Public key cryptography based privacy preserving multi-context RFID infrastructure ☆

Selim Volkan Kaya, Erkay Savaş [1], Albert Levi [*,2], Özgür Erçetin

Faculty of Engineering and Natural Sciences, Sabancı University, Orhanli, Tuzla, 34956 Istanbul, Turkey

Abstract

In this paper, we propose a novel radio frequency identification (RFID) infrastructure enabling multi-purpose RFID tags realized by the use of privacy preserving public key cryptography (PKC) architecture. The infrastructure ensures that the access rights of the tags are preserved based on the spatial and temporal information collected from the RFID readers. We demonstrate that the proposed scheme is secure with respect to cryptanalytic, impersonation, tracking, replay, and relay attacks. We also analyze the feasibility of PKC implementation on passive class 2 RFID tags, and show that the requirements for PKC are comparable to those of other cryptographic implementations based on symmetric ciphers. Our numerical results indicate PKC based systems can outperform symmetric cipher based systems, since the back end servers can identify RFID tags with PKC based systems approximately 57 times faster than the best symmetric cipher based systems.

Keywords: RFID; Privacy; Security; Public key cryptography; Spatio-temporal attacks

1. Introduction

Remote identification of objects based on radio signals is welcomed with an enthusiastic acceptance in various numbers of applications due to its ease of use and efficiency. Compared with previous technologies for object identification such as barcodes and smart cards, radio frequency identification (RFID) does not require the objects to be in the line of vision. The amount and variety of information that can be stored in an RFID tag are unimaginable in the traditional technologies. These features render the use of RFID tags as popular (and inevitable to a great extent) in large and diverse set of applications such as supply chain, toll collection, payment

☆ A preliminary version of this work has been presented at IFIP Networking 2007 Conference, held in Atlanta, Georgia, May 14–18, 2007.

* Corresponding author. Tel.: +90 216 483 9563; fax: +90 216 483 9550.

E-mail addresses: selimvolkan@su.sabanciuniv.edu (S.V. Kaya), erkays@sabanciuniv.edu (E. Savaş), levi@sabanciuniv.edu (A. Levi), oercetin@sabanciuniv.edu (Ö. Erçetin).

[1] Erkay Savaş is supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under project number 105E089.

[2] Albert Levi is supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under project number 104E071.

tokens, etc. A common characteristic of these RFID-based applications is that the tags are used for a single purpose and in a single context, in the sense that only designated readers can challenge/query the tags. This does not necessarily prevent other unauthorized readers from participating in privacy-violating activities such as tracking the movements of the tags, and hence, the individuals associated with them.

Therefore, the usage of RFID in a single context puts certain limitations on the versatility of the tags while adding to the privacy problems. For instance, if an object needs to be identified by different readers for different purposes, multiple RFID tags are required for the same object. However, this approach is clearly not scalable, since not only attaching multiple tags increases the cost of the system, but also makes the management of multiple tags more difficult. Moreover, privacy breaches are more likely to occur with multiple tags, since the attacker has more opportunities to track the movement of the object.

In this paper, we propose a multi-purpose RFID infrastructure, where a single RFID tag is interrogated by various readers for different purposes. This infrastructure is more flexible, since in real life an object does not have a single purpose in a single isolated context, but it is related to multiple parties in some way as a result of cooperative and collaborative structure of the society. For example, consider the example given in Fig. 1, where an RFID tag is used to identify the individuals. In this example, the same RFID tag can be queried in different sites for different purposes. The police department should be able to identify each person to find out whether that person has a crime record or not. The hospital should be able to identify each person in case of health emergency to learn about previous health record of that person. The security department of a building should identify the person to decide whether to give her access to the building. We can easily extend this example with tens of different usage scenarios, where the identification of a person or object (e.g., car) is needed. From the scenarios outlined above, each party with a reader queries the RFID tag to retrieve information of interest for the person or object whose identification information is stored in the tag. Clearly, there must be some rules and limitations that govern the kind of information, and the circumstances under which this information can be obtained for the individuals.

As demonstrated in the previous example, increasing the versatility of RFID tags by enabling multipurpose access by different parties emphasizes the importance of privacy. Note that RFID tags can provide access to a large amount of private information which may be compromised if the access rights of each querying party are not clearly defined, and if the RFID infrastructure is vulnerable to security attacks. In this work, we address several security

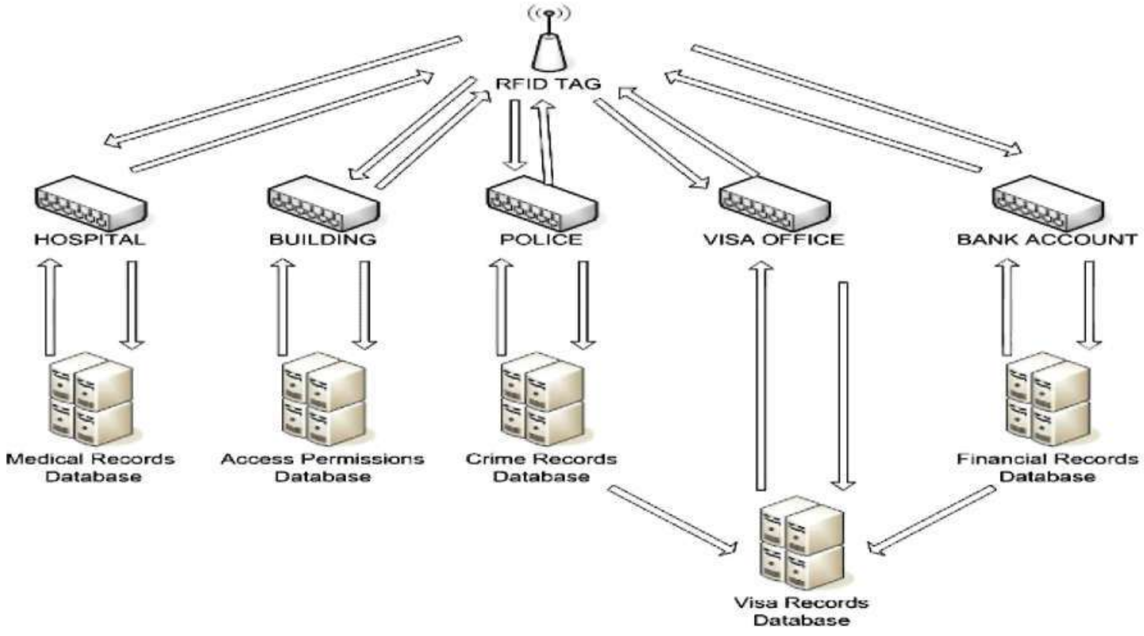


Fig. 1. Multi-context RFID infrastructure.

attacks, and show that these attacks can be prevented by using our proposed multi-purpose RFID infrastructure. Some of the attacks considered include *traceability, impersonation of readers and tags, replay, relay, spatio-temporal, and cryptanalytic attacks.*

Public key cryptography (PKC) is a powerful technique which proves to be indispensable for the proposed infrastructure, where the aforementioned attacks for RFID systems can be thwarted without an adverse effect on the scalability of the overall system. However, PKC is known for its excessive need for resources. However, RFID tags are limited in resources. Nonetheless, PKC can still be implemented in RFID applications provided that the PKC algorithms are efficiently realized in hardware while taking into account the power and chip area constraints. To this end, we show in Section 4 that NTRU public key cryptosystem [11] offers great advantages in terms of power, memory requirements, performance results, and security level. Thus, NTRU is a viable solution even for Class 2 RFID tags.

The paper is structured as follows. In Section 2, previous related work is summarized. The proposed infrastructure, an authentication protocol, and their security analysis are given in Section 3. The feasibility analysis of using PKC in the proposed protocol is provided in Section 4. Section 5 presents the performance analysis and throughput figures. Section 6 provides a discussion on the different aspects of the proposed infrastructure. The paper concludes in Section 7.

