International Journal of Security and its Applications
Vol. 3, No. 1, January, 2009

Design of Multi-Service Smart Card Systems for High Security and Performance

Mei Hong, Hui Guo
School of Computer Science and Engineering
University of New South Wales, Australia
{*meihong, huig*}*@cse.unsw.edu.au*

Abstract

A multi-service smart card system enables users to access different services over an open network with a single smart card. Due to its highly economic and social benefits, the multi-service smart card system has drawn much attention in industrial and academic areas. However, the big hindrance to its wide employment is the risk of breaching users' privacy and their service confidentiality across different service systems.

This paper proposes a secure multi-service system model to overcome such problems. The model allows users to access different services with a single password. The privacy and service confidentiality are achieved through a set of protections—password protection, user identity protection, and service transaction protection—to guarantee the user's anonymity to all service systems and ensure high unlinkability between different services.

1. Introduction

The idea of a single smart card to be used for multiple services has been around for years. Instead of using separate access devices for different services, a user can access multiple services from different service providers by a single smart card. For example, a user can use the same smart card to log on to a remote server system, enter a secure building, and perform a financial transaction.

This kind of design frees people from carrying many cards, bringing users the great convenience and at the same time saving resources and costs by manufacturing and managing less volume of cards. Therefore, multi-service smart card systems exhibit a high potential for economic and social benefits. Such a system is even more convenient if only one password is used for each card so that users do not need to remember and cope with many passwords.

With this design, each user normally shows the same identity (**ID**) to different service systems. Hence, their access behavior to different services can be easily traced, linked and abused by adversaries.

The multi-service smart card system becomes further complicated when a single-password is used for different services. With this one-password scheme, passwords would run a high risk of being stolen and tampered since they are exposed to more parties, hence inviting more potential attacks.

To address these problems, we developed a sophisticated system model. The model fully realizes one-card/one-password access for multiple services (over an open network) while maintaining high levels of security and service performance.

Our contributions are summarized as follows:

- We propose an indirect password authentication scheme, where password authentication is mingled with service authentication. No passwords need to be sent separately over the network, making password leaks over the network impossible. Moreover, with this scheme, passwords are not stored in any server system nor on any smart card, providing little opportunity for offline password attacks.
- Based on the separate user-service ID approach proposed by [17], we propose a robust design for user service ID generation, storage, and management. In comparison to the existing approach, our design only stores a user service ID on the related service system, not on the smart card. The service ID that a user has to present each time to access a service, is dynamically calculated based on the user password, effectively ensuring service independency and unlinkability.
- We develop a mutual authentication protocol that can resist many attacks from both internal users and external hackers while keeping low computation and communication costs.
- To ensure communication confidentiality and efficiency, we propose a key agreement scheme that uses session keys for service transaction such that maintaining a large and growing table of keys is avoided, thus saving memory spaces and improving performance.

The paper is organized as follows. The reminder of this section reviews the related work on security design. Section 2 outlines our design model. The design of this model is detailed in Section 3, with highlighted security features. The performance analysis of the design model is given in Section 4. Section 5 concludes the paper.

1.1. Related Work

Many approaches to improve the system security have been proposed. They can be classified into two groups: identity authentication (to counter impersonation attacks) and communication protection (to counter network related attacks such as man-in-the-middle and replay attacks).

Passwords are commonly utilized for user authentication, where users identify themselves by sending the system their IDs and passwords. User passwords are usually concealed in the system in an encrypted format, such as one-way hashed value. On receiving a password, the system calculates its hashed value and searches for a match in the password table. When a match is found, the user's authenticity is verified [9]. However, the approach is not safe if the password is transferred over an insecure network.

In 1981, Lamport [15] proposed a scheme for remote password authentication with insecure communication. In his scheme, not only are user passwords stored in the hashed format, they are also hashed during transmission to prevent password eavesdropping. To realize such a design, the approach uses a sequence of passwords, each of which is formed by repeatedly hashing a given secret value. For each round of authentication, a different password from the sequence is used. However, the paper does not provide any solution as

to how to apply this design when the sequence of passwords are exhausted. Moreover, like any of its previous password authentication schemes, the approach displays two common weaknesses: 1) it requires the remote system to maintain a password table, which may incur significant performance and memory overheads if the number of users becomes large, and 2) it risks attacks if the password table can be moderated by intruders.

