國立交通大學

資訊科學與工程研究所

博士論文

適用於無線網路之使用者認證協定
User Authentication Protocols in Wireless Networks with Petri Net Verification

研 究 生：曾蕙如

指導教授：簡榮宏 博士

中華民國九十九年五月

適用於無線網路之使用者認證協定

User Authentication Protocols in Wireless Networks with Petri Net Verification

研究生: 曾蕙如　　Student: Huei-Ru Tseng

指導教授: 簡榮宏 博士　　Advisor: Dr. Rong-Hong Jan

國立交通大學

資訊科學與工程研究所

博士論文

A Dissertation
Submitted to Department of Computer Science
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in
Computer Science
May 2010
Hsinchu, Taiwan, Republic of China

中華民國九十九年五月

適用於無線網路之使用者認證協定

研究生：曾蕙如　　　　指導教授：簡榮宏 博士

國立交通大學 資訊科學與工程研究所

摘 要

無線網路產業歷經多年發展，發展重心在於提供客戶獨一無二的資訊應用服務的技術。對於無線網路而言，其技術核心在於資源的存取－滿足使用者隨時隨地皆能存取遠端資源的行動生活應用需求。然而電腦犯罪活動卻隨著資訊科技的發展日益猖獗。因此，建構一個安全的資訊/通訊環境乃為當務之急。針對無線網路資源的存取，伺服器必須能有效地認證遠端使用者的身份。

近年來無線感測網路(wireless sensor networks, WSNs)已經是無線網路研究中重要的議題之一，它是由許多散佈於各地的感測節點(sensor nodes)所組成，主要用以蒐集各種環境資料，例如溼度、壓力、溫度等。每個節點皆有監控偵測物理環境的能力，並藉由無線通訊的方式，將所蒐集之資訊回傳至基地台(base station)或是應用系統的後端平台(backend)。因應無線感測網路無所不在(ubiquity)的應用需求增加，使用者應能即時存取儲存於感測節點的資訊。因此，感測節點所收集之資訊應該採取安全機制來加以保護，避免未經授權的使用者非法取得。

本論文中，我們將闡述無線安全領域的發展現況，與多種無線網路中的使用者認證協定(user authentication protocols)。同時針對使用者認證之安全架構與安全需求加以說明。再者，我們提出數種適用於無線網路之使用者認證協定，其中包括植基於密碼方法的使用者認證協定(password-based user authentication protocols)、植基於生物特徵方法的使用者認證協定(biometrics-based user authentication protocols)，以及自我憑證方法的使用者認證協定

(self-certificate-based user authentication protocols)。

針對植基於密碼方法的使用者認證協定，我們提出兩種認證協定。協定一乃運用 LU 矩陣分解法(LU decomposition)，讓使用者透過開放式通訊網路進行認證與存取網路服務。此協定的特性包括動態更改密碼、相互認證(mutual authentication)、使用者匿名(user anonymity)，與金鑰協議(key agreement)等。協定二主要是適用於無線感測網路的使用者認證協定，能讓使用者以低運算量來即時存取感測節點的資訊。

針對植基於生物特徵方法的使用者認證協定，我們提出一個適用於智慧卡(smart cards)的使用者認證協定。此協定能允許伺服器驗證使用者之生物特徵的同時，亦能保護使用者隱私。此外，我們將此協定與秘密分享方法結合，擴充為多人生物特徵認證協定(multi-party biometrics-based user authentication protocol)－即(t, n)-門檻式多人認證協定，在此協定中，必須提出至少 t 個以上之使用者生物特徵、密碼，與智慧卡，方可重建認證金鑰(authentication key)。

針對植基於自我憑證方法的使用者認證協定，我們提出一個適用於無線感測網路的認證協定，提供使用者與感測節點相互認證與金鑰協議，同時，金鑰分配中心(key distribution center, KDC)亦可撤銷金鑰對。在此協定中，使用者首先傳送資料要求封包予其傳輸範圍內的感測節點，感測節點認證通過，即可回傳使用者所要求之資料。平均而言，我們假設使用者傳輸範圍內有 n 個感測節點。在攻擊者截取 n 個感測節點當中 t 個節點的情況下，此協定仍然可以維持其安全性。此外，我們利用派翠網路(Petri nets)來建立模型並分析所提出的協定，並證明其可抵禦多種攻擊模式。

User authentication protocols in wireless networks with Petri net verification

Student: Huei-Ru Tseng
Advisor: Dr. Rong-Hong Jan

Department of Computer Science
National Chiao Tung University

Abstract

The wireless industry, over the last few years, has undergone a tremendous amount of change, which is brought about through the introduction of a never ending stream of technologies all designed to provide unique services that customers will purchase. For wireless networks, at the heart of all the technologies introduced is access—being able to access services regardless of where the end user is physically located. While wireless networks are very convenient for users, their widespread use creates new challenges from a security point of view. To control access to wireless networks, it is essential for the server to authenticate the remote users.

A variant of the wireless networks is wireless sensor networks (WSNs). In WSNs, there are spatially distributed sensors which cooperatively monitor environmental conditions, such as humidity, pressure, temperature, motion, or vibration, at different locations. Each sensor node has the ability to monitor the physical world and return the sensed information to base stations or at the backend of the application system via wireless communication. With the increasing ubiquity of WSNs, real-time data could be accessed from every sensor node. Hence, security measures should be taken to protect the collected secrets in order to prevent un-authorized users from gaining the information.

In this dissertation, we introduce recent developments in the field of wireless security and investigate several user authentication protocols in wireless networks. A detailed explanation of security frameworks and security requirements for authentication will be given. We design several user authentication protocols in wireless

networks, including two kinds of password-based user authentication protocols, a biometrics-based user authentication protocol, and a self-certificate-based user authentication protocol.

For password-based user authentication, we propose two password-based user authentication protocols, namely protocol-I and protocol-II. The protocol-I is a password-based user authentication protocol using LU decomposition, which authenticates remote users and allows legitimate users to access network services over an open communication network. This protocol possesses many merits, including freely changeable passwords, mutual authentication, user anonymity, and session key agreement. The protocol-II is a password-based user authentication protocol for WSNs, which allows legitimate users to query sensor data at any of the sensor node in an ad hoc manner and imposes very little computational overhead.

For biometrics-based user authentication, we propose a biometrics-based remote user authentication protocol using smart cards. The protocol fully preserves the privacy of the biometric data of each user while allowing the server to verify the correctness of the users' biometric characteristics without knowing the exact values. In addition, the proposed protocol is later extended to a multi-party biometrics-based remote user authentication protocol by incorporating a secret sharing component. This extended protocol is essentially a (t, n)-threshold multi-party authentication protocol. Any group of t or more users can together reconstruct the authentication key with their own biometric data, passwords, and smart cards but no group of less than t users can.

For self-certificate-based user authentication, we propose a self-certificate-based user authentication protocol for WSNs, which can deal with authenticated queries involving multiple sensor nodes, achieve mutual authentication and key agreement between users and sensor nodes, and provide a key distribution center (KDC) to revoke compromised key pairs. In this protocol, a user can send data requests to the sensor nodes within his communication range and receives valid responses if the requests are legitimate. On average, there are n sensors in the communication

range of the user. The proposed protocol still works well even if the adversary captures t nodes out of n nodes in the WSNs. Moreover, security of these proposed protocols is modelled and analyzed with Petri nets. Our analysis shows that the protocols can defend notorious attacks.

誌謝

首先感謝指導教授簡榮宏博士六年來的諄諄教誨與耐心教導，使學生在學術研究與待人處事上受益良多，感謝之情非筆墨所能形容，浩浩師恩，心版永銘，在此獻上最高的敬意與謝忱。同時，學生也非常感謝口試委員交通大學資訊工程學系楊武教授與陳榮傑教授、清華大學資訊工程學系黃能富教授與孫宏民教授、臺灣科技大學資訊管理系吳宗成教授對本論文不吝批評與指正，使本論文更臻完善，在此深表感激。

我也要特別感謝周澤民教授與德國 Passau University 的 Hermann De Meer 教授對於我在學術方面的指導，也感謝 Passau University 的 Computer Networks and Computer Communications 實驗室成員，陪伴我在德國的研究生活。謝謝計算機網路實驗室所有成員在課業的提攜與生活上的照顧，也謝謝在求學生涯中，曾經鼓勵與支持我的朋友。

同時我也感謝財團法人漢儒文教基金會、財團法人宗倬章先生教育基金會、財團法人世豐善智慈善事業基金會、英業達 OKWAP 教育基金會、臺北縣政府、雅虎奇摩無名小站、東森媒體集團，以及臺灣網路認證股份有限公司所提供的獎助學金，讓我在求學生涯中，更能無後顧之憂的專心於研究工作。

最後，感謝我的家人，在我求學生涯中，無微不至的照顧與關懷，讓我毫無牽掛地專注於學業，給予我最溫暖的愛與不間斷的支持與鼓勵。在此願將完成論文的喜悅與大家分享。

Acknowledgements

A lot of people have contributed to this dissertation, and made my entire stay at NCTU as enjoyable and fulfilling as it has been. Most of all, I would like to thank my advisor Professor Rong-Hong Jan for having his door always open and for helping me to direct my research and to stay focused. Then, I would like to express great gratitude to Professor Wuu Yang, Professor Rong-Jaye Chen, Professor Nen-Fu Huang, Professor Hung-Min Sun, and Professor Tzong-Chen Wu, who gave me a lot of great advice in this dissertation.

Next, I would like to convey my thanks to Professor Emery Jou, who gave me constructive suggestions and has taught me many things, especially in security algorithms. I would also like to thank to everyone in DAAD, NSC, and in particular Professor Hermann De Meer and his entire wonderful research group at Passau University for giving me the opportunity to do research in Germany. The summer spent in this group was truly fantastic. Thanks also to all members of Computer Network Lab for their assistance and kindly helping both in the research and the daily life during these years.

Moreover, I would like to thank the Han-Ru Foundation, the Tzong Jwo Jang Educational Foundation, the Sheh Fung Good Deed and Wisdom Charity Foundation, the OKWAP Inc., the Taipei County Government, the Yahoo! Inc., the Eastern Multimedia Group, and the TWCA Inc. for providing scholarships for me.

Finally, I would like to dedicate this dissertation to my family for their great love and support in all stages of my life.

Thank you, NCTU, for the great time I had here!

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The wireless industry, over the last few years, has undergone a tremendous amount of change, which is brought about through the introduction of a never-ending stream of technologies all designed to provide unique services that customers will purchase. However, wireless network security is still a major impediment to further deployment of the wireless networks. Security mechanisms in wireless networks are essential to protect data integrity and confidentiality, authentication, user privacy, quality of service, and continuity of service. For wireless networks, at the heart of all the technologies introduced is access—being able to access services regardless of where the end user is physically located. The rapid growth of wireless communication means that security issues in wireless networks are of increasing practical importance. Therefore, to control access to wireless networks, it is essential for the server to authenticate the remote users. A remote user authentication protocol is a mechanism that authenticates remote users and allows legitimate users to access network services over an open communication network.

A variant of the wireless networks is wireless sensor networks (WSNs). In WSNs, there are spatially distributed sensors which cooperatively monitor environmental conditions, such as humidity, pressure, temperature, motion, or vibration, at different locations. It integrates both wireless and sensor technology into a small device, called a sensor node. Each sensor node has the ability to monitor the physical world and return the sensed information to base stations or at the backend of the application system via wireless communication. The collected data can be presented to users either upon inquiries or upon event detection. In general, most queries in WSN applications are issued at the base stations or at the backend of the application system. However, real-time data may no longer be accessed only at the base stations or the gateway nodes. With the increasing ubiquity of WSNs, real-time data could be accessed from every sensor node. For some applications, such as military surveillance, the collected data is highly sensitive. Hence, security measures should be taken to protect the collected secrets in order to prevent un-authorized users from gaining the information.

Passwords are frequently used in the user authentication protocols because they are easier to remember by users than cryptographic keys. In 1981, Lamport [33] proposed a password authentication protocol that makes use of password tables to verify remote users. However, in Lamport's protocol, password tables are stored in the remote server, which might be broken into and hence the passwords might be stolen. In order to eliminate the risk of password leakage, a great deal of research, including solutions using smart cards, has been proposed.

A smart card is a tamper-resistant device that contains one or more integrated circuits (ICs) and also may employ one or more of the following machine-readable technologies: magnetic stripe, bar code, contactless radio frequency transmitters, biometric information, encryption and authentication, or photo identification [2]. The integrated-circuit chip (ICC) embedded in the smart card can act as a microcontroller or as a computer. Data are stored in the chip's memory and can

Table 1.1: Formal definition of a Petri net

A Petri net is a 5-tuple, (P, T, F, W, M_0) where:
$P = \{P_1, P_2, \cdots, P_m\}$ is a finite set of places,
$T = \{T_1, T_2, \cdots, T_n\}$ is a finite set of transitions,
$F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (flow relation),
$W : F \rightarrow \{1, 2, 3, \cdots\}$ is a weight function,
$M_0 : P \rightarrow \{0, 1, 2, 3, \cdots\}$ is the initial marking,
$P \cap T = \emptyset$ and $P \cup T \neq \emptyset$.
A Petri net structure $N = (P, T, F, W)$ without any specific initial marking is denoted by N.
A Petri net with the given initial marking is denoted by (N, M_0).

be accessed to complete various processing applications. The merits of a smart card for password authentication are the simplicity and efficiency of the login and authentication process [66]. Experience has shown that constructing a secure user authentication protocol with smart cards is not trivial because lots of proposed protocols were subsequently broken. Therefore, how to design robust user authentication protocols for wireless networks is a critical issue.

This dissertation introduces recent developments in the field of wireless security and investigates various user authentication protocols in wireless networks. A detailed explanation of security frameworks and security requirements for authentication will be given. We design several user authentication protocols in wireless networks, including two kinds of password-based user authentication protocols, a biometrics-based user authentication protocol, and a self-certificate-based user authentication protocol. Moreover, Petri nets [53] may be used to infer what an attacker could know if he happens to know certain items in the security protocol.

The formal definition of a Petri net [46] is listed in Table 1.1. Petri nets are composed from graphical symbols designating places (shown as circles), transitions (shown as rectangles), and directed arcs (shown as arrows). The places denote (atomic and composite) data items. The transitions denote decryption or decomposition operations. The directed arcs run between places and transitions. When a transition fires, a composite data item is decomposed or decrypted, resulting in one or more simpler data items. Since we assume an open network environment, all data items in the transmitted messages are assumed to be public, and are known to the attacker. There will be tokens in the places representing the data items in the transmitted messages initially. From this initial marking, we can infer what an attacker can know. Furthermore, we can also experiment what an attacker can know if he knows additional data items from other sources. Therefore, we use Petri nets in the security analysis of the proposed protocols.

The rest of this dissertation is organized as follows. In Chapter 2, we state the basic terms and preliminaries for our dissertation, and briefly review existing user authentication protocols in wireless networks. In Chapter 3, we introduce password-based user authentication protocols. Next, we present a biometrics-based user authentication protocol in Chapter 4. A self-certificate-based user authentication protocol will be described in Chapter 5. Finally, a conclusion is given in Chapter 6.

Chapter 2

Preliminaries

In this chapter, we first state several mathematical problems [43], including the discrete logarithm problem (DLP), the Diffie-Hellman problem (DHP), the elliptic curve discrete logarithm problem (ECDLP), and the computational Diffie-Hellman problem (CDHP). The LU decomposition [68] and secret sharing method [56] will be presented later. Next, we provide a detailed survey of various user authentication protocols. The notations and their corresponding definitions used in this dissertation are listed in Table 2.1.

2.1 Mathematical problems

Now we introduce several mathematical difficult problems as follows.

Definition 1. *The discrete logarithm problem (DLP) is defined as follows: given a prime* p*, a generator* g *of* Z_p^**, and an element* $\beta \in Z_p^*$*, find the integer* α*,* $0 \leq \alpha \leq p-2$*, such that* $g^\alpha \equiv \beta \ (mod\ p)$*.*

Definition 2. *The Diffie-Hellman problem (DHP) is defined as follows: given a prime* p*, a generator* g *of* Z_p^**, and elements* $g^c \ (mod\ p)$ *and* $g^s \ (mod\ p)$*, find* $g^{cs} \ (mod\ p)$*.*

Let G_1 be a group of the prime order q and P be an arbitrary generator of G_1.

Table 2.1: Notations

Symbol	**Definition**
U_i	User i
ID_i	User i's or sensor node i's identity
PW_i	User i's chosen password
TM_i	User i's iris template
(S_i, Q_i)	User i's or sensor node i's private/public key pair
Key	The sensor gateway-node's private key
$K_{i,j}$	The pair-wise key computed by the entity i and entity j
AK	The authentication key composed of each user's password
KDC	The key distribution center
s	The KDC's private key
K_{pub}	The KDC's public key
K_s	The server's secret key
SK_i	The session key computed by a user i and the server
$COMM_i$	The set of sensor nodes within the communication range of the user i
CI_i	User i's certificate information generated by the KDC
n	The number of users that could be supported by the system
T	The timestamp
$A_{n \times n}$	A symmetric key matrix
$h(\cdot)$	A one-way hash function
$\mathcal{E}_{TM_i}(\cdot)$	An encryption function with the biometric template TM_i as the encryption key [16, 59]
t	A threshold value. At least t users are needed to reconstruct AK for authentication
$f(x)$	A (t-1)-degreed polynomial, where $f(x) = (AK + a_1x + a_2x^2 + \cdots + a_{t-1}x^{t-1}) \bmod q$
k_i	A secret share computed by the server, where $k_i = f(ID_i)$
$\oplus$	The exclusive-or (XOR) operation
$\|$	Concatenation

We view G_1 as an additive group.

Definition 3. *The elliptic curve discrete logarithm problem (ECDLP) is defined as follows: given* Q*,* $R \in G_1$*, find an integer* $x \in Z_q^*$ *such that* $R = xQ$*.*

Definition 4. *The computational Diffie-Hellman problem (CDHP) is defined as follows: given* $(P, aP, bP) \in G_1$ *for* $a, b \in Z_q^*$*, find* $abP \in G_1$*.*

2.2 LU decomposition

In the LU decomposition, an $n \times n$ matrix A is written as

$$A = L \cdot U \tag{2.1}$$

where L is a nonsingular lower triangular matrix, and U is a nonsingular upper triangular matrix.

We assume that $a_{ij} = a_{ji}$, for $1 \leq i \leq n$ and $1 \leq j \leq n$. Since A is symmetric, the product of the x-th row of matrix L and the y-th column of matrix U is as same as that of the y-th row of matrix L and the x-th column of matrix U.

For example, given A as follows:

$$A = \begin{pmatrix} 1 & 2 & 4 & 5 \\ 2 & 5 & 8 & 9 \\ 4 & 8 & 15 & 17 \\ 5 & 9 & 17 & 20 \end{pmatrix} \tag{2.2}$$

we perform elementary row operations to get the lower matrix L and upper matrix U as follows:

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 4 & 0 & -1 & 0 \\ 5 & -1 & -3 & -3 \end{pmatrix} \text{ and } U = \begin{pmatrix} 1 & 2 & 4 & 5 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{2.3}$$

Given $x = 2$ and $y = 3$, we can compute a_{23} and a_{32} as follows:

$$a_{23} = L_R(2) \cdot U_C(3) = \begin{pmatrix} 2 & 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 4 & 0 & 1 & 0 \end{pmatrix}^T = 8 \tag{2.4}$$

$$a_{32} = L_R(3) \cdot U_C(2) = \begin{pmatrix} 4 & 0 & -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 2 & 1 & 0 & 0 \end{pmatrix}^T = 8 \tag{2.5}$$

Since matrix A is symmetric, $a_{23} = a_{32}$. Note that $L_R(2)$ denotes the 2nd row of matrix L and $U_C(3)$ denotes the 3rd column of matrix U.