2. Related work

Different methods proposed in the literature that aim to provide privacy in RFID systems basically take one of the three main approaches. In the first approach hash functions are used for their low-cost and computational efficiency. Hash lock scheme [1] is the most primitive representative of this approach and it works by locking RFID tag by the time it receives required response from the interrogator. However, the hash lock scheme fails to prevent tracking since the hashed values of tag IDs can be used as metaIDs, which remain the same over time, to trace each tag. To prevent tracking, randomized hash lock scheme [1] is introduced, where the hashed value of tag ID is changed for each read request by concatenating a random value to tag ID before hashing. The main drawback of this scheme is that reader should try all possible tag IDs with the random number, sent with the hash value, to identify the correct tag. Both hash lock and randomized hash lock schemes provide no forward secrecy, since if a tag is compromised all previous communications of that tag can be determined. To prevent tracking and provide forward secrecy, hash chain scheme [2] is introduced. However, hash chain scheme requires strict synchronization between tag and reader which makes it vulnerable to de-synchronization attacks [17].

The second approach is to use a tree structure [24] to store secrets for each tag. In the tree structure, a tag which has a unique path to the root, keeps multiple secrets defined by this unique path. In the tree structure, the secrets of the interrogated tag could be obtained by using depth-first search. However, the tree structure results in an overlap among the secrets of tags in the system [4]. Thus the compromise of a single tag may cause to reveal secrets of other tags.

The third approach is to use symmetric encryption for challenge–response based authentication as defined in [25]. As reported in [7], symmetric encryption is feasible in RFID systems. However, using symmetric keys comes with a price. The compromise of the secret key even in one tag affects the whole system if every tag uses the same secret to authenticate itself [3]. On the other hand, if each tag uses a different secret, as in the challenge–response protocol proposed in [24], then the problem of matching a secret with a tag during the authentication process becomes computationally prohibitive in large systems with many tags. This is due to the fact that brute force search is needed to search the entire space of secrets [4]. Therefore, we propose a PKC based infrastructure and show in the subsequent sections that it offers significant scalability and security advantages.

3. Our approach

The following sub-sections present a detailed explanation of the multi-context RFID infrastructure, proposed protocol definition, and security analysis of our approach for the most common attacks that can be applied in RFID systems.

3.1. Multi-purpose RFID infrastructure

Our objective is to use a single RFID tag for identification in several different applications. For that purpose, we propose to use a decentralized RFID infrastructure that consists of several sub-domains, where each sub-domain is responsible for only one application with a trusted backend server. The backend server is used to store the information related to that sub-domain. The information kept in each sub-domain corresponds to a set attribute values related to the RFID tag. For instance, if we revisit the example given in Section 1, a person has data on her legal, health, social and financial records kept on corresponding trusted back-end servers of the sub-domains. Before a reader can query an RFID tag, the reader is subscribed to the related sub-domains based on the interests and roles of the reader. A reader may belong to multiple sub-domains, e.g., in Fig. 1, the reader in the visa office may want to access data related to financial and legal sub-domains. Once the reader is subscribed to a sub-domain, the access control list of the sub-domain is updated to include the new reader. As described later in this section, the access control lists define the rights of the readers based on spatio-temporal constraints and the roles of the readers.

The aforementioned multi-context RFID infrastructure has several additional advantages. First, the partitioning of data among sub-domains reduces the security risk if one of the sub-domains is compromised. Second, the management of data is easier in a decentralized system, since the size of data is smaller for each sub-domain. Finally, access control is much simpler in the decentralized model, since each sub-domain has a limited portion of the overall information and the readers subscribe to the sub-domains according to their roles.

The access control model is based on the following three pieces of information: (1) the authenticated ID of the reader, (2) verifiable location of the reader during tag interrogation, and (3) verifiable time of the interrogation. During the interrogation process, the reader should present these three pieces of information to the backend server. The authentication of the reader to the backend sever can be done in a straightforward manner using symmetric or public key cryptography, which is not explicitly shown in the protocol. The reader can obtain the verifiable time and location information in several ways. If the RFID tag is mounted on an object, where the resources are abundant, such as automobiles, this information can be supplied by the tag itself. In such environments, an active tag can be used such that it maintains a built-in clock and can get the location information from a nearby GPS device. If the tag is a passive device and there is no means of getting location information, the infrastructure must provide the verifiable location and time service. If the reader is a wireless device connected to mobile (cellular) network, time and location information of the reader during tag interrogation can be supplemented by the network within certain limits of precision.

In our model, it is assumed that the readers share symmetric and pairwise keys with backend servers before they are set up. These keys are needed to prove readers' credentials to interrogate a tag to the backend server.

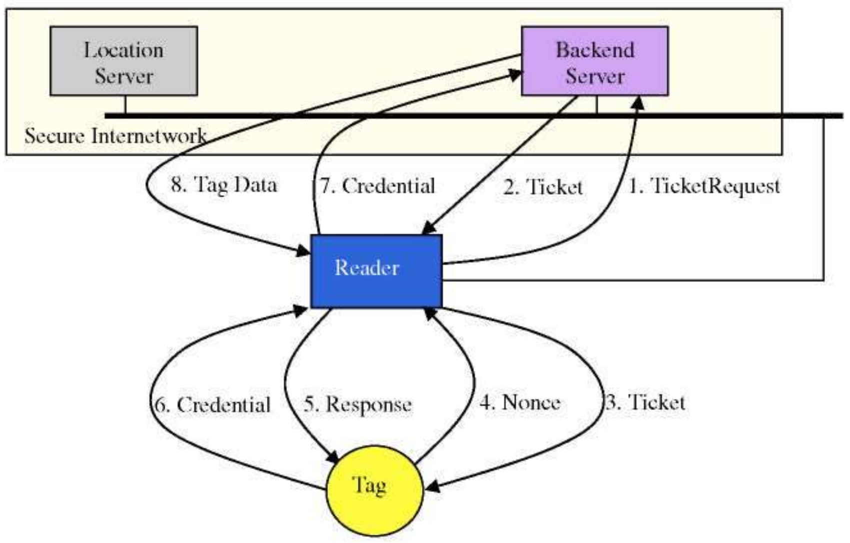
Since the backend servers are trusted in our model, they share the same public key and private key pair. The tags know the public key of the backend servers. In the protocol definition below, we only showed one backend server for the sake of simplicity, and we assume that there is a secured internetwork connecting the backend servers.

3.2. Protocol definition

Fig. 2 outlines the proposed protocol that shows the communication between tag, reader, and backend server in this architecture. The numbers in the figure indicate the order of steps. The protocol can be divided into three phases as explained below.

Phase I: The reader acquires a ticket from the backend server to query tags in its reading range to obtain data pertaining to them. To do so, first the reader sends a ticket request to the backend server (message 1 in Fig. 2). Then, the backend server issues and sends the *Ticket* to the reader (message 2). *Ticket* is the encryption of reader ID, current time and location of the reader under a secret key S known only to the backend server. The backend server obtains the reader location from the location server in a secure way. Since the reader is a self-powered device and it is connected to the network, the location server can locate the reader when necessary. The reader can use the same ticket to query different tags for a certain period of time, as specified by the backend server according to reader ID, its location and the query time. Thus, the reader does not have to repeat the first two steps each time it wants to query tags.