In response to this problem, some authentication approaches using smart cards were proposed [13, 11, 6, 21, 16, 4]. With these approaches, passwords no longer need to be stored in the remote system. They are only possessed by users, and required by the smart card to generate a message to the remote system for authentication.

Juang [14] in 2004 proposed a multi-server password authentication protocol. With this approach, no password verification table is needed; users can freely choose passwords; users and servers can be mutually authenticated.

In 2005, Chang and Kuo [5] proposed another multi-server password authentication scheme using smart cards. The scheme is based on the Chinese Remainder Theorem and a modulus table. The size of the modulus table increases as the number of servers increases. A similar protocol with some improvement was proposed in [20].

Based on previous contributions [14, 5, 20], Hwang and Shiau [12] presented a sophisticated solution to counter-attack most security problems.

To protect user privacy, Schaffer and Schartner in [19] proposed to unlink the user real identity from service systems. In their approach, user confidential personal information is replaced by fully anonymous data using cryptographic function. Fischer and Prouff in [10] presented a group signature method, where users communicate with an outside party in a group identity such that the user's personal information is not revealed.

All of the above methods were proposed for systems where a single ID is used to uniquely identify a user. When they are extended to multiple services, a user's behavior can be easily traced, and the user's privacy cannot be protected.

Nohara et al. [17] proposed an unlinkable ID management method for service unlinkability. Users access different services with a separate ID so that their behavior on different services cannot be linked. However, this system requires each service system maintain a full list of whole system user IDs, which adversely affects the system scalability and security. When the user list increases, the time for searching the list increases, thus reducing performance; with the growth of user list, the possibility of matching a particular ID value increases, thus extending the attacker's chances to impersonate a valid user.

In 2006, Bakdi [3] proposed a security solution for the multi-functional smart card system using public key cryptography. Instead of using an ID for each user, separate private and public key pairs are used for different functions to ensure authenticity and integrity of application processes and maximally protect the user privacy. However, their designs are complicated and entail a large amount of efforts for system maintenance.

In this paper, we will present a design model that combines a set of security design techniques to achieve a high system security for the multi-service smart card system with low design costs.

2. Design Overview

The structure of our multi-service smart card system is given in Fig.1. There are three entities: the Trusted Management Center (**TMC**), **User Group** and **Service Group**.

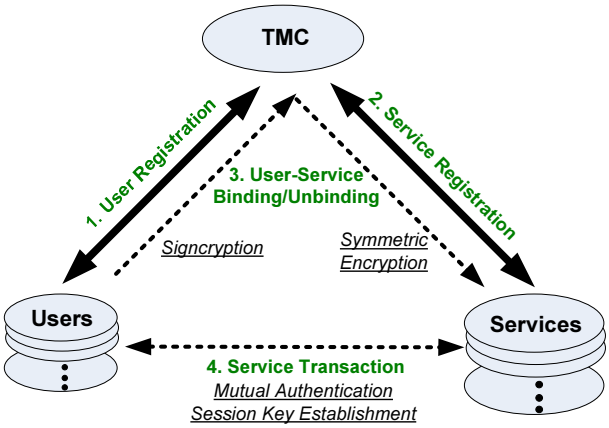

Figure 1: System Overview

The **TMC** is a trusted third party between users and service providers. It handles user and service registrations. **Users** are the smart card holders who access services available in the system. Unlike most existing systems, where user service portfolios are generated by the management center, users in this system can customize and dynamically change their service wish list. There are multiple **Services**; each service must be registered by its provider before it is available to users.

We employ a set of design techniques to best ensure system security, optimize performance and at the same time extensively reduce costs. They are summarized as follows.

- We use hash functions [18] whenever possible to lower computation costs. A hash function is computationally simple and can easily compress a varying size of message to a fixed length string. It is impossible to compute original inputs from hash values, nor to replace them with moderated data for the same outputs within a limited time frame. Hence hash functions have high resistance to online decipher attacks. We use hash functions to encrypt messages and mingle the hashed value with other messages during authentication so that when transferred over networks, the messages are effectively immune to online attacks.

- To protect smart card from being abused by unauthorized people, we introduce a card validation scheme. A key (PIN) is assigned to a smart card. Each time a card is used, it is first validated against the key. Only after the card is validated, can the user use the card together with a correct password to perform service operation. The scheme also effectively enables users to safely change their passwords any time after registration.

