A Framework of Authentic Post-Issuance Program Modification for Multi-Application Smart Cards

Uddin, Mohammad Mesbah
Department of Computer Science and Communication Engineering

Zabir, Salahuddin Muhammad Salim
System LSI Research Center

Nohara, Yasunobu
Department of Computer Science and Communication Engineering

Yasuura, Hiroto
Department of Computer Science and Communication Engineering

出版情報：Proceedings of the International Conference on Wireless Networks. 2008, pp.288-294, 2008-07-16
バージョン：
権利関係：

A Framework of Authentic Post-Issuance Program Modification for Multi-Application Smart Cards

Mohammad Mesbah Uddin*, Salahuddin Muhammad Salim Zabir†,
Yasunobu Nohara* and Hiroto Yasuura*

*Department of Computer Science and Communication Engineering
Kyushu University, Fukuoka, Japan
Email: {mesbah,nohara,yasuura}@c.csce.kyushu-u.ac.jp

†System LSI Research Center
Kyushu University, Fukuoka, Japan
Email: szabir@slrc.kyushu-u.ac.jp

Abstract

Authentic program modification is very important for a multi-application smart card system since applications in the system are realized after the issuance of the smart card. In this paper, we propose a framework for such authentic program modification. In our framework, before issuing a smart card to an individual, the card issuer stores a unique long bit string called PID on the card. From the PID, unique substrings (subPIDs) are generated and used for different authentication purposes. The program modification protocol utilizes the subPIDs along with a one-way hash function and a pseudo random number generator function to verify the identity of the parties and the authenticity of the program. Our proposed framework provides a simple and practical solution to the program modification problem by avoiding direct interaction between the card holder and the issuer. In addition, use of hash functions makes the implementation cost of smart cards low.

Keywords: smart cards, cryptographic protocols for smart cards, multi-application smart cards, security, authentication Keywords

1. Introduction

Security is a key concern In a multi-application smart card system, where different applications are realized by a single card. In a multi-application smart card system, a person receives a smart card from an issuer, and then can freely choose any application provided by some service vendor. We assume a loosely coupled system where programs will be modified (i.e, installed, updated or deleted) on smart cards by service vendors instead of the issuer. This split in trust may incur security risks[1]. It is very important to establish security relations between a cardholder and a corresponding service vendor.

Establishment of security relations is critical especially during post-issuance modification of programs. Since the modification process is carried out by devices owned by service vendors, cardholders cannot control the modification process. Moreover, card will not generally be equipped with display units so that their owners may observe the whole process like a PC program modification. Therefore, during modification of programs, the cardholder must be protected from being modified with an illegal program. The service vendor must also know that it is indeed modifying a legitimate cardholder. In this paper, we address this problem and propose a framework that ensures authentic post-issuance program modification for smart cards.

In our framework, before issuing a smart card to an individual, the card issuer stores in it a unique long bit string called PID. From the PID, unique substrings (subPIDs) are generated and used for different authen-

tication purposes. One subPID is used to mutually authenticate a cardholder and the corresponding service vendor. Another subPID is used to authenticate the program.

The rest of this paper is organized as follows. Section 2 describes multi-application smart card system and defines the problem. Section 3 provides a solution to the problem and section 4 discusses about the proposed system. Section 5 gives some related work and section 6 concludes the paper.

2. Multi-Application Smart Card Environment

2.1. The Smart Card

A multi-application smart card is embedded with an integrated circuit chip that provides memory capacity and computational capability. The chip usually consists of a microprocessor, read only memory, persistent memory which retain its state when the power is removed, and non-static random access memory.

Smart cards became very popular for storing values or credentials in various applications. In the early stages, a smart card could be used only for some dedicated application. However, due to ease of use, many application providers started to adopt smart cards in their business settings. Thus we see a growing necessity for a multi-application smart card environment that can flexibly manage a number of different applications.

