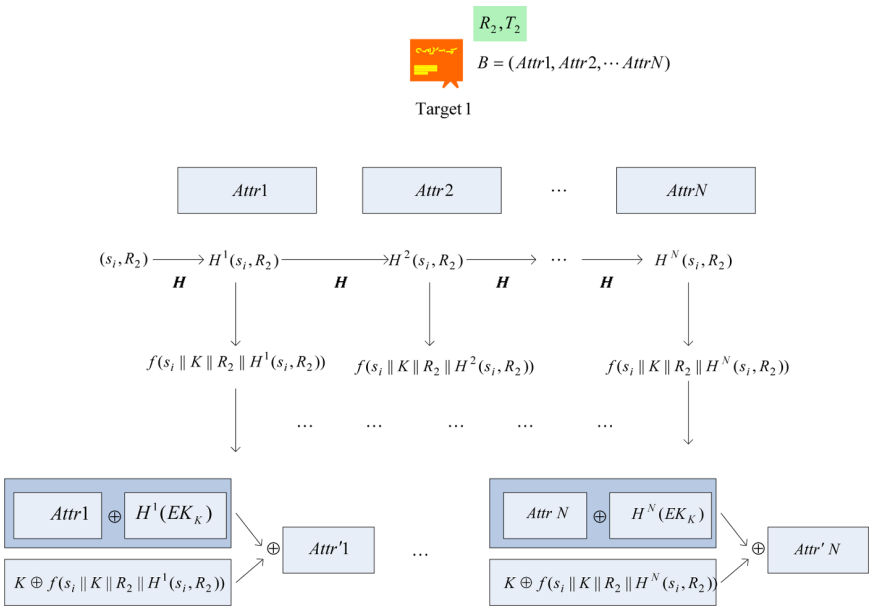
Authorised readers can inversely transform $Attr'K$ back to *AttrK* by computing $AttrK = Attr'K \oplus H^K(EK_i) \oplus K \oplus f(s_i \| K \| R_2 \| H^K(s_i, R_2))$. Because authorised readers can retrieve (s_i, R_2) from the database, they can easily calculate *AttrK* without exposing sensitive and private data during wireless transmission.

A major contribution of SSP is that it ensures the privacy of the remaining attributes in the event that some attributes are compromised. Since $f(s_i \| K \| R_2 \| H^K(s_i, R_2))$ varies by K,

$$Attr'(K+1) = Attr(K+1) \oplus H^{K+1}(EK_i) \oplus (K \oplus f(s_i \| K \| R \| H^{K+1}(s_i, R_2)))$$

will remain secure even when $f(s_i \| K \| R_2 \| H^K(s_i, R_2))$ is compromised.

Figure 5 SSP operations (see online version for colours)



3.2 Data search phase

To search for an attribute *AttrK*, the RFID reader broadcasts an encrypted query $AttrK \oplus H^K(EK_i)$ to all sensor nodes. Next, each sensor node calculates $Attr'K$ by $Attr'K = AttrK \oplus H^K(EK_i) \oplus K \oplus f(s_i \| K \| R_2 \| H^K(s_i, R_2))$ with its own s_i, R_2, and every value of K. The sensor node must calculate an $Attr'K$ for all *K*s because it does not know the value of *K*. If any sensor node finds a match, it returns $Attr'K$ and *K* to the RFID reader. Since data is encrypted, privacy is maintained during the transmission.

4 Security analysis

In this section, we first demonstrate the security of *AMULET* under man-in-the-middle attacks. Second, we provide an analysis that discusses the resources required to break SSP.

Before we proceed to theoretical proof, we first describe the security requirements specifying the attacker's abilities and when the latter is considered successful. The abilities and disabilities of the attackers include:

- the attacker has an arbitrary polynomial-time computation power
- the attacker can eavesdrop to messages in the Air
- the attacker can modify encrypted messages
- the attacker can compromise tags
- the attacker cannot compromise readers
- the attacker cannot know the shared secrets s_i and the encryption keys EK_i.

An attacker is considered to be successful if the attacker can comprise the original messages or attributes, or forge a legal encrypted data. In our system, we consider only passive attacks where attackers can only listen to the messages transmitted in the Air or modify the messages. We do not intend to solve active attackers' problem, as these kinds of attacks are not hard to be solved merely by any cryptographic algorithms.

Before we begin our proof, we give several definitions here.