- Since the user identities are of paramount importance to the overall system security, we minimize the permanent store of identities to limit offline attacks. To protect user privacy, we only store the user ID that can reveal user personal information in the TMC; we use a separate ID, called User Registration ID (UID), for anonymous service access. To achieve service unlinkability, we use different User-Service binding ID, $USID$.

- We employ Signcryption [22] [8] for communication between the user and the TMC.

Signcryption is a public key cryptography (asymmetric) scheme that simultaneously fulfills both digital signature and public key encryption. It is much more secure and has lower computational costs than traditional asymmetric approaches.

- Symmetric-key encryption techniques can achieve the same level of security as asymmetric encryption methods, but they are computationally fast. Since a symmetric-key encryption requires a separate key for each communication pair, the cost of key maintenance becomes significant for large groups of users. Therefore, we only employ the symmetric-key encryption for communications between the TMC and services. The symmetric keys are generated by the TMC and populated to the service systems in a secure environment. A standard encryption/decryption algorithm (such as AES [1]) can be used for the encryption.

- For large, growing user groups, we apply session keys for service transactions between users and services. A session key is an ephemeral secret value, which is restricted to a session. After the session finishes, it is eliminated. Therefore, the average session-key list of a service system is minimized. We propose to use Diffie-Hellman key agreement [7] to generate session key, which can be easily embedded in our mutual authentication protocol so that the key is authenticated for each round of communication, ensuring high integrity of the transferred message.

3. Design Details

In this section, we elaborate the system model. Along with the design description, where appropriate, we provide some remarks to highlight the design feature.

3.1. Terms and Definitions

There are a number of parameters and functions used in the system. We set them into two groups: 1) public information that is available to whole system, and 2) keys used for encryption and verification. For clarity, their definitions are given below.

Public Information:

The public information includes parameters and functions that are generated by the TMC and can be either loaded into smart cards at the card issue stage, or stored in the trusted card access terminals from which the information can be later retrieved by smart cards.

p: a large prime number, recommended to be at least 1024 bits for security.

q: a large prime number, factor of $p - 1$.

g: an integer, with the order q modulo p, chosen randomly from $(1, \cdots, p - 1)$.

$h(.)$: the one-way hash function with a fixed-length output.

$KH(.)$: a keyed one-way hash function. The hash function includes a key in the calculation.

$E_k(.)$: symmetric encryption algorithm with key k.

$D_k(.)$: symmetric decryption algorithm with key k.

Keys:

Keys are generated by one of the entities in the system.

k: long secret key of the TMC. The TMC should keep it confidential at any time. It is used to generate the user registration ID (UID) from the user ID (ID).

x: the TMC's private key, chosen randomly from $(1, \cdots, q-1)$.

y: the TMC's public key, $(y = g^x modp)$.

x_i: the user U_i's private key, chosen randomly from $(1, \cdots, q-1)$.

y_i: the user U_i's public key, $(y_i = g^{x_i} modp)$.

k_j: a symmetric cryptographic key, generated by the TMC and shared between the TMC and service provider for service S_j.

(k_1, k_2): the symmetric keys, used in Signcyption.

PIN: a key for card verification.

3.2. Registrations

To use the multi-service smart card system, both smart card users and service providers need to register. The registration basically sets up parameters required for each system component and establishes the logic connection between users and services. There are three registration phases: user registration, service registration, and user-service binding. Users and services only need to register once, while the user-service binding operation can happen unlimited times. Each phase is detailed below and the related system parameter generation and allocation are recorded in the table shown in Fig.2 for easy reference, where the first column of the table lists all the phases to set up a system, the first row lists the three system entities, and the last column specifies the parameter transfer actions.

Operation Phase	TMC	User U_i (smart card)	Service S_j	Parameter Transmission
Initialization	**k, (x, y)**	∅	∅	∅
User Registration	**k, (x, y)** {**UID$_i$**, y_i}	V_i, **(x_i,y_i)**, **PIN**	∅	TMC=>U_i : smart card(V_i) U_i=>TMC : {UID_i, y_i}
Service Registration	**k, (x, y)** {**UID$_i$**, y_i} {**SID$_j$**, **k$_j$**}	V_i, **(x_i,y_i)**, **PIN**	SID_j, k_j	TMC=>SP_j : {SID_j, k_j}
User-Service Binding	**k, (x, y)** {UID_i, y_i,$USID_{ij}$} {**SID$_j$**, **k$_j$**}	V_i, **(x_i,y_i)**, **PIN**	SID_j, k_j {$USID_{ij}$}	U_i→ TMC: Signcrypt($USID_{ij}$,UID_i,SID_j) TMC→SP_j : E_{kj}($USID_{ij}$)

Figure 2: System Parameters

Before the overall system is set up, no users and services are connected to the TMC. The TMC initially contains only the secret key, k, and its private and public key pair, (x, y); both smart cards and service systems hold empty information, as shown in the second row in the table (see Fig.2).