2.3 Secret sharing method

The *secret sharing method*, was introduced by Shamir in 1979 [56]. Numerous researchers have investigated such methods since then. The goal of sharing a secret is to distribute a secret among a group of users, each of whom is allocated a share of the secret. In a secret sharing method there are one dealer and n users. The dealer gives each user a share of the secret in such a way that any group of t or more users can together reconstruct the secret but no group of less than t users can. Such a system is also called a (t, n)-*threshold scheme*.

Here we illustrate how the secret sharing method works. Consider a (t, n)-threshold scheme and a secret value K. The dealer randomly chooses a large prime q, and selects a (t-1)-degreed polynomial $f(x) = (a_0 + a_1x + \cdots + a_{t-1}x^{t-1}) \bmod q$ in which $a_0 = K$ and $a_1, a_2, \ldots, a_{t-1}$ are randomly chosen from a uniform distribution over the integers in $[0, q)$. Next, the dealer computes the shares for individual users:

$$k_1 = f(ID_1), k_2 = f(ID_2), \ldots, k_n = f(ID_n).$$

Given any subset of t of these k_i values (together with their identities), the users can find the coefficients of $f(x)$ by interpolation, and then obtain the secret $K = f(0)$.

2.4 Related works

In this section, we provide a detailed survey of various user authentication protocols in wireless networks, including password-based user authentication, biometrics-based user authentication, and self-certificate-based user authentication.

2.4.1 Password-based user authentication

In 1981, Lamport [33] proposed the first password authentication protocol for remote users over an insecure channel. Since then, several protocols [11, 12, 13, 15, 21, 22, 23, 25, 29, 31, 32, 37, 38, 39, 45, 48, 49, 57, 64, 65] have been proposed to improve security, efficiency, and functionality. Past experience has shown that constructing a secure user authentication protocol is not trivial because lots of proposed protocols were subsequently broken by well-known attacks [11, 21, 22, 23, 31, 32, 37, 45].

Traditionally, if a remote user wants to log into a server, he has to submit his identity and password to the server. On receiving the login request, the server first checks the validity of the identity and computes a one-way hash value of the received password, and then checks the computed value against the server's verification table. Since this approach clearly incurs the risk of tampering and the cost of managing the table, several protocols [12, 13, 15, 25, 29, 38, 39, 48, 49, 57, 64, 65] have been proposed that do not depend on a verification table.

Due to the constrained resources in smart cards, the computation and communication overhead must be low in practical implementation. Sun [57] proposed an efficient authentication protocol that adopts only simple hashing operations. In 2002, Chien et al. [13] proposed another authentication protocol that improves on Sun's in two ways: it achieves mutual authentication and it allows users to choose their passwords freely.

After a user is authenticated, the messages between the user and the server must be encrypted when transmitted over the public network. They have to agree on a session key. Juang [25] proposed an authentication protocol that provides a key agreement function. In various e-commerce applications, user anonymity is also crucial. Das et al. [15] first proposed a dynamic identity-based authentication protocol that preserves user anonymity. However, Chien and Chen [12] pointed out that Das et al.'s protocol [15] fails to protect user anonymity.

In order to reduce the risk of single-point failures, Choi and Youn [14] proposed a novel data encryption and distribution approach based on LU decomposition in 2004. The protocol allows higher security and availability compared with the mirroring protocol [19, 41, 44], and provides a solution for failures and malicious compromises of storage nodes, client systems, and user account. Pathan et al. [48, 49] also proposed two bilateral authentication protocols based on LU decomposition. However, their protocols have several security weaknesses, including (1) they cannot resist replay attacks; (2) passwords could be revealed by the server; (3) they cannot preserve user anonymity; and (4) the server and users cannot agree on a session key.

Even though a number of user authentication protocols with smart cards have been proposed, these existing protocols cannot be directly applied to user authentication in WSNs due to the limited computational power and energy supply in sensor nodes. In order to achieve better performance, Wong et al. [63] proposed the first password-based user authentication protocol for WSNs. Their protocol is efficient since the protocol participants perform only a few hash operations. Unfortunately, Tseng et al. [58] showed that Wong et al.'s protocol suffers from vulnerabilities to both replay and forgery attacks.

2.4.2 Biometrics-based user authentication

Over the past few years, many researchers have paid a lot of attention to remote user authentication protocols by combining biometrics and passwords [5, 10, 18, 20, 26, 27, 28, 36, 40, 54]. The most commonly used biometric techniques are fingerprint, face, iris, voice, and palm print etc. In 2002, Lee et al. [36] proposed a fingerprint-based remote user authentication protocol using smart cards. In this protocol, the server stores two secret keys and public parameters in a user's smart card. A user can access the smart card by his own fingerprint. However, Hsieh et al. [20] and Lin and Lai [40] pointed out that Lee et al.'s protocol [36] is vulnerable to impersonation attacks. Therefore, Lin and Lai [40] proposed an improved protocol to enhance the security, which allows users to choose and change their password freely.

In 2007, Khan and Zhang [26] demonstrated that Lin and Lai's protocol [40] is susceptible to the server spoofing attack since Lin and Lai's protocol [40] performs only unilateral authentication and there is no mutual authentication between user and remote server. Khan and Zhang [26] proposed an improved protocol which overcomes the weakness of Lin and Lai's protocol [40].

Recently, Fan and Lin [17] proposed a remote user authentication protocol with privacy protection on biometrics. Their protocol fully preserves the privacy of the biometric data of each user while allowing the server to verify the correctness of the users' biometric characteristics without knowing the exact values. However, in Fan and Lin's protocol [17], if an attacker eavesdrops a message sent by a legitimate user and replays it to log to the system in a later session, the server needs to perform one asymmetric decryption operation, one symmetric encryption operation, and two symmetric decryption operations to detect the replay login request. Therefore, dramatic increase in the number of replay login requests will certainly result in exhausting the server's resources. Furthermore, their protocol cannot allow users to change their passwords. If a user's password is compromised or a user wants to change the password for any reasons, there is no way to change the password. The only option for the user is to apply for a new card, which is an inefficient solution. In addition, compared with a regularly changed password, a fixed password is more vulnerable.

2.4.3 Self-certificate-based user authentication

In 2001, Perrig et al. [50] proposed security protocols for WSNs (SPINS), providing important security primitives: authenticated and confidential communication, and authenticated broadcast. They designed an authenticated routing protocol and a secure node-to-node key agreement protocol. User authentication in WSNs was proposed by Benenson et al. [7] in 2004. They investigated several security issues in WSNs, including access control, and also introduced the notion of (t, n)-*threshold authentication*, which means the authentication succeeds if the user can be successfully authenticated with at least $(n - t)$ out of n sensors. The rest of the sensors could be compromised or out of order. Thereafter, Benenson et al. [9] proposed the first solution to the user authentication problem in the presence of node-capture attacks. Their protocol is based on public-key cryptography, and is designed for a sensor node to authenticate the users.

In 2006, Banerjee and Mukhopadhyay [6] proposed authenticated querying in WSNs that is based on symmetric keys. The protocol can deal with queries involving multiple sensors. However, identifying the involved sensor nodes and flooding the access requests turn out to be very challenging for WSNs. Later, Wang and

Li [60] proposed a distributed user access control mechanism under a realistic adversary model for sensor networks. The protocol, which is based on an elliptic-curve cryptosystem (ECC), is divided into local authentication, which is conducted by the *local sensors*, that is, those sensors that are located physically close to the user, and remote authentication, which is based on the endorsement of the local sensors.

In order to achieve better performance, Wong et al. [63] proposed the first password-based user authentication protocol for WSNs. Compared with earlier works, their protocol is efficient since the protocol participants perform only a few hash operations. Unfortunately, Tseng et al. [58] showed that Wong et al.'s protocol suffers from vulnerabilities to both replay and forgery attacks and proposed an improved protocol. However, these protocols [58, 63] can only solve the access-control problem for individual sensor nodes, but not for the whole sensor networks.

Recently, Jiang et al. [24] proposed a user authentication protocol based on the self-certified-key cryptosystem [51] and used ECC to establish pair-wise keys between users and sensor nodes. However, the self-certified-key cryptosystem is not without security flaws. Lee and Kim [35] showed that the self-certified-key cryptosystem cannot provide explicit authentication for the public key. An attacker can produce a seemingly valid self-certified key with a third party's identity. This bogus key cannot be distinguished from a valid one until successful communication with the real owner of the identity. To solve the bogus key problem, they introduced the *self-certificate* for the self-certified key. It is a user-generated certificate for the authentication of the self-certified key.

Chapter 3

Password-based user authentication protocol

In this chapter, we propose two password-based user authentication protocols, namely protocol-I and protocol-II. The protocol-I is a password-based user authentication protocol using LU decomposition, which authenticates remote users and allows legitimate users to access network services over an open communication network. This protocol possesses many merits, including freely changeable passwords, mutual authentication, user anonymity, and session key agreement.

The protocol-II is a password-based user authentication protocol for WSNs, which allows legitimate users to query sensor data at any of the sensor node in an ad hoc manner and imposes very little computational overhead. Moreover, security of the proposed protocols is modelled and analyzed with Petri nets. Our analysis shows that the protocols can defend notorious attacks.

3.1 Protocol-I: Password-based user authentication protocol using LU decomposition

The proposed password-based user authentication protocol is divided into three phases: registration, login-and-authentication, and password-change phases.

3.1.1 Registration phase

Suppose a new user U_i with the identity ID_i wants to register with a server for remote-access services. U_i will take the following steps:

Step R1: U_i randomly chooses his password PW_i and sends the pair $(ID_i, h(PW_i))$ to the server in person or through an existing secure channel.

Step R2: Upon receiving the registration message, the server generates two random numbers x_i, y_i between 1 and n, and selects the x_i-th row from matrix L (denoted as $L_R(x_i)$), the x_i-th column from matrix U (denoted as $U_C(x_i)$), and the y_i-th column from matrix U (denoted as $U_C(y_i)$). Next, the server computes the pair $(K_{x_iy_i}, \theta_i)$ as follows: ($\oplus$ means the exclusive-or operation)

$$K_{x_iy_i} = L_R(x_i) \cdot U_C(y_i) \tag{3.1}$$

$$\theta_i = h(ID_i \oplus K_{x_iy_i}) \oplus h(PW_i) \oplus h(K_s) \tag{3.2}$$

Then the server issues a smart card containing $(K_{x_iy_i}, \theta_i, U_C(x_i), v_i, h(\cdot), g, p)$ to U_i, where $v_i = h(K_s) \oplus y_i$.

In the registration and password-change phases, in order to keep a user's password secret and resist insider attacks, the user transmits his password in hashed form, rather than as plain text. Note that Pathan et al.'s protocols [48, 49] make use of plain text for transmitting passwords. In addition, the system parameters g and p, where g is a generator of order q and p is a prime number which is divisible by $q-1$, used for computing a session key, have to be embedded in the smart card for later use.

3.1.2 Login-and-authentication phase

When U_i wants to log in to the system, U_i first attaches the smart card and inputs his password PW_i^*. The details are presented as follows.

Step L1: The smart card generates a random number r and computes the pair (H_i, S_i) as follows:

$$H_i = K_{x_i y_i} \oplus h(r \oplus T) \tag{3.3}$$

$$S_i = \theta_i \oplus h(PW_i^*) \oplus r \tag{3.4}$$

where T is the current timestamp. Next, the smart card generates a random number a and computes the pair (r_i, R_i):

$$r_i = g^a \bmod p. \tag{3.5}$$

$$R_i = h(\theta_i \oplus r_i) \tag{3.6}$$

After that, the smart card encrypts $(ID_i, r_i, U_C(x_i), v_i, T)$ with R_i and computes C_i:

$$\begin{aligned} C_i &= \theta_i \oplus h(ID_i \oplus K_{x_i y_i}) \oplus h(PW_i^*) \oplus R_i \\ &= h(K_s) \oplus R_i \end{aligned} \tag{3.7}$$

Finally, the smart card sends the login message $M_i = (C_i, E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T), H_i, S_i, T)$ to the server.

Step L2: Upon receiving the login request M_i, the server computes $R_i = C_i \oplus h(K_s)$, and decrypts $E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T)$ with R_i. Then the server checks the validity of ID_i and verifies whether the time interval $(T' - T) \leq \Delta T$, where T' is the current timestamp and ΔT is the allowed time interval for transmission delay. If so, the server computes $(v_i \oplus h(K_s))$, which is denoted as y_i, and computes the triple $(K_{y_ix_i}, t, r')$ as follows:

$$K_{y_ix_i} = L_R(y_i) \cdot U_C(x_i) \tag{3.8}$$

$$t = h(ID_i \oplus K_{y_ix_i}) \tag{3.9}$$

$$r' = S_i \oplus T \oplus h(K_s) \oplus t \tag{3.10}$$

After that, the server checks whether the equation holds as follows:

$$K_{y_ix_i} \stackrel{?}{=} H_i \oplus h(r') \tag{3.11}$$

If equation (3.11) holds, the server generates a random number b and computes r_s:

$$r_s = g^b \bmod p. \tag{3.12}$$

The server constructs the authenticated session key SK_i:

$$SK_i = {r_i}^b = g^{ab} \bmod p. \tag{3.13}$$

Finally, the server sends $E_{R_i}(K_{y_ix_i} \oplus r_s, r_i + 1, T'')$ to U_i.

Step L3: After receiving the message $E_{R_i}(K_{y_ix_i} \oplus r_s, r_i + 1, T'')$, the new user U_i decrypts the message to obtain $K_{y_ix_i} \oplus r_s$, and verifies whether $(T''' - T'') \leq \Delta T$, where T''' is the current timestamp. If so, U_i checks whether decrypted data contains the value $r_i + 1$. If so, U_i uses K_{x_iyi} to compute r_s as follows:

$$r_s = (K_{y_ix_i} \oplus r_s) \oplus K_{x_iyi} \tag{3.14}$$

Next, U_i generates the authenticated session key SK_i as follows:

$$SK_i = {r_s}^a = g^{ba} = g^{ab} \bmod p. \tag{3.15}$$

	User i			Server
L1.	Input PW_i^* Compute $H_i = K_{x_iy_i} \oplus h(r \oplus T)$ $S_i = \theta_i \oplus h(PW_i^*) \oplus r$ $r_i = g^a \bmod p$ $R_i = h(\theta_i \oplus r_i)$ Encrypt $(ID_i, r_i, U_C(x_i), v_i, T)$ with R_i Compute $C_i = \theta_i \oplus h(ID_i \oplus K_{x_iy_i})$ $\oplus h(PW_i^*) \oplus R_i$ Send $M_i = (C_i, E_{R_i}(ID_i, r_i, U_C(x_i)$ $, v_i, T), H_i, S_i, T)$ to the server	→	**L2.**	Receive M_i Compute $R_i = C_i \oplus h(K_s)$ Decrypt $E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T)$ Check ID_i Verify $(T' - T) \leq \Delta T$ Compute $y_i = v_i \oplus h(K_s)$ $K_{y_ix_i} = L_R(y_i) \cdot U_C(x_i)$ $t = h(ID_i \oplus K_{y_ix_i})$ $r' = S_i \oplus T \oplus h(K_s) \oplus t$ Verify $K_{y_ix_i} \overset{?}{=} H_i \oplus h(r')$ Compute $r_s = g^b \bmod p$ Construct $SK_i = {r_i}^b = g^{ab} \bmod p$ Send $E_{R_i}(K_{y_ix_i} \oplus r_s, r_i + 1, T'')$
L3.	Receive $E_{R_i}(K_{y_ix_i} \oplus r_s, r_i + 1, T'')$ Decrypt $E_{R_i}(K_{y_ix_i} \oplus r_s, r_i + 1, T'')$ Verify $(T''' - T'') \leq \Delta T$ Check $r_i + 1$ Compute $r_s = (K_{y_ix_i} \oplus r_s) \oplus K_{x_iyi}$ Construct $SK_i = {r_s}^a = g^{ba} = g^{ab}$	←		

Figure 3.1: The login-and-authentication phase of the password-based user authentication protocol using LU decomposition.

Then U_i uses SK_i to communicate with the server. A high-level depiction of the login-and-authentication phase in the proposed protocol is illustrated in Figure 3.1.

3.1.3 Password-change phase

When U_i wants to change his password PW_i to PW_i', the following steps will be performed.

Step P1: U_i sends the triple $(ID_i, h(PW_i), h(PW_i'))$ to the server. As in the registration phase, these private data should be submitted in person or via a secure channel.

Step P2: Upon receiving the password-change message, the server computes θ_i' as follows:

$$\begin{aligned} \theta_i' &= \theta_i \oplus h(PW_i) \oplus h(PW_i') \\ &= h(ID_i \oplus K_{x_i y_i}) \oplus h(PW_i') \oplus h(K_s) \end{aligned} \tag{3.16}$$

Next, the server replaces θ_i with θ_i' in the smart card.

As in the registration phase, the user has to transmit his password in hashed form in this phase to keep his password secret and withstand insider attacks.

3.2 Protocol-II: Password-based user authentication protocol for WSNs

In the proposed protocol, authorized users can access any of the sensor nodes in WSNs using mobile devices, such as PDAs, PCs, etc. Before issuing a query to a sensor node, a user has to register at the sensor gateway (GW) via a secure channel. Upon successful registration, the user can login to a nearest sensor login-node to retrieve sensor data. The proposed protocol is divided into three phases: registration, login-and-authentication, and password-change phases. Note that the registration and the password-change phases are performed via a secure channel.

3.2.1 Registration phase

Suppose a new user U_i with the identity ID_i wants to register with a GW for retrieving sensor data. U_i will take the following steps:

Step R1: U_i randomly chooses his password PW_i and sends the pair $(ID_i, h(PW_i))$ to the GW through a secure channel.

Step R2: Upon receiving the registration message, the GW stores the dataset $(ID_i, h(PW_i), T)$ in its database. Then, the GW replies to the user successful registration. Finally, the pair (ID_i, T) is then distributed to all the sensor nodes.

3.2.2 Login-and-authentication phase

When U_i wants to retrieve sensor data, U_i first inputs his/her password PW_i^*. The details are presented as follows.

Step L1: U_i computes A as follows:

$$A = h(h(PW_i^*) \oplus T') \tag{3.17}$$

where T' is the current timestamp. Next, U_i sends the triple (ID_i, A, T') to a login-node.

Step L2: Upon receiving the login request (ID_i, A, T'), the login-node first checks whether ID_i is in the list of datasets (ID_i, T). If not, the login-node then sends REJ-LOGIN to U_i. Otherwise it computes C for the user:

$$C = h(A \oplus T'') \tag{3.18}$$

where T'' is the current timestamp. Then, the login-node sends (ID_i, C, T'', T') to the GW for authentication.

Step L3: After receiving the message (ID_i, C, T'', T'), the GW first checks whether (ID_i, T') is in the database. If ID_i is not in the database or (ID_i, T') is already contained in the database, the GW sends REJ-LOGIN to the login-node. Otherwise, it checks whether the transmission delay is within the allowed time interval. If $(T''' - T'') \geq \Delta T$ or $(T'' - T') \geq \Delta T$, the GW sends REJ-LOGIN to the login-node. Otherwise, it computes the pair (A^*, C^*) for verification.

$$A^* = h(h(PW_i) \oplus T') \tag{3.19}$$

$$C^* = h(A^* \oplus T'') \tag{3.20}$$

The GW verifies if $C^* \overset{?}{=} C$. If so, the GW stores T' in the database and sends ACC-LOGIN to the login-node and the login-node also sends ACC-LOGIN to U_i. Otherwise, the GW sends REJ-LOGIN to the login-node. A high-level depiction of the login-and-authentication phase in the proposed protocol is illustrated in Figure 3.2.

3.2.3 Password-change phase

When U_i wants to change his password PW_i to PW_i', the following steps will be performed.

Step P1: U_i sends the triple $(ID_i, h(PW_i), h(PW_i'))$ to the GW. As in the registration phase, these private data should be submitted in person or via a secure channel.

