자바카드 애플릿 설정에 따른 사용자 인증의 다각화

Diversification of User Authentication by Writing Applet on Java Card

Young-Sang Song*, In-Chul Shin**
송 영 상*, 신 인 철**

Abstract

Recently, IC cards are used to protect personal information and to have user verification. Among them, the usage of Java Cards which can contain applications after issuing are increasing and installing several applets on Java card is possible. When Java Cards are used, applet works after completing user identification. In this paper, we designed, embodied and verified the mechanism of user identification process according to PIN setting of applets; Stored_PIN, Install_PIN and Update_PIN. These several applications of Java cards will be used for user identification independently or multiply, while using diverse user identification.

요 약

최근 개인 정보보호 및 사용자 인증에 IC카드가 사용되고 있다. 이 중 발급 후 응용프로그램을 적재 할 수 있는 자바카드의 활용이 증가하고 있으며, 한 장의 자바카드에 여러 개의 애플릿을 설치할 수 있다. 자바카드 사용 시 애플릿은 사용자 인증이 이루어진 후에 동작한다. 본 논문에서는 자바카드 내에서 동작하는 애플릿의 PIN 설정에 따라 Stored_PIN, Install_PIN, Update_PIN으로 구분하여 사용자 인증이 수행되는 메커니즘을 설계 구현하고 이를 검증한다. 사용자 인증의 다각화를 이용하여 자바카드의 여러 응용프로그램이 독립적 또는 복합적으로 사용자 인증에 활용될 수 있을 것이다.

Key words : Smart Card, Java Card, Applet, PIN, User Authentication, Verify

Ⅰ. 서론

최근 정보 통신의 발달로 다양한 서비스가 사용자에게 제공되고 있으며, 이로 인한 개인 정보의 기밀성, 무결성, 인증, 부인봉쇄 등의 정보보호 서비스에 대한 관심도 높아져가고 있다. 이러한 정보보호 서비스를 제공하는 물리적 장치로 IC카드 사용이 급증하고 있다[1~6]. IC카드는 플라스틱카드에 마이크로프로세서, 메모리 및 암호 알고리즘을 수행하기 위한 보조프로세서 등을 내장한 카드이다. 일반적으로 IC카드는 스마트카드라 불리며, 스마트카드의 운영체제에 따라 네이티브카드(native card)와 개방형 플랫폼을 지닌 자바 카드(java card), WFSC(Windows for Smart Card), MULTOS(Multi-OS) 등으로 구분된다. 개방형 플랫폼을 내장한 카드는 발행 후 최종 사용자(end user)가 원하는 응용프로그램을 카드에 추가할 수 있는 발급 후 적재(post-issuance) 개념을 도입했다[4~10].

자바카드는 이진 코드의 이식성(portability), 상호운용성(inter-operability)이 뛰어나고 타입 검사 등에 악의적 코드에 대한 보안성(security)을 지닌 자바 언어를 스마트카드의 실시간 환경에 대해 최적화하고 있는 장점으로 사용이 증가하고 있다. 자바카드는 일반적인 스마트카드에 적용되는 모든 표준을 따르는 전형적인 스마트카드로 하위의 운영체제 위에 존재하는 자바카드 가상기계(JCVM : Java Card Virtual Machine)에서 바이트 코드(bytecode)를 수행하고 메모리, I/O 같은 카드 내의 모든 자원에 대한 접근을 제어하는 애플릿(applet)으로 구성된다[4, 8, 11].

애플릿은 자바카드용 응용프로그램이라 할 수 있으며, 응용프로그램이 구동될 때 마다 자바 가상머신에

* 檀國大學校 電子電氣工學部
** 檀國大學校 電子電氣工學部
接受日:2009年 12月 12日, 修正完了日: 2009年 12月 29日

의해 번역된다. 애플릿은 구동될 때마다 사용자 인증을 위해 PIN(Personal Identification Number)을 사용한다. 그러나 PIN의 재설정 및 갱신을 위해 애플릿을 다시 설치해야하는 번거로움이 있고, 하나의 애플릿에서의 다중 사용자 지원이 되지 않는다.