2.2. Entities, Roles and Trust

A multi-application smart card system comprises of three entities, namely, the **issuer** (**I**), **service vendors** (**V**), and **cardholders** (**H**). In a typical system, there are usually one issuer, several service vendors, and many cardholders (Fig. 1). In a multi-application smart card system, all the participants are associated with some devices. The issuer's device is called the **host system**. Each service vendor has its own **server system** and each cardholder possesses a **personalized smart card**. The smart card is assumed to be **tamper resistant** to physical and side-channel attacks. In this paper, we assume that the issuer is a **trusted** entity. That is, the issuer is trusted by the cardholders and the service vendors. On the other hand, there is no trust between other entities in the system.

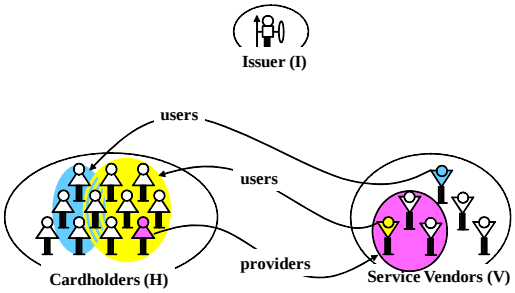

Figure 1. Entities in a Multi-application Smart Card System.

2.3. Smart Card Life Cycle and Application

The life cycle of a multi-application smart card is broadly divided into three phases – namely **issue**, **use** and **dispose** (Fig. 2).

In the **issue phase**, the smart card is first designed, manufactured, packaged. Then then a smart card is personalized [1] for an individual. (Installation of some applications may be done at this level.) The personalized smart card is then given (i.e., **issued**) to the individual.

In the **use phase**, the cardholder will receive services from vendors. The cardholder uses the smart card to install applications, and then execute applications from different service vendors.

Finally, in the **dispose phase**, the smart card is disposed when its possessor no longer need it.

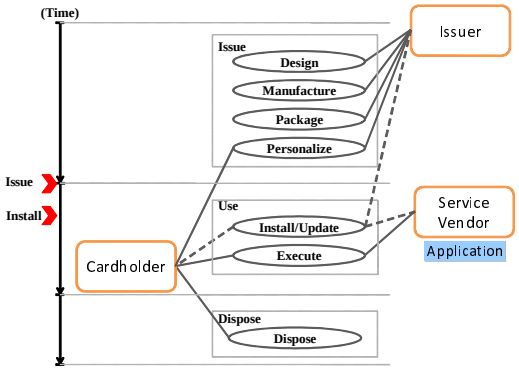

Figure 2. Smart card life cyle.

1. Adds extra functionality to the card and cardholder-specific information which may not be "personal information" such as a cardholder's name, telephone number, etc.

2.4. Post-Issuance Program Modification

In a multi-application environment, applications are realized after the issuance of the smart card. That is, service vendors may launch new applications or application updates, or remove applications after the issuance of the smart card. It is therefore, very important to ensure that application programs can be modified (i.e., installed, updated or deleted) after the issuance.

However, applications are owned by service vendors, who is not in a **trust relation** with the cardholder. In such an open and hostile environment, it is very important that smart cards are modified only with non-malicious programs. In general, it is very difficult to formally verify whether a program is malicious. Therefore, a framework is needed to ensure a cardholder that she can modify a program with enough confidence to receive the service.

3. A Framework for Authentic Post-Issuance Program Modification

3.1. Framework Requirements

In this paper, we assume a **trusted issuer**, who provides necessary devices and mechanisms for the smart card environment. In this framework, a service is based on setting up a security agreement relation through **"registration and licensing"**. That is, an application must be registered and properly certified before it can be used.

Smart cards are small devices with low power, low computational ability and short ranged communication. It is not possible to integrate a **complex installer** module in the smart card. Therefore, the smart card will require a **card acceptance device (CAD)** for processing much of the installation. It is not realistic that cardholders will carry a CAD or learn operating them whenever such a modification is required. It is more likely that service vendors will have those devices as a part of their systems and will modify their application programs upon receiving a request from a cardholder.

In order to install, update, or delete a program, a cardholder gives her card to a service vendor. Therefore, it is very important that smart cards are modified only with authentic programs. In the presented framework, the following requirements need to be fulfilled.

