

Research Report

Version 1.0

Michael Fong
mcfong@iastate.edu

November 24, 2009

Abstract

Like many high-tech device, RFID system has become a necessary technology that every company would like to adopt. While this widely used technology brings convenience to our society, it also raises considerable threats among persons due to their privacy concerns. Many authentication protocols based on symmetric challenge-response scheme have been developed in order to ensure the privacy of the users is preserved. However, many of the existing schemes cannot protect tag privacy in the presence of compromised or malicious RFID readers. In this paper, we investigate the possible security and privacy threats to RFID system. We then propose and implement a new approach to authenticate smart tags without exposing their owners' identities and activities patterns, in the attacking scenario of various outsider attacks, and readers or tags compromises. Extensive implementation and experiments have also been conducted to validate the feasibility of the proposed method.

1 Introduction

Radio Frequency Identification (RFID) is a system that wirelessly enables massive identification and tracking of items. The first RFID tag was developed in 1973, implemented a passive transponder to unlock a door without a key. However, this device does not open up a commercial wealth, until with live stock tracking in 1990s'[1]. Without a standard in the 90s', several different and incompatible types of identifiers and protocols are used, depending on application developer and device vendor. In 1999, the AutoID center at the Massachusetts Institute of Technology was founded to guide and standardize the development of

RFID technology[2], which resulted in the adoption of the Electronic Product Code (EPC) as an international standard. In addition, the wireless nature with no line-of-sight requirement makes RFID ideal for massive inventory control and fast check out. Thus, each tag embedded with a unique identifier follows a standardized electronic product code (EPC), which is anticipated to serve on consumer product as a successor to the ubiquitous "UPC" bar code in near future. The making of EPC standard has benefited the RFID technology to be comprehensively adapted in a broad range of application nowadays, such as electronic toll collection for highways, inventory management, employee badges, and wireless electronic or credit card payment

Two components involved in the systems are RFID tags and readers. The tags, usually attached to item, contain a radio frequency transponder and a read-only (sometimes re-writable) memory chip that was preloaded with a unique identifier. Power source is typically used to distinguish passive and active tags. Active tags includes a small battery to transmit information directly to reader, whereas passive tags work by taking the energy received from the reader through an antenna and using that energy to transmit its secret data back to the reader. Passive tags are likely to be more widely used, because of their low cost production.

When tags get queried by readers, they respond with their unique identifier. After querying the tags, the readers usually transmit the data back to a back-end system, i.e. a database. The readers are designed to query multiple tagged items at once and distinguish between each one of them. Nevertheless, in practice, both parties are constantly exposed in the untrusted environment, due to unencrypted transmission, lack of data integrity, or mutual authentication, and thereby damaging the privacy[3].

1.1 Threats in Real World

Security aspects of RFIDs that work at different layer (Physical Layer[4], Communication Layer[5], or Application Layer [6, 7, 8]), have been proposed. We are presenting some applications for RFID tags and highlight their security concerns.

1. Among all application, the most well known is Supply Chain Management, which manufacturers, retailers, and logistics providers make unprecedented use of RFID technology to track, secure and manage items from the time they were raw materials through the entire cycle of the product. For instance, pharmaceutical industry in United States are making \$32 billion dollars, which represents 10 percents of global market. The recent increase of

counterfeit or diluted drugs has caught the attention of Food and Drug Administration (FDA), and considered as a threat to public health.[9] The RFID technology could immediately impact pharmaceutical supply chain safely through real-time, off line, and item-level authentication, from initially at the point of manufacturing to the stage of dispensing pharmacies.

2. RFID tags also have the potential on the individual level to store personal information that can be used for security check-in. For example, an employee carries an ID card, embedded with RFID chips, could be used to authenticate at the security entry in a high security facility. Another real-life example is the US passport. The US government issued the first passports containing RFID chips in October 2006. The embedded chips in the new passports stores the same personal information as those in the old printed document, including the name, nationality, sex, date of birth, place of birth, fingerprint, and a photo of the passport holder. According to government officials, the use of the RFID chip allows passports to be scanned and cross-referenced with security databases more easily. Due to the nature of communication in RFID, identity theft can be done wirelessly [10]. The personal information is just up there available in the air for hackers, who would hack into the device, snap personal information, and walk away. The consequence of unauthorized duplication of your passport would affect millions of Americans.

1.2 Security Threats

Security in RFID works quite different from conventional network security. Due to the hardware obstacle, it is difficult to directly employ the existing security approaches to the area of RFID networks. Most of the security concerns can be addressed by the services of confidentiality , availability and integrity. When we review a real system in practice, however, authenticity and privacy are also on our list of consideration.

1.2.1 Confidentiality

¹ Each party in the transumption should not leak data to any unauthorized parties (this illegal act is sometimes called skimming), therefore it is extremely important to build a secure channel in the system. Especially in a commercial application, the data stored in the RFID tag is sometimes highly confidential. The standard solution for keeping

¹Confidentiality refers to the concealment of information.

sensitive data secret is to encrypt the data with a secret key, known only to the sender and receiver. The receiver would then decrypt the data, and thus achieve confidentiality.

1.2.2 Integrity

² In the wireless environment, the information exchanged between two parties needs to be confidential when sensitive data, such as personal bio-matrix, must not be collected by an eavesdropper ³. Fortunately, with the implementation of confidentiality, the attacker may be unable to steal information. However, the adversary may modify the message in transit without knowing the message of content. For instance, the lack of authentication in the simple Diffie-Hellman key exchange protocol makes it vulnerable to man in the middle attack. Message authentication codes, hash functions and digital signatures can guarantee message integrity and as well as authenticity.

1.2.3 Availability

⁴

Availability is an important aspect of reliability, especially when a reader needs to be ready to authenticate every incoming user that may enter its communication range at certain time intervals. For example, the functionality of the network must be ensured to resist denial-of-service attacks (DoS). The typical countermeasures include Quality of Service (Qos).

1.2.4 Authenticity

In any network communication, authentication proves the claimed identity of of the other parties, and it is an important security measure for preventing counterfeiting behaviors. Both the reader and user need to confirm the identity of each party involved in the communication. The use of authentication may also be required in applications, such as security entry systems. In addition, a system equipped with strong authentication indicates a system of proving knowledge of a secret of the other party without revealing it.

²Data Integrity ensures that any received data has not been altered in transit.

³Eavesdropping is unauthorized listening/intercepting, through the use of radio receiving equipment, of an authorized transmission to monitor or record data between sender and receiver

⁴Availability refers to the ability to use the information desired

1.3 Threats against Privacy

Privacy, in general, refers to the ability of an entity to stop information about themselves from becoming known to people other than those whom they choose to give the information. In the world of wireless network, while this technology promises to produce a massive amount of data transmission, an adversary can use seemingly irrelevant fragment of data, assemble them, and derive much more sensitive information. Hence, adversaries does not have not be physically present to maintain surveillance, but they can gather information in a low-risk, anonymous manner.

1.3.1 Eavesdropping

By listening to the data, the adversary could easily discover the communication contents. When the traffic conveys the control information about the network configuration, which contains potentially more detailed information about the communication protocol, the eavesdropping can act effectively against the privacy protection.

1.3.2 Traffic Analysis

This is an advanced version of combination with monitoring and eavesdropping. An attacker could potentially monitor the increase in the number of transmitted packets between certain reader or users, and could signal that a specific tag has registered activity. Through the analysis on the traffic, some users with previous activities can be effectively identified. Thus, this act violate the identity privacy.

1.3.3 Camouflage

After an adversaries insert a rogue reader or compromise a legitimate user tags in the network, the attackers impersonate some seemingly normal tags to fool the readers.

authorized to.

2 Related Work

Weis[8] firstly proposed a hash-based scheme, called Hash Lock. Combined with a backend database to perform RFID authentication, each tag uses the hash of a random key as its metaID. When locked, a tag responds to all queries with its metaID to the reader. Then, the reader is able to obtain the real ID of the tag from the database via the looking up the received metaID. However, this scheme does not

provide forward security because the same metaID is used repeatedly. Therefore, Weis extends the original method to another randomized scheme, called Randomized Access Control, which employs a random number generator to prevent from the tracking attack . However, tag impersonation attack is also possible, since an intercepted response can be replayed.