본 논문에서는 다중 사용자 및 다중 PIN을 사용하여 애플릿의 보안성 보장 및 다양한 사용을 가능케 하는 PIN 설정 메커니즘을 제시한다. 이를 위해 자바카드에 애플릿 설치와 애플릿 구동시기에 따른 PIN 설정을 위한 3가지 방법으로 구분하였다. 첫 번째 메커니즘은 애플릿 내에 PIN을 설정하는 Stored_PIN, 두 번째 메커니즘은 애플릿을 카드에 로드하여 설치 시 PIN을 설정하는 Install_PIN, 그리고 마지막은 애플릿 설치 후 PIN을 설정하는 Update_PIN으로 구분한다. 다음과 같은 방법을 이용하여 애플릿 사용자에 따라 또는 사용자의 권한에 따라 인증이 이루어짐으로써 사용자 인증의 다각화로 한 장의 카드에서 동작하는 여러 애플릿간의 독립된 서비스를 제공할 수 있다. II장에서는 자바카드에 대해 살펴보고, III장에서 본 논문에서 설계 및 구현한 애플릿을 살펴보고 이의 동작을 확인한다. 마지막으로 결론 및 향후 과제를 살펴본다.

II. 관련 연구

1. 자바카드

자바카드란 스마트카드 기술에 자바의 기술을 접목시킨 것이다. 일반적으로 자바카드는 메모리, 통신, 보안 그리고 어플리케이션의 실행을 지원하고, 메모리는 RAM과 ROM 그리고 EEPROM으로 구성된다.

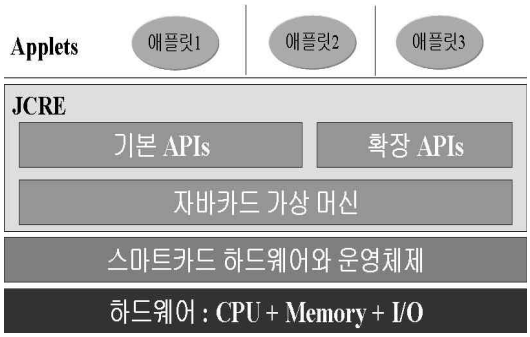

그림 1. 자바카드 구조
Fig. 1. Java Card Architecture

자바카드는 자바 언어의 다차원 배열, 큰 데이터 유형, 쓰레드, 객체 병렬화, 문자와 문자열 등 특징을 제외한 언어서브셋(language subset)을 지원하고 이를 실행시킬 수 있는 가상머신을 필요로 한다. 자바카드 가상머신은 카드외부(off-card) 가상머신과 카드내부(on-card) 가상머신으로 나뉘는 분할 가상기계로 이루어진다. 이는 수행 엔진인 인터프리터를 지칭하며 넓은 의미로는 프래임워크가 중심이 되는 시스템 클래스 API와 인터프리터, 메모리 관리 루틴, 예외 처리 루틴 및 운영체제와의 인터페이스 등을 포함하는 JCRE을 의미한다. 자바카드의 일반적인 구조를 그림 1에 나타내었다. 최하위 단은 자바카드의 하드웨어와 COS(Card Operating System)가 위치한다. 그 윗단에는 JCRE가 위치하여 최상위 애플릿을 동작시키기 위한 작업을 수행하게 된다.

자바카드는 다음과 같은 특징을 제공한다.

- 플랫폼 독립성(platform independence) : 자바카드의 JCRE기반으로 작성되어 상호 호환성이 있다.
- 복수 응용 프로그램(multiapplication program) : 하나 이상의 응용프로그램이 카드에서 동작가능.
- 응용 프로그램의 갱신(application program update) : 카드 발급 후 응용프로그램을 설치 및 갱신가능.
- 융통성(flexibility) : 자바카드기술의 객체지향 기술은 스마트카드 프로그래밍에 융통성을 제공한다.
- 호환성(compatibility) : 자바카드 국제 표준인 ISO 7816과 산업 표준인 EMV와 호환한다.