Phase II: The reader sends the *Ticket* (message 3 in Fig. 2) to start the query process. The tag responds by sending a randomly generated *Nonce* (message 4). The *Nonce* is a challenge for the



Ticket:	$E_S(RID+ReaderLocation+TicketIssuanceTime)$	[+: concatenation]
Nonce:	a random number serves as a challenge	
Response:	$E_R(Nonce)$	
Credential:	$E_{pub}(Ticket+TID+Nonce+E_R(Nonce))$	
RID:	Reader ID	
TID:	Tag ID	
S:	A secret key used by backend server to generate (encrypt) and decrypt *Ticket*	
R:	Key shared between the reader and the server	
pub:	Public key of server	
priv:	Private key of server	

Fig. 2. Multi-context RFID protocol definition.

reader. The reader responds the challenge by *Response* (message 5). *Response* consists of the encryption of the *Nonce* under the reader's secret key, *R*, which is known only to itself and to the backend server. This challenge response part of the protocol is to prove the reader's claimed identity and its correspondence with the tag which in turn proves the reader's and tag's location when combined with *Ticket* in the rest of the protocol; such a proof will be needed for the backend server in Phase III as described below. Message 5 should be returned to the tag within a certain period of time. Otherwise, the transaction is aborted by the tag. This rule is enforced in order to resist against relay attacks as described in Section 3.3. In the final stage of the Phase II, the tag encrypts its ID, *TID*, *Nonce*, E_R (*Nonce*), and *Ticket* using the public key of the backend server, *pub*, and sends it to the reader (message 6). The resulting ciphertext is called *Credential*, and used by the reader to access to the data pertaining to the tag in the backend server in Phase III. The tag verifies neither the *Ticket* nor the readers' *Response* before issuing the *Credential*. This is a design decision in order to take the burden of costly cryptographic operations off the tag. This may seem insecure at the first glance. However, the necessary verifications are performed by the backend server in Phase III and the security of the entire system is achieved. A detailed security analysis of the protocol is given in Section 3.3.

Phase III: The reader submits the *Credential* to the backend server (message 7 in Fig. 2). After that, the backend server performs some cryptographic operations as described in Fig. 3. The backend server first decrypts *Credential* using its private key and obtains the *Ticket*, tag ID (*TID*), *Nonce* (challenge) and *Response*. Moreover, the server decrypts *Ticket* using its secret key *S* to obtain the reader ID, *RID*, and location and time information in *Ticket*. Using *RID*, the server looks up the key *R*, which is shared between the reader and itself, and verifies the challenge/response pair in order to make sure that the reader has contacted the tag. Other than these cryptographic operations, the backend server should also make a validity check of *Ticket*. To do so, the server notes the timeframe between the reception of *Credential* and the time information in *Ticket* as the possible interval of reader's contact to the tag. If the length of this timeframe is small enough to assure that the reader remained around the location given in *Ticket*, then as a final step the backend server makes an access control decision. Using *TID*, *RID*, location and timeframe information, the backend server selectively discloses the tag data to the reader (message 8 in Fig. 2) based on the reader's access rights, which may vary depending on the timeframe of the query and location of the reader and the tag.

As a result of this process, the backend server makes sure about the location of the reader and consequently the location of the tag since the reader proves to the backend server that it contacted the tag via the challenge–response part of the protocol. Moreover, the backend server makes sure that the reader-to-tag contact has been performed within the timeframe between the issuance of *Ticket* and reception of *Credential*. Here there is an implicit assumption that this timeframe should be small enough such that the reader's mobility does not allow it to move away from the location at which it obtained *Ticket*. This assumption can be relaxed via a constant monitoring of reader during the lifetime of *Ticket*, which is set and controlled by the backend server. If the lifetime of *Ticket* is large, then the backend server interrogates the location server to obtain the location information of the reader within the timeframe between *Ticket* issuance and *Credential* reception. If the reader remained around a location which is in proximity of the location information in *Ticket* and the length of the timeframe between the reception time of *Credential* and the time information in *Ticket* is within allowable limits, then the backend server continues with the access control decision as described above.

3.3. Security analysis of the protocol

In this section, we analyze security of our protocol against spatio-temporal, replay, impersonation, relay and cryptanalytic attacks, and tracking.

3.3.1. Assumptions

The protocol assumes that there are trusted location servers that can either track the readers or locate them when asked by the backend servers. The protocol uses a public key encryption scheme to form the credential for the reader in Phase II of the protocol. Although public key cryptography is known to demand large resources in subsequent sections we demonstrate that a certain class of public key algorithms can efficiently be used in our scheme. It is further assumed that only the tags know the public key of the back-end servers, and each reader shares a pairwise key with each backend server. We also assume that there is a dedicated secure and authenticated channel between a reader and backend server, and opponents cannot clone a reader, which is a reasonable assumption since it is always possible to build tamper-proof hardware to protect confidential information (e.g., secret keys) in the reader.

3.3.2. Spatio-temporal attacks

A reader can access to data pertaining to a tag at certain locations and at certain times of the day. For instance, a reader in a hospital should not be able to interrogate a tag outside the hospital. Otherwise dif-

Step 1: $\mathrm{D}_{priv}(\mathit{Credential}) = \mathrm{D}_{priv}(\mathrm{E}_{pub}(\mathit{Ticket}+\mathit{TID}+\mathit{Nonce}+\mathrm{E}_R(\mathit{Nonce})))$
$= \mathit{Ticket}+\mathit{TID}+\mathit{Nonce}+\mathrm{E}_R(\mathit{Nonce})$

Step 2: $\mathrm{D}_S(\mathit{Ticket}) = \mathrm{D}_S(\mathrm{E}_S(\mathit{RID}+\mathit{ReaderLocation}+\mathit{TicketIssuanceTime}))$
$= \mathit{RID}+\mathit{ReaderLocation}+\mathit{TicketIssuanceTime}$

Step 3: Look up *R* from the database and
check if $\mathrm{D}_R(\mathrm{E}_R(\mathit{Nonce})) == \mathit{Nonce}$ obtained in Step 1

See Figure 2 for the notation used.

Fig. 3. Cryptographic operations performed by the backend server in Phase III of the proposed protocol.

ferent readers may combine information pertaining to a tag to compromise its privacy. Moreover, access rights of the reader may change depending on the time and its location.

The location of a reader can be determined by specialized location servers. If the reader is connected to a cellular network, methods such as radiolocation or triangulation can be used to estimate its location. A secure and efficient method that can be applied in any wireless network uses an approximate location estimation technique based on the distance-bounding protocol [18]. Clearly, the spatial access rights of a reader are limited by the precision provided by location service. The mobility of the reader may invalidate the location estimation after a certain time period. Thus, the location estimation should be repeated periodically. The measurement validity period is determined depending on how mobile the reader is and it is known by the reader and the backend server.

The reader uses the location information and measurement time to obtain data pertaining to a tag. The reader needs not see the location information and measurement time, and furthermore the reader must be prevented from counterfeiting them. Therefore, these two pieces of information along with the reader ID is encrypted by the key of the backend server and passed to the reader in *Ticket*. Therefore, a malicious reader cannot modify and fabricate a ticket or pass it to another reader.

A reader with a legitimate ticket queries a tag, which, in turn, embeds *Ticket* in *Credential* that is sent back to the reader. The reader delivers *Credential* to the backend server to access the data pertaining to the queried tag. The server compares the reader ID, location and measurement time against the access rights of the reader. There may be several cases when the backend server denies access to the reader: (1) reader has no access rights at all, (2) the reader has no rights in the specified location as written in the ticket, and (3) the measurement time of the location has expired.

3.3.3. Impersonation attacks

A malicious reader cannot impersonate another reader and get access to information pertaining to a tag, to which is not allowed, since there is a secure and authenticated channel between the reader and the backend server. Our protocol also prevents impersonation of tags since no tag ID is sent in clear text, but encrypted within the credential with public key of the backend server. Thus, no malicious tag, in collaboration with a malicious reader, can claim to be an honest tag as tag IDs are known only to the backend servers and to the tags. Note that tag IDs must be generated in a secure way so that it is not feasible to fabricate one by unauthorized parties.