3.2.1. User Registration: To register, a user, $\mathbf{U_i}$, provides the TMC with his/her personal information and password, PW_i. After verifying his/her identity, the TMC generates a unique user ID, ID_i, based on the user information, and creates a user registration identity (UID_i) and a parameter (V_i) by using the following formulas:

$$UID_i = h(ID_i \parallel k) \tag{1}$$

$$V_i = UID_i \oplus PW_i \tag{2}$$

where k is a secret key of the TMC, $||$ the concatenation operation, and $\oplus$ the logic exclusive OR (XOR) operation.

The TMC then issues the user a smart card that is loaded with parameter V_i.

Remark 1: *The user's personal information can only be revealed with UID_i on the TMC. To maximally protect UID_i as well as PW_i, we do not store them directly in the smart card; instead, an indirect value (V_i) is stored. Given a smart card, UID_i can only be obtained with a correct PW_i. In case of card loss or theft, the chance for an adversary to gain UID_i or PW_i by exploring information on the card is almost nil since both UID_i and PW_i are unknown to the adversary.*

A new card must be activated before its first use. To activate a card, the user inserts his/her card into a secure terminal at the TMC, inputs a password, PW_i, at the prompt. Then the card computes UID_i based on the following formula derived from Formula 2:

$$UID_i = V_i \oplus PW_i \tag{3}$$

Next the card generates a private key x_i ($x_i \in [1, p-1]$) and calculates the corresponding public key y_i, ($y_i = g^{x_i} mod p$). The key pair (x_i, y_i) are stored into a tamper-resistant memory on the smart card, and the data set $\{UID_i, y_i\}$ are transferred back to the TMC.

On completion of this phase, the user is issued with a personal identification number (PIN) for the valid card.

The parameters generated in this phase are summarized in row 3 of the table in Fig.2. The TMC has parameters $\{UID_i, y_i\}$ for user $\mathbf{U_i}$ and the smart card contains $\{V_i, (x_i, y_i), PIN\}$.

Note that in the table we use $\{\}$ to denote a list, and symbols $\Rightarrow$ and $\rightarrow$ to represent secure and insecure channel transfer respectively; we indicate the origin of a parameter by highlighting it in the column of its generator. For example, y_i is highlighted in the third column since it is generated by **User**. The notion in the last column $U_i \Rightarrow TMC : (UID_i, y_i)$ stands for **User** transfers two parameters $\{UID_i, y_i\}$ to the TMC over a secure channel.

3.2.2. Service Registration: When a service provider, SP_j, registers its service, $\mathbf{S_j}$, with the TMC, the TMC generates a unique service identity SID_j for $\mathbf{S_j}$ and a symmetric key k_j for later communication. These two parameters are transferred to SP_j through a secure channel and saved on both sides of the system.

Service identities $SIDs$ are available to any user in the system. However, both the TMC and service providers should keep their communication symmetric-key confidential.

3.2.3. User-Service Binding: A service binding enables a user to use a service. It consists of following five steps:

Step 1. At a trusted card reader terminal provided by the TMC, user, $\mathbf{U_i}$, attaches his/her smart card, and keys in the PIN. After the card is validated, the user is prompted for a password PW_i. With the user's password, the smart card computes UID_i based on Formula 3; Then, the user can select a service from the service list displayed on the terminal screen. When a service, $\mathbf{S_j}$, is selected, its service identity SID_j is transferred to the smart card.

Step 2. The smart card generates binding identity $USID_{ij}$ for user $\mathbf{U_i}$ and service $\mathbf{S_j}$ using

$$USID_{ij} = h(UID_i \oplus SID_j) \tag{4}$$

Remark 2: *The hash function ensures that with knowledge of $USID_{ij}$, it is computationally infeasible to find input UID_i, which is a secret to identify $\mathbf{U_i}$.*

Step 3. The $USID_{ij}$ is sent to the TMC with Signcryption. $USID_{ij}$ together with UID_i and SID_j is encrypted into a cypher message (c, r, s, y_i) which is transferred to the TMC. The details of Signcryption are listed in the left part of the table in Fig.3.