Step P2: Upon receiving the password-change message, the GW first checks whether $(ID_i, h(PW_i))$ is correct. If ID_i is not in its database or $h(PW_i)$ is not

User i		**GW**		**Login-node**
L1. Input PW_i^* $A = h(h(PW_i^*) \oplus T')$ Send (ID_i, A, T') to the login-node				
			→	**L2.** Receive (ID_i, A, T') Check ID_i $C = h(A \oplus T'')$ Send (ID_i, C, T'', T') to the GW
		L3. Receive (ID_i, C, T'', T') Check ID_i Check transmission delay $A^* = h(h(PW_i) \oplus T')$ $C^* = h(A^* \oplus T'')$ $C^* \overset{?}{=} C$ Store T' Send ACC-LOGIN to the login-node and U_i	←	
Receive ACC-LOGIN	←		→	Receive ACC-LOGIN

Figure 3.2: The login-and-authentication phase of the password-based user authentication protocol for WSNs.

correct, the GW sends REJ-CHANGE to U_i. Otherwise, it updates the corresponding dataset with $(ID_i, h(PW_i'), T^*)$, where T^* is the current timestamp. Then, the GW replies to U_i successful password change. Finally, the new pair (ID_i, T^*) is then distributed to all the sensor nodes.

3.3 Security analysis

In this section, we use Petri nets [53] to model and analyze the proposed protocols. Security properties of the protocols will be specified. We also show that our proposed protocols can resist several notorious attacks. In addition, we provide a comparative study with other authentication protocols.

3.3.1 Correctness

According to equation (3.10), we first derive the equation as follows:

$$r' = S_i \oplus T \oplus h(K_s) \oplus t$$

$$\begin{aligned}
&= \theta_i \oplus h(PW_i^*) \oplus r \oplus T \oplus h(K_s) \oplus t \\
&= h(ID_i \oplus K_{x_iy_i}) \oplus h(PW_i) \oplus h(K_s) \oplus h(PW_i^*) \oplus r \oplus T \oplus h(K_s) \oplus t \\
&= h(ID_i \oplus K_{x_iy_i}) \oplus r \oplus T \oplus t \\
&= h(ID_i \oplus K_{x_iy_i}) \oplus r \oplus T \oplus h(ID_i \oplus K_{y_ix_i}) \\
&= r \oplus T \qquad (3.21)
\end{aligned}$$

Since the protocol-I employs LU decomposition, $K_{x_iy_i} = K_{y_ix_i}$. That is, $h(ID_i \oplus K_{x_iy_i}) \oplus h(ID_i \oplus K_{y_ix_i}) = 0$. Therefore, $r' = r \oplus T$.

Using equation (3.21), we verify equation (3.11) as follows:

$$\begin{aligned}
K_{y_ix_i} &= H_i \oplus h(r') \\
&= K_{x_iy_i} \oplus h(r \oplus T) \oplus h(r \oplus T) \\
&= K_{x_iy_i} \qquad (3.22)
\end{aligned}$$

3.3.2 Petri net model

The Petri net models of the protocol-I and protocol-II are illustrated in Figure 3.3 and Figure 3.5, respectively. We also construct attack scenarios in Figure 3.4 and Figure 3.6 for the protocol-I and protocol-II, respectively. The definitions of the places and transitions used in these models are listed in Table 3.1, Table 3.2, Table 3.3, and Table 3.4, respectively. We use the platform independent Petri net editor 2 (PIPE2) [1] to simulate the proposed protocols. The simulation results for the protocol-I and the protocol-II are bounded, which could be realized in hardware [52].

3.3.3 Security properties

The security of the protocol-I is based on the difficulty of DLP and DHP,

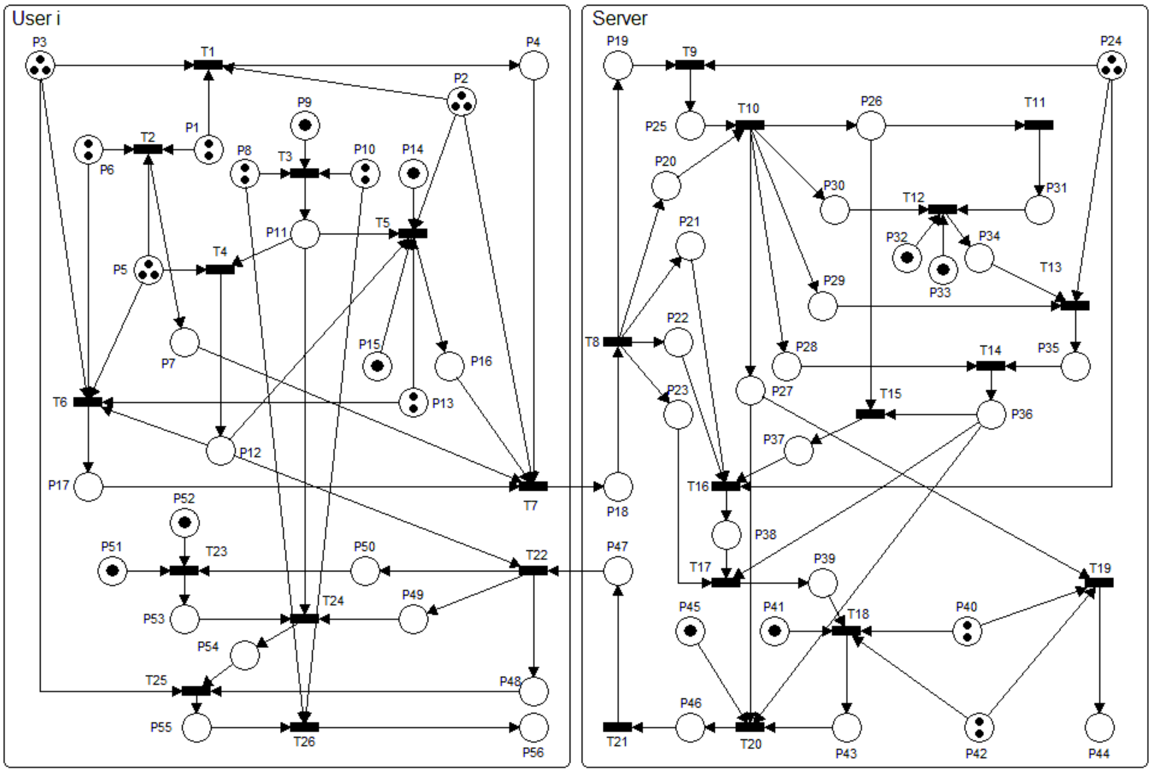

Figure 3.3: A Petri net model of the password-based user authentication protocol-I.

which are believed infeasible to solve in polynomial time. We will show that the protocol-I can resist replay attack, forgery attack, insider attack, reflection attack, and parallel session attack. We will also analyze the following security properties: anonymity, mutual authentication, forward secrecy, and known-key security.

Theorem 1. *The proposed protocol-I can resist a replay attack.*

Proof. Assume an adversary eavesdrops the login message sent by U_i and uses it to impersonate U_i when logging into the system in a later session. However, the replay of U_i's previous login message will be detected by the server since the user has already bound the timestamp T into the login message according to equation (3.3), and the server will verify the validity of the timestamp T used by U_i. As shown in Figure 3.3, computing H_i is defined in transition T_1, which has three

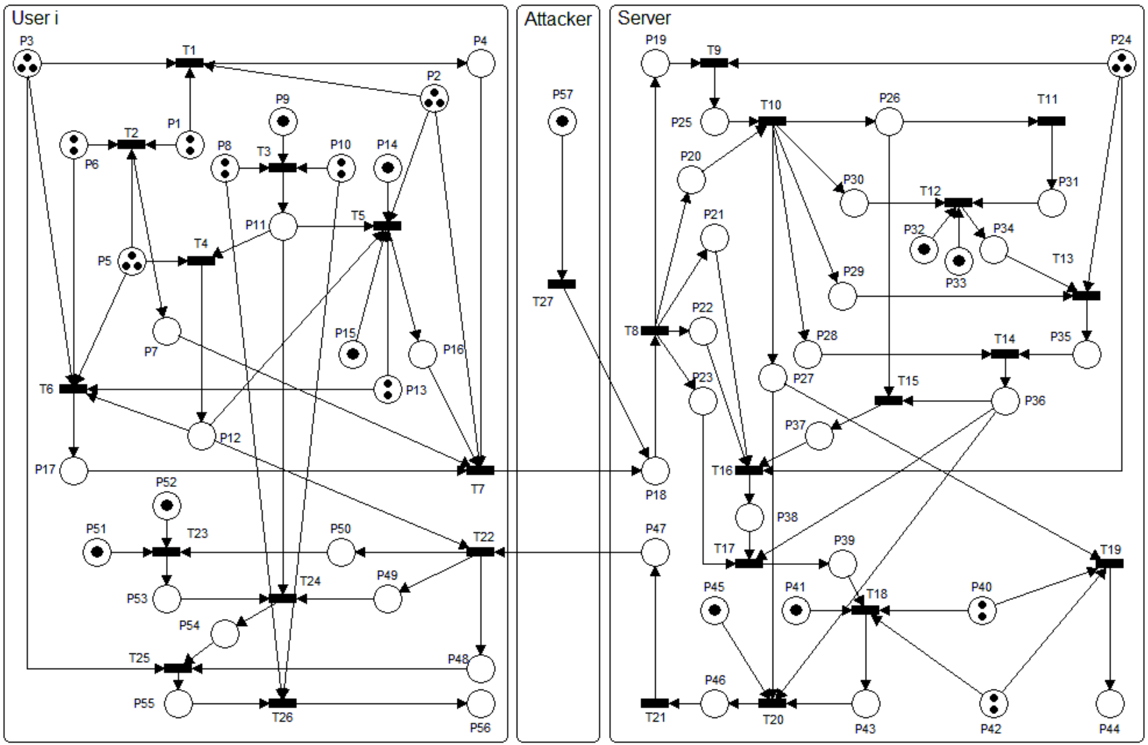

Figure 3.4: A Petri net model of the password-based user authentication protocol-I under an attack scenario.

input places, P_1, P_2, and P_3. Place P_2 is the value of T.

In Figure 3.4, when the adversary replays U_i's login message (P_{57}), the firing sequence is given below: $T_{27} \rightarrow T_8 \rightarrow T_9 \rightarrow T_{10} \rightarrow T_{11}$. However, there is a deadlock in the transition T_{12} since the server detects that the timestamp in the login message is not fresh. Therefore, the adversary cannot replay the login message. However, there seems to be one potential security threat common to most existing timestamp-based user authentication protocols. That is, an adversary could impersonate a legitimate user by replaying that user's previous login message within the allowed time interval ΔT. This threat can be solved by the additional requirement that T is not reused by U_i within ΔT. $\square$

Theorem 2. *The proposed protocol-I can resist a forgery attack.*

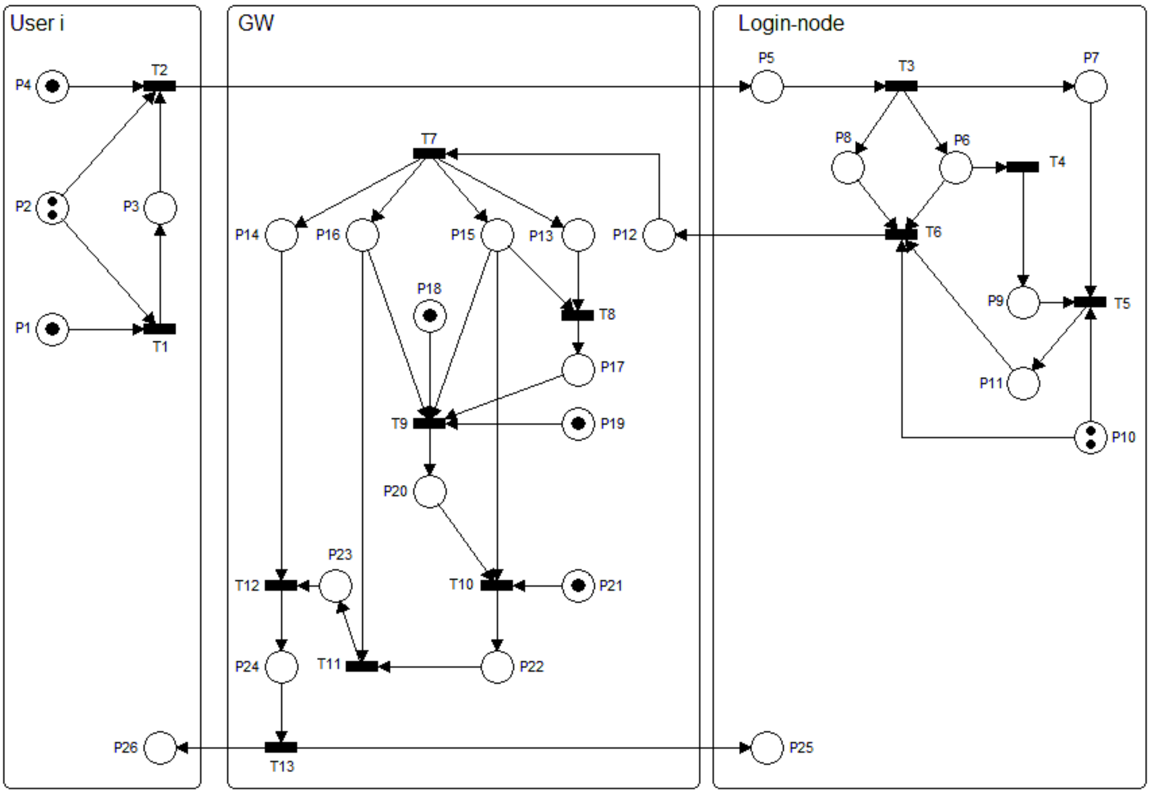

Figure 3.5: A Petri net model of the password-based user authentication protocol-II.

Proof. If the adversary wants to impersonate U_i, he has to create a valid login message $(C_i^*, E_{R_i^*}(ID_i, r_i^*, U_C(x_i), v_i, T^*), H_i^*, S_i^*, T^*)$, where T^* is the current timestamp. First he has to choose a random number r^* and compute the pair (H_i^*, S_i^*) as follows.

$$H_i^* = K_{x_i y_i} \oplus h(r^* \oplus T^*) \tag{3.23}$$

$$S_i^* = \theta_i \oplus h(PW_i) \oplus r^* \tag{3.24}$$

As shown in Figure 3.3, computing H_i is defined in transition T_1, which has three input places, P_1, P_2, and P_3. Place P_3 is the value of $K_{x_i y_i}$. Computing S_i is defined in transition T_2, which has three inputs, P_1, P_5 and P_6. Place P_5 is the

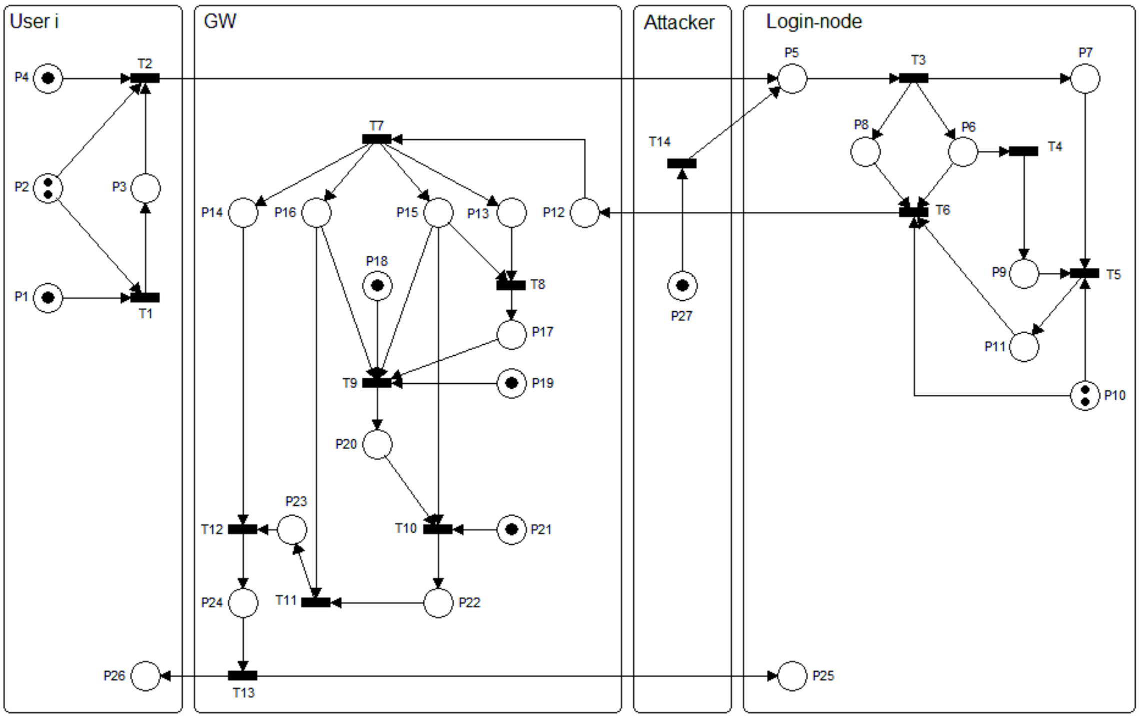

Figure 3.6: A Petri net model of the password-based user authentication protocol-II under an attack scenario.

value of θ_i and place P_6 is the value of PW_i^*.