1) Only a legal cardholder should be able to receive a program.
2) Only a legal service vendor should be allowed to modify its program.
3) The modified program must be legitimate.
4) The program must be installed in a non-overlapping space (memory) inside the card.

A cardholder is said to be legal for a service vendor if there exists a trust or agreement relation between the cardholder and the service vendor. Similarly, a service vendor is legal w.r.t. a cardholder it is in trust or agreement relation with. A legitimate program, on the other hand, is a program that is certified by the trusted issuer. In this paper, we assume mutually exclusive program and data space (memory) for different applications.

3.2. PID System

PID (Personal Identifier) system is proposed in [2], [3]. A PID is a unique long bit string the issuer issues to an individual. The issuer generates a unique PID, stores in a smart card and gives it to the person. A unique substring of PID is called a **subPID**, and this subPID is used as a means of authentication during an application modification or execution.

3.2.1. Trust. The PID System has an extremely simple mechanism which is based on the existing social trust relationship. The system consists of a PID issuer, individuals, and service vendors. The PID issuer is an organization that individuals belong to (government, communities, companies, schools, etc). Issuers are basically assumed to have a duty to protect their individuals. This social trust is the basis for the PID system.

3.2.2. Agreement. A registration process intervened by the issuer is carried out for a (cardholder, service vendor) pair so as to give eligibility to the cardholder for using the application provided by the service vendor (and to the service vendor to serve the cardholder). Let, $h \in \mathbf{H}$ (cardholder) and $v \in \mathbf{V}$ (service vendor). Define a relation of **agreement**, **A** over **H**, **V** as the following:

$\mathbf{A} = \{(h, v) \mid registration\ is\ done\ for\ (h, v)\ pair\}$

Given $h \in \mathbf{H}$, $v \in \mathbf{V}$, we say h and v is in agreement relation if $(h, v) \in \mathbf{A}$; h is referred to as a **user** of v, and v is referred to as a **provider** of h (Fig. 1).

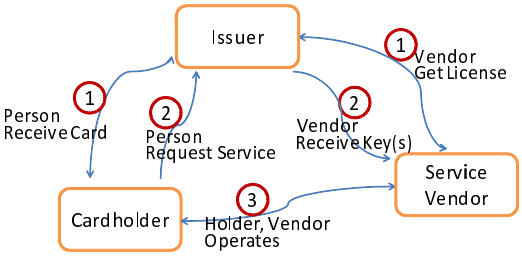

Figure 3. Overview of smart card business framework.

3.2.3. Operation Scenario in the Framework.

1) Registration and Licensing
   An individual asks the issuer for a smart card. The issuer examines the person's identity, and gives a personalized smart card to her (at this stage, she becomes a cardholder). The smart card is capable of computing public functions. During personalization, a unique PID pid is created for the cardholder and is stored in the smart card. A function (or lookup table) to determine the subPID to be shared with a service vendor with a given identity value is also installed in the smart card. The issuer stores the PID and functions (or lookup tables) into a database.
   When a service vendor wants to provide some application for cardholders, it applies to the issuer for permission. Upon examining the service vendor, the issuer grants permission with a service vendor ID.
2) Establishment of Service Agreement
   When a cardholder wants to use an application from a service vendor, she needs to set up an agreement with the service vendor. The agreement is set up with an intervention of the issuer following a registration process. The issuer selects from the pid a unique subPID (substring s_1) and gives to the service vendor. This s_1 is used as a shared secret between the service vendor and the cardholder for authentication purpose. The service vendor receives subPIDs and usually as a list $SLIST_v$.
3) Operation
   As the cardholder wants to receive a service from a service vendor, proper credential established in the above steps are used to do so.

3.3. Proposed Protocol

We propose a PID-based protocol that uses **two distinct subPIDs** for a cardholder and a corresponding service vendor. One of the subPIDs will be used for mutual authentication between the cardholder and the service vendor. The other subPID will be used for checking whether an illegal program is being sent during program modification.

Below, we give the mechanism of our proposed protocol. We assume that the program of the service vendor is P. We also assume the following public functions — a one-way hash function G, and a pseudo-random number generator function R.