Signcryption by U_i	**Signcryption Verification by TMC**
1. Choose a random r in *(1,..., q-1)* 2. *$(k_1, k_2)=h(y^r \bmod p)$* 3. *$c=E_{k1}(USID_{ij},UID_i,SID_j)$* *4. $e=KH_{k2}(USID_{ij},UID_i,SID_j)$* *5. $s=r/(e+x_i) \bmod q$* 6. Sends *(c,e,s,y_i)* to TMC	1. Computes *$(k_1,k_2)=h(y_i*g^e)^{s*x} \bmod p$* 2. *$(USID_{ij},UID_i,SID_j)=D_{k1}(c)$* 3. Verifies if *$KH_{k2}(USID_{ij},UID_i,SID_j)=e$*

Figure 3: Signcryption and Verification Process

Step 4. After receiving the message, the TMC un-Signcryptes it to obtain data ($USID_{ij}$, UID_i, SID_j). The details of Signcryption verification are given in the right part of the table in Fig.3. The TMC confirms the UID_i by searching a match in its user list and verifies the correctness of the received $USID_{ij}$. Then it passes $USID_{ij}$ to service provider SP_j over an open network. The message transferred is encrypted with their shared symmetric key k_j.

Remark 3: *The Signcryption ensures security with both a public key encryption and a digital signature. The encryption ensures message transmission is secure over the open network. The digital signature certifies the binding identity $USID_{ij}$ is originated from $\mathbf{U_i}$, which confirms the true service binding by the user.*

Step 5. Upon receiving of $USID_{ij}$, the service provider SP_j stores the ID in its service customer list for $\mathbf{S_j}$—a new user now has been successfully linked to the service.

Remark 4: *Access activities of $\mathbf{U_i}$ to service $\mathbf{S_j}$ are linked and only linked with $USID_{ij}$. Neither users' real identities nor their activities on other services are exposed to SP_j. Therefore, the service unlinkability is realized. In addition, the binding scheme allows a user to bind many services at the same time and only to signcrypt them once, thus improving the system efficiency.*

3.3. Service Transaction

After binding, a user can access a certain service with a valid smart card. To ensure the authenticity of both communication parties, we employ a mutual authentication scheme. To protect the confidentiality of transferred messages, we encrypt the messages with session keys.

The service transaction process consists of a series of tasks, which are summarized in Fig.4. The process can be divided into three steps: the first step is user authentication; if it is successful, then service authentication is performed in the next step; a session key is established after the mutual authentication succeeds. The session key is used in the following service transaction over an insecure channel. The completed task flow is elaborated as follows.

User Authentication:

To access a service $\mathbf{S_j}$, a user $\mathbf{U_i}$ attaches his/her smart card to a terminal device. After the card is verified with the PIN, the user inputs his/her password, PW_i. Then, the smart card performs a number of tasks:

1. Calculating UID_i using Formula 3, retrieving service $\mathbf{S_j}$'s identity SID_j from the terminal, and calculating the user-service binding identity $USID_{ij}$ for $\mathbf{S_j}$ using Formula 4;
2. Selecting a random exponent, $a \in Z_p^*$, and computing $A = g^a modp$;
3. Calculating $C1 = h(USID_{ij} \parallel t \parallel A)$ (t is the current time stamp of the device);
4. Sending authentication request message $(SID_j, C1, t, A)$ to SP_j.

On receiving the message from the user, the service provider SP_j performs the following sequence of tasks:

1. Checking the time interval between t and the receiving time; if the interval is beyond a given reasonable transmission delay, the message may be tampered by some intruders during the transmission over an insecure network, and the authentication request will be rejected. Otherwise, continue to the next step;
2. Checking if $USID_{ij}$ is in the service customer list by exhaustively searching a $USID$ such that $C1' = h(USID_{ij} \parallel t \parallel A) = C1$. If not found, the provider assumes the request comes from an invalid user, and rejects the request. Otherwise, continue to the service provider authentication.

Service Provider Authentication & Session Key Generation:

On the successful authentication of the user, the service provider continues with the following tasks:

3. Selecting a random exponent $b \in Z_p*$, and computing $B = g^b modp$;
4. Computing $C2 = h(USID_{ij} \parallel t' \parallel A \parallel B)$, where t' is the current time stamp of SP_j;
5. Generating session key, $K = A^b modp$, for later service transactions;
6. Sending $(C2, t', B)$ to $\mathbf{U_i}$.