Because having no idea about $K_{x_i y_i}$, θ_i, and PW_i, the adversary cannot forge a valid login message and hence cannot launch a forgery attack. □

Theorem 3. *The proposed protocol-I can resist an insider attack.*

Proof. In the protocol-I, when U_i wants to resigter with a server for remote-access services, he has to submit $(ID_i, h(PW_i))$ instead of (ID_i, PW_i), as in Pathan et al.'s protocols [48, 49]. Due to the employment of the one-way hash function h, it is considered practically impossible for the server to derive the user's password PW_i from the hashed value [55]. That is, even the server does not know PW_i. Obviously, the protocol-I can prevent the insider attack. □

Theorem 4. *The proposed protocol-I can resist a reflection attack.*

Table 3.1: Definitions of places for protocol-I

Place	**Definition**	**Place**	**Definition**
P_1	r	P_{29}	v_i
P_2	T	P_{30}	T
P_3	$K_{x_i y_i}$	P_{31}	Success verification message
P_4	H_i	P_{32}	T'
P_5	θ_i	P_{33}	ΔT
P_6	PW_i^*	P_{34}	Success verification message
P_7	S_i	P_{35}	y_i
P_8	a	P_{36}	$K_{y_i x_i}$
P_9	g	P_{37}	t
P_{10}	p	P_{38}	r_i'
P_{11}	r_i	P_{39}	Success verification message
P_{12}	R_i	P_{40}	b
P_{13}	ID_i	P_{41}	g
P_{14}	$U_C(x_i)$	P_{42}	p
P_{15}	v_i	P_{43}	r_s
P_{16}	$E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T)$	P_{44}	SK_i
P_{17}	C_i	P_{45}	T''
P_{18}	M_i	P_{46}	$E_{R_i}(K_{y_i x_i} \oplus r_s, r_i + 1, T'')$
P_{19}	C_i	P_{47}	$E_{R_i}(K_{y_i x_i} \oplus r_s, r_i + 1, T'')$
P_{20}	$E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T)$	P_{48}	$K_{y_i x_i} \oplus r_s$
P_{21}	T	P_{49}	$r_i + 1$
P_{22}	S_i	P_{50}	T''
P_{23}	H_i	P_{51}	T'''
P_{24}	K_s	P_{52}	ΔT
P_{25}	R_i	P_{53}	Success verification message
P_{26}	ID_i	P_{54}	Success verification message
P_{27}	r_i	P_{55}	r_s
P_{28}	$U_C(x_i)$	P_{56}	SK_i

Proof. A reflection attack is one in which, when a user sends a login message to a server, the adversary eavesdrops the message and sends it (or a modified version of the message) back to the user. In the proposed-I, the adversary cannot fool the server since he has to know the server's secret key K_s in computing R_i, which is used to decrypt the ciphertext $E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T)$ sent by U_i. As

Table 3.2: Definitions of transitions for protocol-I

Trans.	**Definition**	**Trans.**	**Definition**
T_1	Compute H_i	T_{14}	Compute $K_{y_i x_i}$
T_2	Compute S_i	T_{15}	Compute t
T_3	Compute r_i	T_{16}	Compute r'
T_4	Compute R_i	T_{17}	Verify $K_{y_i x_i} \overset{?}{=} H_i \oplus h(r')$
T_5	Compute $E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T)$	T_{18}	Compute r_s
T_6	Compute C_i	T_{19}	Compute SK_i
T_7	Transmit M_i	T_{20}	Compute $E_{R_i}(K_{y_i x_i} \oplus r_s, r_i + 1, T'')$
T_8	Split M_i	T_{21}	Transmit $E_{R_i}(K_{y_i x_i} \oplus r_s, r_i + 1, T'')$
T_9	Compute R_i	T_{22}	Decrypt $E_{R_i}(K_{y_i x_i} \oplus r_s, r_i + 1, T'')$
T_{10}	Decrypt $E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T)$	T_{23}	Check $(T''' - T'') \leq \Delta T$
T_{11}	Check ID_i	T_{24}	Check $r_i + 1$
T_{12}	Check $(T' - T) \leq \Delta T$	T_{25}	Compute r_s
T_{13}	Compute y_i	T_{26}	Compute SK_i

Table 3.3: Definitions of places for protocol-II

Place	**Definition**	**Place**	**Definition**
P_1	PW_i^*	P_{14}	C
P_2	T'	P_{15}	T'
P_3	A	P_{16}	T''
P_4	ID_i	P_{17}	Success verification message
P_5	(ID_i, A, T')	P_{18}	ΔT
P_6	ID_i	P_{19}	T'''
P_7	A	P_{20}	Success verification message
P_8	T'	P_{21}	$h(PW_i)$
P_9	Success verification message	P_{22}	A^*
P_{10}	T''	P_{23}	C^*
P_{11}	C	P_{24}	Success verification message
P_{12}	(ID_i, C, T', T'')	P_{25}	ACC-LOGIN
P_{13}	ID_i	P_{26}	ACC-LOGIN

illustrated in Figure 3.3, computing R_i is defined in transition T_9, which has two input places, P_{19} and P_{24}. Place P_{24} is the value of K_s. Therefore, it is ensured

Table 3.4: Definitions of transitions for protocol-II

Trans.	**Definition**	**Trans.**	**Definition**
T_1	Compute A	T_8	Check ID_i, T'
T_2	Transmit (ID_i, A, T')	T_9	Check the transmission delay
T_3	Split (ID_i, A, T')	T_{10}	Compute A^*
T_4	Check ID_i	T_{11}	Compute C^*
T_5	Compute C	T_{12}	Verify $C^* \overset{?}{=} C$
T_6	Transmit (ID_i, C, T'', T')	T_{13}	Store T' and transmit ACC-LOGIN
T_7	Split (ID_i, C, T'', T')		

that the protocol-I can withstand the reflect attack. □

Theorem 5. *The proposed protocol-I can resist a parallel-session attack.*

Proof. In the protocol-I, an adversary cannot impersonate a legitimate user by creating a valid login message in another on-going run from the honest run since the server's response message $E_{R_i}(K_{y_i x_i} \oplus r_s, r_i + 1, T'')$ is encrypted with R_i, which is unknown to the adversary. Therefore, the protocol-I can resist the parallel-session attack. □

Theorem 6. *The proposed protocol-I can provide user anonymity.*

Proof. If an adversary eavesdrops the login message, he cannot extract the user's identity from the ciphertext $E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T)$ since it is encrypted with R_i, which is unknown to the adversary. In addition, due to the use of the nonce and the timestamp in the login phase, the login messages submitted to the server are different in the login sessions. As shown in Figure 3.3, computing $E_{R_i}(ID_i, r_i, U_C(x_i), v_i, T)$ is defined in transition T_5, which has six places, P_2, P_{11}, P_{12}, P_{13}, P_{14}, and P_{15}. Place P_2 is the value of T and place P_{11} is the value of $r_i = g^a \bmod p$. Hence, it is difficult for the adversary to discover a user's identity. Clearly, the protocol-I can provide user anonymity. □

Theorem 7. *The proposed protocol-I can provide mutual authentication.*

Proof. The protocol-I uses the Diffie-Hellman key exchange algorithm to achieve mutual authentication between the server and a user. U_i and the server securely exchange r_i and r_s in the login and authentication phases, respectively. As a result, the authenticated session key is established as follows:

$$SK_i = r_i^b = r_s^a = g^{ab} \bmod p \tag{3.25}$$

As illustrated in Figure 3.3, computing SK_i is defined in transition T_{19} and T_{26}, which are computed by the server and U_i, respectively. Therefore, U_i and the server can use the authenticated session key SK_i in subsequent communications. □

Theorem 8. *The proposed protocol-I can provide perfect forward secrecy.*

Proof. Perfect forward secrecy means that the disclosure of the long-term secret key material (e.g., server's secret key K_s and user's password PW_i) does not compromise the secrecy of the agreed keys in earlier runs. In the protocol-I, perfect forward secrecy is ensured since the Diffie-Hellman key exchange algorithm is used to establish the authenticated session key g^{ab}. Even if the adversary knows the server's secret key K_s, he is only able to obtain g^a and g^b from earlier runs. As shown in Figure 3.3, computing $r_i(= g^a \bmod p)$ and $r_s(= g^b \bmod p)$ is defined in transition T_3 and T_{18}, respectively.

However, based on the difficulty of the discrete logarithm problem and the Diffie-Hellman problem, it is computationally infeasible to compute the authenticated session key g^{ab} from g^a and g^b. Thus, the protocol-I provides perfect forward secrecy. □

Theorem 9. *The proposed protocol-I can provide known-key security.*

Proof. Known-key security means that the compromise of a session key will not lead to further compromise of other secret keys or session keys. Even if a session key g^{ab} is revealed to an adversary, he still cannot derive other session keys since they are generated from the random numbers $g^{a'}$ and $g^{b'}$ based on Diffie-Hellman key exchange algorithm. Hence, the protocol-I can achieve known-key security. □

Now we will show that the protocol-II can resist replay attack, forgery attack, and insider attack.

Theorem 10. *The proposed protocol-II can resist a replay attack.*

Proof. Assume an adversary eavesdrops the login message sent by U_i and uses it to impersonate U_i when logging into the system in a later session. However, the replay of U_i's previous login message will be detected by the server since the user has already bound the timestamp T' into the login message according to equation (3.17), and the GW will checks whether (ID_i, T') exists in the database. If (ID_i, T') is already in the database, it means that this user has already login to this system at time T'. The GW then rejects the user's login request. As shown in Figure 3.5, computing A is defined in transition T_1, which has two input places, P_1 and P_2. Place P_2 is the value of T'.

In Figure 3.6, when the adversary replays U_i's login message (P_{27}), the firing sequence is given below: $T_{14} \rightarrow T_3 \rightarrow T_4 \rightarrow T_5 \rightarrow T_6 \rightarrow T_7$. However, there is a deadlock in the transition T_8 since the server detects that the timestamp in the login message is not fresh. Therefore, the adversary cannot replay the login message. Hence, the attacker cannot launch a replay attack. □

Theorem 11. *The proposed protocol-II can resist a forgery attack.*

Proof. In the protocol-II, even if the attacker gains the list stored in the sensor login-node, the protocol is still secure since there is no secret information stored in the sensor login-node. The hash values are useless to an attacker. In order to forge a login message, the attacker has to know the user's password, due to equation (3.17). However, it is difficult to derive the user's password from the hashed value A. It is considered practically impossible for an attacker to derive the user's password from the hashed value [55]. Because having no idea about the user's password, the adversary cannot forge a valid login message and hence cannot launch a forgery attack. □

Theorem 12. *The proposed protocol-II can resist an insider attack.*

Proof. In the protocol-II, when U_i wants to resigter with a GW for retrieving sensor data, he has to submit $(ID_i, h(PW_i))$. Due to the employment of the one-way hash function h, it is considered practically impossible for the GW to derive the user's password PW_i from the hashed value [55]. That is, even the server does not know PW_i. Obviously, the protocol-II can prevent the insider attack. □

3.3.4 Functionality

We summarize the functionality of the proposed-I in this subsection. The crucial criteria in a user authentication protocol are listed below:

C1. *Freely chosen password*: A user can choose his password freely in the registration phase.

C2. *Mutual authentication*: The server and a user can authenticate each other.

C3. *User anonymity*: A user's identity is protected when he logs into the system. No one knows the user's identity except the server.

C4. *Session key agreement*: While mutual authentication is established between the server and a user, they can agree on a session key for use in subsequent communications.

Table 3.5: Comparison of authentication protocols

	C1	**C2**	**C3**	**C4**	**C5**
Protocol-I	Yes	Yes	Yes	Yes	Yes
Pathan et al.'s protocol [49]	Yes	Yes	No	No	No
Hu et al.'s protocol [23]	Yes	Yes	Yes	Yes	Yes
Pathan et al.'s protocol [48]	Yes	Yes	No	No	Yes*
Chien and Chen's protocol [12]	Yes	Yes*	Yes	Yes	No
Das et al.'s protocol [15]	Yes	No	Yes*	No	No
Juang's protocol [25]	Yes	Yes	No	Yes	No
Chien et al.'s protocol [13]	Yes	Yes	No	No	No

C1: freely chosen password; C2: mutual authentication; C3: user anonymity; C4: session key agreement; C5: secure password change.
Yes*: Authors claimed such a security property but the property actually failed.

Table 3.6: Evaluation parameters

Symbol	**Definition**
T_H	Time for performing a one-way hash function
T_M	Time for performing a vector multiplication operation
T_{XOR}	Time for performing an XOR operation
T_{EXP}	Time for performing an exponentiation operation
T_{ENC}	Time for performing a symmetric encryption operation
T_{DEC}	Time for performing a symmetric decryption operation

C5. *Secure password change*: After the registration, a user can change his password freely.

We summarized the functionality of related authentication and key distribution protocols in Table 3.5.

3.4 Efficiency analysis

Now we first examine the performance of the protocol-I. The evaluation parameters are defined in Table 3.6. The time requirement of the protocol-I is summarized

Table 3.7: Performance of the protocol-I

Phase	**The server**	**A user**
Registration	$1T_M + 2T_H + 4T_{XOR}$	$1T_H$
Login-and-authentication	$1T_M + 2T_H + 8T_{XOR} + 2T_{EXP} + 1T_{ENC} + 1T_{DEC}$	$3T_H + 11T_{XOR} + 2T_{EXP} + 1T_{ENC} + 1T_{DEC}$
Password-change	$2T_{XOR}$	$2T_H$
Total	$2T_M + 4T_H + 14T_{XOR} + 2T_{EXP} + 1T_{ENC} + 1T_{DEC}$	$6T_H + 11T_{XOR} + 2T_{EXP} + 1T_{ENC} + 1T_{DEC}$

in Table 3.7. We use the computational overhead as the metrics to evaluate the performance of the protocol-I. In the protocol-I, only one hashing operation is required for a user to register and get his smart card. In the login-and-authentication phase, three hashing operations, eleven exclusive-or operations, two exponentiation operation, one symmetric encryption operation, and one symmetric decryption operation are needed for a user.

We can see from Table 3.7 that the exponentiation operations are required by the server and the user due to the requirements of key agreement and perfect forward secrecy. These operations might be expensive for smart cards nowadays. However, with an increasing demand for information security as today's security systems still have plenty of room for improvement, it is expected that the complicated computations will be widely adopted as a necessary security measure and hardware security enhancement for smart cards will become prevalent in the near future.

Now we examine the performance of the protocol-II. We can see from Table 3.8 that the computations between Wong et al.'s protocol [63] and our proposed protocol-II in the three phases (registration, login-and-authentication, and

Table 3.8: Performance comparison between Wong et al.'s protocol and the protocol-II

Phase	**Wong et al.'s protocol**	**Protocol-II**
Registration	$3T_H$	$1T_H$
Login-and-authentication	$4T_H + 4T_{XOR}$	$4T_H + 4T_{XOR}$
Password-change	Not supported	$2T_H$
Total	$7T_H + 4T_{XOR}$	$7T_H + 4T_{XOR}$

password-change) are very similar. Clearly, in these phases, our proposed protocol-II does not add additional computational cost. Compared with their protocol, the proposed protocol is also efficient.

Chapter 4

Biometrics-based user authentication protocol

In this chapter, we propose a biometrics-based remote user authentication protocol using smart cards. The protocol fully preserves the privacy of the biometric data of each user while allowing the server to verify the correctness of the users' biometric characteristics without knowing the exact values. The crucial merits include (1) it allows users to choose and change their passwords freely and hence gives users more convenience and security; (2) it achieves mutual authentication between a server and a user; (3) a server and a user can generate authenticated sessions keys so that later communication between them can proceed efficiently in protected mode to fulfill desired confidentiality.

In addition, the proposed protocol is later extended to a multi-party biometrics-based remote user authentication protocol by incorporating a secret sharing component [56]. Moreover, security of the proposed protocol is modelled and analyzed with Petri nets. Our analysis shows that the proposed protocol can successfully defend notorious attacks, including replay attacks, forgery attacks, stolen-smart-card attacks, reflection attacks, parallel-session attacks, and insider attacks, and suitable for smart cards with limited computing capability.

4.1 Proposed protocol

The proposed protocol is divided into three phases: registration, login-and-authentication, and password-change. Firstly, the server randomly chooses a string K_s as its secret key for symmetric encryption. Then, the server keeps the secret key K_s secret.

4.1.1 Registration phase

Suppose a new user U_i (with identity ID_i) wants to register with a server for remote-access services. He/she will take the following steps:

Step R1: User U_i randomly chooses his/her password PW_i, two random strings b_i and r_i, performs an iris scan, and computes S_i with his/her iris template TM_i:

$$S_i = r_i \oplus TM_i \tag{4.1}$$

Next, U_i sends the triple $(ID_i, h(b_i \oplus PW_i), S_i)$ to the server via a secure channel.

Step R2: Upon receiving the registration message, the server computes the triple (y_i, z_i, w_i):

$$y_i = E_{K_s}(ID_i \| S_i) \tag{4.2}$$

$$z_i = h(ID_i \| K_s) \oplus h(b_i \oplus PW_i) \tag{4.3}$$

$$w_i = h(h(ID_i \| K_s) \| h(b_i \oplus PW_i)) \tag{4.4}$$

Then, the server stores the tuple $(ID_i, y_i, z_i, w_i, h(\cdot))$ in a smart card and issues it to U_i via a secure channel.

Step R3: Finally, U_i encrypts b_i and r_i with the biometrics template TM_i and stores the sketch $\mathcal{E}_{TM_i}(b_i \| r_i)$ in the smart card. At this time, the smart contains the following information: ID_i, y_i, z_i, w_i, $h(\cdot)$, and $\mathcal{E}_{TM_i}(b_i \| r_i)$.

4.1.2 Login-and-authentication phase

When user U_i wants to login to the system, U_i first inputs his/her password PW_i^* and performs an iris scan to obtain TM_i^*. The details are presented as follows.

Step L1: The smart card retrieves $(b_i \| r_i)$ by decryption the sketch $\mathcal{E}_{TM_i}(b_i \| r_i)$ with TM_i^*, and then computes C_0 and checks whether the equation holds as follows:

$$C_0 = z_i \oplus h(b_i \oplus PW_i^*) \tag{4.5}$$

$$w_i \overset{?}{=} h(C_0 \| h(b_i \oplus PW_i^*)) \tag{4.6}$$

If equation (4.6) holds, U_i is a legitimate user and the smart card proceeds to the next step, otherwise, it rejects the login request. Next, the smart card computes the pair (S_i^*, C_1):

$$S_i^* = r_i \oplus TM_i^* \tag{4.7}$$

$$C_1 = C_0 \oplus u_i \tag{4.8}$$

where u_i is a string randomly chosen by the smart card. Then the smart card sends (y_i, C_1) to the server as a login request.

Step L2: After receiving the login request (y_i, C_1), the server first decrypts y_i to obtain $(ID_i \| S_i)$. The server checks the validity of ID_i. If so, the server keeps S_i for later use and computes C_2 to obtain u_i' as follows:

$$C_2 = h(ID_i \| K_s) \tag{4.9}$$

$$u_i' = C_1 \oplus C_2 \tag{4.10}$$

Next, the server computes the pair (C_3, C_4):

$$C_3 = h(C_1 \| u_i') \tag{4.11}$$

$$C_4 = C_2 \oplus v_i \tag{4.12}$$

where v_i is a string randomly chosen by the server. Then the server sends (C_3, C_4) back to the smart card.

Step L3: The smart card checks whether the equation holds as follows:

$$C_3 \stackrel{?}{=} h(C_1 \| u_i) \tag{4.13}$$

If equation (4.13) holds, the smart card can ensure that C_3 indeed comes from the original server. Then, the smart card computes the tuple (v_i', SK_i, C_5, C_6):

$$v_i' = C_4 \oplus C_0 \tag{4.14}$$

$$SK_i = h(u_i \| v_i') \tag{4.15}$$

$$C_5 = h(C_4 \| v_i') \tag{4.16}$$

$$C_6 = v_i' \oplus S_i^* \tag{4.17}$$

Finally, the smart card sends (C_5, C_6) to the server.

Step L4: Upon receiving (C_5, C_6), the server checks whether the equation holds as follows:

$$C_5 \stackrel{?}{=} h(C_4 \| v_i) \tag{4.18}$$

If so, the server computes S_i^*:

$$S_i^* = C_6 \oplus v_i \tag{4.19}$$

Finally, the server checks whether the matching score $\Delta(S_i^*, S_i)$ is beyond a pre-defined threshold value If so, the server accepts the login request of the smart card and computes SK_i:

$$SK_i = h(u_i' \| v_i) \tag{4.20}$$

A high-level depiction of the login-and-authentication phase in the proposed protocol is illustrated in Figure 4.1.

4.1.3 Password-change phase

When U_i wants to change his password PW_i to PW_i', he/she has to input the old password PW_i^* and perform an iris scan to obtain TM_i^*. The following steps will be performed.

Step P1: The smart card retrieves $(b_i \| r_i)$ by decryption the sketch $\mathcal{E}_{TM_i}(b_i \| r_i)$ with TM_i^*, and then computes C_0 and checks whether the equation holds as follows:

$$C_0 = z_i \oplus h(b_i \oplus PW_i^*) \tag{4.21}$$

$$w_i \overset{?}{=} h(C_0 \| h(b_i \oplus PW_i^*)) \tag{4.22}$$

If equation (4.22) holds, U_i is a legitimate user and the smart card proceeds to the next step, otherwise, it rejects the request.