2. 자바카드 APDU

자바카드와 애플릿 간의 통신을 위한 APDU (Application Program Data Unit)교환으로 카드의 동작이 수행된다. 애플릿과 외부 응용 프로그램 간의 통신은 명령어(command) APDU와 응답(response) APDU로 구성되어 APDU 교환을 통해서 이루어진다. APDU의 구조는 필수 항목인 헤더(header) 부분과 선택 항목인 바디(body)부분으로 나눌 수 있다.

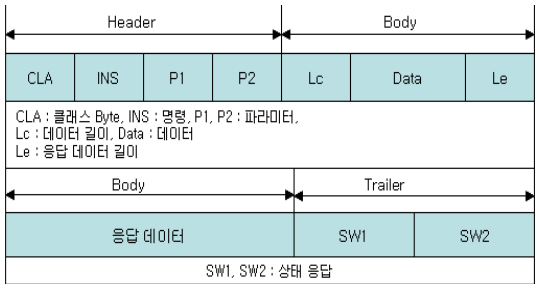

그림 2. 명령, 응답 APDU
Fig. 2. Command and Response APDU

그림 2에 APUD의 구조 형태를 나타내고 있다. 명령 APDU는 필수부분인 헤더부분과 선택부분인 바디로 나눠진다. 헤더 부분은 클래스(CLA : class), 명령

어(INS : Instruction), 파라미터(P1, P2)로 총4byte로 나타내며, 바디 부분는 전송데이터의 길이(Lc), 데이터(Data), 응답데이터 길이(Le)로 구성되며, 데이터의 길이는 가변적이다. 명령 APDU는 바디부분의 구성에 따라 총 4가지의 형태를 갖게 된다. 응답 APDU는 필수부분인 꼬리(trailer)와 선택 부분인 바디로 구성되며, 바디에 따라 2가지 형태의 응답으로 구성된다.

3. 애플릿

애플릿은 자바카드 플랫폼에서 동작하는 자바 응용 프로그램이다. 하나의 카드에 여러 개의 애플릿이 탑재되며, AID(Application IDentifier)를 이용하여 여러 개의 애플릿을 구분한다. 애플릿은 JCRE에 위치하여 카드 내에서 정해진 규칙에 따라 동작한다. 또한 ROM에 설치될 필요 없이 카드의 EEPROM에 다운로드 하여 사용한다.

기존 애플릿 작성은 카드 외부에서의 동작과 카드 내에서 동작으로 크게 두 부분으로 나누어진다. 애플릿 동작을 위한 코드는 SUN사에서 제공되는 API를 이용하여 install(), select(), deselect(), process()의 필수 메소드를 포함하여 작성하고 이를 컴파일 하여 클래스 파일을 만들게 된다. 또한 생성된 클래스 파일과 익스포트(export)파일을 컨버트(convert)시켜 CAP (Converted APplet)파일을 만들어 카드에 로딩 시키게 된다. 로드된 CAP 파일은 인스톨하여 사용가능한 애플릿 상태가 되어 외부 어플리케이션 프로그램으로부터 그림 2와 같은 형식의 명령 APDU를 받아 처리하고, 그에 따른 응답 상태를 외부 어플리케이션 쪽으로 보내 처리 결과를 알려준다. 그림 3은 애플릿 동작 절차를 나타내었다.

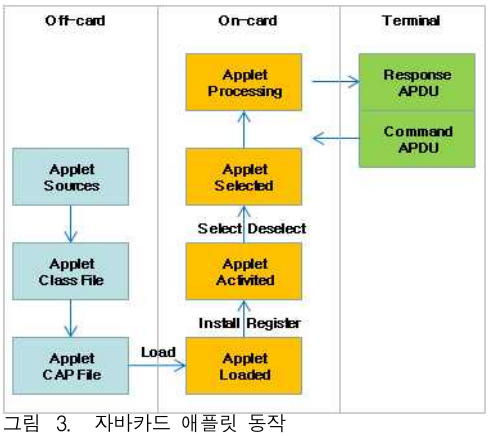

그림 3. 자바카드 애플릿 동작
Fig. 3. Applet Process on Java Card

자바카드의 애플릿 작성을 위해 고려해야 할 4개의 필수 메소드의 역할을 나타내고 있다.