3.3.1. Set UP. A One-time set up using a secure channel is done for a (cardholder, service vendor) pair who is in agreement and remains valid until they are in the relation.

1) Registration and Licensing
   The issuer provides the cardholder with a PID, pid, from which two subPIDs, s_1 for mutual authentication and s_2 for program modification, can be determined using f_1 and f_2 respectively. The service vendor will give the program P to the issuer who will return a service id id_v.
2) Establishment of Service Agreement
   At the first instance of interaction between the service vendor and the cardholder, the service vendor receives the credential (s_1) for the cardholder from the issuer using a secure channel. A realization of this process is illustrated in Fig.4.

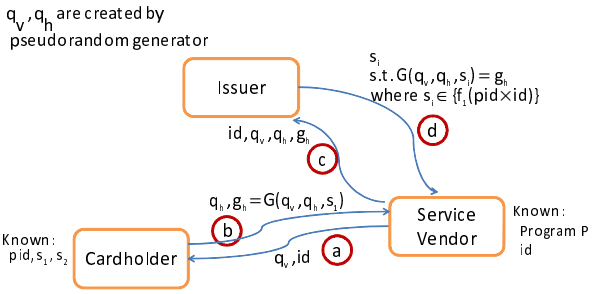

Figure 4. Establishment of service agreement.

3) Authentic Modification of Programs
   Upon receiving a request from the cardholder, the issuer will send s_1 and the signature g of the program P to the corresponding vendor. The

signature g will be generated applying G for P, with s_1 and s_2 as the key.

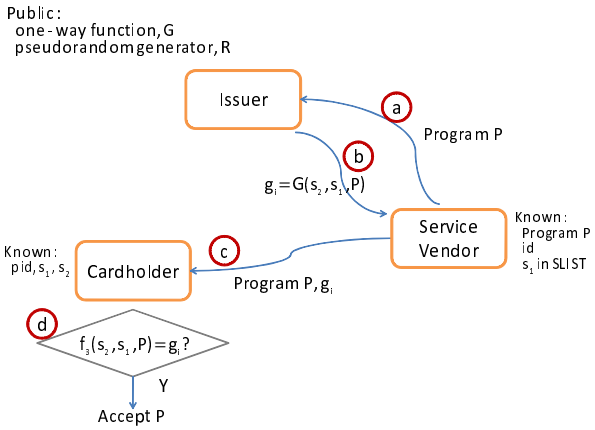

Figure 5. An outline of the proposed protocol.

3.3.2. Protocol Messages. Each round has messages between the smart card (d_h) and the vendor's device (d_v) as follows.

$d_h \rightarrow d_v$: Request (for modification) (1)
$d_h \leftarrow d_v$: id_v, r_v [2] (2)
$d_h \rightarrow d_v$: $r_h, g_h = G(r_v||r_h||s_1)$ [3] (3)
$d_h \leftarrow d_v$: $g_v = G(s_1||r_h)$ (4)
$d_h \rightarrow d_v$: Ok (5)
$d_h \leftarrow d_v$: P, g [4] (6)
$d_h \rightarrow d_v$: Accept/Reject (7)

3.3.3. Protocol Actions. The protocol is carried out whenever an application needs to be modified. It has the following steps.

1) The smart card (d_h) generates a request for modification and sends it to the service vendor's device (d_v).
2) d_v sends id_v and a pseudo-random number r_v to d_h.
3) d_h finds s_1 using id_v, generates a pseudo-random number r_h, computes g_h and sends r_h, g_h to d_v.
4) d_v checks if s_1 is in the subPID list ($SLIST_v = \{s_h|(h,v) \in \mathbf{A}\}$) by computing $G(r_v||r_h||s_h)$ for each $s_h \in SLIST_v$ and comparing with g_h; if a match is found, such that $G(r_v||r_h||s_{h'}) = g_h$, then $s_{h'}$ will equal s_1. d_v computes and sends $g_v = G(s_1||r_h)$ to d_h.
5) The smart card computes $G(s_1||r_h)$ and checks if it equals g_v. If it is equal, the protocol implies a success in authentication and modification of application is permitted.
6) When d_v receives Ok from d_h (protocol message 5), it sends P and g[5] to d_h.
7) d_h stores P in its temporary memory, determines s_1 and s_2, computes $G(s_1, s_2, P)$ and then compares it with g. If the signature values match, a success is realized and P is copied to the application memory. An $Accept$ message is sent to d_v upon completion of the copy. A $Fail$ message is sent otherwise.
8) The protocol round terminates at receiving $Fail$ as protocol message (7) by d_v.