Step P2: U_i inputs the new password PW_i'. The smart card computes the pair (z_i', w_i'):

$$z_i' = z_i \oplus h(b_i \oplus PW_i^*) \oplus h(b_i \oplus PW_i') \tag{4.23}$$

	User i			**Server**
L1.	Input PW_i^* and TM_i^*			
	Retrieve (b_i, r_i) with TM_i^*			
	Compute $C_0 = z_i \oplus h(b_i \oplus PW_i^*)$			
	Verify $w_i \overset{?}{=} h(C_0 \| h(b_i \oplus PW_i^*))$			
	Compute $S_i^* = r_i \oplus TM_i^*$			
	$C_1 = C_0 \oplus u_i$			
	Send (y_i, C_1) to the server	→	**L2.**	Receive (y_i, C_1)
				Obtain (ID_i, S_i) by decryption y_i
				Check ID_i
				Compute $C_2 = h(ID_i \| K_s)$
				$u_i' = C_1 \oplus C_2$
				$C_3 = h(C_1 \| u_i')$
				$C_4 = C_2 \oplus v_i$
L3.	Receive (C_3, C_4)	←		Send (C_3, C_4) to U_i
	Verify $C_3 \overset{?}{=} h(C_1 \| u_i)$			
	Compute $v_i' = C_4 \oplus C_0$			
	$SK_i = h(u_i \| v_i')$			
	$C_5 = h(C_4 \| v_i')$			
	$C_6 = v_i' \oplus S_i^*$			
	Send (C_5, C_6) to the server	→	**L4.**	Receive (C_5, C_6)
				Verify $C_5 \overset{?}{=} h(C_4 \| v_i)$
				Compute $S_i^* = C_6 \oplus v_i$
				Check the matching $\Delta(S_i^*, S_i)$
				Compute $SK_i = h(u_i' \| v_i)$

Figure 4.1: The login-and-authentication phase of the proposed protocol.

$$w_i' = h(C_0 \| h(b_i \oplus PW_i')) \tag{4.24}$$

Then the smart card replaces the old z_i and w_i with the new z_i' and w_i' in the smart card.

4.2 Multi-party biometrics-based authentication protocol

In this section, we extend the above biometrics-based authentication protocol for authenticating multiple parties. This extended protocol is essentially a (t, n)-threshold multi-party authentication protocol. Any group of t or more users can together reconstruct the authentication key with their own biometric data, passwords, and smart cards but no group of less than t users can. For the sake of brevity, the password-change phase is not provided in the multi-party biometrics-based authentication protocol.

4.2.1 Registration phase

Suppose a group of n users want to register with a server for remote-access services. Each of them will take the following steps at the same time.

Step R1: User U_i (with identity ID_i) randomly chooses a password PW_i, two random strings b_i and r_i, performs an iris scan, and computes S_i with his/her iris template TM_i:

$$S_i = r_i \oplus TM_i \tag{4.25}$$

Next, U_i sends the triple $(ID_i, h(b_i \oplus PW_i), S_i)$ to the server via a secure channel.

Step R2: Upon receiving n users' registration messages, the server first computes AK:

$$AK = h(b_1 \oplus PW_1) \oplus h(b_2 \oplus PW_2) \oplus \cdots \oplus h(b_n \oplus PW_n) \tag{4.26}$$

Then, the server randomly chooses a (t-1)-degreed polynomial $f(x)$ and computes the tuple (k_i, y_i, z_i, w_i) as follows:

$$f(x) = (AK + a_1 x + \cdots + a_{t-1} x^{t-1}) \bmod q \tag{4.27}$$

$$k_i = f(ID_i) \tag{4.28}$$

$$y_i = E_{K_s}(ID_i \| S_i \| k_i) \tag{4.29}$$

$$z_i = h(ID_i \| K_s) \oplus h(b_i \oplus PW_i) \tag{4.30}$$

$$w_i = h(h(ID_i \| K_s) \| h(b_i \oplus PW_i)) \tag{4.31}$$

Then, the server stores $(ID_i, y_i, z_i, w_i, h(\cdot), n)$ in a smart card and issues it to U_i via a secure channel.

Step R3: Finally, U_i encrypts b_i and r_i with the biometrics template TM_i and stores the sketch $\mathcal{E}_{TM_i}(b_i \| r_i)$ in the smart card. At this time, the smart contains the following information: ID_i, y_i, z_i, w_i, $h(\cdot)$, n, and $\mathcal{E}_{TM_i}(b_i \| r_i)$.

4.2.2 Login-and-authentication phase

When user U_i wants to login to the system, U_i first inputs his/her password PW_i^* and performs an iris scan to obtain TM_i^*. The details are presented as follows.

Step L1: The smart card retrieves $(b_i \| r_i)$ by decryption the sketch $\mathcal{E}_{TM_i}(b_i \| r_i)$ with TM_i^*, and then computes C_0 and checks whether the equation holds as follows:

$$C_0 = z_i \oplus h(b_i \oplus PW_i^*) \tag{4.32}$$

$$w_i \overset{?}{=} h(C_0 \| h(b_i \oplus PW_i^*)) \tag{4.33}$$

If equation (4.33) holds, U_i is a legitimate user and the smart card proceeds to the next step, otherwise, it rejects the login request. Next, the smart card computes the pair (S_i^*, C_1):

$$S_i^* = r_i \oplus TM_i^* \tag{4.34}$$

$$C_1 = C_0 \oplus u_i \tag{4.35}$$

where u_i is a string randomly chosen by the smart card. Then the smart card sends (y_i, C_1) to the server as a login request.

Step L2: After receiving the login request (y_i, C_1), the server first decrypts y_i to obtain $(ID_i \| S_i \| k_i)$. The server checks the validity of ID_i. If so, the server keeps S_i and k_i for later use and computes C_2 to obtain u'_i as follows:

$$C_2 = h(ID_i \| K_s) \tag{4.36}$$

$$u'_i = C_1 \oplus C_2 \tag{4.37}$$

Next, the server computes the pair (C_3, C_4):

$$C_3 = h(C_1 \| u'_i) \tag{4.38}$$

$$C_4 = C_2 \oplus v_i \tag{4.39}$$

where v_i is a string randomly chosen by the server. Then the server sends (C_3, C_4) back to the smart card.

Step L3: The smart card checks whether the equation holds as follows:

$$C_3 \overset{?}{=} h(C_1 \| u_i) \tag{4.40}$$

If equation (4.40) holds, the smart card can ensure that C_3 indeed comes from the original server. Then, the smart card computes the tuple (v'_i, SK_i, C_5, C_6):

$$v'_i = C_4 \oplus C_0 \tag{4.41}$$

$$SK_i = h(u_i \| v'_i) \tag{4.42}$$

$$C_5 = h(C_4 \| v'_i) \tag{4.43}$$

$$C_6 = v_i' \oplus S_i^* \tag{4.44}$$

Finally, the smart card sends (C_5, C_6) to the server.

Step L4: Upon receiving (C_5, C_6), the server checks whether the equation holds as follows:

$$C_5 \overset{?}{=} h(C_4 \| v_i) \tag{4.45}$$

If so, the server computes S_i^*:

$$S_i^* = C_6 \oplus v_i \tag{4.46}$$

Finally, the server checks whether the matching score $\Delta(S_i^*, S_i)$ is beyond a pre-defined threshold value If so, the server computes SK_i:

$$SK_i = h(u_i' \| v_i) \tag{4.47}$$

After receiving t login requests, the server reconstructs AK:

$$AK = f(0) = \sum_{s=1}^{t} k_s \prod_{j=1, j \neq s}^{t} \frac{-ID_j}{ID_s - ID_j} \bmod q \tag{4.48}$$

Finally, the server accepts the login request. A high-level depiction of the login-and-authentication phase in the multi-party authentication protocol is illustrated in Figure 4.2.

4.3 Security analysis

In this section, we use Petri nets [53] to model and analyze the proposed protocol, and show that our protocol can withstand the notorious attacks. In addition, we provide a comparative study with Fan and Lin's protocol [17]. In comparison with Fan and Lin's protocol, our protocol achieves better time efficiency.

User i		Server
L1. Input PW_i^* and TM_i^* Retrieve (b_i, r_i) with TM_i^* Compute $C_0 = z_i \oplus h(b_i \oplus PW_i^*)$ Verify $w_i \overset{?}{=} h(C_0 \| h(b_i \oplus PW_i^*))$ Compute $S_i^* = r_i \oplus TM_i^*$ $C_1 = C_0 \oplus u_i$		
Send (y_i, C_1) to the server	→	**L2.** Receive (y_i, C_1) Obtain (ID_i, S_i, k_i) by decryption y_i Check ID_i Compute $C_2 = h(ID_i \| K_s)$ $u_i' = C_1 \oplus C_2$ $C_3 = h(C_1 \| u_i')$ $C_4 = C_2 \oplus v_i$
L3. Receive (C_3, C_4) Verify $C_3 \overset{?}{=} h(C_1 \| u_i)$ Compute $v_i' = C_4 \oplus C_0$ $SK_i = h(u_i \| v_i')$ $C_5 = h(C_4 \| v_i')$ $C_6 = v_i' \oplus S_i^*$	←	Send (C_3, C_4) to U_i
Send (C_5, C_6) to the server	→	**L4.** Receive (C_5, C_6) Verify $C_5 \overset{?}{=} h(C_4 \| v_i)$ Compute $S_i^* = C_6 \oplus v_i$ Check the matching $\Delta(S_i^*, S_i)$ Compute $SK_i = h(u_i' \| v_i)$ After receiving t login requests, the server reconstructs AK: $AK = f(0)$ $= \sum_{s=1}^{t} k_s \prod_{j=1, j \neq s}^{t} \frac{-ID_j}{ID_s - ID_j} \bmod q$

Figure 4.2: The login-and-authentication phase of the multi-party authentication protocol.

4.3.1 Petri net model

The Petri net model of the proposed protocol is illustrated in Figure 4.3. We also construct attack scenarios in Figure 4.4. The definitions of the places and transitions used in this model are listed in Table 4.1 and Table 4.2, respectively.

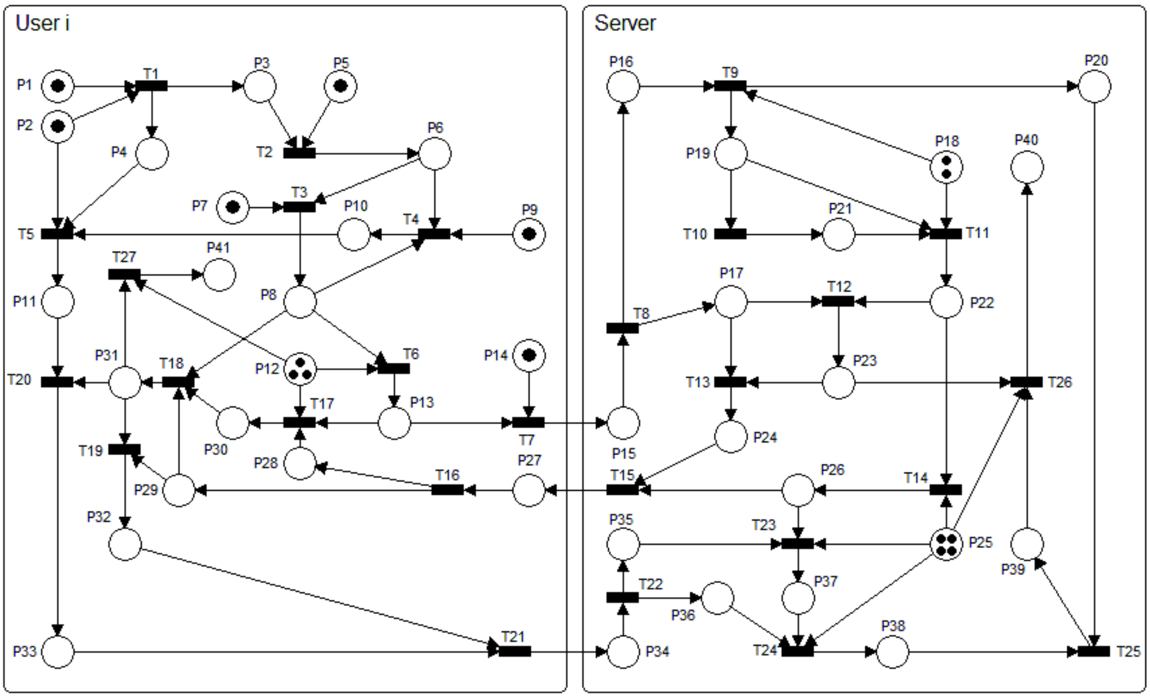

Figure 4.3: A Petri net model of the proposed biometrics-based user authentication protocol.

We use the platform independent Petri net editor 2 (PIPE2) [1] to simulate the protocol. The simulation result for the protocol is bounded, which could be realized in hardware [52].

4.3.2 Security properties

We now analyze the security properties of our protocol. We will show that our protocol can resist replay attacks, forgery attacks, stolen-smart-card attacks, reflection attacks, parallel-session attacks, and insider attacks. We will also analyze the following security properties: mutual authentication and known-key security.

Theorem 1. *The proposed protocol can resist a replay attack.*

Proof. Assume an adversary A eavesdrops the message (y_i, C_1) sent by U_i and replays it to log to the system in a later session. Upon receiving the replay message,

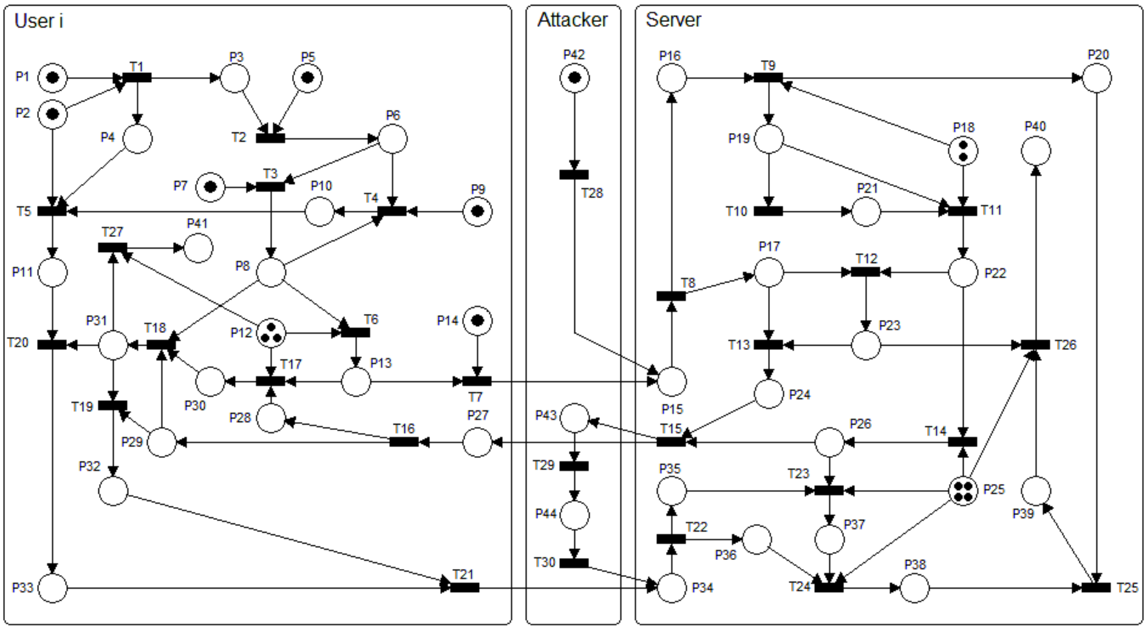

Figure 4.4: A Petri net model of the proposed biometrics-based user authentication protocol under an attack scenario.

the server first decrypts y_i to obtain $(ID_i \| S_i)$. The server checks the validity of ID_i, and then computes C_2 to obtain u_i' as follows:

$$C_2 = h(ID_i \| K_s) \tag{4.49}$$

$$u_i' = C_1 \oplus C_2 \tag{4.50}$$

Next, the server chooses a random string v_i^*, computes the pair (C_3, C_4^*), and sends (C_3, C_4^*) back to A.

$$C_3 = h(C_1 \| u_i') \tag{4.51}$$

$$C_4^* = C_2 \oplus v_i^* \tag{4.52}$$

After receiving the message, A has to recover v_i^* for constructing (C_5^*, C_6^*). However, A is unable to compute v_i^* due to lack of $C_0 (= h(ID_i \| K_s))$. In addition, A cannot just replay the message (C_5, C_6) obtained in the previous session directly

Table 4.1: Definitions of places

Place	**Definition**	**Place**	**Definition**
P_1	$\mathcal{E}_{TM_i}(b_i\|r_i)$	P_{22}	C_2
P_2	TM_i^*	P_{23}	u_i'
P_3	b_i	P_{24}	C_3
P_4	r_i	P_{25}	v_i
P_5	PW_i^*	P_{26}	C_4
P_6	$h(b_i \oplus PW_i^*)$	P_{27}	(C_3, C_4)
P_7	z_i	P_{28}	C_3
P_8	C_0	P_{29}	C_4
P_9	w_i	P_{30}	Success verification message
P_{10}	Success verification message	P_{31}	v_i'
P_{11}	S_i^*	P_{32}	C_5
P_{12}	u_i	P_{33}	C_6
P_{13}	C_1	P_{34}	(C_5, C_6)
P_{14}	y_i	P_{35}	C_5
P_{15}	(y_i, C_1)	P_{36}	C_6
P_{16}	y_i	P_{37}	Success verification message
P_{17}	C_1	P_{38}	S_i^*
P_{18}	K_s	P_{39}	Success verification message
P_{19}	ID_i	P_{40}	SK_i
P_{20}	S_i	P_{41}	SK_i
P_{21}	Success verification message		

since the random nonce v_i embedded in C_5 is different from v_i^* in this session. As shown in Figure 4.3, computing v_i' is defined in transition T_{18}, which has three input places, P_8, P_{29}, and P_{30}. Place P_8 is the value of C_0 and place P_{29} is the value of C_4.