Mulnar and Wagner[11] suggested a server-less authentication system that both parties use a shared secret and individually contribute a random number to protection the messages communicated in the channel. since the reader knows the shared secret, its own nonce, and previously tag-generated nonce, thus, the reader can obtain the tag ID. They have also built a tree-based protocol to provide scalable authentication with search complexity $O(\log n)$. Each tag represents a leaf nodes in the tree, and each edge is associated with a secret between two nodes. In addition, a tag may be loaded with multiple secrets corresponding to the path from the tag to the root. However, this protocol does not guarantee backward untraceability, especially when a reader was compromised. In that case, the adversary compromising the reader can learn the secret keys toe very tag the reader has access to. On the other hand, if a tag is compromised, the attacker can back trace the previous communication from this tag.

Similarly, Dimitriou[12] uses the similar secret-sharing authentication protocol, in which both reader and tags employs their own random nonce. In this challenge-response scheme, when a reader query comes in , the tag response with a hash of its identifier, which the reader gives to the secure server. After authenticating the reader, the secure server sends back the valid message to reader, and reader response this received message back to the tag. The tag will verify the message sent by the reader. If the value match, then the tag knows the reader has been authentication by the server, and updated its secret id. Otherwise, the tag remains the old secret id. One obvious vulnerability in Dimitriou’s protocol is attack against id synchronization. Moreover, the scheme is also prone to tag impersonation attack, because the the same hashed tag identifier could be reused between valid sessions.

Tsudik[13] describes a simple RFID authentication protocol, which he calls YA-TRAP (Yet Another Trivial RFID Authentication Protocol). The author aims this novel approach at preassumption that tag information is processed in batch, and additionally RFID tags have its own power source to keep track of time. In this scheme, the reader send a time-stamp of the current reader time to a tag, which authenticate reader’s identity via evaluating the received time-stamp. If the time-stamp is invalid, the tag will output a random reply, otherwise, it will return an encrypted reply based on the received time-stamp and

its own internal time-stamp. Last, the reader sends this reply back to a backend server to obtain the real tag data. Although Tsudik has not formally analyzed the security properties YA-TRAP, other experiments have proved that this scheme is prone to several security threats. For example, the protocol is vulnerable to "future-time" attack in which the adversary queries the tag off line with several valid time periods in the near future. Then, the adversary can capture the tag's responses and use their responses for online authentication when these time periods. Therefore, this protocol does not provide forward untraceable privacy.

To overcome the aforementioned limitations in the existing solutions to the privacy-preserving authentication scenario, we propose a new scheme that is based on computational hardness of well known elliptic curve discrete logarithmic problem (ECDLP).

3 Evaluation Criterion

We will describe some of the common attack strategies that relate to unauthorized authentication or stealing private information inside the user smart tag

- Eavesdropping activities via a rogue reader:
An outside eavesdropper may attempt to figure out the secrecy hidden in the communication message, then the user privacy is no longer under protection.
- Impersonation of innocent smart tags:
An adversary may attempt to clone a fake tag with the information fetched from a physically compromised smart tag or a eavesdropping to a legitimate smart tag. The attacker may conduct misbehaving activities in the wrong name of innocent victim.
- Reply Attack:
An adversary that monitors on a legitimate communication channel and replay its message at some later time engages in a reply attack. Since the replayed message originated from the authorized node, the same receiver will accept it again. In that situation, the attacker could pass the security without knowing the innocent's secret.

3.1 Design goal

Our design goals are set to achieve as follows:

- Defending against impersonation activities:
It is required to be computationally impossible for the adversary, such as eavesdropper, malicious inside attacker, or malicious owner of RFID reader, to impersonate a smart tag that has not been compromised. In addition, if a smart tag has been reported loss or stolen, the stolen tag should be revoked such that the adversary should not be able to use this stolen tag to impersonate the identity of original owner to gain access.
- Strong Authentication
Even when the RFID readers verify the legitimacy of smart tags, the identity of smart tags should remain anonymous during any communication transmission involved with the tag.
- Low-resource Consumption
The proposed scheme has low computational, communication and storage overheads.

3.2 Assumption

The following assumptions are made regarding the system:

- The open nature of the wireless communication between smart tags and RFID readers enables outside attackers to monitor the communication.
- Smart tags can be stolen, and then attackers can use the stolen tags to impersonate legitimate identity to access resources.
- RFID readers can also be stolen, and the attackers can use the stolen reader to impersonate some other innocent (uncompromised) smart tags.
- The trusted third-party authority is always trustworthy and will not deviate from its normal operations.
- Each smart tag has a rewritable memory, and is computationally powerful to perform a light-weight cryptographic function, such as SHA-1.

4 Proposed Scheme

In this section, we will walk through revolted method at each phase, propose an efficient attack on each of them, and provide a patch to fix the vulnerability. For each method, we describe in which system parameters are initialized or preloaded to smart tags and RFID readers. After that, the authentication process will be explained.

4.1 Preliminary

The trusted authority determines three polynomials $C(x)$, $D(x)$ and $E(x)$ over prime finite field F_q for the readers, where

$$C(x) = c_1x + c_0 \quad (1)$$

$$D(x) = d_1x + d_0 \quad (2)$$

$$E(x) = e_2x^2 + e_1x + e_0 \quad (3)$$

Then, the trusted authority chooses universal polynomials $A(x, y)$ and $B(x, y)$ over the same prime finite field F_q for every tags, where

$$A(x, y) = a_{1,1}xy + a_{1,0}x + a_{0,1}y + a_{0,0} \quad (4)$$

$$B(x, y) = b_{1,1}xy + b_{1,0}x + b_{0,1}y + b_{0,0} \quad (5)$$

, such that

$$A(x, y) \cdot C(x) + B(x, y) \cdot D(x) = E(x) \quad (6)$$

In order to satisfy the previous equation 6, the equalities in the following linear system need to hold:

$$a_{1,1}c_1 + b_{1,1}d_1 = 0 \quad (7)$$

$$a_{1,1}c_0 + a_{0,1}c_1 + b_{1,1}d_0 + b_{0,1}d_1 = 0 \quad (8)$$

$$a_{0,1}c_0 + b_{0,1}d_0 = 0 \quad (9)$$

$$a_{1,0}c_1 + b_{1,0}d_1 - e_2 = 0 \quad (10)$$

$$a_{1,0}c_0 + a_{0,0}c_1 + b_{1,0}d_0 + b_{0,0}d_1 - e_1 = 0 \quad (11)$$

$$a_{0,0}c_0 + b_{0,0}d_0 - e_0 = 0 \quad (12)$$

$C(x)$, $D(x)$ and $E(x)$ are remains the same value when they are preloaded on to a reader, i.e., $c_1, c_0, d_1, d_0, e_2, e_1, e_0$ have been predetermined. Therefore, the authority can always find $A(x, y)$ and $B(x, y)$ to satisfy equation 6, since equations 7 through 12 become a linear equation system regarding unknown coefficients of $A(x, y)$ and $B(x, y)$. The number of unknowns, i.e., 8 unknown variables, is greater than the number of equations, i.e., 6 equations. Thus, it is infeasible for a naive attacker to break this linear equation system.

4.1.1 Hash

A hash function h is an one-way function that maps an arbitrary length input to a k -bit output, i.e. $h : \{0, 1\}^* \rightarrow \{0, 1\}^k$. The typical requirement for this cryptographic checksum functions are

- Pre-image resistance: for any given input x , it is computationally efficient to compute $h(x)$. However, given an arbitrary output y , it is computationally infeasible to find an input such that $h(x) = y$.
- 2nd pre-image resistance: given x , it is computationally infeasible to find $x' \neq x$, such that $h(x) = h(x')$
- Collision resistance: it is computationally infeasible to find any pair of distinct inputs x and x' , such that $h(x) = h(x')$

4.1.2 Symbols and Notations

The bolded symbols are distinctly chosen in random for each session, e.g., γ_2, λ . The specifically underlined parameter are transmitted from the user side, for instance, $A(x, u)$. A good example of combination would be $A(\underline{\gamma_2}, u)$, which represents a user function that need to feed an input at different session.

A function with a hat means it is a elliptic curve point multiplied with output of the function. An example of this would be $\hat{C}(x) = \alpha \times C(x)$, where α is an elliptic curve point.