On the user side, user $\mathbf{U_i}$ performs tasks as follows:

1. Similarly checking the time interval between t' and the receiving time; if the interval is beyond the reasonable transmission delay, U_i aborts the protocol. Otherwise, continue
2. Computing $C2' = h(USID_{ij} \parallel t' \parallel A \parallel B)$, checking if $C2' = C2$; If it is not, the authentication fails; otherwise, the authentication successes;
3. Calculating the session key $K' = B^a modp$ for the service transaction.

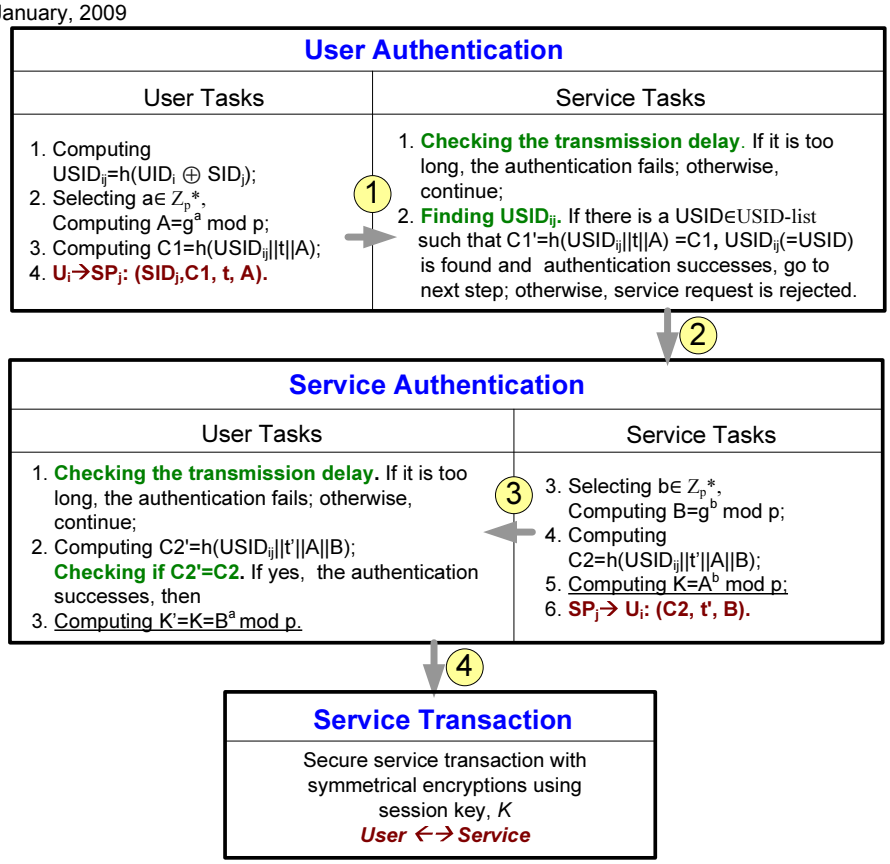

Figure 4: Service Authentication Task Flow

K and K' are same ($K = K' = B^a modp = A^b modp = g^{ab} modp$). The session key $K(K')$ will be used in the upcoming communication between $\mathbf{U_i}$ and $\mathbf{S_j}$ for secure service transactions over an insecure communication channel.

Remark 5: *This authentication scheme achieves mutual authentication through an insecure channel with low computational complexity and communication costs. Throughout the whole process, there are only two message transmissions (as can be seen from the task flow in Fig.4: Task 4 from the user and Task 6 from the service provider), which incurs a very small network traffic.*

3.4. User-Service Unbinding

If user $\mathbf{U_i}$ wants to unbind service $\mathbf{S_j}$, he/she can send a Signcrypted message to the TMC. The message includes user identity UID_i, service identity SID_j, related user-service binding identity $USID_{ij}$, together with an unbinding request.

Similar to service binding, the Signcryption performs both digital signature and message encryption functions.

After receiving and verifying the request, the TMC informs the service provider SP_j to unbind its service from the user. If there is no pending payment and other contract issues, the service provider will delete the user's service ID, $USID_{ij}$, from its customer list. $\mathbf{U_i}$ will no longer be able to access this service.