In Figure 4.4, when the adversary replays U_i's login message (P_{42}), the firing sequence is given below: $T_{28} \rightarrow T_8 \rightarrow T_9 \rightarrow T_{10} \rightarrow T_{11} \rightarrow T_{12} \rightarrow T_{13} \rightarrow T_{14} \rightarrow T_{15} \rightarrow T_{29} \rightarrow T_{30} \rightarrow T_{22}$. However, there is a deadlock in the transition T_{23} since the random nonce v_i is different from v_i^* in this session. Therefore, the adversary cannot launch a replay attack. □

Table 4.2: Definitions of transitions

Trans.	**Definition**	**Trans.**	**Definition**
T_1	Retrieve (b_i, r_i) with TM_i^*	T_{15}	Transmit (C_3, C_4)
T_2	Perform hash function to compute $h(b_i \oplus PW_i^*)$	T_{16}	Split (C_3, C_4)
T_3	Compute C_0	T_{17}	Check $C_3 \overset{?}{=} h(C_1 \| u_i)$
T_4	Check $w_i \overset{?}{=} h(C_0 \| h(b_i \oplus PW_i^*))$	T_{18}	Compute v_i'
T_5	Compute S_i^*	T_{19}	Compute C_5
T_6	Compute C_1	T_{20}	Compute C_6
T_7	Transmit (y_i, C_1)	T_{21}	Transmit (C_5, C_6)
T_8	Split (y_i, C_1)	T_{22}	Split (C_5, C_6)
T_9	Decrypt y_i with K_s	T_{23}	Check $C_5 \overset{?}{=} h(C_4 \| v_i)$
T_{10}	Check ID_i	T_{24}	Compute S_i^*
T_{11}	Compute C_2	T_{25}	Check $\Delta(S_i^*, S_i)$
T_{12}	Compute u_i'	T_{26}	Compute SK_i
T_{13}	Compute C_3	T_{27}	Compute SK_i
T_{14}	Compute C_4		

Theorem 2. *The proposed protocol can resist a forgery attack.*

Proof. If an adversary A wants to impersonate U_i, he has to create a valid login message (y_i, C_1^*). First A has to choose a random string u_i^* and compute C_1^* as follows.

$$\begin{aligned} C_1^* &= C_1 \oplus u_i^* && (4.53) \\ &= C_0 \oplus u_i \oplus u_i^* && (4.54) \end{aligned}$$

Then, A sends (y_i, C_1^*) to the server. After decryption y_i to check the validity of ID_i, the server computes C_2 to obtain u_i' as follows:

$$C_2 = h(ID_i \| K_s) \tag{4.55}$$

$$\begin{aligned} u_i' &= C_1^* \oplus C_2 && (4.56) \\ &= u_i \oplus u_i^* && (4.57) \end{aligned}$$

Next, the server chooses a random string v_i^*, computes the pair (C_3^*, C_4^*), and sends (C_3^*, C_4^*) back to A.

$$C_3^* = h(C_1^* \| u_i') \tag{4.58}$$

$$C_4^* = C_2 \oplus v_i^* \tag{4.59}$$

After receiving the message, A has to recover v_i^* for constructing (C_5^*, C_6^*). However, A is unable to compute v_i^* due to lack of $C_0(= h(ID_i \| K_s))$. As shown in Figure 4.3, computing v_i' is defined in transition T_{18}, which has three input places, P_8, P_{29} and P_{30}. Place P_8 is the value of C_0 and place P_{29} is the value of C_4. Because having no idea about C_0 for constructing (C_5^*, C_6^*), the adversary has no chance to login by launching a forgery attack. □

Theorem 3. *The proposed protocol can resist a stolen-smart-card attack.*

Proof. Assume an adversary A obtains U_i's smart card and intercepts the messages (y_i, C_1), (C_3, C_4), and (C_5, C_6) transmitted between U_i and the server in the login-and-authentication phase. That is, the protocol is only under the protection of the password and the biometric data. Due to lack of U_i's biometric template TM_i^* to retrieve b_i from $\mathcal{E}_{TM_i}(b_i \| r_i)$ to pass the password verification (equation (4.6)), A will fail at the beginning of the login-and-authentication phase. As a result, it is difficult for A to derive the password. As shown in Figure 4.3, retrieving b_i is defined in transition T_1, which has two input places, P_1 and P_2. Place P_2 is the value of TM_i^*. The password verification is defined in transition T_4, which has three input places, P_6, P_8, and P_9. Place P_6 is the value of $h(b_i \oplus PW_i^*)$, place P_8 is the value of C_0, and place P_9 is the value of w_i. Without the user's biometrics template TM_i^*, the illegal request will be rejected. Obviously, the proposed protocol is secure against the stolen-smart-card attack. □

Theorem 4. *The proposed protocol can resist a reflection attack.*

Proof. When an honest user sends a login message to a server, an adversary A eavesdrops/intercepts the message and sends it (or a modified version of it) back to the original user. However, A cannot impersonate a legitimate server successfully since he/she must know the secret key K_s for computing $C_2(= h(ID_i \| K_s))$. As shown in Figure 4.3, computing C_2 is defined in transition T_{11}, which has three input places, P_{18}, P_{19}, and P_{21}. Place P_{18} is the value of K_s and place P_{19} is the value of ID_i. According to the above analysis, it is ensured that our protocol can withstand the reflection attack. $\square$

Theorem 5. *The proposed protocol can resist a parallel-session attack.*

Proof. In our proposed protocol, an adversary A cannot impersonate a legitimate user by creating a valid login message in another on-going run since the server responses different v_i in C_4 in each session. As shown in Figure 4.3, computing C_4 is defined in transition T_{14}, which has two input places, P_{22} and P_{25}. Place P_{22} is the value of C_2 and place P_{25} is the value of v_i. Therefore, the proposed protocol can resist the parallel-session attack. $\square$

Theorem 6. *The proposed protocol can resist an insider attack.*

Proof. In our proposed protocol, when U_i wants to register with a server for remote-access services, he has to submit $(ID_i, h(b_i \oplus PW_i), S_i)$ instead of $(ID_i, h(PW_i), S_i)$, as in Fan and Lin's protocol [17]. Due to the employment of the one-way hash function $h(\cdot)$, it is considered practically impossible for the server to derive the user's password PW_i from the hashed value [55]. Moreover, as b_i is not revealed to the server, the insider of the server cannot obtain PW_i by performing an offline guessing attack on $h(b_i \oplus PW_i)$. That is, even the server does not know PW_i. In addition, the proposed protocol does not maintain any verifier table. Obviously, the proposed protocol can prevent the insider attack. $\square$

Theorem 7. *The proposed protocol can provide mutual authentication.*

Proof. An adversary A cannot impersonate U_i or a server since the adversary does not have U_i's biometrics template TM_i, U_i's password PW_i, and the server's secret key K_s to obtain the correct u_i and v_i, which are randomly chosen by U_i and the server in messages C_1 and C_4, respectively. Using equation (4.20), the session key between U_i and the server is established as follows:

$$\begin{aligned} SK_i &= h(u_i' \| v_i) && (4.60) \\ &= h(u_i \| v_i') && (4.61) \end{aligned}$$

As shown in Figure 4.3, computing a session key is defined in transition T_{26} and T_{27}. Therefore, the proposed protocol achieves mutual authentication between a user and a server. □

Theorem 8. *The proposed protocol can provide known-key security.*

Proof. Known-key security means that the compromise of a session key will not lead to further compromise of other secret keys or session keys. Even if a session key SK_i is revealed to an adversary, he still cannot derive other session keys since each key generated in one protocol round is independent. Hence, the proposed protocol can achieve known-key security. □

4.4 Efficiency analysis

In this section, we summarize the performance of our proposed protocol. The evaluation parameters are defined in Table 4.3. The performance comparison between Fan and Lin's protocol [17] and the proposed protocol is presented in Table 4.4 and Table 4.5. We use the computational overhead as the metric to evaluate the performance of authentication protocols. Table 4.4 and Table 4.5 show the efficiency comparisons of the two protocols required by the users and the server, respectively.

In Table 4.4, the computation overhead between Fan and Lin's protocol and our proposed protocol in the registration phase is very similar. For the login-and-authentication phase, there is no need to perform asymmetric encryption operation in smart card for a user in our proposed protocol. Only five hash operations, six exclusive-or operations, and one symmetric decryption operation for a user in our protocol. Therefore, from the user's perspective, our proposed protocol achieves better time efficiency than Fan and Lin's protocol [17]. For the password-change phase, four hash operations, four exclusive-or operations, and one symmetric decryption operation are needed for a user in our protocol.

From Table 4.5, for the login-and-authentication phase, four hash operations, three exclusive-or operations, and one symmetric decryption operation are needed for the server in our proposed protocol. Obviously, our proposed protocol achieves better time efficiency than Fan and Lin's protocol [17]. As the number of login-and-authentications increases, the performance differences between the Fan and Lin's protocol [17] and the proposed protocol will be significant. Due to the energy constraint of smart cards and the cost of implementation, the lower the computational overhead, the greater the chance of success in practical implementation.

Table 4.3: Evaluation parameters

Symbol	**Definition**
T_H	Time for performing a one-way hash function
T_{XOR}	Time for performing an XOR operation
T_{AENC}	Time for performing an asymmetric encryption operation
T_{ADEC}	Time for performing an asymmetric decryption operation
T_{SENC}	Time for performing a symmetric encryption operation
T_{SDEC}	Time for performing a symmetric decryption operation

Table 4.4: Performance comparison between Fan and Lin's protocol and the proposed protocol (per user)

Phase	**Fan and Lin's protocol**	**The proposed protocol**
Registration	$1T_H + 1T_{XOR} + 1T_{SENC}$	$1T_H + 2T_{XOR} + 1T_{SENC}$
Login-and-Authentication	$2T_H + 1T_{XOR} + 1T_{AENC} + 1T_{SENC} + 2T_{SDEC}$	$5T_H + 6T_{XOR} + 1T_{SDEC}$
Password-change	Not supported	$4T_H + 4T_{XOR} + 1T_{SDEC}$

Table 4.5: Performance comparison between Fan and Lin's protocol and the proposed protocol (for the server)

Phase	**Fan and Lin's protocol**	**The proposed protocol**
Registration	$1T_{SENC}$	$2T_H + 1T_{XOR} + 1T_{SENC}$
Login-and-Authentication	$1T_H + 1T_{ADEC} + 1T_{SENC} + 2T_{SDEC}$	$4T_H + 3T_{XOR} + 1T_{SDEC}$
Password-change	Not supported	No computation cost*

No computation cost*: The proposed protocol allows users to change the passwords in local without notifying the server.

Chapter 5

Self-certificate-based user authentication protocol

To control access to WSNs, it is essential for sensor nodes to authenticate the users. Compared with symmetric-key cryptography widely used in WSNs, public-key cryptography provides a more flexible interface that requires no complicated key pre-distribution and management as in symmetric-key protocols [60, 61]. Over the past few years, elliptic-curve cryptosystem (ECC) has attracted considerable attention as ECC devices have higher strength per key bit, lower power consumption, and smaller bandwidth compared to RSA cryptosystems [30, 34]. For example, an elliptic curve over a 163-bit field gives the same level of security as a 1024-bit RSA modulus [34]. In addition, the recent progress in 160-bit ECC implementation shows that an ECC point multiplication takes less than one second, which proves that ECC is feasible for resource-constrained platforms such as wireless devices [42, 61, 62].

As completely preventing any physical captures is a costly option, it is cheaper to design security protocols for WSNs that can tolerate a certain number of node captures [6]. Therefore, we propose a user authentication protocol for WSNs based on ECC. This protocol can withstand capture of up to t sensor nodes. The proposed protocol is based on self-certificates, which enable users to generate their own certificates and to change their key pairs without the involvement of the KDC. A self-certificate is first generated by a user A and is encrypted with A's private key. The receiver of the self-certificate verifies the self-certificate with A's public key. The receiver can trust A's public key because it is endorsed by a trusted third party, such as a KDC.

Additionally, the proposed protocol provides many desired features: (1) it can deal with authenticated queries involving multiple sensor nodes; (2) it achieves mutual authentication and key agreement between users and sensor nodes; (3) it provides a KDC to revoke compromised key pairs. Moreover, Petri nets [53] may be used to infer what an attacker could know if he happens to know certain items in the security protocol. We used Petri nets in the security analysis of the proposed protocol. Our analysis shows that the proposed protocol can successfully defend several notorious attacks, including replay attacks, forgery attacks, and node-capture attacks.

5.1 Proposed protocol

We assume a public-key infrastructure (PKI) for ECC [9, 24, 42, 60, 61, 62]. There is a KDC in WSNs, which has a private/public key pair and is responsible for generating the private/public key pairs for users and sensor nodes. Prior to deployment, each user and sensor node has the public key of the KDC preloaded. With that public key, each entity can verify the certificates endorsed by the KDC.

In addition, we assume a large static sensor network. Each sensor node is assumed to have the same transmission range and communicates with each other

Table 5.1: Formal definition of a self-certificate

Let (S_i, Q_i) be entity (sensor or user) i's private/public key pair issued by the KDC, and CI_i be entity i's certificate information. Entity i signs on (CI_i, Q_i) with his private key S_i to generate:

$$\textit{Self-Cert}_i = \textit{Sign}_{S_i}(CI_i, Q_i)$$

Then $\textit{Self-Cert}_i$ is called a self-certificate for the public key Q_i.

via bi-directional wireless channels. A user can send data requests to the sensor nodes within his communication range and receives valid responses if the requests are legitimate. Note that when a node of WSNs is physically captured by an adversary, all the secrets stored in that node could be revealed. Because completely preventing any physical captures is a costly option, it is cheaper to design security protocols for WSNs that can tolerate a certain number of node captures [6]. On average, there are n sensors in the communication range of the user. Of these, t sensors are allowed to be malicious or to fail. It is assumed that $t < n/2$, i.e. the majority of sensors are honest. The assumption is reasonable since compromising sensors takes time and effort. Therefore, the user can rely on communication among at least a half of sensors in his communication range. Our proposed protocol still works well even if the adversary captures t nodes out of n nodes in the WSNs. We call the proposed protocol a (t, n)-*threshold authentication protocol.*

The proposed protocol is divided into four phases: pre-deployment, login-and-authentication, user-controlled key change, and key revocation. We define a self-certificate in Table 5.1.

5.1.1 Pre-deployment phase

Firstly, the KDC defines an elliptic curve over a prime Galois field $GF(q_1)$ and chooses a base point P with order q_2 belonging to this elliptic curve group. Then, it randomly selects a number $s \in GF(q_2)$ as its private key and performs the point multiplication $s \cdot P$ on the elliptic curve to compute its public key K_{pub}.

For every entity (sensor or user) i, the KDC generates its identity and private/public key pair as follows:

1. Randomly choose $ID_i \in GF(q_2)$ as entity i's identity.

2. Perform the point multiplication $r_i \cdot P$ to compute R_i, where r_i is a random number, i.e. $R_i = r_i \cdot P$.

3. Prepare the certificate information CI_i as follows:

   $$CI_i = [CertNo||ID_i||ID_{KDC}||R_i||P||K_{pub}||ValidPeriod] \tag{5.1}$$

   where $CertNo$ is the certificate serial number and $ValidPeriod$ is the valid time period of the certificate.

4. Generate entity i's private key S_i and perform the point multiplication to compute the corresponding public key Q_i as follows:

   $$S_i = s \cdot h(CI_i) + r_i \tag{5.2}$$

   $$Q_i = S_i \cdot P \tag{5.3}$$

5. Send (CI_i, S_i, Q_i) to entity i via a secure channel.

Upon receiving (CI_i, S_i, Q_i), entity i signs (CI_i, Q_i) with its private key S_i and generates the self-certificate of the public key Q_i as follows:

$$\textit{Self-Cert}_i = Sign_{S_i}(CI_i, Q_i) \tag{5.4}$$

The overall operation of the pre-deployment phase is illustrated in Figure 5.1.

	KDC		**Entity (sensor or user)** i
1.	Choose $ID_i \in GF(q_2)$		
2.	Compute $R_i = r_i \cdot P$		
3.	Prepare CI_i		
4.	Generate $S_i = s \cdot h(CI_i) + r_i$		
5.	Generate $Q_i = S_i \cdot P$		
6.	Send (CI_i, S_i, Q_i)	$\rightarrow$	Receive (CI_i, S_i, Q_i)
7.			Generate $\mathit{Self\text{-}Cert}_i = Sign_{S_i}(CI_i, Q_i)$

Figure 5.1: The pre-deployment phase of the proposed protocol.

5.1.2 Login-and-authentication phase

When user i wishes to query sensor data, he communicates with the sensor nodes within his communication range. The detailed steps are as follows.

1. $U_i \rightarrow WSNs : \{CI_i, Q_i, R_i, \mathit{Self\text{-}Cert}_i\}$

   U_i broadcasts his certificate information CI_i, public key Q_i, signature parameter R_i, and the self-certificate $\mathit{Self\text{-}Cert}_i$. Let $COMM_i$ denote the set of sensor nodes within the communication range of U_i.

2. Every $j \in COMM_i$: verify Q_i and $\mathit{Self\text{-}Cert}_i$

   Each sensor node $j \in COMM_i$ checks the validity of U_i's public key Q_i and the self-certificate $\mathit{Self\text{-}Cert}_i$. Sensor node j computes $K_{pub} \cdot h(CI_i) + R_i$ and checks if $Q_i = S_i \cdot P$ as follows:

   Note that

$$\begin{aligned} K_{pub} \cdot h(CI_i) + R_i &= s \cdot P \cdot h(CI_i) + r_i \cdot P \\ &= (s \cdot h(CI_i) + r_i) \cdot P \\ &= S_i \cdot P \end{aligned} \qquad (5.5)$$

The operations in equation (5.5) are performed on the elliptic curve. Sensor node j then extracts CI_i and Q_i from $\textit{Self-Cert}_i$ with the public key Q_i and checks if CI_i and Q_i are correct.

3. Every $j \in \textit{COMM}_i : j \rightarrow U_i : \{CI_j, Q_j, R_j, \textit{Self-Cert}_j, MAC_{K_{j,i}}(m_j)\}$

   If sensor node j successfully authenticates U_i, it performs the point multiplication $S_j \cdot Q_i$ to compute the pair-wise key $K_{j,i}$, i.e. $K_{j,i} = S_j \cdot Q_i$. Then, it chooses a random nonce m_j and calculates the message authentication code (MAC) [43] with $K_{j,i}$.

4. U_i : verify Q_j and $\textit{Self-Cert}_j$

   U_i verifies whether sensor node j's public key Q_j and the self-certificate $\textit{Self-Cert}_j$ are valid. If so, he performs the point multiplication $S_i \cdot Q_j$ to compute the pair-wise key $K_{i,j}$, i.e. $K_{i,j} = S_i \cdot Q_j$.

5. $U_i \rightarrow \textit{WSNs}$: compute and broadcast $\{v\}$

   U_i decrypts the MAC with the corresponding pair-wise key $K_{i,j}$ and obtains the nonce m'_j. This is because:

$$\begin{aligned} K_{i,j} &= S_i \cdot Q_j \\ &= S_i \cdot S_j \cdot P \\ &= Q_i \cdot S_j \\ &= K_{j,i} \end{aligned} \tag{5.6}$$

   The operations in equation (5.6) are performed on the elliptic curve. Upon collecting all the nonces, he constructs the authentication value $v = m'_1 || \cdots || m'_n$ and then broadcasts $\{v\}$.

	User i		**Sensor node** $j \in COMM_i$
1.	Broadcast $\{CI_i, Q_i, R_i, \textit{Self-Cert}_i\}$	$\longrightarrow$	Receive $\{CI_i, Q_i, R_i, \textit{Self-Cert}_i\}$
2.			Verify Q_i and $\textit{Self-Cert}_i$
3.			Generate m_j
4.			Compute $K_{j,i} = S_j \cdot Q_i$
5.			Compute $MAC_{K_{j,i}}(m_j)$
6.	Receive $\{CI_j, Q_j, R_j, \textit{Self-Cert}_j,$	$\longleftarrow$	Send $\{CI_j, Q_j, R_j, \textit{Self-Cert}_j,$
7.	$MAC_{K_{j,i}}(m_j)\}$		$MAC_{K_{j,i}}(m_j)\}$
8.	Verify Q_j and $\textit{Self-Cert}_j$		
9.	Compute $K_{i,j} = S_i \cdot Q_j$		
10.	Compute $\{v\}$		
11.	Broadcast $\{v\}$	$\longrightarrow$	Receive $\{v\}$
12.			Verify $m_j \in v$

Figure 5.2: The login-and-authentication phase of the proposed protocol.

6. Every $j \in COMM_i$: verify $m_j \in v$

   Each sensor node $j \in COMM_i$ verifies whether U_i correctly responds to the challenge by checking whether m_j is in v. If so, the sensor node broadcasts to other nodes its *yes* vote. Otherwise, it remains silent. If $(n - t)$ or more *yes* votes are collected, the sensor node believes U_i is a legitimate user. Note that in some situations, there could be bogus votes. To deal with the bogus-vote problem, the sensor nodes could use the pair-wise keys to encrypt the votes and related information, such as sensor nodes' identities and the timestamps, before broadcasting the encrypted messages.

The overall operation of the login-and-authentication phase is illustrated in Figure 5.2.

5.1.3 User-controlled key change phase

A fixed key pair is much easier to attack than a frequently changing one. In certificate-based protocols, changing a key pair usually requires complicated interaction between a user and a KDC. In our protocol, a user can change his key pair without interaction with the KDC.

Let (S_i, Q_i) be user i's private/public key pair issued by the KDC and let $\textit{Self-Cert}_i$ be the self-certificate issued by U_i. He can generate a new key pair (S_i', Q_i') and a new certificate $\textit{Self-Cert}_i'$ with the following operations.

1. Perform the point multiplication $r_i' \cdot P$ to compute R_i', where r_i' is a random number, i.e. $R_i' = r_i' \cdot P$.

2. Generate a new private key S_i' and perform the point multiplication to compute the corresponding public key Q_i' as follows:

$$S_i' = S_i \cdot h(CI_i || R_i') + r_i' \tag{5.7}$$

$$Q_i' = S_i' \cdot P \tag{5.8}$$

3. Generate the self-certificate $\textit{Self-Cert}_i'$ by signing (CI_i, Q_i') with his new private key S_i' as follows:

$$\textit{Self-Cert}_i' = Sign_{S_i'}(CI_i, Q_i') \tag{5.9}$$

Once the new public key Q_i' and the self-certificate $\textit{Self-Cert}_i'$ are generated, U_i will broadcast $\{CI_i, Q_i', R_i', \textit{Self-Cert}_i'\}$. Every sensor node $j \in COMM_i$ computes $K_{pub} \cdot h(CI_i) \cdot h(CI_i || R_i') + R_i \cdot h(CI_i || R_i') + R_i'$ and checks if $Q_i' = S_i' \cdot P$.