- install() Method
애플릿 사용 환경을 만들기 위해 카드에 설치
- select() Method
애플릿이 선택되었을 때 동작을 활성화
- deselect() Method
애플릿 선택을 해제될 때 활성화시킬 동작 정의
- process() Method
외부의 응용 프로그램으로 부터 명령을 받아 처리

일반적으로 자바 애플릿 프로그램은 그림 3에서와 같이 애플릿 로드(applet loaded) 후 인스톨과정에서 사용자 PIN을 설치한다. 사용자 인증 절차는 카드에 설치되어 있는 애플릿을 선택 후(applet selected APDU) 사용자 인증이 이루어진다. 사용자는 설정된 PIN을 제출하고 일치할 경우 사용자 인증이 수행 후 애플릿 사용이 이뤄진다. 그러나 PIN 값이 3번 연속으로 일치할지 않을 때 카드 사용 제한을 한다. 사용자 PIN 길이는 일반적으로 8바이트를 사용한다.

III. PIN설정에 따른 애플릿

기존 애플릿은 하나의 애플릿에 PIN을 설정하여 사용자 인증을 한다. 그러나 다중 사용자 지원 및 PIN의 갱신 및 재설정을 위해 애플릿을 재 설치해야 하는 단점이 있다. 본 논문에서는 이를 해결하기위해 PIN설정을 위한 3가지 메커니즘을 제시한다. 이는 애플릿 설정 시기에 따라 Stored_PIN, Install_PIN, Update_PIN으로 구분하였으며, 이를 그림 4에 설정 시기에 따른 메커니즘을 나타내고 있다.

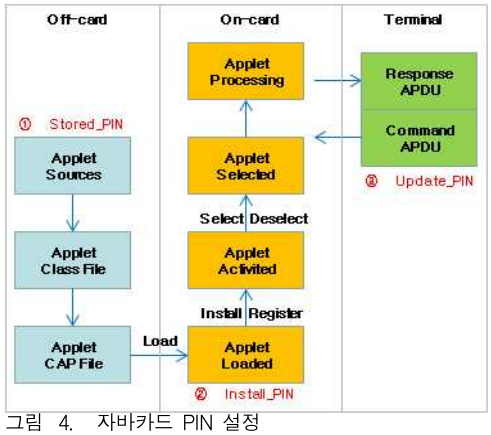

그림 4. 자바카드 PIN 설정
Fig. 4. PIN Setting on Java Card

PIN설정은 8바이트의 PIN과 3번의 입력 제한을 갖도록 설계하고, 또한 자바카드 API에서 제공하는

OwnerPIN 클래스를 이용하여 EEPROM영역에 PIN 설정했다.

본 논문에서 제시하는 PIN설정 메커니즘을 이용하여 이를 지원하는 각각의 애플릿을 설계 구현하여 검증하였다. 3가지 메커니즘은 사용자 PIN을 검증하는 명령APDU로 CLS는 0x00, INS는 0x20으로, 파라메터 P1, P2는 0x00으로 모두 동일하게 설정하였다. 애플릿에서의 사용자 인증은 process() 메소드 동작 시 verify() 메소드를 호출하여 사용자 인증이 이루어진다. 사용자 인증 실패 시 오류 명령어는 0x6300으로 예외처리 한다.

1. PIN의 다각화 설계 및 구현

가. Stored_PIN

Stored_PIN은 그림 4와 같이 애플릿 소스작성 시 정해진 8바이트의 PIN값을 애플릿 소스에 설정한다.