4.2 The Naive Authentication Protocol

Set-up Phase

When each tag is manufactured, each smart tag is associated with a secret identifier, u , and two polynomial functions:

$$\begin{aligned}
 A(\mathbf{x}, u) &= A_{u,1} \cdot \mathbf{x} + A_{u,0} \\
 A_{u,1} &= a_{11} \cdot u + a_{01} \\
 A_{u,0} &= a_{10} \cdot u + a_{00} \\
 B(\mathbf{x}, u) &= B_{u,1} \cdot \mathbf{x} + B_{u,0} \\
 B_{u,1} &= b_{11} \cdot u + b_{01} \\
 B_{u,0} &= b_{10} \cdot u + b_{00}
 \end{aligned}$$

, where different users have different coefficients ($\{a_{11}, a_{10}, a_{01}, a_{00}\}, \{b_{11}, b_{10}, b_{01}, b_{00}\}$).

When a RFID reader is initialized, the reader is preloaded with an elliptic point α , and three different polynomial functions

$$\begin{aligned}
 C(\mathbf{x}) &= c_1 \cdot \mathbf{x} + c_0 \\
 D(\mathbf{x}) &= d_1 \cdot \mathbf{x} + d_0 \\
 E(\mathbf{x}) &= e_2 \cdot \mathbf{x}^2 + e_1 \cdot \mathbf{x} + e_0
 \end{aligned}$$

, where the coefficients remains the same among different readers.

Communication Phase

Each transmission starts, when the reader first sends a random session nonce, γ_0 to the tag. After receiving data from reader, the tag generate a random session nonce of its own, called γ_1 , and hash the concatenations of these two random session. Afterwards, the user computes its secret functions A and B given γ_2 . That is, $\gamma_2 = H(\gamma_1|\gamma_0)$. Last, the tag sends back a packet of data $\langle \gamma_1, \underline{A(\gamma_2, u)}, \underline{B(\gamma_2, u)} \rangle$ to reader side.

Authentication Phase

With the carefully selected coefficients, the authentication process could be achieved by checking the following equality, as shown in the figure 1 below⁵:

$$\begin{aligned}
& \hat{C}(\gamma_2) \times \underline{A(\gamma_2, u)} + \hat{D}(\gamma_2) \times \underline{B(\gamma_2, u)} \\
& = \alpha \times C(\gamma_2) \cdot \underline{A(\gamma_2, u)} + \alpha \times D(\gamma_2) \cdot \underline{B(\gamma_2, u)} \\
& = \alpha \times \left(C(\gamma_2) \cdot \underline{A(\gamma_2, u)} + D(\gamma_2) \cdot \underline{B(\gamma_2, u)} \right) \\
& = \alpha \times E(\gamma_2) \tag{Equation 6}
\end{aligned}$$

4.2.1 Security Analysis

Vulnerability to Tag Compromise Attack

All tags are preloaded with the same coefficient $\langle a_{11}, a_{10}, a_{01}, a_{00}, b_{11}, b_{10}, b_{01}, b_{00} \rangle$. Assume an attacker has compromised k tags, and each secret function in the tag could leak out 2 pieces of meaningful information about the tag secret u . In other words, the attacker could obtain the following equations:

$$\begin{aligned}
& \text{Tag 1: } \left\{ \begin{array}{l} (a_{11}u_1 + a_{10}) \cdot x \\ (a_{01}u_1 + a_{00}) \\ (b_{11}u_1 + b_{10}) \cdot x \\ (b_{01}u_1 + b_{00}) \end{array} \right. \\
& \dots \\
& \text{Tag k: } \left\{ \begin{array}{l} (a_{11}u_k + a_{10}) \cdot x \\ (a_{01}u_k + a_{00}) \\ (b_{11}u_k + b_{10}) \cdot x \\ (b_{01}u_k + b_{00}) \end{array} \right.
\end{aligned}$$

⁵The specifically underlined parameter are transmitted from the user side, and the bolded symbols are distinctly chosen in random for each session.

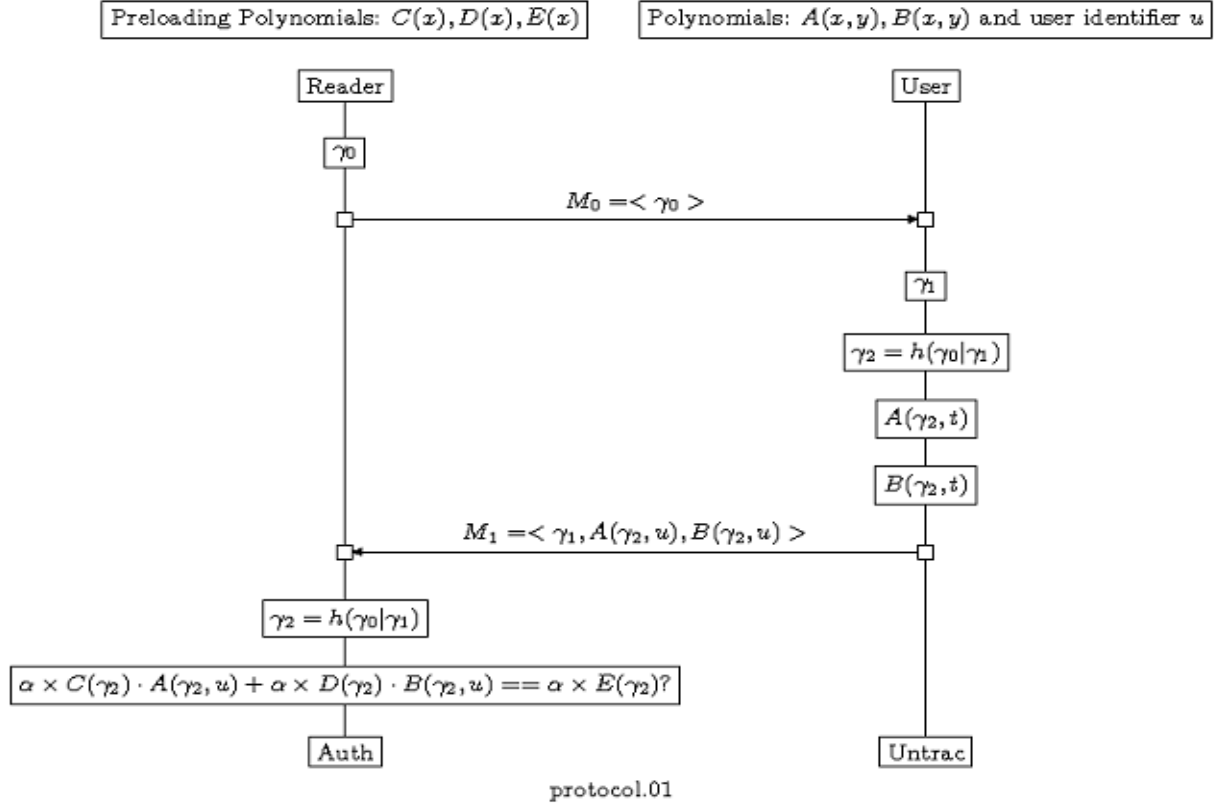


Figure 1: Naive Protocol

If the attacker could fix the dynamic input x , such as having a rogue reader interrogating the compromised tags, then x would no longer be an effect. In that case, there would be in total $4k$ equations for k different tags, and the unknowns include $\underbrace{\{u_1, \dots, u_k, a_{00}, a_{01}, a_{10}, a_{11}, b_{00}, b_{01}, b_{10}, b_{11}\}}_{(k+8) \text{ unknowns}}$.

Since the number of variables is less than the number of equation in the linear system, thus, the attacker is able to reveal the secret parameters by solving this linear system.

4.3 The Security Enhanced Protocol

Set-up Phase

In order to protect the secret function $A(x, u)$ and $B(x, u)$, we need to introduce several variables in the system, $s_u, \bar{a}_u, \bar{b}_u, \lambda$. The tag-specified secret variables $s_u, \bar{a}_u, \bar{b}_u$ are established at the time when the RFID tag was manufactured, while λ were randomly chosen at each transmit session. In order to reduce the computational cost on the tag

side, points parameter, including $\alpha_u, \beta_{u,1}, \beta_{u,0}$, were pre-computed and pre-loaded in the code at tag-creation time. Therefore, it is necessary to have a data structure to maintain such list of points and provide efficient look-up.