Note that

$$\begin{aligned} & K_{pub} \cdot h(CI_i) \cdot h(CI_i || R_i') + R_i \cdot h(CI_i || R_i') + R_i' \\ &= (s \cdot h(CI_i) \cdot h(CI_i || R_i') \cdot P) + (r_i \cdot h(CI_i || R_i') \cdot P) + R_i' \end{aligned}$$

$$
\begin{aligned}
&= (s \cdot h(CI_i) + r_i) \cdot h(CI_i||R'_i) \cdot P + R'_i \\
&= S_i \cdot h(CI_i||R'_i) \cdot P + r'_i \cdot P \\
&= (S_i \cdot h(CI_i||R'_i) + r'_i) \cdot P \\
&= S'_i \cdot P \qquad (5.10)
\end{aligned}
$$

The operations in equation (5.10) are performed on the elliptic curve. Sensor node j then extracts CI_i and Q'_i from *Self-Cert*$'_i$ with the public key Q'_i and checks if CI_i and Q'_i are correct. If both conditions hold, sensor node j performs step 3 in the login-and-authentication phase.

5.1.4 Key revocation phase

When a certified key pair is found compromised, the KDC can revoke it with a *certificate revocation list* (CRL). The KDC publishes CRL containing the serial numbers of all the certificates for the revoked key pair. Anyone who wants to verify a self-certificate should check the CRL first. Once the certificates of the compromised key are revoked, the compromised key can no longer be used to gain access to sensor data. More details on certificate revocation and certificate update can be found in [47].

5.2 Security analysis

In this section, we show that our protocol can resist several notorious attacks. In addition, we provide a comparative study with other user authentication protocols.

5.2.1 Petri net model

The Petri net model is illustrated in Figure 5.3. We also construct attack scenarios in Figure 5.4. The definitions of the places and transitions used in this

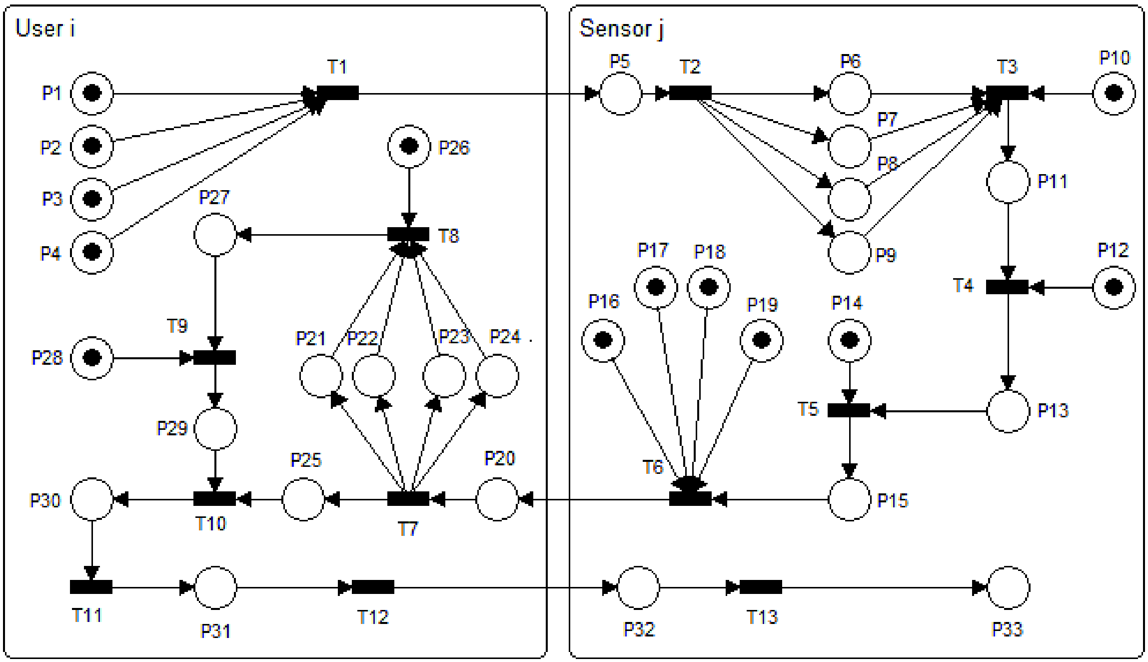

Figure 5.3: A Petri net model of the proposed self-certificate-based user authentication protocol.

model are listed in Table 5.2 and Table 5.3, respectively. The model is simulated with the platform independent Petri net editor 2 (PIPE2) [1]. The simulation result for the protocol is bounded, which could be realized in hardware [52].

5.2.2 Security properties

The security of the proposed protocol is based on the difficulty of the elliptic-curve discrete logarithm problem (ECDLP), which is believed to be unsolvable in polynomial time. Let G_1 be a group of the prime order q and P be an arbitrary generator of G_1. We view G_1 as an additive group.

Now we show that the proposed protocol can resist replay attacks, forgery attacks, and node-capture attacks, and also analyze the security property: mutual authentication.

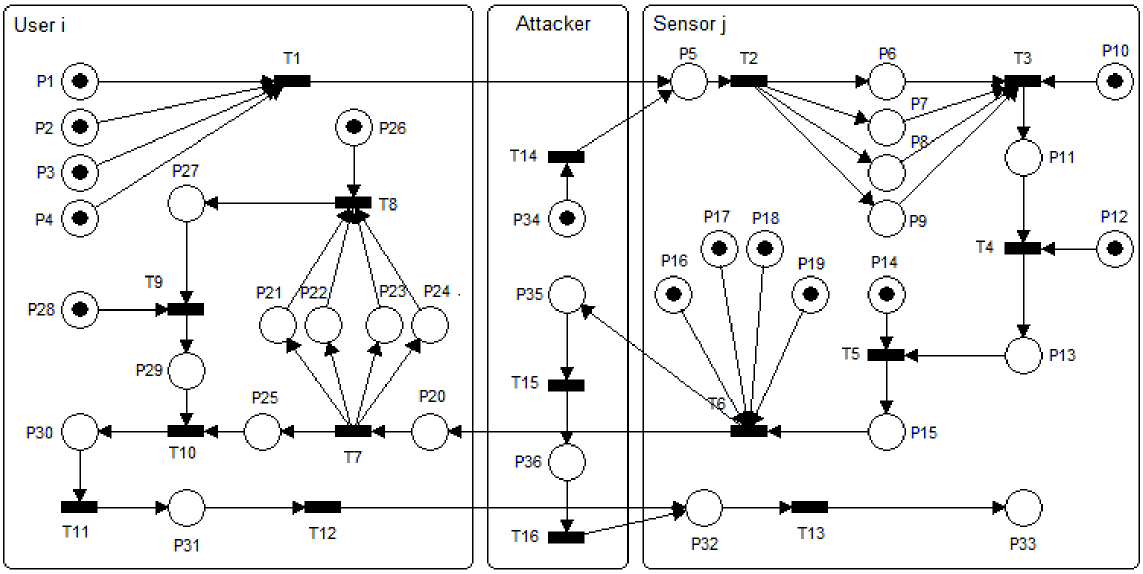

Figure 5.4: A Petri net model of the proposed self-certificate-based user authentication protocol under an attack scenario.

Theorem 1. *The proposed protocol can resist a replay attack.*

Proof. Assume an adversary A eavesdrops the messages $\{CI_i, Q_i, R_i, \textit{Self-Cert}_i\}$ and $\{v\}$ sent by U_i and replays them to log in to the system in a later session. Upon receiving the replay message, sensor node j first verifies Q_i and $\textit{Self-Cert}_i$, and then chooses a random nonce m_j^*. Next, j computes $MAC_{K_{j,i}}(m_j^*)$ and sends $\{CI_j, Q_j, R_j, \textit{Self-Cert}_j, MAC_{K_{j,i}}(m_j^*)\}$ back to A. After receiving the message, A has to compute $v^* = m_1''||\cdots||m_n''$ and broadcast $\{v^*\}$ back to the WSNs. However, A cannot just replay the message $\{v\}$ directly since the random nonce m_j embedded in $MAC_{K_{j,i}}(m_j)$ is different from m_j^* in this session. As shown in Figure 5.3, computing m_j is defined in transition T_{10}, which has two input places, P_{25} and P_{29}. Place P_{25} is the value of $MAC_{K_{j,i}}(m_j)$ and place P_{29} is the value of $K_{i,j}$.

In Figure 5.4, when the adversary replays U_i's login message (P_{34}), the firing sequence is given below: $T_{14} \rightarrow T_2 \rightarrow T_3 \rightarrow T_4 \rightarrow T_5 \rightarrow T_6 \rightarrow T_{15} \rightarrow T_{16}$. However,

Table 5.2: Definitions of places

Place	**Definition**	**Place**	**Definition**
P_1	CI_i	P_{18}	R_j
P_2	Q_i	P_{19}	$\textit{Self-Cert}_j$
P_3	R_i	P_{20}	$Packet\{CI_j, Q_j, R_j, \textit{Self-Cert}_j,$
P_4	$\textit{Self-Cert}_i$		$MAC_{K_{j,i}}(m_j)\}$
P_5	$Packet\{CI_i, Q_i, R_i, \textit{Self-Cert}_i\}$	P_{21}	CI_j
P_6	CI_i	P_{22}	Q_j
P_7	Q_i	P_{23}	R_j
P_8	R_i	P_{24}	$\textit{Self-Cert}_j$
P_9	$\textit{Self-Cert}_i$	P_{25}	$MAC_{K_{j,i}}(m_j)$
P_{10}	K_{pub}	P_{26}	K_{pub}
P_{11}	Success verification message	P_{27}	Success verification message
P_{12}	S_j	P_{28}	S_i
P_{13}	$K_{j,i}$	P_{29}	$K_{i,j}$
P_{14}	m_j	P_{30}	m'_j
P_{15}	$MAC_{K_{j,i}}(m_j)$	P_{31}	$v = m'_1 \| \cdots \| m'_n$
P_{16}	CI_j	P_{32}	$Packet\{v\}$
P_{17}	Q_j	P_{33}	Success verification message

there is a deadlock in the transition T_{13} since the random nonce m_j embedded in $MAC_{K_{j,i}}(m_j)$ is different from m_j^* in this session. Because having no idea about $K_{i,j}$ to correctly respond the challenge m_j^*, the adversary cannot launch a replay attack. □

Theorem 2. *The proposed protocol can resist a forgery attack.*

Proof. Assume an attacker A impersonates user i by submitting $\{CI_i, Q_i, R_i, \textit{Self-Cert}_i\}$ obtained in a previous session. Upon receiving the message, sensor node j first performs the authentication operations. Then j sends $\{CI_j, Q_j, R_j, \textit{Self-Cert}_j, MAC_{K_{j,i}}(m_j^*)\}$ back to A. However, A cannot decrypt $MAC_{K_{j,i}}(m_j^*)$ since he does not have user i's private key, which is needed for computing the pair-wise key $K_{i,j}$. As shown in Figure 5.3, computing the pair-wise key $K_{i,j}$ is

Table 5.3: Definitions of transitions

Trans.	**Definition**	**Trans.**	**Definition**
T_1	Transmit	T_7	Split the packet
	$\{CI_i, Q_i, R_i, \mathit{Self\text{-}Cert}_i\}$	T_8	Verify Q_j and $\mathit{Self\text{-}Cert}_j$
T_2	Split the packet	T_9	Compute $K_{i,j}$
T_3	Verify Q_i and $\mathit{Self\text{-}Cert}_i$	T_{10}	Decrypt $MAC_{K_{j,i}}(m_j)$
T_4	Compute $K_{j,i}$		with $K_{i,j}$
T_5	Compute $MAC_{K_{j,i}}(m_j)$	T_{11}	Compute $v = m'_1 \| \cdots \| m'_n$
T_6	Transmit $\{CI_j, Q_j, R_j,$	T_{12}	Broadcast $\{v\}$
	$\mathit{Self\text{-}Cert}_j, MAC_{K_{j,i}}(m_j)\}$	T_{13}	Check $m_j \overset{?}{=} m'_j$

defined in transition T_9, which has two input places, P_{27} and P_{28}. Place P_{28} is the value of S_i. If A could compute U_i's private key somehow, he would have broken the elliptic-curve discrete logarithm problem (ECDLP) as defined in Definition 3. The discrete logarithm problem can be reduced to the problem of computing the private key S_i from the public key $Q_i = S_i \cdot P$. In addition, even if the adversary obtains multiple pair-wise keys $K_{i,j}$, it is intractable to compute S_i due to the hardness of the ECDLP problem. Thus, we claim that computing the private key from the public key and the pair-wise key is at least as difficult as the elliptic-curve discrete logarithm problem. As a result, our protocol is secure against the forgery attacks.□

Theorem 3. *The proposed protocol can resist a node-capture attack.*

Proof. It is assumed that $t < n/2$, i.e. the majority of sensors are honest. Due to the voting stage in the login-and-authentication phase, if a sensor node can collect at least $(n - t)$ *yes* votes, the sensor node believes the user is legitimate. Hence, our protocol can tolerate up to t nodes being captured. □

Theorem 4. *The proposed protocol can provide mutual authentication.*

Proof. The security of the pair-wise key is based on the difficulty of ECDLP, which are believed to be unsolvable in polynomial time. Using equation (5.6), the pair-wise key between U_i and sensor node j is established as follows:

$$K_{i,j} = S_i \cdot Q_j = S_i \cdot S_j \cdot P = Q_i \cdot S_j = K_{j,i} \tag{5.11}$$

As shown in Figure 5.3, computing a pair-wise key is defined in transition T_4 and transition T_9. Therefore, U_i and sensor node j can use the pair-wise key $K_{i,j}$ in subsequent communications. □

5.2.3 Functionality

We summarize the functionality of our proposed protocol in this subsection. The crucial requirements for a user authentication protocol are listed below:

C1. (t, n)*-threshold authentication*: A protocol can deal with authenticated queries involving multiple sensor nodes and still works well even if the adversary captures t nodes out of n nodes in the WSNs.

C2. *Mutual authentication*: A user and a sensor node can authenticate each other.

C3. *Key agreement*: After successful authentication, a user and a sensor node mutually agree upon pair-wise keys.

C4. *User-controlled key change*: A user can change his key pair without interaction with a key distribution center.

C5. *Key revokability*: An issued key pair can be revoked, say, when it is found compromised.

We summarize the functionality of related authentication protocols in Table 5.4.

5.3 Efficiency analysis

Now we examine the performance of our proposed protocol. We use the com-

Table 5.4: Comparison of user authentication protocols for WSNs

	C1	**C2**	**C3**	**C4**	**C5**
Our proposed protocol	Yes	Yes	Yes	Yes	Yes
Benenson et al.'s protocol [9]	No	No	No	No	No
Benenson et al.'s protocol [8]	Yes	No	No	No	No
Banerjee et al.'s protocol [6]	Yes	No	No	No	No
Wang et al.'s protocol [60]	Yes	No	Yes	No	No
Jiang et al.'s protocol [24]	Yes	Yes	Yes	No	No
Wong et al.'s protocol [63]	No	No	No	No	No
Tseng et al.'s protocol [58]	No	No	No	No	No
Yu et al.'s protocol [67]	No	No	No	No	No

C1: (t, n)-threshold authentication; C2: mutual authentication; C3: key agreement; C4: user-controlled key change; C5: key revokability.

putational and communication overhead as the metric to evaluate the performance of the proposed protocol. Due to the similarity of network scenarios, we compare our proposed protocol with Jiang et al.'s protocol [24], which is presented in Table 5.5, Table 5.6. We only compare the computational overhead in two phases (pre-deployment and login-and-authentication) since Jiang et al.'s protocol did not include the user-controlled key change and key revocation phases. As illustrated in Table 5.5, the computational overhead in Jiang et al.'s protocol and our protocol in the pre-deployment phase is very similar. The only difference is that each entity needs to generate a self-certificate in our protocol.

As shown in Table 5.6, one certificate verification is required for each sensor node during the login-and-authentication phase in our protocol. If a user generates a new key, it takes one more hash operation and two more point multiplications for each sensor node in order to verify the new key. Hence, compared with Jiang et al.'s protocol, our protocol provides various functionalities at the cost of one

Table 5.5: Performance comparison in the pre-deployment phase

Computational type	**Jiang et al.'s protocol**		**Our protocol**	
	KDC	Each entity	KDC	Each entity
Random number generation	3	0	3	0
Hash operation	1	0	1	0
Point multiplication	3	0	3	0
Certificate generation*	–	–	0	1

Certificate generation*: Jiang et al.'s protocol [24] provides no certificate generation.

certificate verification for each sensor node.

The communication overhead is in terms of the following three aspects: the communication overhead incurred by broadcasting the messages from a user to sensors within his transmission range, the overhead incurred by delivering a response from a sensor to a user, and the overhead incurred by transmitting *yes* votes between sensors. In our analysis, we assume a key length of 160 bits in the ECC cryptosystem. As stated in Section 5.1.2, the user broadcasts $\{CI_i, Q_i, R_i, \textit{Self-Cert}_i\}$ in step 1 and $\{v\}$ in step 5. The length of the certificate information CI_i is 184 bytes, as shown in Figure 5.5. Q_i and R_i each costs 40 bytes. Assume the $\textit{Self-Cert}_i$ is constructed by the elliptic-curve digital signature algorithm (ECDSA) [3, 4]. The length of the $\textit{Self-Cert}_i$ is 40 bytes. Thus, the communication overhead incurred by broadcasting the messages from a user to sensors is $(304 + |v|)$ bytes.

As stated in Section 5.1.2, when a sensor transmits $\{CI_j, Q_j, R_j, \textit{Self-Cert}_j, MAC_{K_{j,i}}(m_j)\}$ to a user in step 3, as shown in Figure 5.6, it will cost each sensor 324 bytes. Upon correctly verifying the user, the sensor broadcasts a *yes* vote to other nodes, which costs $(n-1) \times |\textit{yes}\ \text{vote}|$ bytes. Note that the sensor nodes could use the pair-wise keys to encrypt the votes and related information to avoid the bogus-vote problem. The total communication overhead is listed in Table 5.7.

Table 5.6: Performance comparison in the login-and-authentication phase

Computational type	**Jiang et al.'s protocol**		**Our protocol**	
	Each node	Each user	Each node	Each user
Random number generation	1	0	1	0
Hash operation	1	n^*	$1\ (2)^{**}$	n
Symmetric encryption	1	0	$1\ (n)^{***}$	0
Symmetric decryption	0	n	$0\ (n)^{***}$	n
Point multiplication	2	$2n$	$2\ (4)^{**}$	$2n$
Certificate verification****	–	–	1	n

n^*: Assume there are n sensors in the communication range of the user.
$(2)^{**}$: If a changed key is used, it takes one more hash operation and two more point multiplications for each sensor node.
$(n)^{***}$: To deal with the bogus-vote problem, the sensor nodes could use the pair-wise keys to encrypt and decrypt the votes and related information.
Certificate verification****: Jiang et al.'s protocol [24] does not include certificate verification.