3.3.4. Replay attacks

The proposed protocol is safeguarded against replay attacks by means of measurement time of location that serves as a timestamp and random nonce values that change in every tag query. A reader, however, can use the same credential repeatedly until the measurement time expires, which is, in fact, one of the aims of the proposed protocol. A reader cannot eavesdrop and use a credential intended for another reader since the credential contains ID of the intended reader. As mentioned before the readers maintain secure and authenticated channels with backend server, and thus, they cannot impersonate other readers.

3.3.5. Traceability

The fact that a reader or a group of collaborating readers is able to track a tag violates the privacy of the tag. If tag ID is sent in clear text or the tag responds to the reader's queries always in the same way (e.g., sending the same nonce or same credential in messages 4 and 6), then readers can violate the privacy by tracking the tags. However, tag IDs are always encrypted, and both nonce and credential change in every transaction in a nondeterministic way. Only way that a reader can track a tag is when it collaborates with the backend server. In our assumptions, the backend server is trusted and when it is compromised all system security is lost.

3.3.6. Relay attacks

A relay attack can be applied by employing two connected readers: (1) a legitimate reader located inside the authorized area, where it can query tags, and (2) a dummy reader located outside this area, where the legitimate reader has no rights to interrogate tags. The legitimate reader establishes a secure channel with the backend server while the tag is in the reading range of the *dummy* reader. The attack scenario is depicted in Fig. 4. One can easily observe that the legitimate reader can deceive both the backend server and the tag into believing that it is in its authorized region for interrogating the tag with the assistance of a dummy reader whose only

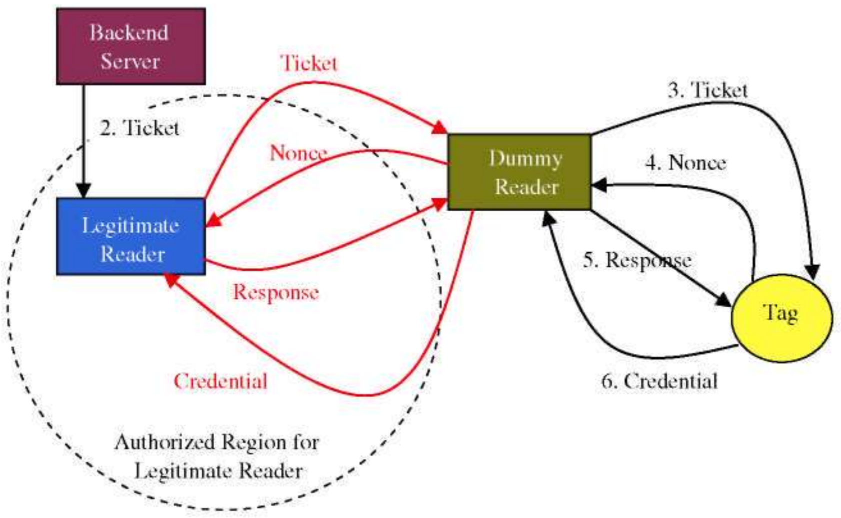


Fig. 4. Relay attack scenario.

functionality is relaying messages between the tag and the legitimate reader.

The only way to prevent this attack is to enforce the dummy reader to respond the challenge (*Nonce*) in message 4 within a given period of time, which should be just sufficient for the reader to encrypt the challenge and send the ciphertext (i.e., the response) back to the tag. Since the dummy reader has to relay the challenge to the legitimate reader, which in turn sends the response back to the dummy reader in a relay attack scenario, the response will take longer to arrive at the tag. Note that we cannot clone the legitimate reader and the dummy reader functions as a simple relay. The tag starts a counter as soon as it sends the nonce and stops it when it receives the response. Since ISO 15693 regulates that tag operates at 13.56 MHz and the communication speed between the tag and reader is 26.48 kbit/s, we can calculate how much it takes to send the Nonce and receive the Response. Sending messages 4 and 5 using 55-bit Nonce and 55-bit Response takes about 2.1 ms each while the encryption time of Nonce at reader is negligible comparing to this figure. For instance, in [26] 128-bit block encryption using AES is reported to take 400 clock cycles with 128 bit key on 32-bit RISC processor. Even with a very conservative 16 MHz MC68328 "DragonBall" processor, the encryption/decryption time is only about 25 μs. As an extreme case, we assume that the attacker employs a very fast hypothetical encryption engine which can perform encryption in 0 s. This gives the attacker 25 μs to relay Nonce and Response, which necessitates about 10 Mbit/s network speed between the dummy reader and the legitimate reader. The overhead due to communication primitives such as send and receive only add to the difficulty of mounting the relay attack. Apparently, this attack needs special equipment and cannot be applied using ordinary readers.

One difficulty of safeguarding the system against the relay attacks is to detect the disruptive attempts of the reader on the correct functioning of the tag. Since the tag relies on the reader for powering up its circuit, the reader can do two things while the tag is measuring the time elapsed between sending *Nonce* and receiving *Response*: (1) Shut off the power after it receives *Nonce* so that the tag cannot count in the mean time, and (2) Lower the frequency so that the tag slows down the counting process giving the dummy reader extra time to communicate with the legitimate reader. Clearly, these disruptive behaviors necessitate special equipment. Moreover, the tag can also take some precautions to thwart the former attack. The nonce generated by the tag is stored in a volatile memory while the tag is waiting for the response to the nonce. If the power is off meanwhile, the nonce in the volatile memory is lost, resulting in automatic abortion of the protocol. However, it is difficult to thwart the relay attack without the help of some point of reference for the time if the dummy reader slows down the frequency.

3.3.7. Cryptanalytic attacks

Malicious readers can mount these attacks using the tags as either encryption or decryption oracles and the tags under attack is usually unaware of the attack. In chosen plaintext attack (CPA), the reader chooses a plaintext and uses the tag to encrypt the chosen plaintext. Since the Credential is always a function of random nonce and therefore the reader has no liberty of choosing the plaintext, the CPA-based attacks are not efficient against the proposed scheme. Note that the readers are assumed not to know the public key of the backend server. If a reader, somehow, learns the public key, then it can mount CPA. However, in this case the resilience of the protocol against CPA is basically determined by the strength of underlying public key algorithm. A reader may try to apply CPA against the backend server by guessing the plaintext in Step 2 (e.g., asking for Ticket in a specific place at a specific time). The backend can detect this attack since the communication between the tag and reader is secure and authenticated, or the Ticket can also be randomized.

In chosen ciphertext attack (CCA), the reader chooses a ciphertext and uses the tag to decrypt it. This attack cannot be applied in our protocol since the tag does not perform decryption. The proposed scheme is also secure against adaptive chosen ciphertext attacks (CCA2) because of the same reason.

4. Public key cryptography for RFID systems

As showed in [13], the main problem in low-cost RFID is the scarcity of resources since low-cost RFID tags are passive devices with limited number of gates for application-specific computation, and the power induced from the electromagnetic field of the reader is usually low. In previous studies [14,15], applying public key cryptography is emphasized to be infeasible in low-cost RFID applications since they are known to use excessive amounts of memory, power, and space. In order to validate the applicability of our proposed scheme that utilizes public key cryptography, we provide a feasibility analysis with respect to these restrictions in the forthcoming section.

4.1. Power, space, energy and time requirements for the proposed scheme

In our feasibility analysis, we use the findings of [5], where state-of-the-art hardware implementations of several light-weight PKC algorithms are compared. In [5], three schemes with moderate level of security for RFID applications are analyzed: (1) Rabin's scheme [19], (2) elliptic curve cryptography (ECC) [20] and (3) NTRU cryptography [11]. For Rabin's scheme, a modulus of 512 bits is chosen, which provides a security level of around 60 bits according to [6]. ECC algorithm that is constructed over a prime field of 100 bits in size provides a security level between 56 and 60 bits. In NTRU algorithm, when the parameters are chosen as $(N,p,q) = (167,3,128)$, the algorithm provides a security level of around 57 bits [5].