EEPROM 영역에 설정할 PIN을 저장하기 위해서 공간을 할당하고 PIN을 다루기 위한 생성자를 만들어 CAP파일 코드가 생성된다. 생성된 코드는 카드로 로드 후 인스톨하여 사용가능하게 된다. 사용자 인증을 위해 제출한 PIN은 미리 설정해놓은 PIN과 비교하여 사용자 인증이 이루어진다. PIN은 코드를 수정하여 다시 로드 후 인스톨할 때 까지는 수정이 불가능하다. 보안성이 높은 PIN을 설정하고자 할때 유용하다. 애플릿 제공자의 인증 시 유용하다. 아래 코드는 사용자 인증을 위해 APDU의 명령어 정의와 인증 실패했을 경우 예외처리 및 PIN의 사용제한과 크기 설정코드를 나타내고 있다. 그다음 코드는 Stroed_PIN을 이용하여 PIN 저장 코드를 나타내고 있다.

```
// Command APDU Instruction byte Constants
final static byte VERIFY              = (byte) 0x20;

// PIN Constants
final static byte PIN_TRY_LIMIT   =(byte)0x03;
final static byte MAX_PIN_SIZE    =(byte)0x08;

// PIN Verification Fail
final static short SW_VERIFICATION_FAILED  = 0x6300;
```

```
// Store PIN
final static byte[] mypin={(byte)0x11, (byte)0x22,
        (byte)0x33, (byte)0x44, (byte)0x55, (byte)0x66,
        (byte)0x77, (byte)0x88};
```

나. Install_PIN

Install_PIN은 기존 자바카드 애플릿에서 사용하는 방법으로 그림 4에서와 같이 애플릿 작성 후 카드에 CAP 파일을 로드하여 애플릿 인스톨 시 PIN값을 파라미터 값으로 주고, 이를 셋팅 한다. 이러한 동작은 CAP 파일 로드 시 파라미터 값으로 PIN값을 같이 제공해야하므로 PIN값의 노출 될 수 있어 보안성이 요구된다. PIN값을 수정하기 위해서는 애플릿을 다시 설치하고 새로운 PIN을 제출하여 셋팅하게 된다.

Install_PIN은 애플릿 제공자의 인증 시 유용하게 사용되고, 애플릿 설치 시 제공되는 파라미터 값을 갖고 명령 APDU의 헤더부분과 고유 애플릿 AID, PIN값을 구분하여 설정하게 된다. 아래 코드는 install() 메소드에서 인스톨 시 호출하는 생성자 및 제출 PIN을 추출하는 코드를 나타내고 있다.

```
private InstallPIN (byte[] data, short Offset, byte
bLength){
 pin = new OwnerPIN(PIN_TRY_LIMIT, MAX_PIN_SIZE);
 byte iLen = data[Offset]; // aid length
 Offset = (short)(Offset+iLen+1);
 byte cLen = data[Offset]; // info length
 Offset = (short)(Offset+cLen+1);
 byte aLen = data[Offset]; // applet data length

 pin.update(data, (short)(Offset+1), aLen);
 register();}
```

다. Update_PIN

Update_PIN은 그림 4와 같이 애플릿에 인스톨 후 PIN 값을 설정시키기 위한 별도의 명령어를 두어 PIN값을 설정 하게 된다. 애플릿이 선택이 되면 가장 먼저 사용자의 PIN을 설정 하는 작업이 선행되어야 한다. PIN값을 수정하기 위해서는 애플릿을 다시 설치하거나 또는 PIN값을 수정 할 수 있는 명령어를 두어 계속 적으로 PIN값을 변경하게 할 수 있다. 이 방법은 애플릿 사용자가 유용하게 사용할 수 있다.