Definition 1 (*Instance*)**:** We can formally describe a target by its *ID* and attributes, where $B = (ID_B, Attr1, Attr2, \ldots, AttrN)$. An *instance* X_B is defined as $X_B = (Attr1, Attr2, \ldots, AttrN)$, and a verification function V_f is defined as

$$V_f(X_B) = \sum_{i=1}^{n} Attri.$$

Each instance is a part of the distributed database, and the verification function is used to distinguish one instance from another.

Definition 2 (*Distinguishable*)**:** Two instances of a target are distinguishable if any attribute has different values.

Definition 3 (*R*-Breakable)**:** Let an instance $X_R = (Attr1, Attr2, \ldots, AttrN)$. If X_B can be derived from *R* ($R \le N$) attributes, then it is *R*-Breakable. Under the same condition, a system is *R*-Breakable if it needs *R* resources to break the system.

4.1 Security of AMULET

We classify man-in-the-middle attacks into three categories: type-1 attack modifies $R1$ only, type-2 attack modifies R_2 only, and type-3 attack modifies R_1, R_2 and α. We will show that these three types of attacks fail against our authentication protocol.

Type-1 attacker, shown in Figure 6, eavesdrops on R_1, generates a false value R_1', and delivers it to the tag. The tag then uses R_2 to generate $\alpha = ID \oplus f(s_i \| R_1' \| R_2 \| T_2)$ and sends R_2, T_2 and α back to the reader. Since $R_1 \neq R_1'$, the reader will find that $f(s_i \| R_1 \| R_2 \| T_2) \neq f(s_i \| R_1' \| R_2 \| T_2)$. As a result, the readers can prevent type-1 man-in-the-middle attacks.

As depicted in Figure 7, a type-2 attacker eavesdrops on R_2, produces a false value R_2', and transmits R_2 and α back to the reader. Because $R_2 \neq R_2'$, the reader will find that $f(s_i \| R_1 \| R_2 \| T_2) \neq f(s_i \| R_1' \| R_2 \| T_2)$, thus thwarting type-2 man-in-the-middle attacks.

In Figure 8, a type-3 attacker generates false R_1', R_2', and α' back to the reader and the tag separately. Since s_i remains secret, the reader will observe that $f(s_i \| R_1 \| R_2 \| T_2) \neq f(s_i \| R_1 \| R_2' \| T_2)$ and $ID_i \neq ID'$. causing type-3 man-in-the-middle attacks to fail.

Figure 6 Type-1 man-in-the-middle attack (see online version for colours)

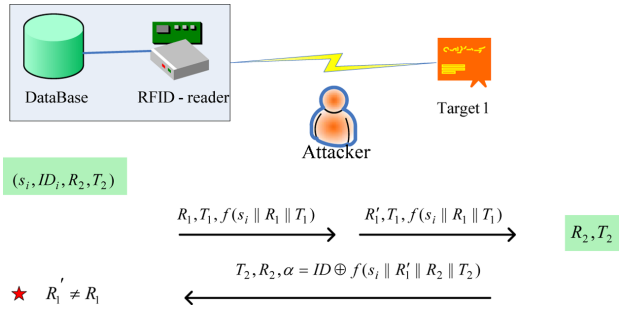


Figure 7 Type-2 man-in-the-middle attack (see online version for colours)