The results of [5] clearly demonstrate that ASIC implementation of NTRU algorithm outperforms the other two algorithms. NTRU encryption requires a power of about 20 μW and 3000 NAND equivalent gates at the working frequency of 500 kHz. Thus, NTRU requires at least 7.75 times less power and 5.4 times less number of gates as compared to other algorithms. One drawback of NTRU cryptosystem is that it causes a ciphertext expansion of seven times compared to the corresponding plaintext.

In order to secure the proposed scheme against attacks, it is necessary to implement a true random number generator (TRNG) in the tag. The authors in [27] report that a TRNG can be realized using digital circuit artifacts such as meta-stability with only 300 NAND equivalent gates. Note that TRNG is considered as a standard feature for secure RFID operation regardless of the cryptographic algorithm used (public or secret key cryptography). In summary, the area needed to realize our protocol in an RFID tag, we only need about 3300 NAND equivalent gates, which is almost the same (even slightly less) as the AES realization proposed in [7] for RFID usage.

If we consider the Class 2 RFID tags with costs in the vicinity of 50¢ that have about 10,000 gates [28] and power requirement of 10 μW, the proposed scheme appears to be feasible even for the tags in this price range. Note that we can always halve the power requirement by reducing the operation frequency at the expense of higher execution times. One can argue that the space requirements exceed the area that can be allocated for cryptographic functionality since significant portion of logical gates will be required for basic tag functionality. Nevertheless, one might consider that Moore's law will ensure that there will be more processing power for the cryptographic functionality in coming years.

Table 1
Ticket (Spatio-temporal verification data) format

Reader ID	Location	Time	Total
59 bits	42 bits	27 bits	128 bits

4.2. Memory requirement for the proposed scheme

In this section, we determine the memory requirements for NTRU public key cryptosystem. As pointed out earlier, each RFID tag in the proposed infrastructure should store the public key of the backend server. If we use NTRU to provide moderate level of security as stated in the previous section, the parameters can be chosen as $N = 167$, $q = 128$, and $p = 3$ resulting in a public key of $167 \times 7 = 1169$ bits (146 bytes) stored in each RFID tag.

We also need to calculate the total length of data to be encrypted by RFID tag in step (6) of Fig. 2. We assume that the location data is represented in NMEA format [8] used in GPS systems. GPS data is ASCII coded and its total length is 70 bytes. To reduce length of this data format, we can extract latitude and longitude information which are adequate to define location of an RFID reader. The length of altitude and longitude data is 21 bits each, where 9 bits are used for degree, 6 bits for minute, and 6 bits for second. Therefore, we need 42 bits to store the location data. In the proposed scheme, the time precision of minutes is sufficient for all practical purposes and a data structure of 27 bits can be used to store time information: i.e., 7 bits for years, 4 bits for months, 5 bits for days, 5 bits for hours, and 6 bits for minutes.

We also assume that a reader ID of 59 bits can be used to identify all the readers in the system.[3] Consequently, we can conclude that a total of 128 bits is needed to form a Ticket that incorporates location and time data along with the reader ID (See Table 1). We assume that the backend server uses 128-bit symmetric key to encrypt spatio-temporal verification data resulting in a 128-bit ciphertext. Thus only one block of encryption would suffice to create the Ticket.

Upon receiving the ticket of 128 bits, the RFID tag generates the *Credential*. It concatenates tag ID of 96 bits, *Ticket* of 128 bits, *Nonce* of 55 bits, and *Response* (E_R (*Nonce*)) of 55 bits,[4] resulting in a 334-bit long plaintext. The tag encrypts the plaintext with the public key of the backend server using NTRU encryption algorithm. Encryption operation has to be done on two blocks since one block in NTRU is 167 (with chosen parameters (167, 3, 128)). Encryption operation results in a ciphertext of total $167 \times 2 \times 7 = 2338$ bits due to ciphertext expansion of seven times. Accordingly, the requirement for *volatile* user memory to perform cryptographic operations is found to be 2338 bits.

Considering that the NTRU public key is 1169 bits while the tag ID is 96 bits, we need 1265 bits of *nonvolatile* system memory for permanent data. These memory requirements are feasible with respect to current RFID technology for low-cost passive RFID tags [22]. Also advances in the memory technology like FRAM [7], which has the same cost as EEPROM but has more durability, speed, and lower power consumption, will make memory requirements even less of a concern in the near future.

4.3. Time and energy complexity of the proposed scheme

The proposed protocol consists of eight steps as shown in Fig. 2, and it also includes the process of getting verifiable time and location data. However, since the RFID tag is the only party with severely constrained resources, in our complexity analysis we only consider the time needed to interact with the tag, namely time spent on steps 3, 4, 5, and 6. The energy complexity of the proposed scheme is not explicitly analyzed as the energy requirements of the mentioned protocol steps and encryption operations are implicitly covered in time complexity analysis.

If we assume our scheme works with high frequency (HF) tags, operating at 13.56 MHz in accordance with ISO 15693, having ranges up to a meter or more [16], our tags will have data rate of 26.48 kbit/s uplink (tag to reader) and downlink (reader to tag) [23]. Therefore, the total duration of time covering the transmission of *Ticket, Nonce, Response,* and *Credential* (totaling 2576 bits) is 97.3 ms.

Encrypting one block (167 bits) of ciphertext with NTRU takes 58.45 ms at 500 kHz [5]. In the proposed algorithm the tag encrypts two blocks of plaintext, resulting in 116.9 ms total encryption time. Therefore, the total time spent on the protocol steps between the tag and reader takes about

[3] For all practical purposes, it is sufficient to use 59 bits for Reader ID assuming that the tag ID size is 96 bits [21].

[4] Fifty-five bits of ciphertext is sent as a response.

214.2 ms. If we use 250 kHz to reduce the power consumption during the encryption operation, the total time becomes 331.1 ms. Note that the frequency of 13.56 MHz must be divided for the encryption circuit to conserve power. Implementation details are given in Table 2.

4.4. Public key cryptography versus secret key cryptography in RFID tags

The foremost motivation of using public key cryptography in RFID-based systems is for better scalability when the privacy is of a concern. Pseudonym based schemes [2,24,30], where tags respond with different, random looking pseudonyms at each read, have been received wide acceptance in the research community. Informally speaking, a pseudonym is obtained by encrypting tag ID padded with a random string using a symmetric key algorithm (e.g., AES). For better security, each tag has a separate key rather than a single key used by all the tags in the system. In order to link a pseudonym to a tag ID, one must posses the secret key used in pseudonym generation process. Since the secret key is only known to tag itself and the backend server, and the pseudonym changes at every read in unpredictable manner, readers can link a pseudonym neither to a tag ID nor to any other pseudonym generated by the same tag. Consequently, readers rely on the backend servers to link pseudonyms to tag IDs.

One major drawback with pseudonym based schemes using symmetric key cryptography is that the backend servers have to perform decryption operation with the corresponding key to link a pseudonym to a tag ID. The backend server may have to try all possible keys to decrypt the pseudonym since every tag is stipulated to use a different key for security reasons, resulting in a serious scalability problem as the number of tags increase. Assuming that N is the number of tags, the number of symmetric encryption operations to be performed is in the order $O(N)$ in [2], $O(N^{2/3})$ in [30], and $O(\log N)$ in [24,29]; not to mention the storage requirements. Any information attached to or included in the pseudonym that facilitates the linking operation at backend server before decryption operation would also benefit readers for tracking tags. An obvious remedy for better scalability is to use public key cryptography, where tags generate pseudonyms by encryption with the public key of the backend server. When compared to symmetric key cryptography based systems, the advantage of public key cryptography is self-evident as shown in Table 3.