2. r_v is a pseudo-random number generated by d_v.

3. r_h is a pseudo-random number generated by d_h and "||" denotes concatenation operation.

4. Sent if only message (5) is generated; sending may require a sequence of messages.

4. Discussion

4.1. Correctness

At step 3 of protocol action, d_h determines s_1 using the received id_v. at step 4, d_v determines s_1 with exclusive "compute-hash and check-for-match" procedure. The collision resistance property of a hash function ensures that only one match will be found. When this match is found, d_v knows that the message came from a party who indeed possesses the subPID s_1. At step 5, d_h, upon confirming the equality of g_v, knows that g_v came from a party who knows s_1 and r_h sent previously in the protocol. Thus, after the protocol round, both the parties have successfully confirmed the identity of their counter-party.

At step 7, the properties of a keyed hash function ensures that where the cardholder shares keys s_1 and s_2 with the service provider, and the issuer respectively, modification with an illegal program by an illegal party is not possible.

4.2. Attacks

In the proposed protocol, operations between the issuer and the cardholder or between the issuer and

5. The signature obtained from the issuer.

the service vendor is done using a secure channel. Therefore we need to consider only the operations between the cardholder and the service vendor.

The authentication phase (step 1 to step 5) will be compromised if an attacker can reveal s_1, or generate g_h or g_v. One-way or pre-image resistance properties of a hash function ensures that by intercepting all the messages between d_h and d_v, it is impossible to determine s_1. It is also impossible to generate a g_h or g_v. **Replay attack** is not possible because *fresh* random values are used in each round of the protocol.

The program modification phase will be compromised if an attacker can find s_1, s_2 or can find a program P' ($P' \neq P$) to generate the signature g. One-way or pre-image resistance properties of a hash function ensures that by intercepting all the messages between d_h and d_v, it is impossible to determine s_1, s_2, or find a P'. It is also impossible to find a program P'.

The attacker may try to modify the massage (P, g) in step 6. However, without s_1, s_2 it is not possible to create a valid signature. Moreover, s_1, s_2 cannot be recovered from the signature (hash values). Thus any altered message in step 6 will result in a **denial-of-service**.

The attacker, may try to use a previously recorded message (P, g), but it is not clear how beneficial that will be.

4.3. Performance and Cost

As an alternative to the proposed protocol, public key based scheme (Public Key infrastructure, PKI) can also be adopted which will reduce the computation time for the server system. However, this will lead to an increase in the computation time needed by the smart card, and an extra device cost. In general, public-key algorithms are at least 1000 times slower than symmetric algorithms and requires more area ([4],[5]).

The proposed protocol uses shared secret and a hash function. The shared secret (subPID) in the scheme is a considerably long bit string with random properties and therefore cannot be determined easily using brute-force attacks. Known hash functions can be implemented with exclusive-or (XOR) operations that can provide low cost for smart card devices. In general, hash computations require lighter computation than public key based computations.

However, prior to each program modification, the proposed protocol requires to execute step (2) of one-time setup procedure in order to notify the signature. This incurs cost for setting up secure channels and an extra inconvenience.

On the other hand, in a PKI-based scheme, a Trusted Third Party (TTP) generates the signature for a program of a service vendor. The smart card receives the signature from the corresponding server system and then validates the signature using the public key of the TTP. The public key of the TTP is known to the smart card during the personalization phase and is fixed throughout the life cycle. Therefore, establishment of a secure channel between the smart card and the issuer is not necessary. However, generation and exchange of signatures between the issuer and the service vendor can not be avoided. Moreover, known public key based algorithms use exponentiation and modular arithmetic that require high computation and device cost.