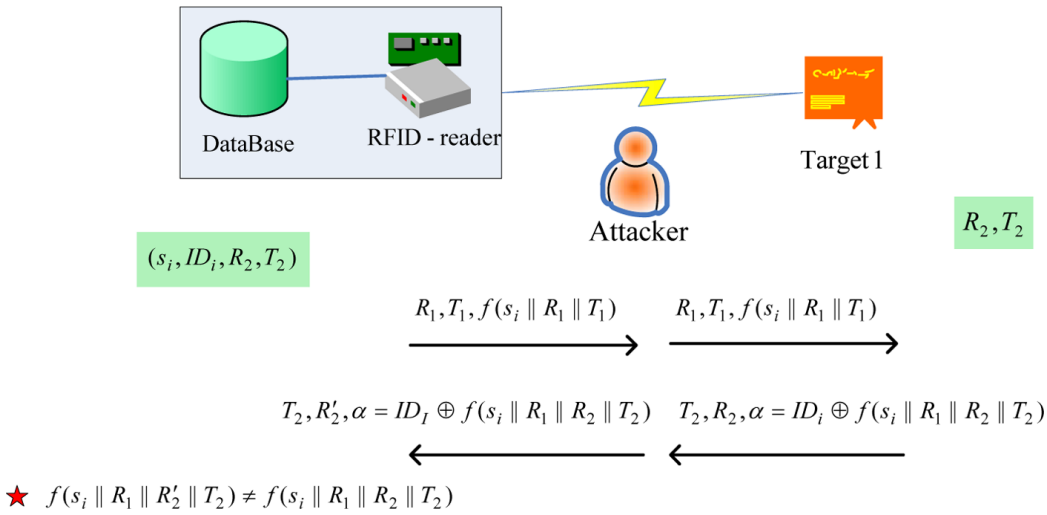
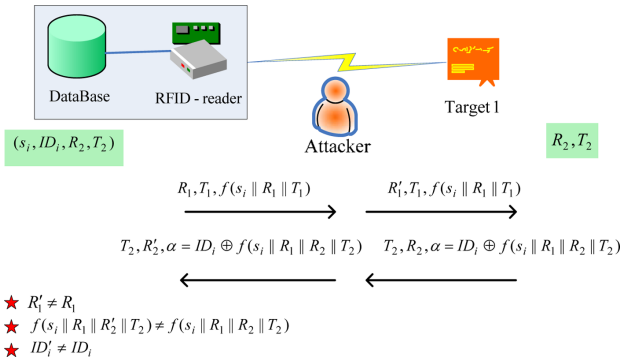


Figure 8 Type-3 man-in-the-middle attack (see online version for colours)



4.2 Security of SSP

We provide a proof of SSP's security strength in terms of the secrecy of its attributes. By establishing the number of resources required to compromise a system, we can evaluate its security strength. Theorem 1 states that an attacker must have knowledge of both s and R_2 to compromise $Attr'K$, where

$$Attr'K = AttrK \oplus H^K(EK_i) \oplus (K \oplus f(s_i \| K \| R_2 \| H^K(s_i, R_2))). \quad (1)$$

Theorem 1: $f(s_i \| K \| R_2 \| H^K(s_i, R_2))$ *is* (s_i, R_2) – *breakable.*

Proof: Since attackers may extract the values of N and K, only s and R_2 must be kept secret. Attackers must know both s_i and R_2 to compromise $f(s_i \| K \| R_2 \| H^K(s_i, R_2))$. Thus, $f(s_i \| K \| R_2 \| H^K(s_i, R_2))$ is (s_i, R_2) – breakable.

An instance is a collection of all attributes of a tag whose security strength is defined by the number of attributes needed to compromise the tag. Thus, as the number of distinguishable attributes increases, the instance will attain a higher security level.

Theorem 2: *Given an instance of any two attributes Attr′I, Attr′J, where I ≠ J, there does not exist a different instance Attr″I, Attr″J, such that the verification function evaluates to the same value* $V_f(Attr'I, Attr'J) = V_f(Attr''I, Attr''J)$.

Proof: Let *AttrI*, *AttrJ* be two original attributes such that $I > J$, *Attr′I*, *Attr′J* be their transformed attributes, and $V_f(Attr'I, Attr'J)$ be the verification of the transformed attributes. We will prove that an attacker cannot generate attributes *Attr″I*, *Attr″J* that satisfies $V_f(Attr'I, Attr'J) = V_f(Attr''I, Attr''J)$.

From equation (1), we know that

$$V_f(Attr'I + Attr'J) = (AttrI \oplus H^I(EK_i) \oplus (K \oplus f(s_i \| K \| R_2 \| (H^I(s_i, R_2)))) + (AttrJ \oplus H^J(EK_i) \oplus (K \oplus f(s_i \| K \| R_2 \| (H^J(s_i, R_2))))).$$

An important property of our protocol is that *AttrI* can be used to authenticate *AttrJ* by checking that

$$H^J(s_i, R_2) = H^{J-I}(H^I(s_i, R_2)) \tag{2}$$

If an attacker generates attributes *Attr″I*, *Attr″J*, $H^I(s_i, R_2)$ and $H^J(s_i, R_2)$ can be calculated by the following two equations.

$$Attr''I \oplus AttrI = I \oplus f(s_i \| I \| R_2 \| H^I(s_i, R_2)) \tag{3}$$

$$Attr''J \oplus AttrJ = J \oplus f(s_i \| J \| R_2 \| H^J(s_i, R_2)). \tag{4}$$

Because only authorised readers and tags know s_i and R_2, the attacker cannot falsify $H^I(s_i, R_2)$ and $H^J(s_i, R_2)$. This property is vital because if the attacker could successfully generate false attributes and the readers or tags are not aware of the falsity, the attacker could, therefore, inject unnecessary data or operation to tags. This makes readers or tags unreliable.