The scheme in [24] fails to scale well due to the fact that it necessitates at least 3 and possibly as many as $O(\log N)$ rounds of communication between tag and reader. The scheme in [29], on the other hand, requires that tag stores $O(\log N)$ keys and perform $O(\log N)$ symmetric key operations, which renders the original scheme impractical in RFID systems. The optimized scheme of [29] reduces tag storage, computation

Table 3
Comparison of proposed scheme to symmetric key based schemes

Scheme	Time complexity	Storage complexity
[2]	$O(N)$	$O(N)$
[30]	$O(N^{2/3})$	$O(N^{2/3})$
[24]	$O(\log N)$	$O(1)$
[29]	$O(\log N)$	$O(1)$
Proposed	$O(1)$	$O(1)$

Table 2
Implementation details of the proposed scheme

Memory requirement (bit)				
Volatile			Nonvolatile	
Credential = 2338 bit (=293 B)			Public key + tag ID = 1265 bit (=159 B)	
Area (# of NAND equivalent gates)				
NTRU encryption circuit + TRNG = ~3300				
Power requirements				
20 μW @ 500 kHz			10 μW @ 250 kHz	
Time requirements				
Communication time	Encryption time		Total	
97.3 ms	116.9 ms @ 500 kHz	233.8 ms @ 250 kHz	~214.2 ms @ 500 kHz	~331.1 ms @ 250 kHz

and communication overhead at the expense of more computation at the backend server. We substantiate this discussion using concrete figures on a realistic example below.

One other caveat before giving the example is about possible confusion due to Table 3. The comparison in the table would be at least unfair (if not misleading), if we failed to adjust complexity of a symmetric key operation with respect to that of a public key operation; the latter is known to be categorically much more expensive than the former. In what follows, we give a fair comparison of our scheme against symmetric cipher based systems.

In order to compare the performance of NTRU public key cryptosystem against that of a fast symmetric cipher, we use the performance results on NTRU decryption operation given in [34] and used the crypto++ package [31], which is the most widely known cryptographic library. Sosemanuk stream cipher algorithm, which is the fastest symmetric key algorithm in crypto++ package, turns out to decrypt 89 times faster than NTRU decryption.[5] Although this figure seems to favor the symmetric key cryptography, the NTRU cryptosystem is superior so far as the computation cost at the backend server is concerned. Assuming that there are 2^{20} (about a million) tags, the number of operations needed in proposed schemes and scheme in [29] is given in Table 4.

As depicted in Table 4, using public key cryptography significantly benefits the RFID system as far as the scalability is concerned.

Yet, we still need to show that NTRU encryption operation can be implemented in RFID tags and its implementation in ASIC consumes only comparable resources to those needed by symmetric key systems. To this end, we include a comparison of results of NTRU cryptosystem implementation against one of the state-of-the-art implementation of AES algorithm intended for RFID tags [7] in Table 5.

In terms of area and power requirements, the NTRU is, in fact, slightly better than AES. On the other hand, AES is significantly faster than NTRU operation as expected (about 29 times faster at 100 kHz). Nevertheless, the time spent on encryption operation in tag can be masked, if reader interrogates other tags meanwhile. Note also that the comparison in Table 5 is not exactly fair since the security levels of two algorithms, feature sizes used in the implementations, and operating frequencies are different. However, these two implementations represent the state-of-the-art and our main aim is to show that the requirements in two cases are comparable. For example, another NTRU implementation with higher security [33] (i.e., $N = 503$ is stronger than 4096 bit RSA) requires about 3000 gates with feature size 0.35 μm which is the same as [7]. However, since the work in [33] does not provide power requirement analysis, we did not include it in our comparisons.

[5] The performance comparison is estimated using raw encryption/decryption speed. For a conservative estimate, we do not take into account the initialization time of the stream cipher and access time for decryption keys, needed in case of symmetric ciphers.

Table 4
Comparison of proposed scheme to symmetric key based schemes

Scheme	Number of cryptographic operation	Adjusted number of cryptographic operations	Speedup over [29]
[29]	5120 Sosemanuk decryptions	5120	1
Proposed	1 NTRU decryption	89	57

Table 5
Comparison of NTRU against symmetric cipher in RFID tags

	NTRU [5]		AES [7]
Frequency	500 kHz	250 kHz	100 kHz
Power	20 μW	10 μW	12.2 μW
Time per encryption operation	58.45 ms (167 bit block)	116.9 ms (167 bit block)	10 ms (128 bit block)
Gate count	3000		3595
Feature size	0.13 μm		0.35 μm

5. Performance and throughput of proposed protocol

We now demonstrate that the proposed protocol can be implemented in conformance to ISO-IEC 18000-3 standard [32] and that the latency of cryptographic operation can be masked by reader interrogating other tags. The ISO-18000 reference protocol for inventory request, given in Fig. 5, takes 10.228 ms.

First, the reader sends an inventory requests (R: inventory) to determine the tags in the reading range. The tag waits TTA (0.3209 ms) seconds before sending its ID. In our protocol, the tag uses the Nonce to generate a temporary ID for the rest of

R: inventory	Time to Answer (TTA)	**T**: ID	Time to next Request (TTR)	**R**: Stay Quiet	TTR

Fig. 5. ISO 18000 reference protocol.

the protocol. The reader sends a Stay Quiet message and waits TTR seconds (0.3092 ms) before the next request. The total time spent on the reference protocol depends on the number of tags in its reading range. Collisions are managed by anti-collision protocol given in [32].

Once the reader gets the inventory, it can start the proposed protocol with any one of the tags. By adding appropriate TTA and TTR delays between the messages to the encryption and message transmission times in Table 2, tag-reader communication of our protocol (messages 3, 4, 5 and 6 of Fig. 2) takes approximately 215.1 ms at 500 kHz, and 332.1 ms at 250 kHz. These values give throughput of 4.65 and 3.01 tags per second (tags/s), respectively. However, we can increase this rate, if the reader idle time, during which tag performs NTRU encryption, is used to run the protocol for other tags in the reading range. In this case, the NTRU encryption time is masked and reader can

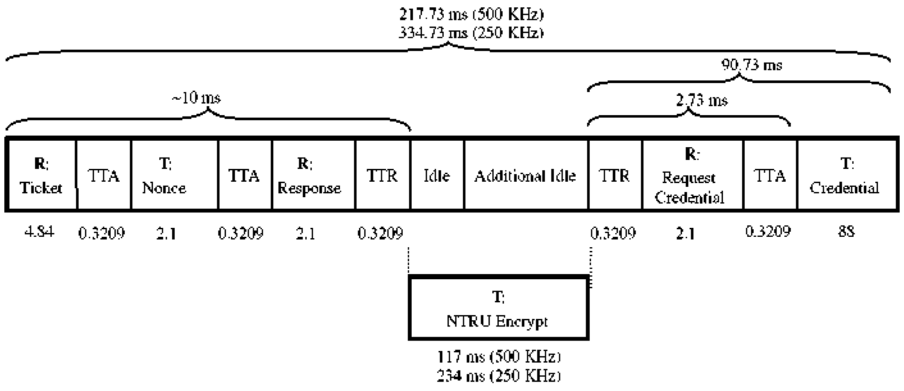
R: *Ticket*	TTA	**T:** *Nonce*	TTA	**R:** *Response*	**R:** Idle (**T**: NTRU encryption)

(a) Phase 1

TTR	**R:** *Request Credential*	TTA	**T**: *Credential*

(b) Phase 2

Fig. 6. Tag-reader communication in proposed protocol in ISO 18000 form.