Remark 6: *With the available unbinding facility, the system can be easily expanded and contracted over the time, without the need to maintain redundant users on each service system, as compared to the existing similar designs [17] [3].*

3.5. Password Change

To change a password, the user inserts his/her smart card in a system card reader. After the card is validated, the user selects the password change function. He/she is then prompted for an old password. With the old password, the smart card calculates the user ID, UID_i, using Formula 3. The UID_i is used to calculate a new value of V_i for the new password. The old V_i is then replaced by the new value on the smart card.

Remark 7: *Anyone except the legitimate user cannot change a password without knowing both the PIN and the old password, which effectively enables password change by users themselves.*

4. Performance Analysis

We judiciously combine a set of security design techniques of different computational complexity to enhance system performance without sacrificing security.

In the user and service registration phases, the hash function and XOR operation are used. Both have small computation costs. These computations are mainly performed by the TMC which normally has a superior computing power. Therefore, the performance concern in these registration phases can be eliminated.

In the user-service binding/unbinding phase, since high complexity operations for digital signature and asymmetric encryption are necessary for security, and they need to be performed by low-end smart cards, we apply Signcryption to speed up performance. The signcryption on average takes 50% less computation time than a normal signature-then-encryption approach.

In the service transaction phase, fast hash functions are used for the mutual authentication and the performance-efficient Diffie-Hellman key agreement protocol is used for the session key generation.

Service transactions are frequently operated and dominate the system performance. Therefore, to verify the performance effectiveness of our design, we compare the authentication process of service transactions in our model with some existing authentication designs: Juang's (J's) [14], Tsaur's (T's) [20], Chang-Kuo's (C-K's) [5], Hwang-Shiau's (H-S's) [12], as shown in the table in Fig.5.

Since the computation cost of XOR, hash function, and modular addition are much lower than that of the symmetric encryption/decryption, modular multiplication, and modular exponentiation, we only consider the later set of operations performed by the user (i.e. the smart card in the smart card based authentication, or client software in a general client-server based authentication) and the server system.

The computation costs are measured in the number of blocks of symmetric encryptions/decryptions (as given in row 3 of the table in Fig.5), modular exponentiation operations (row 4), and modular multiplication operations (row 5). Note that with T's scheme, the number of modular multiplication operations is related to the number of servers in the

	Ours		J's		T's		C-K's		H-S's	
	User	Server	User	Server	User	Server	User	Server	User	Server
Symm. Enc/Dec (blocks)	0	0	12	14	0	0	14	16	0	2
Mod.Exp Oper. (times)	2	2	0	0	0	0	0	0	0	0
Mod.Multi Oper. (times)	0	0	0	0	m+[m/2]+[3m/2]+2	$2m^2+3m+4$	0	0	1	1
Trans. Rounds	1	1	2	1	1	0	2	1	2	1
Size of Message (bits)	256+256+1024	256+1024	256+256+256*6	256*6	(m+3)*1024+256	0	256*10	256*6	256*5	256*3

Figure 5: Comparison of Computation and Communication Costs in Authentication Processes

system, m. The communication costs are measured in number of network transmission rounds and the size of each transmission in bits. They are given in the last two rows in the table.

The costs for those designs were derived from the results presented in [12] with the following upgraded security settings:

- We use SHA-256 [2] as the one-way hash function and AES [1] as the symmetric cryptosystem.
- The output size of SHA-256 and the size of random number are 256 bits, while block size of AES is 256 bits (not 128 bits as used in [12]).
- The size of the identity is 256 bits.
- Parameter p is 1024 bits rather than 512 bits as used in [12].

From the table (Fig.5), we can see that the computation cost of our scheme is low, just trailing behind T's. Our design also requires less rounds of network transmissions than J's, C-K's and H-S's. For each transmission the message size is smaller than the other designs except H-S's. Overall, our design is the most efficient when computation and communication costs are combined.

5. Conclusions

In the paper, we proposed a secure and efficient multi-service system model where a user can use a smart card to access multiple services with a single password.

We developed an effective ID model and a series of design techniques to best enhance system security and performance, and at the same time to reduce costs on memory consumption and network traffic volumes.

To maximally protect the system, we propose to minimize the physical storage of secret information (such as user service identities and passwords) on smart cards and service systems, and to implement a strict measure on smart card uses. We also apply mutual authentication on service transactions. To ensure service unlinkability, we employ a separate

service identity scheme similar to an existing approach, but we enhance the scheme by improving the security in service ID generation, transmission, and management.