Thus, each smart tag is preloaded with a secret identifier, u , and two polynomial functions:

$$\begin{aligned}
A'_u(\mathbf{x}) &= s_u \cdot A(\mathbf{x}, u) - \bar{a}_u \\
&= s_u \cdot A_{u,1} \cdot \mathbf{x} + s_u \cdot A_{u,0} - \bar{a}_u \\
B'_u(\mathbf{x}) &= s_u \cdot B(\mathbf{x}, u) - \bar{b}_u \\
&= s_u \cdot B_{u,1} \cdot \mathbf{x} + s_u \cdot B_{u,0} - \bar{b}_u \\
\alpha_u &= \alpha \times s_u \\
\beta_{u,1} &= \alpha \cdot \lambda \cdot (\bar{a}_u \cdot c_1 + \bar{b}_u \cdot d_1) \\
\beta_{u,0} &= \alpha \cdot \lambda \cdot (\bar{a}_u \cdot c_0 + \bar{b}_u \cdot d_0)
\end{aligned}$$

Communication Phase

The reader initiates the communication by sending a random session nonce, γ_0 to the tag. After receiving data from reader, the tag responds a random session nonce of its own, called γ_1 , and hash the concatenations of these two random session. That is, $\gamma_2 = H(\gamma_1|\gamma_0)$. Following by that, the user computes its secret function given γ_2 . Last, the tag sends back a packet of data $\langle \gamma_1, \lambda \cdot A'_u(\gamma_2), \lambda \cdot B'_u(\gamma_2), \alpha_u \cdot \lambda, \beta_{u,1}, \beta_{u,0} \rangle$ to reader side.

Authentication Phase

With the carefully selected coefficients, the authentication process could be achieved by checking the following equality, as shown in the

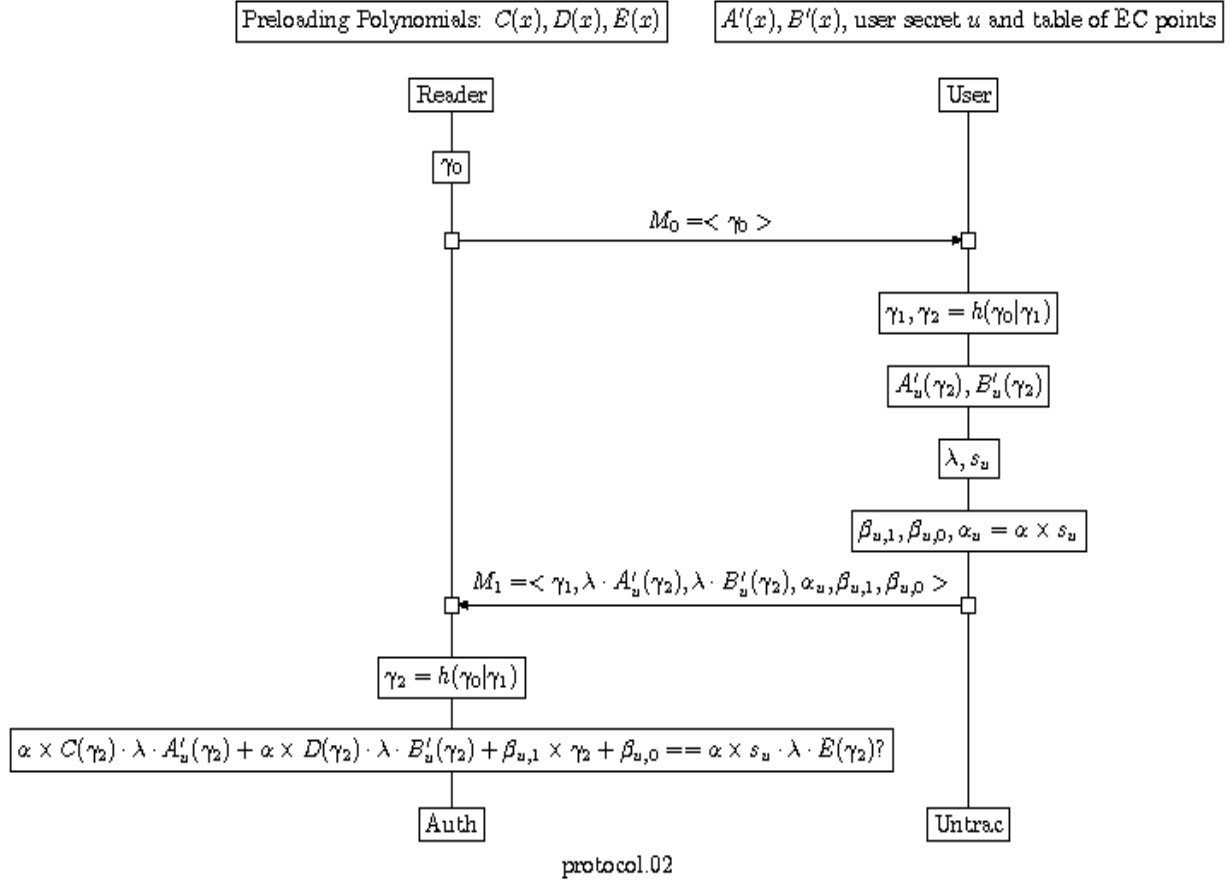


Figure 2: Security Enhanced Protocol

next figure 2⁶:

$$\begin{aligned}
& \hat{C}(\gamma_2) \times \underline{\lambda} \cdot \underline{A}'_u(\gamma_2) + \hat{D}(\gamma_2) \times \underline{\lambda} \cdot \underline{B}'_u(\gamma_2) + \underline{\beta}_{u,1} \cdot \gamma_2 + \underline{\beta}_{u,0} \\
& = \alpha \times C(\gamma_2) \cdot \underline{\lambda} \cdot (s_u \cdot A(\gamma_2, u) - \underline{\bar{a}}_u) \\
& + \alpha \times D(\gamma_2) \cdot \underline{\lambda} \cdot (s_u \cdot B(\gamma_2, u) - \underline{\bar{b}}_u) \\
& + \underline{\alpha} \times s_u \cdot \underline{\lambda} \cdot (\underline{\bar{a}}_u \cdot c_1 + \underline{\bar{b}}_u \cdot d_1) \cdot \gamma_2 \\
& + \underline{\alpha} \times s_u \cdot \underline{\lambda} \cdot (\underline{\bar{a}}_u \cdot c_0 + \underline{\bar{b}}_u \cdot d_0) \\
& = \underline{\alpha} \times \underline{s}_u \cdot \underline{\lambda} \cdot \left(C(\gamma_2) \cdot \underline{A}(\gamma_2, u) + D(\gamma_2) \cdot \underline{B}(\gamma_2, u) \right) \\
& = \underline{\alpha} \times \underline{s}_u \cdot \underline{\lambda} \cdot \underline{E}(\gamma_2)
\end{aligned} \tag{Equation 6}$$

⁶The specifically underlined parameter are transmitted from the user side, and the bolded symbols are distinctly chosen in random for each session.

4.3.1 Security Analysis

Resistance against Tag Compromise Attack

In order to obfuscate the coefficients in the secret functions $A(x,u)$ and $B(x,u)$, we use three tag-based parameters, $s_u, \bar{a}_u, \bar{b}_u$. Similarly, the attack could retrieve 4 pieces of information for each tag. However, this scheme can successfully defend against k-tag compromise attack, and following equations will tell you why.

$$\begin{aligned} \text{Tag 1} & \begin{cases} s_{u_1} \cdot (a_{11}u_1 + a_{10}) \cdot x \\ s_{u_1} \cdot (a_{01}u_1 + a_{00}) - \bar{a}_{u_1} \\ s_{u_1} \cdot (b_{11}u_1 + b_{10}) \cdot x \\ s_{u_1} \cdot (b_{01}u_1 + b_{00}) - \bar{b}_{u_1} \end{cases} \\ & \dots \\ \text{Tag k} & \begin{cases} s_{u_k} \cdot (a_{11}u_k + a_{10}) \cdot x \\ s_{u_k} \cdot (a_{01}u_k + a_{00}) - \bar{a}_{u_k} \\ s_{u_k} \cdot (b_{11}u_k + b_{10}) \cdot x \\ s_{u_k} \cdot (b_{01}u_k + b_{00}) - \bar{b}_{u_k} \end{cases} \end{aligned}$$

There are also $4k$ equations, however, the unknown variables have turned into

$$\underbrace{\{u_1, \dots, u_k, s_{u_1}, \dots, s_{u_k}, a_{u_1}, \dots, a_{u_k}, b_{u_1}, \dots, b_{u_k}, a_{00}, a_{01}, a_{10}, a_{11}, b_{00}, b_{01}, b_{10}, b_{11}\}}_{(4k+8) \text{ unknowns}}$$

Since there is no other way to solve a linear system where the number of unknown variables is greater than the number of equation. Therefore, the system is secure.