In an alternative settings, it is also possible for a service vendor to delegate the task of program installation to the trusted issuer. This enables the cardholder to obtain the program and its signature from a single entity. However, such a centralized system is prone to congestions during program installation and requires a proper scheduling scheme that is accepted by the service vendors.

5. Related Work

MULTOS

MULTOS (Multi-application Operating System) [6] was the first platform for multi-application smart cards. As the most mature technology, it has an infrastructure in place to guarantee secure loading/modification of programs either by the issuer or by a service vendor with a issuer-certified asymmetric key. However, MULTOS uses a low-level programming language called MEL, which does not have a large base of developers programming in it. Also involvement of several entities makes the system complex and costly.

NICE

Developed by NTT, NICE (Network-based IC card Environment) [7] uses a Java-based proprietary multi-application smart card (called ELWISE card) to support an application-oriented business model and network-based operations and management. Although

NICE architecture claims to accommodate different platforms, at present only NTT-developed platforms are realized. NICE uses asymmetric encryption scheme.

PIPM

In our previous work [8], a system for post issuance program modification, PIPM, has been proposed. In this scheme, for every program update or loading, a direct communication between the issuer and the cardholder is established. A specialized device allows the processor in the smart card to receive credentials 'signed' by the issuer. If the signature matches that generated from the program under consideration, a write permission is granted to the vendor. Despite its novelty, the direct communication between the issuer and the card holder turns out to be unrealistic in practical scenarios. In addition, the authentication and program uploading procedures are complex.

In this paper, we propose a simpler mechanism that avoids the direct communication between the issuer and the cardholder. Our current proposal is therefore, simple and more practical. In addition, the implementation of the system is simple and can be readily done using Java Card Technology[9].

6. Conclusion

In this paper, we presented a framework for multi-application smart cards that allows authentic post-issuance program modification on smart cards. The proposed system has a very simple mechanism based on existing social trust and primitive secret sharing schemes that requires low cost to provide moderate security. Our proposed framework does not require direct issuer-cardholder interaction for obtaining new services. Therefore, it is easily deployable. In addition, the implementation of the system is simple and can be readily done using Java Card Technology.

Acknowledgment

This work has been supported by the Grant-in-Aid for Scientific Research (A) No.19200004 of the Ministry of Education, Science, Sports and Culture (MEXT) from 2007 to 2010. We are grateful for their support. The authors would like to thank Prof. Masayoshi Yoshimura of Kyushu University for valuable suggestions on preparing this manuscript.

References

[1] B. Schneier and A. Shostack, "Breaking up is hard to do: Modeling security threats for smart cards," in *In USENIX Workshop on Smart Card Technology*. USENIX Press, 1999, pp. 175–185.

[2] H. Yasuura, "Towards the digitally named world - challenges for new social infrastructures based on information technologies," in *Proceedings of Euromicro Symposium on Digital System Design - Architectures, Methods and Tools-(DSD2003)*, Sep. 2003, pp. 17–22.

[3] Y. Hamasaki and H. Yasuura, "A proposal of secure information infrastructure based on pid (in japanese)," in *DICOMO2002 Symposium*, Jun. 2002.

[4] B. Schneier, *Applied Cryptography*. John Wiley & Sons, 1996), 1996.

[5] P. v. O. A. Menezes and S. Vanstone, *Handbook of Applied Cryptography*. Boca Raton CRC Press, 1997), 1997.

[6] MULTOS, "http://www.multos.com/." [Online]. Available: http://www.multos.com/

[7] R. Toji, Y. Wada, S. Hirata, and K. Suzuki, "A network-based platform for multi-application smart cards," in *Proceedings of the 5th IEEE Int'l Conference on Enterprise Distributed Object Computing (EDOC 2001)*, Sep. 2001, p. 34.

[8] M. M. Uddin, Y. Nohara, D. Ikeda, and H. Yasuura, "A multi-application smart card system with authentic post-issuance program modification," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E91-A, no. 1, pp. 229 – 235, 2008.

[9] J. C. Technology. [Online]. Available: http://java.sun.com/products/javacard/index.jsp