The next theorem stipulates that an attacker must compromise all attributes of an instance to deceive readers. If only a portion of the attributes are compromised, the reader can still verify the instance. We will use induction to show that an instance of a target *B* is *N*-breakable and distinguishable, where *N* is the number of attributes of *B*.

Theorem 3: Let $V_f(B) = \sum_{i=0}^{n} Attri = Attr'1 + Attr'2 + \cdots + Attr'N$. *B* is *N*-breakable and distinguishable.

Proof: Let $B = (Attr1, Attr2, \ldots, AttrN)$ be the original attributes and $B' = (Attr'1, Attr'2, \ldots, Attr'N)$ be the attributes after transformation.

For $N = 2$, *B* is 2-breakable by Theorem 2.

Suppose when $N = P$, *B* is *P*-breakable. We want to prove *B* is *P*-breakable when $N = P + 1$.

Let $B_1 = (Attr1, Attr2, \ldots, AttrN, AttrN + 1)$.

From Theorem 2, we know that every pair of attributes is distinguishable. Therefore, $AttrN + 1$ and *AttrM* are distinguishable for $M = 1, 2, \ldots, N$ by verifying $H^{N+1}(s_i, R_2)$ and $H^1(s_i, R_2), H^2(s_i, R_2), \ldots, H^N(s_i, R_2)$, respectively. Since all $N + 1$ attributes are distinguishable, we have shown that an instance of a target is *N*-breakable.

If the new attribute *AttrK* is inserted between *Attr*1 and *AttrN*, *AttrK* can be verified by both its predecessor attribute *Attr*($K - 1$) and its successor attribute *Attr*($K + 1$) through equations (5) and (6).

$$H(H^{K-1}(s_i, R_2)) = H^K(s_i, R_2) \tag{5}$$

$$H(H^K(s_i, R_2)) = H^{K+1}(s_i, R_2). \tag{6}$$

If both equations (5) and (6) are satisfied, the added attribute *AttrK* is valid. Otherwise, *AttrK* is invalid and should be discarded. Since an instance *B* is *N*-breakable, it needs to compromise entire *N* attributes to achieve falsity. Moreover, if only one attribute is compromised, the attacker cannot use this attribute to generate other false attributes.

5 Discussion

In this section, we discuss some practical considerations for the proposed schemes and give comparisons with related work.

5.1 Practical considerations

The proposed SSP is feasible for a network with sensor nodes and RFID tags as SSP uses low-computation operations, i.e., a hash function and a random number generating function, to encrypt and search data. Unlike other schemes, SSP gets clients (tags) involved in the data encryption process. The reader uses the random number R_2 generated by each tag as a parameter in encryption process. It is clear that in the scheme, each tag generates different R_2, and this enhances security and privacy strength of the data encryption process. It can be verified by the readers if the data is being copied to another tag if the tag does not know R_2.

5.2 Supporting fixed-index search queries

A problem occurred in private search schemes (Gertner et al., 1998) is that private search schemes are hard to provide fixed-index search among different clients. Most private search schemes use different encryption keys for each client (databases) to provide better security strength when a client is compromised. However, using different encryption keys causes inconvenience in searching data. The private search scheme provides fixed-index search queries, however, it leads to a security and privacy problem, i.e., all encrypted data are identical in clients.

In SSP, a reader can use the same encryption key *EK* for every tag, but still the encrypted data are different for every tag as long as these tags do not use the same R_2. In this case, the data encryption process can be reduced to:

$$\text{ciphertext} = ATTR\ K \oplus H(EK) \oplus K \oplus f(s_i \| K \| R_2 \| H^K(s_i, R_2)). \tag{7}$$

Since each tag owns different R_2, the ciphertext will be different according to different tags. To search, the reader uses $ATTR\,K \oplus H(EK)$ to search all tags (clients), and each tag will generate their own $K \oplus f(s_i \| K \| R_2 \| H^K(s_i, R_2))$ and check if there is any attribute satisfying equation (7). Our proposed encryption scheme simplifies search query by supporting fixed-index search but remains data secrecy and privacy for different clients (tags).