Vulnerability to Eavesdropping Attack

Unfortunately, if an eavesdropper had been tailing and monitoring the same tag for k successful sessions, then number of the session-based variables would be the factor to break the system.

$$\begin{aligned} \text{Session 1} & \begin{cases} A'_1(x) = s_u \cdot (a_{11} \cdot u + a_{10}) \cdot x_1 + (a_{01} \cdot u + a_{00}) - \bar{a}_u \\ B'_1(x) = s_u \cdot (b_{11} \cdot u + b_{10}) \cdot x_1 + (b_{01} \cdot u + b_{00}) - \bar{b}_u \end{cases} \\ & \dots \\ \text{Session k} & \begin{cases} A'_k(x) = s_u \cdot (a_{11} \cdot u + a_{10}) \cdot x_k + (a_{01} \cdot u + a_{00}) - \bar{a}_u \\ B'_k(x) = s_u \cdot (b_{11} \cdot u + b_{10}) \cdot x_k + (b_{01} \cdot u + b_{00}) - \bar{b}_u \end{cases} \end{aligned}$$

There would be $2k$ equations, and the unknowns now become $\underbrace{\{x_1, \dots, x_k, u, s_u, \bar{a}_u, \bar{b}_u\}}_{(k+4) \text{ unknowns}}$,

since $u, s_u, \bar{a}_u, \bar{b}_u$ live through the tag lifetime. Security wise, this system is vulnerable against eavesdropping attack.

4.4 Privacy-Preserving Authentication Protocol (Current Design)

Set-up Phase

In order to protect data against eavesdropping attack, each smart tag is preloaded with another secret identifier, μ , generated distinctly at each session. In order to reduce the computational cost on the tag side, points parameter, including $\langle \alpha \times u \cdot \lambda, \beta_{u,1}, \beta_{u,0} \rangle$, were pre-computed and hard-coded in the source code on tag side. Therefore, it is necessary to have a data structure to maintain such list of points and provide efficient look-up.

$$\begin{aligned}
 \bar{A}_u(\mathbf{x}) &= A'_u(\mathbf{x}) - \mu \\
 &= s_u \cdot A(\mathbf{x}, u) - \bar{a}_u - \mu \\
 \bar{B}_u(\mathbf{x}) &= B'_u(\mathbf{x}) - \mu \\
 &= s_u \cdot B(\mathbf{x}, u) - \bar{b}_u - \mu \\
 \alpha_u &= \alpha \times s_u \\
 \beta_{u,1} &= \alpha \cdot \lambda \cdot ((\bar{a}_u \cdot c_1 + \bar{b}_u \cdot d_1) + (d_1 + c_1) \cdot \mu) \\
 \beta_{u,0} &= \alpha \cdot \lambda \cdot ((\bar{a}_u \cdot c_0 + \bar{b}_u \cdot d_0) + (d_0 + c_0) \cdot \mu)
 \end{aligned}$$

Communication Phase

Each transmission starts, when the reader first sends a random session nonce, γ_0 to the tag. After receiving data from reader, the tag generate a random session nonce of its own, called γ_1 , and hash the concatenations of these two random session. That is, $\gamma_2 = H(\gamma_1|\gamma_0)$. Last, the tag sends back a packet of data $\langle \gamma_1, \lambda \cdot \bar{A}_u(\gamma_2), \lambda \cdot \bar{B}_u(\gamma_2), \alpha_u \cdot \lambda, \beta_{u,1}, \beta_{u,0} \rangle$ to reader side.

Authentication Phase

With the carefully selected coefficients, the authentication process could be achieved by checking the following equality, as shown in the

figure 3 below⁷:

$$\begin{aligned}
& \alpha \times C(\gamma_2) \cdot \underline{\lambda} \cdot \bar{A}_u(\gamma_2) + \alpha \times D(\gamma_2) \cdot \underline{\lambda} \cdot \bar{B}_u(\gamma_2) + \underline{\beta_{u,1}} \cdot \gamma_2 + \underline{\beta_{u,0}} \\
= & \alpha \times C(\gamma_2) \cdot \underline{\lambda} \cdot (s_u \cdot A(\gamma_2, u) - \bar{a}_u - \mu) \\
+ & \alpha \times D(\gamma_2) \cdot \underline{\lambda} \cdot (s_u \cdot B(\gamma_2, u) - \bar{b}_u - \mu) \\
+ & \alpha \times \underline{\lambda} \cdot (\bar{a}_u \cdot c_1 + \bar{b}_u \cdot d_1 + (c_1 + d_1) \cdot \mu) \cdot \gamma_2 \\
+ & \alpha \times \underline{\lambda} \cdot (\bar{a}_u \cdot c_0 + \bar{b}_u \cdot d_0 + (c_0 + d_0) \cdot \mu) \\
= & \alpha \times \underline{s_u} \cdot \underline{\lambda} \cdot (C(\gamma_2) \cdot A(\gamma_2, u) + D(\gamma_2) \cdot B(\gamma_2, u)) \\
= & \underline{\alpha \times s_u} \cdot \underline{\lambda} \cdot E(\gamma_2)
\end{aligned} \tag{Equation 6}$$

4.4.1 Security Analysis

Resistance against Tag Compromise Attack

This scheme inherit the resistance against tag compromise attack from the previous scheme. In order to disturb the coefficients in the secret functions $A(x,u)$ and $B(x,u)$, we have introduced four tag-based parameters, $s_u, \bar{a}_u, \bar{b}_u, \mu$. The following shows that this scheme can defend against k-tag compromise attack.

$$\begin{aligned}
\text{Tag 1: } & \begin{cases} s_{u_1} \cdot (a_{11}u_1 + a_{10}) \cdot x \\ s_{u_1} \cdot (a_{01}u_1 + a_{00}) - \bar{a}_{u_1} - \mu_{u_1} \\ s_{u_1} \cdot (b_{11}u_1 + b_{10}) \cdot x \\ s_{u_1} \cdot (b_{01}u_1 + b_{00}) - \bar{b}_{u_1} - \mu_{u_1} \end{cases} \\
& \dots \\
\text{Tag k: } & \begin{cases} s_{u_k} \cdot (a_{11}u_k + a_{10}) \cdot x \\ s_{u_k} \cdot (a_{01}u_k + a_{00}) - \bar{a}_{u_k} - \mu_{u_k} \\ s_{u_k} \cdot (b_{11}u_k + b_{10}) \cdot x \\ s_{u_k} \cdot (b_{01}u_k + b_{00}) - \bar{b}_{u_k} - \mu_{u_k} \end{cases}
\end{aligned}$$

There are also 4k equations, however, the unknown variable has turned into

$$\underbrace{\{u_1, \dots, u_k, s_{u_1}, \dots, s_{u_k}, a_{u_1}, \dots, a_{u_k}, b_{u_1}, \dots, b_{u_k}, \mu_{u_1} \dots \mu_{u_k}, a_{00}, a_{01}, a_{10}, a_{11}, b_{00}, b_{01}, b_{10}, b_{11}\}}_{(5k+8) \text{ unknowns}}$$

Since there is no other way to solve a linear system where the number of unknown variables is grater than the number of equation. Therefore, the system is secure.

⁷The specifically underlined parameter are transmitted from the user side, and the bolded symbols are distinctly chosen in random for each session.