The legitimate use of smart cards is ensured by double passes: 1) card validation where a card is verified against a PIN each time it is used, and 2) user indirect password authentication. In our user password authentication, the password authentication is mingled with service authentication; the user password cannot be revealed either through offline attacks to the smart card or via online eavesdropping over the network. User passwords are much more secure than any other existing smart card password designs.

To improve performance, we employ a fast mutual authentication process for service transactions, use fast design solutions, such as Signcryption, to mitigate performance bottlenecks in the system, and reduce network communication.

The reduced network communication also helps relieve network traffic and reduces the hardware costs that would otherwise be incurred by extra network resources. The hardware costs are also saved by the reduced memory consumption in each service system.

Our design model is scalable, highly secure and more efficient in performance and cost.

References

[1] Advanced encryption standard (aes), national institute of standards and technology. *U.S. Department of Commerce, Gaithersburg, MD, USA*, page 197, 2001.

[2] Secure hash standard, national institute of standards and technology. *U.S. Department of Commerce, Gaithersburg, MD, USA*, pages 180–2, 2002.

[3] I. Bakdi. Towards a secure and practical multifunctional smart card. In *Smart Card Research and Advanced Applications, 7th IFIP WG 8.8/11.2 International Conference, CARDIS 2006*, pages 16–31. Springer Berlin / Heidelberg, 2006.

[4] Z.-C. Chai, Z.-F. Cao, and R.-X. Lu. Efficient password-based authentication and key exchange scheme preserving user privacy. *Lecture Notes in Computer Science*, 4138:467–477, 2006.

[5] C.-C. Chang and J.-Y. Kuo. An efficient multi-server password authenticated key agreement scheme using smart cards with access control. In *Proceedings. 19th International Conference on Advanced Information Networking and Applications*, pages 257–260, 2005.

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

[7] W. Diffie and M.-E. Hellman. Multiuser cryptographic techniques. In *Proceedings of AFIPS National Computer Conference*, pages 109–112, 1976.

[8] H.-M. Elkamchouchi, A.-A.-M. Emarah, and E.-A.-A. Hagras. A new public key dynamic signcrypted identification (pk-ds-id) protocol using smart cards. In *Proceedings of National Radio Science Conference*, pages 1–10, 2007.

[9] A.-J. Evans and E. Weiss. A user authentication scheme not requiring secrecy in the computer. *Communication of the ACM*, 17:437–442, 1974.

[10] J.-B. Fischer and E. Prouff. Off-line group signatures with smart cards. *Lecture Notes in Computer Science*, 3928:263–277, 2006.

[11] M.-S. Hwang and L.-H. Li. A new remote user authentication scheme using smart cards. *IEEE Transactions on Consumer Electronics*, 46:28–30, 2000.

[12] R.-J. Hwang and S.-H. Shiau. Provably efficient authenticated key agreement protocol for multi-servers. *Computer Journal*, 50:602–615, 2007.

[13] T. Hwang, Y. Chen, and C.-J. Laih. Non-interactive password authentications without password tables. In *1990 IEEE Region 10 Conference on Computer and Communication Systems*, pages 429–431, 1990.

[14] W.-S. Juang. Efficient multi-server password authenticated key agreement using smart cards. *IEEE Transactions on Consumer Electronics*, 50:251–255, 2004.

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

[16] I.-E. Liao, C.-C. Lee, and M.-S. Hwang. A password authentication scheme over insecure networks. *Journal of Computer and System Sciences*, 72:727 – 740, 2006.

[17] Y. Nohara, S. Inoue, and H. Yasuura. Toward unlinkable id management for multi-service environments. In *Proceedings. Third IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 115–119, 2005.

[18] B. Preneel. Cryptographic hash functions. *Proceedings. European transactions on Telecommunications*, 5:431–448, 1994.

[19] M. Schaffer and P. Schartner. Anonymous authentication with optional shared anonymity revocation and linkability. *Lecture Notes in Computer Science*, 3928:206–221, 2006.

[20] W.-J. Tsaur, C.-C. Wu, and W.-B. Lee. An enhanced user authentication scheme for multi-server internet services. *Applied Mathematics and Computation*, 170:258–266, 2005.

[21] W. Yang and S. Shieh. Password authentication schemes with smart cards. *Computers and Security*, 18:727–733, 1999.

[22] Y. Zheng. Signcryption and its applications in efficient public key solutions. *Lecture Notes in Computer Science*, 1396:291 – 312, 1997.