For 500 KHz Clock Frequency:

117 ms NTRU Encryption time	
$\lfloor 117/10 \rfloor = 11$	Masked transactions (12 in total)
$90.73 \times 12 =$ 1088.76 ms	Sending time of 12 credentials
1088.76+10+117= 1215.76 ms	Total time for 12 tags
9.87 tags/sec	

For 250 KHz Clock Frequency:

234 ms NTRU Encryption time	
$\lfloor 234/10 \rfloor = 23$	Masked transactions (24 in total)
$90.73 \times 24 =$ 2177.52 ms	Sending time of 24 credentials
2177.52+10+234= 2421.52 ms	Total time for 24 tags
9.91 tags/sec	

Fig. 7. Performance and throughput of proposed protocol.

use this time to deal with other tags. To implement this idea, the tag-reader communication part of our protocol is divided into two phases as shown in Fig. 6. In Phase 1, the reader sends *Ticket*, the tag sends *Nonce*, and the reader sends *Response*. After that, the reader essentially gets into an idle state, but it can use this idle time to run Phase 1 with other tags. After the first tag finishes computing the credential, the reader goes into Phase 2 and receives the *Credentials* from all tags one by one. However, in order to obtain credentials from the tags one by one, the readers should send an extra *Credential Request* message that does not exist in the basic protocol for one tag.

Our analysis, which is given in the Fig. 7, showed that the abovementioned two-phase version of the protocol takes 217.73 ms at 500 kHz and 334.73 ms at 250 kHz for one tag case. We also observed that at 500 kHz, a reader can run the protocol with 12 tags in 1215.76 ms rendering a throughput of 9.87 tags/s. Interesting to note that when 250 kHz clock frequency is used for lower power consumption, the throughput becomes 9.91 tags/s, which is slightly better than 500 kHz case. As clearly seen form these throughput values, the time spent on public key encryption operation turns out to be less of a concern when there are several tags to query. Yet, the transmission time of the long credential has much more influence on the performance. The reason for a long credential is the ciphertext expansion in NTRU encryption operation. NTRU inventors claim that the message expansion can be avoided in certain situations. We leave the investigation of this claim and other solutions to obtain a shorter credential as a future work. Since we intend to use the proposed protocol for high-value merchandize (e.g., cars) or identification cards of individuals, we believe that the observed performance of our protocol is acceptable.

6. Discussions

As stated in [9], the main advantage of PKC is to use asymmetric keys for reader and tag communication in order to overcome the main security vulnerability of symmetric key encryption: use of one master key for both tag and reader. In symmetric encryption, compromise of an RFID tag and master key stored in that tag means compromise of the whole RFID system which depends on that secret. As demonstrated in our work, another important advantage of using PKC in RFID applications is that our proposed scheme provides a more scalable solution since the number of decryptions that the backend server has to perform is only one per transaction.

Conservative space and power requirements of NTRU with respect to legacy PKC schemes make it more appropriate for RFID applications. Originally, the NTRU is proposed to be implemented in software for RFID tags [10] considering the additional cost of adding cryptographic co-processor. However, software implementation of NTRU may not be suitable for applications, where high performance and less power consumption is needed provided that certain amount of increase in manufacturing cost is tolerable. Since hardware implementation of NTRU is possible with about 3000 gates and less than 20 μW power consumption [5], the additional cost of NTRU hardware to RFID tag is affordable. Additionally, the increase in manufacturing costs for NTRU specific hardware may be tolerable. This is due to the fact that efficient and secure PKC in RFID tags promotes the use of multi-purpose RFID tags, where a single powerful tag may replace several standard tags. In other words, an increase in quality of RFID tags allows decrease in the quantity of tags.

There are also some disadvantages of NTRU over legacy PKC schemes. First of all, the length of ciphertext can be up to seven times of the plaintext size as a result of NTRU encryption. Expansion of ciphertext means more data to be transmitted from tag to reader in step (6) in the proposed protocol, and hence longer transmission time. However Section 4.2 shows that the message expansion can be tolerated in RFID systems under realistic assumptions. NTRU inventors claim that the message expansion can be avoided if the Credential is formed as two parts, which are sent as separate packets. While the message expansion is still necessary for the first part, we can eliminate it from the second part. Further work is needed to assess the efficiency of this technique and to investigate other similar techniques. Secondly, NTRU will cause decryption failures which will occur with probability of 2^{-40} as stated in [12]. Decryption failure causes limitations on security analysis of NTRU, and attacks based on decryption failure can be performed on NTRU cryptosystem.

A malicious party may produce fraudulent clones of a tag by means of physical replication. We did not propose any solution to cloning attacks within our infrastructure.

The cryptographic key lengths that the state-of-the-art implementations of NTRU primitives [5] use can provide only moderate security level. As the use of RFID tags become more wide-spread, the key lengths will have to be longer in the near future. For example, Another NTRU implementation with higher security [33], where $N = 503$ (stronger than 4096 bit RSA), requires about 3000 gates. The only reason we did not include this particular implementation is that [33] does not provide power requirements, which is essential in our feasibility analysis.

Passive tags are vulnerable for certain relay attacks that can only be applied using specialized equipment discussed in Section 4. Active tags, which are used in high value merchandize and for individuals, with reliable source of power and clock, can easily thwart these attacks.

7. Conclusion

We proposed a privacy-aware multi-context RFID infrastructure that employs public key cryptography (PKC). In this infrastructure, different readers can interrogate RFID tags for different purposes. It is not possible for the readers to track RFID tags, therefore their privacy is preserved. During interrogation, tags encrypt their IDs with the public key of the backend server, which performs only one decryption to access the ID of the interrogated tag. In symmetric cipher-based schemes, the backend server has to try many symmetric keys since it cannot know the ID beforehand and choose the corresponding symmetric key. Therefore, employing PKC makes the proposed scheme more scalable compared to other symmetric cipher-based schemes.

We analyzed the feasibility of PKC for our protocol. We determined that with NTRU cryptosystem, PKC is suitable in the proposed protocol with power requirement of no more than 20 μW, chip area of about 3300 gates, user memory size of 293 bytes, system memory size of 159 bytes, and total execution time of 217.73 ms at 500 kHz (334.73 ms at 250 kHz).

We also addressed several security attacks such as impersonation, tracking, replay, cryptanalytic attacks and we showed that the protocol is secure against these attacks. In addition, we introduced a novel type of attack, for which we coined the term spatio-temporal attacks. In this type of attack, the malicious readers can try to interrogate tags beyond their authorized interrogation area and time interval. We showed that ordinary readers or group of readers cannot mount spatio-temporal attacks since special equipment with considerable resources is necessary to mount them.

References

[1] S. Weis, S. Sarma, R. Rivest, D. Engels, Security and privacy aspects of low-cost radio frequency identification systems, in: D. Hutter, G. Muller, W. Stephan, M. Ullmann (Eds.), International Conference on Security in Pervasive Computing – SPC 2003, LNCS, vol. 2802, Springer-Verlag, 2003, pp. 454–469.

[2] M. Ohkubo, K. Suzuki, S. Kinoshita, Cryptographic approach to privacy-friendly tags, in: RFID Privacy Workshop, MIT, 2003.

[3] A.J. Menezes, van P.C. Oorschot, S.A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997. Available online at <http://www.cacr.math.uwaterloo.ca/hac/>.

[4] G. Avoine, E. Dysli, P. Oechslin, Reducing time complexity in RFID systems, in: B. Preneel, S. Tavares (Eds.), Selected Areas in Cryptography – SAC 2005, LNCS, vol. 3897, Springer-Verlag, 2006, pp. 291–306.

[5] G. Gaubatz, J.P. Kaps, E. Öztürk, B. Sunar, State of the art in ultra-low power public key cryptography for wireless sensor networks, in: 2nd IEEE International Workshop on Pervasive Computing and Communication Security – PerSec 2005, Kauai Island, Hawaii, 2005.