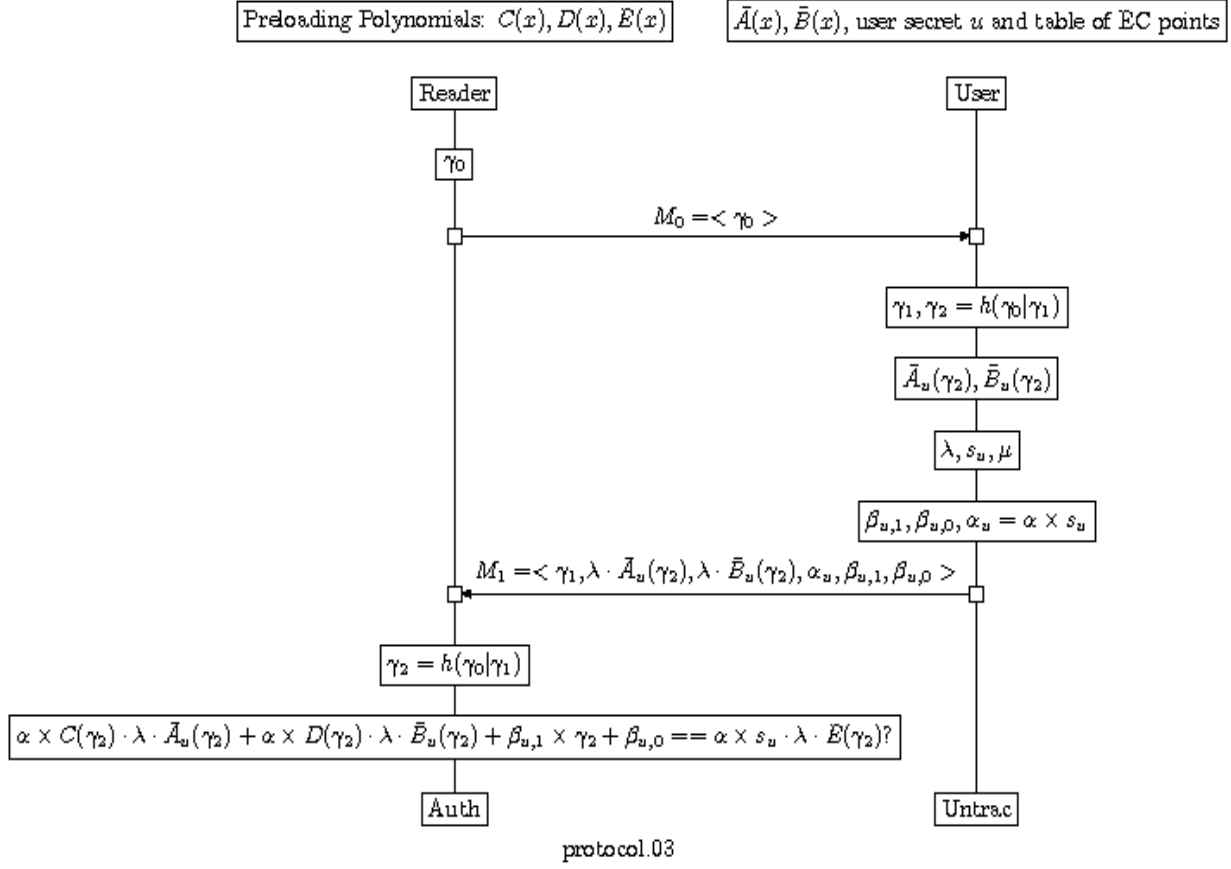


Figure 3: Privacy-Preserving Authentication Protocol

Resistance against Eavesdropping Attack

Introduce a session based random, μ . If an eavesdropper had been monitoring for k successful sessions,

$$\text{Session 1: } \begin{cases} \bar{A}_1(x) = s_u \cdot (a_{11} \cdot u + a_{10}) \cdot x + (a_{01} \cdot u + a_{00}) - \bar{a}_u - \mu_1 \\ \bar{B}_1(x) = s_u \cdot (b_{11} \cdot u + b_{10}) \cdot x + (b_{01} \cdot u + b_{00}) - \bar{b}_u - \mu_1 \end{cases}$$

...

$$\text{Session k: } \begin{cases} \bar{A}_k(x) = s_u \cdot (a_{11} \cdot u + a_{10}) \cdot x_k + (a_{01} \cdot u + a_{00}) - \bar{a}_u - \mu_k \\ \bar{B}_k(x) = s_u \cdot (b_{11} \cdot u + b_{10}) \cdot x_k + (b_{01} \cdot u + b_{00}) - \bar{b}_u - \mu_k \end{cases}$$

In this imported scheme, there are still $2k$ equations, but the number of unknowns has increased to $\underbrace{\{x_1, \dots, x_k, \mu_1, \dots, \mu_k, u, s_u, \bar{a}_u, \bar{b}_u\}}_{(2k+4) \text{ unknowns}}$.

Thus, the system is secure.

Resistance against Brute Force Attack

If an adversary attempts to exhaustively break the data, $\langle \bar{A}(\gamma_2), \bar{B}(\gamma_2) \rangle$, transmitted from a user smart tag, he may find it to be a desperate approach. First, γ_2 is a 160-bit hashed value. Second, the bit length of both function output depends on the size of the finite field, which is around 2^{160} in our implementation.

Resistance against Replay Attack

Another common approach for an attacker is to overhear the communication, and replay the authentication data in the past session. Nevertheless, this attack will not be effective, because γ_2 is defined to be session-distinctly different from time to time.

Resistance against Reader Compromise Attack

If an attacker has compromised the reader, then $\hat{C}(x), \hat{D}(x), E(x)$ are compromised. Due to the NP-hard ECDLP problem, it is computationally infeasible to find out the secret coefficients, such as c_1, c_0, d_1, d_0 . In addition, without those coefficients, knowing $E(x)$ would not help to compute any user's secret functions, $A(x, u)$ and $B(x, u)$ that satisfy the equality for authentication. Therefore, it is impossible to impersonate an innocent smart tags in finite time.

4.5 Point Compression

In our authentication protocol, user needs to transmit several sensitive information wrapped with an elliptic curve points. In reality, wireless sensor does not carry rich resource such as network bandwidth at a premium. Thus, it is desirable to represent those points in a possible minimum form, which is usually referred as point compression.

In a full representation of point compression, an affine point (x_a, y_a) requires $2n$ bits, where $n = \lceil \log_2(p) \rceil$, assuming the prime field is \mathbb{F}_p . The compressed data is trivially reduced to $n+1$ bits by given the x-coordinate of a point plus an additional bit that is used to distinguish two different solutions $(\pm y)$ of recovering the correct y coordinate. Precisely, we need to check for the least significant bit of the least significant coefficient of y coordinate.

In our proposed implementation, we adapted SECG secp160r1[14] recommended parameters to define our elliptic curve. In that standard, each element in the elliptic curve consumes 160 bits.

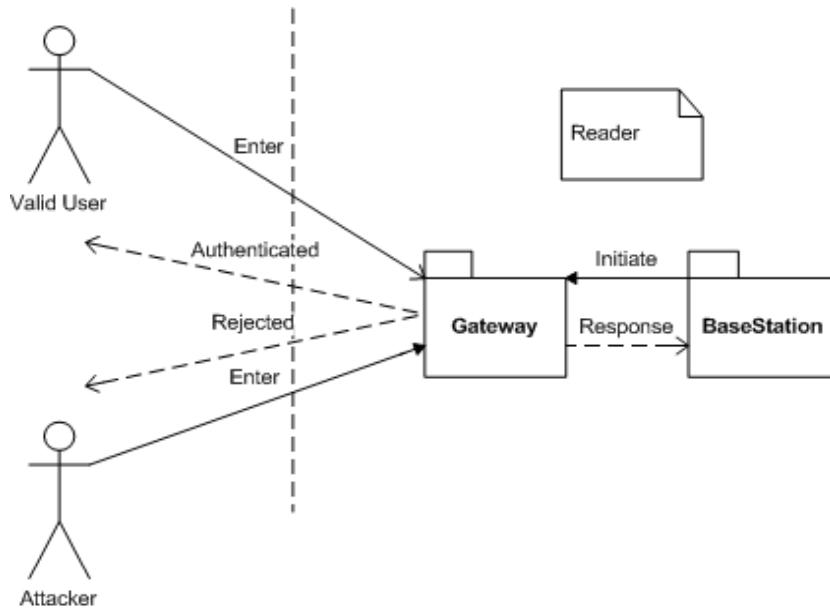


Figure 4: User Case

5 Software Implementation

In our software implementation, we split the reader functions into two components; one processes the authenticated data for each incoming user, and the other acted as a gateway sensor communicating between the user and the base station. All implementations on sensor are programmed with NesC[15] code under TinyOS operation system⁸ and authentication procedure and graphic user interface on base station (PC) are developed in Java. In order to help the reader comprehend our design, we will decompose the system with respect to each different modules.