Update_PIN은 초기 애플릿 구동 시 PIN을 설정해야한다. 그러므로 설정할 수 있는 명령을 따로 설계하였다. 설정 후 사용자 인증은 앞에서 소개한 방법으로 이루어진다. 아래 코드는 PIN설정 및 갱신을 위해 사용되는 APDU 명령을 나타내고 있다.

```
// Update PIN
final static byte INS_PUT_PIN       = (byte) 0x26;
```

라. Multi_PIN

앞에서 언급한 3가지 방법을 하나의 애플릿에 적용하면 안전하고 다양하게 카드 제공자, 애플릿 제공자, 사용자 간의 상호 인증을 통한 안전한 통신을 하게

된다. 또한 Multi_PIN은 애플릿을 어떻게 구성하느냐에 따라 실행 방법의 순서가 결정되며, 사용자 인증 동작도 결정되게 된다. 애플릿의 활용을 사용자에 따라 차등할 경우 제출 PIN에 따라 애플릿 참조 객체 및 동작을 제한 할 수 있다.

Ⅳ. PIN 동작 및 검증

애플릿 PIN 설정에 따른 메커니즘은 Stored_PIN, Install_PIN, Update()_PIN로 동일한 환경으로 설계 구현하였다. 자바카드 개발 환경은 애플릿 구현을 위해 자바카드 개발 툴이 플러그인된 이클립스를 이용하였으며, 테스트카드는 IBM의 JCOP 3.0 자바카드를 이용하여 결과를 확인 하였다. 본 장에서는 이들의 동작 절차와 동작에 따른 결과를 나타내었다.

1. PIN설정 다각화 동작

본 논문에서 애플릿 코딩은 8바이트의 PIN과 사용자 인증 명령 APDU 및 예외 처리를 동일화 하여 설계 구현하였다. 아래 표 1은 오프카드에서 설계된 애플릿 코드를 온카드로 로드되는 CAP파일의 코드 크기 와 동작 절차 및 장단점을 나타내고 있다.

Stored_PIN이 가장 작은 코드 크기를 갖고 있으며, Update_PIN 애플릿이 PIN을 설정하기 위한 코드를 포함하고 있어 코드 크기는 가장 큰 것을 확인하였다. 동일한 환경에서 4개의 메소드와 PIN을 검사하는 메소드 및 APDU의 체크 루틴을 동일하게 구성하였다.

표 1. PIN 메커니즘에 따른 애플릿 크기와 단계
Table 1. PIN mechanism based on size and step to applet.

종류 / 항목	Stored_PIN	Install_PIN	Update_PIN
CAP Size (byte)	203	253	284
Process Step	1	1	Initial 2 after 1
장점	보안성이 높고 코드수정 용이	보안성이 높다	PIN 재설정, 갱신이 쉽다
단점	PIN 갱신이 힘들다	PIN 갱신이 힘들다	코드 Size가 크다

애플릿 동작은 카드를 리더기에 삽입 후 사용 애플릿을 선택하기 위한 명령APDU를 전송하게 된다.

전송되는 APDU는 CLA(00), INS(A4), P1(04), P2(00), Lc(AID Length), Data(AID)이고, 응답APDU (9000) 이 전송되면 애플릿 선택이 성공적으로 선택이 된 것을 확인할 수 있다. 표 2는 PIN 설정에 따른 각각의 애플릿 AID를 나타내고 있다.

표 2. 각 PIN의 AID
Table 2. AID for each PIN

PIN종류	Applet AID
Stored_PIN	53746F726550494E417070
Install_PIN	496E7374616C6C50494E417070
Update_PIN	55706461746550494E417070

다음 동작은 사용자 인증을 위해 PIN을 제출하게 된다. PIN을 제출하는 명령APDU는 CLA(00), INS(20), P1P2(0000), Lc(08), Data(1122334455667788)을 전송하여 응답APDU가 “9000”이면 사용자 인증이 이루어진다. 다음과 같은 절차를 그림 5에서 나타내고 있다.