5.3 Supporting ciphertext update

It is obvious that if a reader is compromised, the compromised reader can retrieve all encrypted data in clients and recover all private information stored in clients. In AMULET, this can be solved, as tags can support ciphertext update if a tag was notified that a reader was compromised. The tag then can choose a new random number R_2' and update the original ciphertext

$$\text{ciphertext} = ATTR\,K \oplus H(EK_i) \oplus K \oplus f(s_i \| K \| R_2 \| H^K(s_i, R_2)) \text{ to}$$

$$\text{(updated) ciphertext} = ATTR\,K \oplus H(EK_i) \oplus K \oplus f(s_i \| K \| R_2' \| H^K(s_i, R_2')).$$

Then, the tag just notifies all but the compromised readers to update their (s_i, R_2) to (s_i, R_2') to finish the ciphertext update.

It is our advantage that even the random number R_2 is changed to R_2', all readers can still use $ATTR\,K \oplus H(EK_i)$ to query data. The query process remains the same. However, when a compromised reader receives

$$ATTR\,K \oplus H(EK_i) \oplus K \oplus f(s_i \| K \| R_2' \| H^K(s_i, R_2')),$$

since the compromised reader still stored unmodified (s_i, R_2) in its database, the compromised reader cannot retrieve *ATTR K*. The data remains secure and private.

6 Conclusion

In this paper, we present the *ARIES* architecture to solve the distance limitation problem in RFID applications by utilising RFID-aware sensor nodes to monitor distant targets. We also propose an authentication protocol, *AMULET*, which mutually authenticates readers and tags. *AMULET* can resist man-in-the-middle attacks and reduce re-authentication overhead. *AMULET* can also provide a fixed-index search query among different servers. This property brings a higher level of efficiency to data search. To invoke queries on encrypted data, an SSP is proposed, which searches secrets in an encrypted form without the need to decrypt it. SSP is feasible for sensor nodes and RFID tags, as it uses low-computation operations, i.e., hash function and random number generating function, to encrypt and search data. SSP prevents the disclosure of information during the transmission or search process. SSP supports ciphertext update that compromised readers cannot decrypt the encrypted data but authorised readers still can recovery original data. In this way, all readers do not need to change the way to invoke queries. Furthermore, SSP uses a key chain to improve data security. As the security analysis shows, even if some attributes are compromised, the rest of attributes remain private.

Acknowledgements

This work was supported in part by NSC, ITRI, III, Chung Shan Institute of Science and Technology, the International Collaboration for Advancing Security Technology (iCAST), Investigation Bureau of Taiwan, Taiwan Information Security Center (TWISC), and Chunghwa Telecomm., respectively.

References

Alon, N., Galil, Z. and Yung, M. (1995) 'Efficient dynamic-resharing verifiable secret sharing against mobile adversary', *Proceedings of European Symposium on Algorithms*, Spain, pp.523–537.

Ballard, L., Green, M., de Medeiros, B. and Monrose, F. (2005) 'Correlation-resistant storage via keyword-searchable encryption', *Proceedings of EUROCRYPT*, Denmark, pp.457–473.

Bennett, K., Grothoff, C., Horozov, T. and Patrascu, I. (2002) 'Efficient sharing of encrypted data', *Proceedings of the 7th Australian Conference on Information Security and Privacy*, Australia, pp.107–120.

Chor, B., Goldreich, O., Kushilevitz, E. and Sudan, M. (1998) 'Private information retrieval', *Proceedings Journal of the ACM*, pp.965–981.

Chow, S.S.M. (2005) 'Exclusion-intersection encryption and its application to searchable encryption', *Proceedings of EUROCRYPT*, Denmark, pp.141–158.

Clarke, I., Sandberg, O., Wiley, B. and Theodore, T.W. (2000) 'Freenet: A distributed anonymous information storage and retrieval system', *Proceeding of the ICSI Workshop on Design Issues in Anonymity and Unobservability*, California, USA, pp.311–320.

Dabek, F., Brunskill, E., Kaashoek, M.F. and Karger, D. (2001) 'Building peer-to-peer systems with chord, a distributed lookup service', *Proceedings of the 8th Workshop on Hot Topics in Operating System,* Germany, May, p.81.