5.1 Module Decomposition

There are three modules in this system: Base Station, Gateway and User.

User Module

This module is implemented in the smart tag carried with each user. When each tag is manufactured, some tag-specific secrets are

⁸TinyOS is an open-source operating system designed for wireless embedded sensor networks. It features a component-based architecture with minimized code size as required by the severe memory constraints inherent in sensor networks.

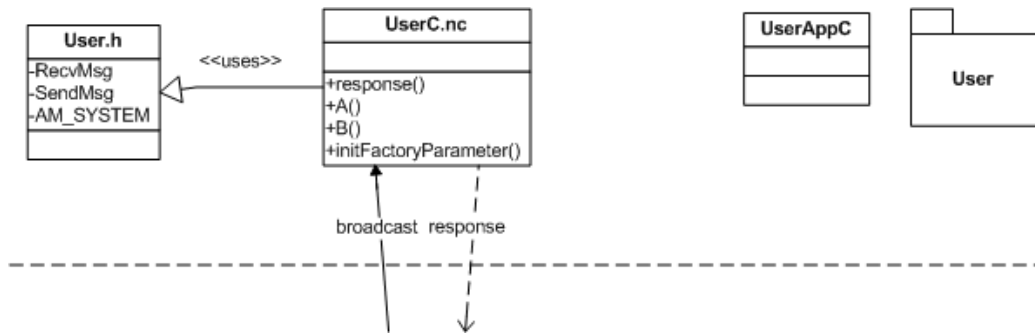


Figure 5: User Module

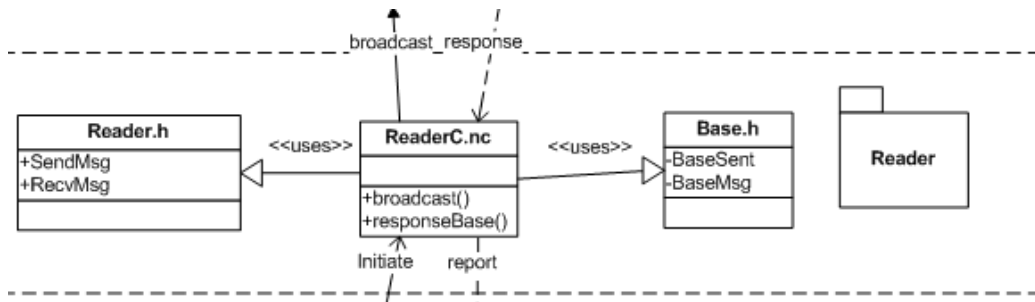


Figure 6: Gateway Module

also embedded during the installation, in addition to some confidential data, such as biometric data, personal private information, and etc. This module is also responsible to communicate with a gateway sensor on a pre-installed channel.

In practice, when the user entered the security entrance, he or she will be authenticated based on the valid parameter that were pre-loaded at installation time. During the time when the message is exchanged wirelessly, the system is guaranteed not to leak out any private information to unknown third party.

Gateway Module

This module takes care of all communicability operations from user smart tags and base station. When the base station first initiates a request, the gateway deliver this requests to an incoming user. Soon after the user responses back, the gateway passes its response back to the base station for authentication. In order to reduce the payload size at each session, this module is also responsible to concatenate two random number generated by base station and user smart tag respectively.

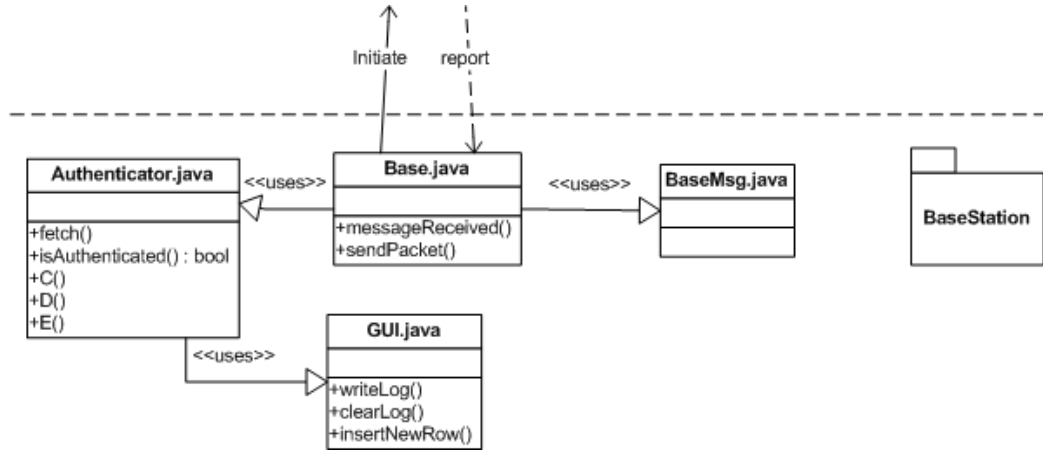


Figure 7: Base Station Module

Base Station Module

This module requires to run on a computational powerful device, e.g. a personal computer.

The first sub-module, Base and BaseMsg take care of all communication tasks to the gateway module, including the generation of the first half of session key to initiate a request, and marshalling the incoming NesC data to a Java-compliant format.

The second sub-module, Authenticator, is responsible to determine the authenticity of an incoming tag, and displayed the result to the graphic user interface, which is the the third sub-module.

5.2 Pre-Computation Storage

Due to the resource limitation in smart tag, it is important to carefully adjust the strength of the security in way that provides reasonable protection while limiting overhead. In our suggested solution, several elliptic curve points, as guardians to protect sensitive data, are preloaded in the memory of smart tags. More precisely, we pre-install 32 points in each table for $\langle \alpha_u, \beta_{u,0}, \beta_{u,1} \rangle$ respectively. As result, we have also observed that the RAM usage elevated, while the number of points used increases. A more detailed observation on storage consumption will be explained in the experimental result section.

5.3 Required Package

Here is the list of libraries that we have used in this project,

- WM-ECC
we adapted the WM-ECC package ⁹ to perform operation on elliptic curve or big integer computation on user's smart tag.
- FlexiProvider (v. 1.6)
Elliptic curve computation on personal computer is implemented with this java-based cryptographic toolkit, FlexiProvider ¹⁰.
- CoDec ASN.1 En-/Decoder Library
it is required to install this CoDec library¹¹ to run FlexiProvider.

6 Experiments

In our experiment, we use a PC, equipped with Duo Core 1.6GHz Pentium(R) CPU and 1.0GB of RAM, to emulate a RFID reader. Each PC is attached with a TI TelosB mote as a gateway, communicating with a user's smart tag, which is also emulated by TI TelosB mote ¹², supported by UC Berkeley's 22.x TinyOS[16] operating system and the NesC programming language.

6.1 Experimental Result

Computational Overhead

In the proposed scheme, the most computationally expensive operation is the point multiplication. In order to reduce the computation cost, we pre-compute an enumerated list of multiplication on point α and pre-load this list in the smart tag at manufacturing time. When the RFID reader initiates a request, the smart tags picks a point in the list corresponding to the parameters s_u and λ and responses back to the reader. In our experiment, With the pre-computation technique, the we have measured authentication spends 140 milliseconds in average out of 100 tries.

Storage Consumption

Like we described in section 5.2, we preloaded some lists of points for

⁹WM-ECC is an Elliptic Curve Cryptography suite developed exclusively for wireless sensor motes. Official website: <http://www.cs.wm.edu/wanghd/>

¹⁰A powerful toolkit that provides varieties of cryptographic modules to any application on top of the Java Cryptographic Architecture

¹¹CoDec is a Java package for encoding and decoding ASN.1 data structures, which is used in varieties of cryptographic standards.