[6] A.K. Lenstra, E.R. Verheul, Selecting cryptographic key sizes, Journal of Cryptology 14 (4) (2001) 255–293.

[7] M. Feldhofer, S. Dominikus, J. Wolkerstorfer, Strong authentication for RFID systems using the AES algorithm, in: M. Joye, J.J. Quisquater (Eds.), CHES 2004, LNCS, vol. 3156, Springer-Verlag, 2004, pp. 357–370.

[8] NMEA web site, 2006, Referenced 2006 at <http://www.kh-gps.de/nmea-faq.htm>.

[9] NTRU RFID data sheet, 2006. <http://www.ntru.com/products/NtruRFID.pdf>.

[10] NTRU RFID white paper, 2006. <http://www.ntru.com/products/RFID_White_paper_FNL.pdf>.

[11] J. Hoffstein, J. Pipher, J.H. Silverman, NTRU: a ring-based public key cryptosystem, in: J.P. Buhler (Ed.), Algorithmic Number Theory (ANTS III), Portland, OR, June 1998, LNCS, vol. 1423, Springer-Verlag, Berlin, 1998, pp. 267–288.

[12] N.H. Graham, P. Nguyen, D. Pointcheval, J. Proos, J.H. Silverman, A. Singer, W. Whyte, The impact of decryption failures on the security of NTRU encryption, in: Proceedings of the Crypto 2003, Santa Barbara, USA, 2003.

[13] D. Ranasinghe, D. Engels, P. Cole, Low-cost RFID systems: confronting security and privacy, in: Auto-ID Labs Research Workshop, Zurich, Switzerland, 2004.

[14] D. Ranasinghe, D. Engels, P. Cole, Security and Privacy: Modest Proposals for Low-Cost RFID Systems, Auto-ID Labs Research Workshop, Zurich, Switzerland, 2004.

[15] A. Juels, RFID security and privacy: a research survey, IEEE Journal on Selected Areas in Communication 24 (2) (2006) 381–394.

[16] G.P. Hancke, M.G. Kuhn, An RFID distance bounding protocol, in: Proceedings of IEEE/CreateNet SecureComm, 2005, pp. 67–73.

[17] T. Dimitriou, A Lightweight RFID Protocol to protect against traceability and cloning attacks, in: Proceedings of IEEE/CreateNet SecureComm, 2005, pp. 59–66.

[18] S. Capkun, J.P. Hubaux, Secure positioning of wireless devices with application to sensor networks, in: Proceedings of the 24th IEEE Conference on Computer Communications – INFOCOM, 2005.

[19] M.O. Rabin, Digitalized Signatures and Public Key Functions as Intractable as Factorization, Massachusetts Institute of Technology, 1979, MIT/LCS/TR-212.

[20] N. Koblitz, Elliptic curve cryptosystems, Mathematics of Computation 48 (177) (1987) 203–209.

[21] EPCglobal Inc. EPCTM generation 1 tag data standards version 1.1 rev.1.27, 10 May 2005, Referenced 2005 at <http://www.epcglobalinc.org>.

[22] EDN's news about advances in low-cost RFID technology 2006, Referenced 2006 at <http://www.edn.com/article/CA438036.html>.

[23] RFID-Handbook website 2006, Referenced 2006 at <http://www.rfidhandbook.de/rfid/standardization.html>.

[24] D. Molnar, D. Wagner, Privacy and security in library RFID: Issues, practices, and architectures, in: B. Pfitzmann, P. Liu (Eds.), Conference on Computer and Communications Security – CCS'04, ACM Press, Washington DC, USA, 2004, pp. 210–219.

[25] National Institute of Standards and Technology (NIST), FIPS-197: Advanced Encryption Standard, November (2001). Available online at <http://www.itl.nist.gov/fipspubs/>.

[26] D. Carman, P. Kruus, B. Matt, Constraints and approaches for distributed sensor network security, NAI Labs Technical Report #00-010, 2001.

[27] M. Epstein, L. Hars, R. Krasinski, M. Rosner, H. Zang, Design and implementation of a true random number generator based on digital circuits artifacts, in: Cryptographic Hardware and Embedded Systems – CHES 2001, LNCS, vol. 2779, Springer-Verlag, Berlin, 2001, pp. 152–165.

[28] S.E. Sarma, S.A. Weis, D.W. Engels, Radio frequency-identification security risks and challenges, CryptoBytes 6 (1) (2003).

[29] D. Molnar, A. Soppera, D. Wagner, A Scalable, Delegatable pseudonym protocol enabling ownership transfer of RFID tags, in: Workshop on RFID and Light-Weight Crypto, July 14–15, Graz, Austria, 2005.

[30] G. Avoine, P. Oeschlin, A Scalable Protocol for RFID Pseudonyms, in: IEEE Persec, 2004.

[31] W. Dai, Crypto++, a Free C++ Library for Cryptography, 2004, <http://www.eskimo.com/~weidai>.

[32] International Organization for Standardization (ISO), ISO/IEC 18000-3, Information Technology AIDC Techniques – RFID for Item Management, March 2003.

[33] C. O'Rourke, B. Sunar, Achieving NTRU with montgomery multiplication, IEEE Transactions on Computers 52 (4) (2003).

[34] NTRU Performance Comparisons, available at <http://www.ntru.com/products/toolkits_performance.htm>.

Selim Volkan Kaya is a Master of Science student in the Program of Computer Science and Engineering at Sabanc? University, Istanbul, Turkey. He has received his B.S. degree in Computer Science and Engineering from Sabancı University in 2004. His research interests include secure multi-party computation, privacy preserving data mining, security in ad hoc sensor networks, and Radio Frequency Identification (RFID) technology.

Erkay Savaş received the B.S. (1990) and M.S. (1994) degrees in electrical engineering from the Electronics and Communications Engineering Department at Istanbul Technical University. He completed the Ph.D. degree in the Department of Electrical and Computer Engineering (ECE) at Oregon State University in June 2000. He worked for various companies and research institutions before he joined Faculty of Engineering and Natural Sciences of Sabanci University as a faculty member in 2002. He is the director of the Cryptography and Information Security Group (CISEC) of Sabanci University. His research interests include cryptography, data and communication security, high performance computing, computer arithmetic, privacy preserving data mining, and distributed systems. He is a member of IEEE, ACM, the IEEE Computer Society, and the International Association of Cryptologic Research (IACR).

Albert Levi received B.S., M.S. and Ph.D. degrees in computer engineering from Boğaziçi University, Istanbul, Turkey, in 1991, 1993 and 1999, respectively. He served as a visiting faculty member in the Department of Electrical and Computer Engineering, Oregon State University, OR, between 1999 and 2002. He was also a postdoctoral research associate in the Information Security Lab of the same department. Since 2002, he is a faculty member of Computer Science and Engineering in Sabanci University, Faculty of Engineering and Natural Sciences, Istanbul, Turkey and co-director of Cryptography and Information Security Group (CISEC). His research interests include computer and network security with emphasis on mobile and wireless system security, public key infrastructures (PKI), and application layer security protocols.

Ozgur Ercetin received his B.S. degree in Electrical and Electronics Engineering from the Middle East Technical University, Ankara, Turkey in 1995, and his M.S. and Ph.D. degrees in Electrical Engineering from the University of Maryland College Park in 1998 and 2002, respectively. He worked for as a research assistant at University of Maryland and in HRL Laboratories, Malibu CA. He is currently working as an Assistant Professor at Sabanci University, Turkey. His research interests are in the field of computer and communication networks with emphasis on fundamental mathematical models, architectures and protocols of wireless systems, sensor networks, high-speed Internet, and satellite communications.