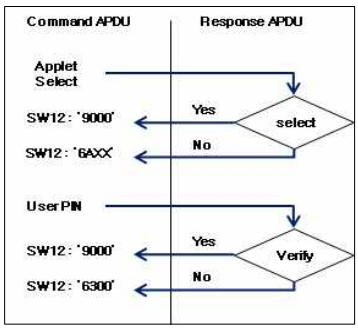

그림 5. 사용자 인증 절차
Fig. 5. Proceeding Verification of User

2. PIN설정 다각화 검증

본 논문에서 제시한 PIN 메커니즘은 Stored _PIN, Install_PIN, Update_PIN이다. 애플릿 동작 검증을 위해 IBM JCOP Bio3.1 자바카드를 이용하여 검증을 확인 하였다. 그림 6은 각각의 PIN설정에 따른 애플릿을 카드에 로딩 후 인스톨된 자바카드의 목록을 보여주고 있다.

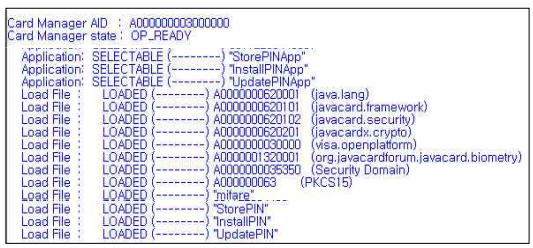

그림 6. 애플릿 로딩 및 설치
Fig. 6. Applet Loading and Install

그림 7은 Stored_PIN 애플릿 선택과 PIN인증을 보여주고 있다. Stored_PIN의 AID는 StoredPIN을 Hex으로 표현하여 선택하였고, PIN값은 8바이트 11~88까지 사용하였다. 마지막 명령은 응답APDU가 사용자 인증 실패를 표시하고 있다.

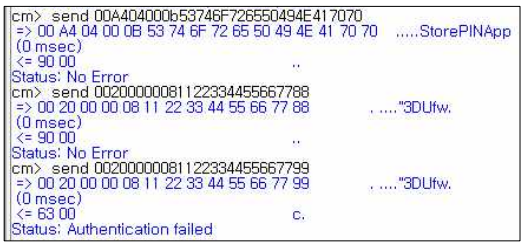

그림 6. Stored_PIN 애플릿의 PIN 검증
Fig. 6. Verify PIN of Stored_PIN Applet

V. 결론

자바카드에서 동작되는 응용프로그램을 애플릿이라 하고, 이는 오프카드에서 프로그래밍되어 카드에서 읽을 수 있는 파일 형태인 CAP 파일로 변환 후 카드에 로드 및 인스톨된다. 애플릿 사용은 애플릿 선택 후 정당한 사용자인지에 대한 확인 절차를 위해 PIN을 제출하여 올바른 PIN일 경우 사용자 인증이 이루어진다. 그러나 하나의 애플릿에 한명의 사용자가 셋팅 되고, PIN의 재설정 및 갱신을 위한 동작을 위해 애플릿을 재설치하는 번거로움 또는 외부에서 악의적으로 PIN값을 확인 할 수 있는 보안성의 취약함이 있다.

본 논문에서는 하나의 애플릿에 한명이상의 사용자 설정뿐만 아니라 보안 정도에 따라 PIN설정을 안전하고, 다양하게 PIN을 설정할 수 있는 애플릿 PIN 설정 메커니즘을 설계, 구현 및 검증했다.

PIN설정 메커니즘은 애플릿에 PIN 설정하는 방법에 따라 3가지로 제시하고, 이들을 독립적 및 복합적으로 사용할 수 있도록 설계하였다. 애플릿의 인스톨이 이루어지는 시점을 기준으로 애플릿 소스 작성 시 PIN을 설정하는 Stored_PIN, 애플릿 인스톨 시 PIN을 설정하는 Install _PIN 방법, 애플릿을 설치 후 PIN을 설정하는 Update_PIN 방법을 제안한다. 위에서 제시하는 3가지 방법을 복합적으로 사용하여 주체에 따라 하나의 애플릿의 사용 권한 및 보안성이 제공된다.