¹²TelosB mote is integrating an IEEE 802.15.4/ZigBee compliant RF transceiver at 250 kbps data rate, an integrated onboard antenna, and a 16-bit, 8MHz TI MSP430 microcontroller with 10kB RAM plus 1MB external flash for data logging, programming and data collection via USB. The mote is battery-chargeable of two AA batteries

	< 8, 8, 8 > points	< 16, 16, 16 > points	< 32, 32, 32 > points
RAM	3430 bytes	4926 bytes	6838 bytes
ROM	31556 bytes	33052 bytes	36464 bytes

Table 1: Memory Usage for Smart Tag

	Gateway Mote	User Tag Mote w/ 96 points
ROM	25734 bytes	31556 bytes
RAM	1916 bytes	3430 bytes

Table 2: Memory Consumption for Gateway Mote and Smart Tag Mote

each tag at the time, when the tag is made. We compare the storage space needed, when the size of the lists increases. In table 1, we show the of RAM size required increases with the respect to the growth of the list for points $\langle \alpha_u, \beta_{u,0}, \beta_{u,1} \rangle$.

The next table 2 below measures the code size of gateway sensor and smart tag sensor required:

Communication Overhead

Under WM-ECC specification, each element relies on the size of finite field on the elliptic curve, which occupies 160 bits (20 bytes). Moreover, each elliptic curve point, consisted of two elements for x and y coordinate,s consumes 320 bits (40 bytes) of memory in total. In order to reduce the bandwidth, we apply the point compression technique for a transmission packet that involves elliptic curve points in communication. Once the compressed data is retrieved on the reader side, it is immediately decompressed back to the original form for further authentication evaluation. In our experiment, the size of payload, built by three EC points $\langle \alpha_u, \beta_0, \beta_1 \rangle$, is dramatically brought down to 112 bytes. This size reduction is smaller than the maximum payload size of 128 bytes, as specified by IEEE 802.15.4 standard. Therefore, the proposed scheme satisfied the constraint of low communication expense.

7 Future Work and Conclusion

The proposed scheme requires some storage space for enumerated list of elliptic curve point α to pre-load in each tag when they were created. Hence, specific implementation that provides fast look-up and space-saving storage mechanism would make improvements in the futures work. In addition, even if the 112-byte payload size is smaller then the the maximum buffer size could be in the standard specification,

however, it would be practical to decrease the packet size down to a reasonable number in the real world.

As we have shown in this project, our proposed approach protects tag's identity from various types of attack. In addition, extensive experiment has been conducted to evaluate its impact on bandwidth, storage and authentication time, while preserving the strength of security. Finally, the results show that its low resource consumption is indeed affordable for the devices with extreme resource limitations.

Acknowledgements

We would like to thank Chuang Wang for many useful discussion on EC computations and uses operation in WM-ECC library.

APPENDIX

A A Short Manual: How to run this program

We give a simple walkthrough on how to run this program. In order to run this program successfully, make sure your computer runs the Java compiler of version 1.5 or higher.

A.1 Pre-install Set Up

Here are several environments parameter you need to set up before compiling this program. The export command is (bash) shell specific. Please use the appropriate command according to your operating system.

```
1 export MOIECOM="serial@/dev/ttyUSB0:telosb"  
2 export CLASSPATH=/opt/tinyos-2.1.0/support/sdk/java/  
   tinyos.jar:lib/FlexiProvider-1.6p5.jar:lib/CoDec-  
   build17.jar:lib/liquidInf.jar:.
```

A.2 Installation

All of these instructions could also be found in the Makefile under gateway/user directory.

A.2.1 Install a User Tag Mote

Each of the tag has unique coefficients, when it is created at manufacturing time. Please locate the following lines in UserC.nc source code to change coefficients, and these are the default settings used in the demonstration:

```
1 //A(x,y) = a11 * x * y + a10 * x + a01 * y + a00  
2 void A(word_t* x, word_t* u, word_t* res)  
3 a11[0] = 1;  
4 a10[0] = 1;  
5 a01[0] = 1;  
6 a00[0] = 1;  
7 //B(x,y) = b11 * x * y + b10 * x + b01 * y + b00  
8 void B(word_t* x, word_t* u, word_t* res)  
9 b11[0] = 1;  
10 b10[0] = 2;  
11 b01[0] = 1;  
12 b00[0] = 1;
```

Here is the command needed to compile and install a user program on a TelosB mote.

```
1 make telosb install.1000 MSG_SIZE=128
```

A.2.2 Install a Gateway Mote

After you have locate the source code under gateway director, use the following to compile and install a gateway program on a TelosB mote.

```
1 make telosb install.1 MSG_SIZE=128
```

A.2.3 Install a Base Station program

Here is the command needed to compile and install the base station program on a Java-supported computer.

```
1 rm *.class
2 rm BaseMsg.java
3 mig java -target=null -java-classname=BaseMsg Base.h
   BaseMsg -o BaseMsg.java
4 make telosb
```

A.3 Running the Program

If you wish to run this program in plain text mode (without GUI support), execute as following,

```
1 java Base -comm serial@/dev/ttyUSB0:telosb
```

Conversely, a graphic user interface is provided with the following command:

```
1 java GUI -comm serial@/dev/ttyUSB0:telosb
```

Do not run this command, unless it is for testing purposes.

```
1 java net.tinyos.tools.PrintfClient -comm serial@/dev/
   ttyUSB0:telosb
```

References

- [1] T. I. Corp, “Ti celebrates 10 year anniversary of rfid, http://www.ti.com/rfid/docs/manuals/rfidnews/tiris_n120.pdf.” RFID News, Issue 20, Texas Instruments., 2000.
- [2] D. B. S. Sarma and K. Ashton, “The networked physical world - proposals for engineering the next generation of computing, commerce & automatic identification. white paper, mit: Auto-id center,,” Oct 2000.
- [3] A. Juels, “Rfid security and privacy: a research survey,” *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 2, pp. 381–394, 2006.
- [4] M. R. Rieback, B. Crispo, and A. S. Tanenbaum, “Keep on blockin’ in the free world: Personal access control for low-cost rfid tags,” in *Security Protocols Workshop*, pp. 51–59, 2005.
- [5] A. Juels, R. L. Rivest, and M. Szydlo, “The blocker tag: selective blocking of rfid tags for consumer privacy,” in *CCS ’03: Proceedings of the 10th ACM conference on Computer and communications security*, (New York, NY, USA), pp. 103–111, ACM Press, 2003.
- [6] Y. Nohara, S. Inoue, K. Baba, and H. Yasuura, “Quantitative evaluation of unlinkable id matching schemes,” in *WPES ’05: Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, (New York, NY, USA), pp. 55–60, ACM, 2005.
- [7] B. Song and C. J. Mitchell, “Rfid authentication protocol for low-cost tags,” in *WiSec ’08: Proceedings of the first ACM conference on Wireless network security*, (New York, NY, USA), pp. 140–147, ACM, 2008.
- [8] S. A. Weis, S. E. Sarma, R. L. Rivest, and D. W. Engels, “Security and privacy aspects of low-cost radio frequency identification systems,” in *Security in Pervasive Computing*, vol. 2802 of *Lecture Notes in Computer Science*, pp. 201–212, 2004.
- [9] Food and D. Association, “Combating counterfeit drugs: A report of the food and drug,” tech. rep., Administration Annual Update, May 18, 2005.
- [10] K. Koscher, A. Juels, and T. Kohno, “Epc rfid tags in security applications: Passport cards, enhanced drivers licenses, and beyond,”
- [11] D. Molnar and D. Wagner, “Privacy for rfid through trusted computing,” in *In Procs. Workshop on Privacy in the Electronic Society WPES05*, pp. 31–34, Press, 2005.

- [12] T. Dimitriou, “A lightweight rfid protocol to protect against traceability and cloning attacks,” September, 2005.
- [13] G. Tsudik, “Ya-trap: yet another trivial rfid authentication protocol,” pp. 4 pp.+, 2006.
- [14] C. Research, “Recommended elliptic curve domain parameter,” tech. rep., Certicom Copr., September 20, 2005.
- [15] R. v. B. M. W. E. B. David Gay, Phil Levis and D. Culler, “The nesc language: A holistic approach to networked embedded systems,” *In Programming Language Design and Implementation (PLDI)*, June 2003.
- [16] “Tinyos, <http://www.tinyos.net/>.”