Devanbu, P.T. and Stubblebine, S.G. (2002) 'Stack and queue integrity on hostile platforms', *Proceedings of IEEE Transactions on Software Engineering*, pp.100–108.

Eschenauer, L. and Gligor, V.D. (2002) 'A key-management scheme for distributed sensor networks', *Proceedings of the 9th ACM Conference on Computer and Communication Security*, USA, pp.41–47.

Estrin, D., Govindan, R., Heidemann, J. and Kumar, S. (1999) 'Next century challenges: scalable coordination in sensor networks', *Proceedings of the 5th annual ACM/IEEE International Conference on Mobile Computing and Networking*, Washington, USA, pp.263–270.

Feldman, P. (1987) 'A practical scheme for non-interactive verifiable secret sharing', *Proceedings of the 28th IEEE Symposium on Foundations of Computer Science*, USA, pp.427–438.

Frank, J., Cheeseman, P. and Stutz, J. (1992) 'On the complexity of blocks-world planning', *Proceedings of Artificial Intelligence*, pp.139–403, CA, USA.

Generiwal, S., Kumar, R. and Srivastava, M.B. (2003) 'Time-sync protocol for sensor networks', *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, USA, pp.138–149.

Gertner, Y., Ishai, Y. and Kushilevitz, E. (1998) 'Protecting data privacy in private information retrieval schemes', *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, Dallas, USA, pp.151–160.

Gu, C., Zhu, Y. and Zhang, Y. (2006) 'Efficient public key encryption with keyword search schemes from pairings', *Proceedings of EUROCRYPT*, Saint Petersburg, Russia, pp.372–383.

Huang, S-I. and Shieh, S. (2005) *Secret Searching in Wireless Sensor Networks with RFIDs, Information Security Conference*, Taiwan, pp.237–244.

Kusilevitz, E. and Ostrovsky, R. (1997) 'Replication is not needed: single database, computationally-private information retrieval', *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, Florida, USA, pp.364–373.

Ledlie, J., Taylor, J., Serban, L. and Seltzer, M. (2002) 'Self-organization in peer-to-peer systems', *Proceedings of 10th SIGOPS European Workshop*, Saint-Emilion, France, pp.125–132.

Molnar, D. and Wagner, D. (2004) 'Privacy and security in library RFID issues, practices, and architectures', *Proceedings of ACM Conference on Computer and Communication Security*, Washington, USA, pp.210–219.

Ostrovsky, R. and Skeith III, W.E. (2005) 'Private searching on streaming data', *Proceedings of Advances in Cryptology*, Aarhus, Denmark, pp.314–328.

Pottie, G.J. (1998) 'Wireless sensor networks', *Proceedings of Information Theory Workshop*, Texas, USA, pp.139–140.

Song, D., Wagner, D. and Perrig, A. (2000) 'Practical techniques for searches on encrypted data', *Proceedings of IEEE Symposium on Security and Privacy*, pp.44–55, California, USA.

Sun, H-M. and Shieh, S-P. (1994) 'On dynamic threshold schemes', *Proceedings of Information Processing Letters*, pp.201–206.

Sun, H-M. and Shieh, S-P. (1996) 'An efficient construction of perfect secret sharing schemes for graph-based access structures', *Proceedings of Computers and Mathematics with Applications*, pp.129–135.

van Greunen, J. and Rabaey, J. (2003) 'Lightweight time synchronization for sensor networks', *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications* San Diego, USA, pp.11–19.

Wagner, D. (2004) 'Resilient aggregation in sensor networks', *ACM Workshop on Security of Ad Hoc and Sensor Networks*, Washington DC, USA, pp.78–87.

Waters, D., Balfanz, D., Durfee, G. and Smetters, V. (2004) 'Building an encrypted and searchable audit log', *Proceedings of 11th Annual Network and Distributed Security Symposium (NDSS)*, San Diego, USA.

Weis, S.A., Sarma, S.E., Rivest, R.L. and Engels, D.W. (2004) 'Security and privacy aspects of low-cost radio frequency identification systems', *Proceedings of Pervasive Computing*, pp.201–212.

Zheng, Y., Hardjono, T. and Seberry, J. (1992) 'How to recycle shares in secret sharing schemes', *Proceedings of Austral Computer Science* Communications, Australia, pp.1053–1064.