본 논문에서 제시하는 PIN 설정 방법은 명령APDU와 예외처리 및 PIN길이를 동일한 환경으로 작성하였다. 작성된 CAP파일의 코드 사이즈는 Stored_PIN (204bytes), Install_PIN (253bytes), Update_PIN (284bytes)의 차이가 있었으며, 애플릿 동작은 Stored_PIN이 가장 간편하고, Update_PIN이 PIN을 설정하는 단계가 추가되어 동작된다. 또한 PIN은 본 논문에서 제시하는 PIN 설정 방법을 복합적으로 사용하여 PIN설정, 갱신을 안전하게 제공한다.

자바카드의 애플릿 개발 시 본 논문에서 제시하는 PIN설정 방법을 독립적으로 또는 복합적으로 사용하여 개발의 시간의 단축 및 코드 단축을 할 수 있을 것으로 기대된다.

참고문헌

[1] W. Rankl, W.Effing, Smart Card HandBook, WILEYVCH, 2000.
[2] Timothy M. Jurgensen, Scott B.Guthery, “Smart Cards”, Person Education, 2002.
[3] S. Oaks, JAVA Security, O'REILLY, 1998
[4] Z. Chen, Java Card Technology for Smart Cards, Addison Wesley, 2000.
[5] V. Hassler, M. Manninger, M. Gordeev, C. Muller, “Java Card for E-Payment Application”, Artech House, 2002.
[6] M. Oestreicher, “Transactions in Java Card”, Annual Computer Security Application Conference, pp.291-298, 1999.
[7] L. Casset, L. Burdy, A. Requet, “Formal Development of an Embedded Verifier for Java Card Byte Code” International Conference on Dependable System and Networks, pp.51-56, 2002.
[8] Jinyoung Moon, Jongyoul Park, Euihyun Paik, "JavaCard-based Two-Level User Key Manage -ment for IP Conditional Access Systems", ICON2007, Networks, 19-21 Nov., pp.72-76, 2007.
[9] MacDonald, J.A.; Mitchell, C.J, "Using The GSM/UMTS SIM to Secure Web Services", Mobile Commerce and Services, WMCS '05. The Second IEEE International Workshop on 19-19 July pp.70 - 78, 2005.
[10] 김호원, 최용제, 김무섭, 박영수, “비대칭키 암호 알고리즘을 고속으로 수행하는 자바카드 구현 및 성능 평가”, 대한전자공학회 하계종합학술대회 논문집 제24권 제1호, pp.55-58, 2001.
[11] 문상재, 이필중, “차세대 IC 카드를 이용한 정보보호 신기술 시스템 개발”, 정보통신부 보고서, 1997
[12] 김연선, 이창욱, “자바카드 애플릿 설계 및 검증에 관한 연구”, 한국통신정보보호학회 종합학술 발표회논문집, Vol.10, No.1, pp.805, 2000.

[13] 김성준, 이희규, 조한진, 이재광, “자바카드 기반 공개키 암호 API를 위한 임의의 정수 클래스 설계 및 구현”, 정보처리학회, 9권 2호, pp.163-172, 2002.
[14] 김도우, 정민수, “자바카드 플랫폼상에서 자바 클래스 파일의 최적화 연구”, 멀티미디어학회 논문지, 6권 7호, pp.1200-1208, 2003

저 자 소 개

송 영 상 (정회원)

1998년 : 삼척산업대학교 전자공학과 졸업 (공학사)
2000년 : 단국대학교 대학원 전자컴퓨터공과 (공학석사)
2007년 : 단국대학교 대학원 전자컴퓨터공과 (공학박사)
2008년 8월~현재 : 단국대학교 전자전기공학부 강의전임강사
<주관심분야> 정보보호, 스마트카드, 자바카드, 전자상거래, 마이크로프로세서, SoC설계

신 인 철 (정회원)

1973년 : 고려대학교 전자공학과 졸업 (공학사)
1978년 : 고려대학교 대학원 전자공학과 (공학석사)
1986년 : 고려대학교 대학원 전자공학과 (공학박사)
1979년 9월~현재 : 단국대학교 전자전기공학부 교수
<주관심분야> 병렬처리, 정보보안, 스마트카드, 자바카드, 전자상거래