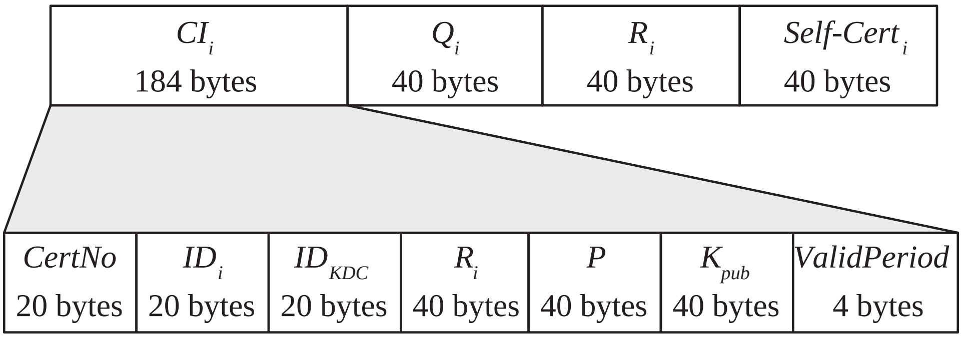

Figure 5.5: Broadcasting message format from a user to sensors in the login-and-authentication.

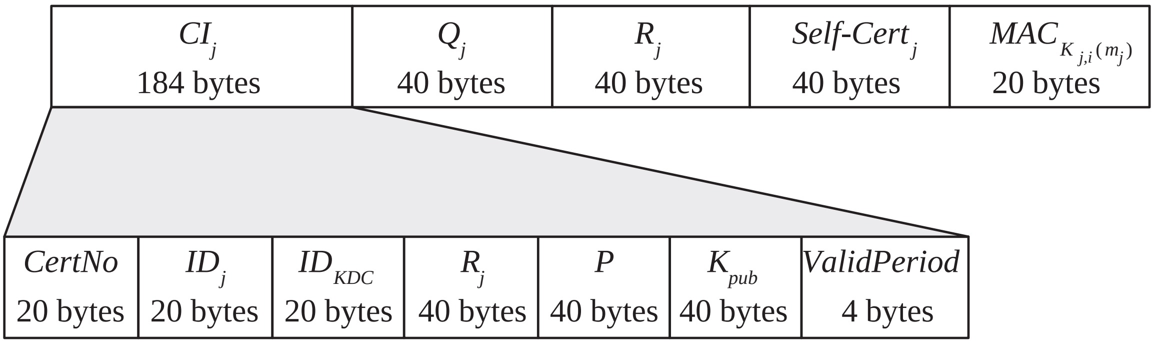

Figure 5.6: Transmitting message format from a sensor to a user in the login-and-authentication phase.

Table 5.7: Communication overhead in the login-and-authentication phase

	Each user	**Each sensor**
Communication overhead	$(304 + \|v\|^{*})$ bytes	$(324{+}(n{-}1)^{**}{\times}\|yes$ vote$\|)$ bytes

$|v|^{*}$: $|v|$ denotes the length of the challenge response sent from a user to sensors.
$(n-1)^{**}$: Assume there are $(n-1)$ sensors in the communication range of the sensor.

Chapter 6

Conclusion and future works

In this dissertation, we introduced recent developments in the field of wireless security and investigated several user authentication protocols in wireless networks. A detailed explanation of security frameworks and security requirements for authentication was given. We designed several user authentication protocols in wireless networks, including two kinds of password-based user authentication protocols, a biometrics-based user authentication protocol, and a self-certificate-based user authentication protocol.

For password-based user authentication, we proposed two password-based user authentication protocols, namely protocol-I and protocol-II. The protocol-I is a password-based user authentication protocol using LU decomposition and the protocol-II is a password-based user authentication protocol for WSNs. For biometrics-based user authentication, we proposed a biometrics-based remote user authentication protocol using smart cards. We also extended the protocol to a multi-party biometrics-based remote user authentication protocol by incorporating a secret sharing component. For self-certificate-based user authentication, we proposed a self-certificate-based user authentication protocol for WSNs, which still works well even if the adversary captures t nodes out of n nodes in the WSNs. Moreover, security of these proposed protocols was modelled and analyzed with Petri nets.

There are still various uncovered security issues in wireless networks. For example, in vehicular ad hoc networks (VANETs), security issues of VANETs are very challenging due to the scale of the network, the speed of the vehicles, their geographic positions, and the very sporadic connectivity between them, especially on how to construct secure inter-vehicle communications (IVC) and roadside-to-vehicle communications (RVC). The above issues might be interesting for possible future work.

Bibliography

[1] Platform Independent Petri net Editor 2 (PIPE2), available: http://pipe2.sourceforge.net/index.html.

[2] *Government Smart Card Handbook*, U. S. General Services Administrator, 2004.

[3] *Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*, American National Standards Institute ANSI X9.62, 2005.

[4] *Digital Signature Standard (DSS)*, National Institute of Standards and Technology FIPS PUB 186-3, 2009.

[5] J. Armington, P. Ho, P. Koznek, and R. Martinez. Biometric authentication in infrastructure security. In *Proceedings of International Conference on Infrastructure Security (InfraSec 2002)*, 2002.

[6] S. Banerjee and D. Mukhopadhyay. Symmetric key based authenticated querying in wireless sensor networks. In *Proceedings of the First International Conference on Integrated Internet Ad Hoc and Sensor Networks*, 2006.

[7] Z. Benenson, F. C. Gärtner, and D. Kesdogan. User authentication in sensor networks (extended abstract). In *Proceedings of Informatik 2004 (Workshop on Sensor Networks)*, 2004.

[8] Z. Benenson, F. C. Gärtner, and D. Kesdogan. An algorithmic framework for robust access control in wireless sensor networks. In *Proceedings of the European Workshop on Wireless Sensor Networks (EWSN 2005)*, 2005.

[9] Z. Benenson, N. Gedicke, and O. Raivio. Realizing robust user authentication in sensor networks. In *Workshop on Real-World Wireless Sensor Networks (REALWSN 2005)*, 2005.

[10] C. C. Chang and I. C. Lin. Remarks on fingerprint-based remote user authentication scheme using smart cards. *ACM SIGOPS Operating Systems Review*, 38(4):91–96, 2004.

[11] Y. F. Chang, C. C. Chang, and Y. W. Su. A secure improvement on the user-friendly remote authentication scheme with no time concurrency mechanism. In *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications (AINA 2006)*, volume 2, 2006.

[12] H. Y. Chien and C. C. Chen. A remote authentication scheme preserving user anonymity. In *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications (AINA 2005)*, 2005.

[13] H. Y. Chien, J. K. Jan, and Y. M. Tseng. An efficient and practical solution to remote authentication: smart card. *Computers and Security*, 21(4):372–375, 2002.

[14] S. J. Choi and H. Y. Youn. A novel data encryption and distribution approach for high security and availability using LU decomposition. In *Proceedings of the International Conference on Computational Science and Its Applications (ICCSA 2004)*, 2004.

[15] M. L. Das, A. Saxena, and V. P. Gulati. A dynamic ID-based remote user authentication scheme. *IEEE Transactions on Consumer Electronics*, 50(2):629–631, 2004.

[16] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In *Proceedings of Advances in Cryptology - Eurocrypt 2004*, 2004.

[17] C. I. Fan and Y. H. Lin. Provably secure remote truly three-factor authentication scheme with privacy protection on biometrics. *IEEE Transactions on Information Forensics and Security*, 4(4):933–945, 2009.

[18] Y. H. Gil, D. S. Moon, S. B. Pan, and Y. W. Chung. Fingerprint verification system involving smart card. In *Proceedings of International Conference on Information Security and Cryptology (ICISC 2002)*, 2002.

[19] H. I. Hsiao and D. J. DeWitt. A performance study of three high availability data replication strategies. In *Proceedings of the First International Conference on Parallel and Distributed Information Systems (ICPDIS)*, 1991.

[20] B. T. Hsieh, H. T. Yeh, and H. M. Sun. Cryptanalysis of a fingerprint-based remote user authentication scheme using smart cards. In *Proceedings of the IEEE International Carnahan Conference on Security Technology*, 2003.

[21] C. L. Hsu. Security of Chien et al.'s remote user authentication scheme using smart cards. *Computer Standards and Interfaces*, 26(3):167–169, 2004.

[22] C. L. Hsu. A user friendly remote authentication scheme with smart cards against impersonation attacks. *Applied Mathematics and Computation*, 170(1):135–143, 2005.

[23] L. Hu, Y. Yang, and X. Niu. Improved remote user authentication scheme preserving user anonymity. In *Proceedings of the IEEE International Conference on Communication Networks and Services Research (CNSR 2007)*, 2007.

[24] C. Jiang, B. Li, and H. Xu. An efficient scheme for user authentication in wireless sensor networks. In *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications Workshops (AINAW 2007)*, 2007.

[25] W. S. Juang. Efficient password authenticated key agreement using smart cards. *Computers and Security*, 23(2):167–173, 2004.

[26] M. K. Khan and J. Zhang. Improving the security of a flexible biometrics remote user authentication scheme. *Computer Standards and Interfaces*, 29(1):82–85, 2007.

[27] M. K. Khan, J. Zhang, and X. Wang. Chaotic hash-based fingerprint biometric remote user authentication scheme on mobile devices. *Chaos, Solitons and Fractals*, 35(3):519–524, 2008.

[28] H. S. Kim, S. W. Lee, and K. Y. Yoo. ID-based password authentication scheme using smart cards and fingerprints. *ACM SIGOPS Operating Systems Review*, 37(4):32–41, 2003.

[29] K. W. Kim, J. C. Jeon, and K. Y. Yoo. Efficient and secure password authentication schemes for low-power devices. In *Proceedings of International Conference on Mobile Ad-hoc and Sensor Networks (MSN 2005)*, 2005.

[30] N. Koblitz, A. Menezes, and S. Vanstone. The state of elliptic curve cryptography. *Designs, Codes and Cryptography*, 19(2-3):173–193, 2000.

[31] W. C. Ku, S. T. Chang, H. H. Chen, and M. J. Tsaur. Weakness and simple improvement of a password authentication scheme based on geometric approach. In *Proceedings of the IEEE Conference on Local Computer Networks (LCN 2005)*, 2005.

[32] W. C. Ku, H. M. Chuang, and M. J. Tsaur. Vulnerabilities of Wu-Chieu's improved password authentication scheme using smart cards. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E88-A(11):3241–3243, 2005.

[33] L. Lamport. Password authentication with insecure communication. *Communications of the ACM*, 24(11):770–772, 1981.

[34] K. Lauter. The advantages of elliptic curve cryptography for wireless security. *IEEE Wireless Communications*, 11(1):62–67, 2004.

[35] B. Lee and K. Kim. Self-certificate: PKI using self-certified key. In *Proceedings of the Conference on Information Security and Cryptology (CISC 2000)*, 2000.

[36] J. K. Lee, S. R. Ryu, and K. Y. Yoo. Fingerprint-based remote user authentication scheme using smart cards. *Electronics Letters*, 38(12):554–555, 2002.

[37] S. W. Lee, H. S. Kim, and K. Y. Yoo. Efficient password authenticated key agreement using smart cards. *Computer Standards and Interfaces*, 27(2):181–183, 2005.

[38] Y. Lee, J. Nam, S. Kim, and D. Won. Two efficient and secure authentication schemes using smart cards. In *Proceedings of International Conference on Computational Science and its Applications (ICCSA 2006)*, 2006.

[39] H. T. Liaw, J. F. Lin, and W. C. Wu. An efficient and complete remote user authentication scheme using smart card. *Mathematical and Computer Modelling*, 44(1-2):223–228, 2006.

[40] C. H. Lin and Y. Y. Lai. A flexible biometrics remote user authentication scheme. *Computer Standards and Interfaces*, 27(1):19–23, 2004.

[41] D. D. E. Long. A technique for managing mirrored disks. In *Proceedings of the IEEE International Conference on Performance, Computing, and Communications*, 2001.

[42] D. J. Malan, M. Welsh, and M. D. Smith. A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography. In *Proceedings of the IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, 2004.

[43] A. J. Menezes, P. C. Oorschot, and S. A. Vanstone. *Handbook of applied cryptography*. CRC Press, Boca Raton, Florida, 1997.

[44] J. Menon, J. Riegel, and J. Wyllie. Algorithms for software and low-cost hardware RAIDs. In *Proceedings of the 40th IEEE Computer Society International Conference (COMPCON)*, 1995.

[45] C. J. Mitchell and Q. Tang. Security of the Lin-Lai smart card based user authentication scheme. Technical report, Royal Holloway, University of London, 2005.

[46] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.

[47] M. Naor and K. Nissim. Certificate revocation and certificate update. *IEEE Journal on Selected Areas in Communications*, 18(4):561–570, 2000.

[48] A. K. Pathan and C. S. Hong. An efficient bilateral remote user authentication scheme with smart cards. In *Proceedings of the 33rd Korea Information Science Society Fall Conference*, 2006.

[49] A. K. Pathan, C. S. Hong, and T. Suda. A novel and efficient bilateral remote user authentication scheme using smart cards. In *Proceedings of the IEEE International Conference on Consumer Electronics (ICCE 2007)*, 2007.

[50] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. In *Proceedings of International Conference on Mobile Computing and Networking (Mobicom)*, 2001.

[51] H. Petersen and P. Horster. Self-certified keys - concepts and applications. In *Proceedings of the 3rd Conference on Communications and Multimedia Security*, 1997.

[52] J. L. Peterson. *Petri net theory and the modeling of systems*. Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[53] C. A. Petri. *Kommunikation mit automaten.* PhD thesis, University of Bonn, 1962.

[54] L. Rila and C. J. Mitchell. Security protocols for biometrics-based cardholder authentication in smartcards. In *Proceedings of Applied Cryptography and Network Security (ACNS 2003)*, 2003.

[55] B. Schneier. *Applied cryptography.* John Wiley & Sons Inc. Publication, New York, 1996.

[56] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[57] H. M. Sun. An efficient remote user authentication scheme using smart cards. *IEEE Transactions on Consumer Electronics*, 46(4):958–961, 2000.

[58] H. R. Tseng, R. H. Jan, and W. Yang. An improved dynamic user authentication scheme for wireless sensor networks. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM 2007)*, 2007.

[59] U. Uludag, S. Pankanti, S. Prabhakar, and A. K. Jain. Biometric cryptosystems: Issues and challenges. *Proceedings of the IEEE*, 92(6):948– 960, 2004.

[60] H. Wang and Q. Li. Distributed user access control in sensor networks. In *Proceedings of the IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS 2006)*, 2006.

[61] H. Wang, B. Sheng, and Q. Li. Elliptic curve cryptography-based access control in sensor networks. *International Journal of Security and Networks*, 1(3-4):127–137, 2006.

[62] R. Watro, D. Kong, S. F. Cuti, C. Gardiner, C. Lynn, and P. Kruus. TinyPK: Securing sensor networks with public key technology. In *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2004.

[63] K. H. M. Wong, Y. Zheng, J. Cao, and S. Wang. A dynamic user authentication scheme for wireless sensor networks. In *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 2006)*, 2006.

[64] S. T. Wu and B. C. Chieu. A user friendly remote authentication scheme with smart cards. *Computers and Security*, 22(6):547–550, 2003.

[65] E. J. Yoon and K. Y. Yoo. New authentication scheme based on a one-way hash function and Diffie-Hellman key exchange. In *Proceedings of International Conference on Cryptology and Network Security (CANS 2005)*, 2005.

[66] E. J. Yoon and K. Y. Yoo. Robust secret key based authentication scheme using smart cards. In *Proceedings of Pacific Rim Conference on Multimedia (PCM 2005)*, 2005.

[67] S. Yu, K. Ren, and W. Lou. FDAC: Toward fine-grained distributed data access control in wireless sensor networks. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM 2009)*, 2009.

[68] C. J. Zarowski. *An introduction to numerical analysis for electrical and computer engineers*. John Wiley & Sons, Inc., Hoboken, New Jersey, 2004.

Vita

Huei-Ru Tseng received the B.S. and M.S. degrees in Information Management from National Taiwan University of Science and Technology, Taipei, Taiwan, in 2002 and 2004, respectively. She is currently finishing her Ph.D. program in the Department of Computer Science at National Chiao Tung University, Hsinchu, Taiwan. Her research interests include wireless networks, network security, and cryptography.

Publication List

Journal Papers

[1] Huei-Ru Tseng, Rong-Hong Jan, and Wuu Yang, “A Robust User Authentication Scheme with Self-Certificates for Wireless Sensor Networks”, accepted and to appear in *Security and Communication Networks* (*SCN*), 2010. **[SCI-E, EI]**

[2] Yi-Chi Wu, Huei-Ru Tseng, Wuu Yang, and Rong-Hong Jan, “DDoS Detection with Decision Tree and Traceback with Grey Relational Analysis,” accepted and to appear in *International Journal of Ad Hoc and Ubiquitous Computing* (*IJAHUC*), 2010. **[SCI-E, EI]**

[3] Huei-Ru Tseng, Rong-Hong Jan, and Wuu Yang, “Bilateral Remote User Authentication Scheme Preserving User Anonymity”, *Security and Communication Networks* (*SCN*), vol. 1, no. 4, Jul./Aug. 2008, pp. 301-308. **[SCI-E, EI]**

[4] Wuu Yang, Huei-Ru Tseng, Rong-Hong Jan, and Bor-Yeh Shen, “Broadcasting with the Least Energy is an NP-complete Problem,” *International Journal of Multimedia and Ubiquitous Engineering* (*IJMUE*), vol. 3, no. 3, Jul. 2008, pp. 55-66.

[5] Huei-Ru Tseng, Rong-Hong Jan, and Wuu Yang, “A Robust Password-based User Authentication Scheme for Heterogeneous Sensor Networks”, *Communications of Institute of Information & Computing Machinery* (*IICM*), vol. 11, no. 3, Sep. 2008, pp. 1-13.

[6] Huei-Ru Tseng, Rong-Hong Jan, and Wuu Yang, “Time-bound Anonymous Routing in Clustered Multihop Wireless Ad Hoc Networks,” submitted to the *KSII Transactions on Internet and Information Systems*, 2009. **[SCI-E]**

Conference Papers

[1] Wuu Yang, Huei-Ru Tseng, and Rong-Hong Jan, “Two-counter Garbage Collection in the Heap,” the 5th *International Conference on Software and Data Technologies* (*ICSOFT* 2010), Athens, Greece, Jul. 2010.

[2] Huei-Ru Tseng, Rong-Hong Jan, and Wuu Yang, “A Chaotic Maps-based Key Agreement Protocol that Preserves User Anonymity,” *IEEE International Conference on Communications* (*ICC* 2009), Dresden, Germany, Jun. 2009. [EI]

[3] Yi-Chi Wu, Huei-Ru Tseng, Wuu Yang, and Rong-Hong Jan, “DDoS Detection with Decision Tree and Traceback with Grey Relational Analysis,” the *3rd International Conference on Multimedia and Ubiquitous Engineering* (*MUE* 2009), Qingdao, China, Jun. 2009. [EI]

[4] Wuu Yang, Huei-Ru Tseng, Rong-Hong Jan, and Bor-Yeh Shen, “Broadcasting with the Least Energy is an NP-complete Problem,” the 2nd *International Conference on Multimedia and Ubiquitous Engineering* (*MUE* 2008), Busan, Korea, Apr. 2008. [EI]

[5] Huei-Ru Tseng, Rong-Hong Jan, and Wuu Yang, “An Improved Dynamic User Authentication Scheme for Wireless Sensor Networks,” *IEEE Global Communications Conference* (*GLOBECOM* 2007), Washington D. C., USA, Nov. 2007, pp. 986-990. [EI]

[6] Huei-Ru Tseng, Rong-Hong Jan, Wuu Yang, and Emery Jou, “A Secure Aggregated Message Authentication Scheme for Vehicular Ad Hoc Networks,” submitted to the *IEEE Global Communications Conference* (*GLOBECOM* 2010), Miami, Florida, USA, Dec. 2010.

Book Chapters

[1] “Wireless LAN Security” (with Rong-Hong Jan and Wuu Yang), invited chapter in “Wireless Ad Hoc Networking: Personal-Area, Local-Area, and Sensory-Area Networks” (edited by Yu-Chee Tseng and Shih-Lin Wu), Auerbach Publications, ISBN: 0-8493-9254-3, 2007.