스마트카드 응용프로그램의 다중 서비스 기법 구현에 관한 제안†

(On the Implementation of the Multiple Service for the Smart Card Application Programs)

김 시 관*, 임 은 기*
(Si-Gwan Kim, Eun-Ki Lim)

요 약 최근 다중 응용 프로그램이 적재된 스마트카드는 여러 가지 장점들로 인해서 많은 관심을 끌고 있다. 일반 사용자들은 지갑에 가지고 다니는 다양한 기능을 가지고 있는 카드의 수를 줄이기를 원하고 있으며 카드 발행자는 카드의 발행 이 후 새로운 응용 프로그램을 추가하거나 기존의 응용 프로그램을 업그레이드시키기 위해서, 개발자들은 개발 기간 단축 가능성등의 요구로 인해 다중 응용 프로그램을 지원하는 스마트카드의 사용이 활성화되고 있는 추세에 있다. 또한, 다중응용프로그램 스마트카드 서비스는 사업 파트너 상호간의 비즈니스 상승효과 측면에서 볼 때도 아주 많은 장점을 가지고 있는 실정이다. 최근에는 자바카드, MULTOS 와 같은 다중응용프로그램 스마트카드를 지원하기 위한 운영체제가 제안되었다. 본 논문에서는 스마트카드의 산업계 표준으로 자리잡아 가고 있는 자바카드를 중심으로 다중응용프로그램 스마트카드의 작동 원리를 연구하고 이와 관련된 여러 가지 보안 측면에서의 문제점을 파악한 뒤 키관리 기법, 응용프로그램 다운로드 방안, 보안 해결 기법등을 제안한다.

핵심주제어 : 스마트카드, 다중응용프로그램, 키관리

Abstract Recently, smart cards with multi-applications loaded are becoming popular owing to many advantages. As many casual users would like to reduce the number of plastic cards, and card issuers want to upgrade the existing applications or add the new applications, and developers wish to reduce the development turn-around time, multi-applications smart cards are becoming important. In addition, many advantages exist between the business partners as smart card applications can share critical information. New operating systems such as JavaCard and MULTOS are suggested for the multi-applications smart card service recently. In this paper, after we review the principles of operations of smart card, we propose the various security mechanisms for the multi-application JavaCard service environment, which is becoming de facto standard in the industry.

Key Words : Smart Card, Multiple Service Program, Key Management

1. 서 론

스마트카드[10]는 1968년 Jergen Dethloff가 마이크로프로세서의 휴대 수단으로써 IC카드에 대한 개념을 제안하고, 이를 1974년 프랑스의 Roland Moreno에 의해서 탄생하였다. 당시 카드는 메모리 기능만을 가진 메모리카드(memory card)로써, 카드 내에서 연산기능을 수행하는 마이크로프로세서는 탑재하지 않았다. 1977년 미국 모토롤라와 프랑스 불(Bull)사가 마이크로프로세서와 독립된 메모리를 가진 카드를 생산하여 프랑스 금융서비스 분야에 적용하였다. 이어 1980년대에 접어들면서 스마트카드의 응용이 가속화되고, 1982

† 본 연구는 금오공과대학교 학술연구비에 의하여 연구되었음.
* 금오공과대학교 컴퓨터 공학부

년 프랑스의 리용 등 도시를 중심으로 French Bank, French Telecom 등이 참여하여 비디오텍스의 원격 요금지불 및 홈뱅킹, POS시스템의 지불수단으로 스마트카드를 사용하였다.

미국의 경우는 1980년대에 스마트카드를 프랑스로부터 도입하기 시작하여 1986년 마스터카드(Master Card)사가 컬럼비아, 메릴랜드, 플로리다 주 등의 지역에서 아메리칸 은행을 발행자로 하여 스마트카드를 시험 운영하였으나, 프랑스의 성공적인 확대보급과는 달리 스마트카드 제조사인 SCI(Smart Card International)의 재정적인 어려움, 서비스 시스템 구축 및 부대 비용의 고가로 인하여 실패하였다. 1990년대에 접어들면서 스마트카드 산업은 컴퓨터를 이용한 인터넷 사용의 급증과 정보통신 환경의 변화 등으로 빠르게 성장하고 있다. 스마트카드의 활용은 통신, 금융, 교통 그리고 최근에는 전자상거래 등 여러 분야에 걸쳐 응용되고 있는 추세다.

스마트카드를 응용한 시장에서의 가장 중요한 분야는 인터넷과 같은 통신망을 이용한 보안성이 강화된 부가가치 서비스라고 할 수 있다. 이러한 서비스를 설계하고 관리하는 도구의 개발도 중요한 분야로서 자리잡고 있다. 인터넷은 본질적으로 보안성이 결여되어 있다고 할 수 있으며 무선망 뿐만 아니라 기업망도 보안에 취약성을 가지고 있다. 이러한 측면에서 스마트카드는 통신망에서 복잡하고도 분산 트랜잭션 처리에서 양끝단(end-to-end)에서의 보안성을 제공하는 비용면이나 성능면에서 아주 각광을 받고 있다[16, 17].

자바카드는 미국 선 마이크로시스템이 제안한 스마트카드의 산업계 표준이라 할 수 있다. 자바카드 기술은 자바로 작성된 프로그램이 스마트카드나 혹은 그 밖에 제한적인 자원을 가진 장치에서의 동작을 가능하게 한다[6, 7, 8, 9]. 기존의 스마트카드의 가장 큰 특징은 한 개의 응용프로그램만을 수용한다는 것이다. 따라서 여러 개의 응용 프로그램을 적재할 때에 발생하는 프로그램 간의 데이터 보호, 비밀키 관리등에 대한 문제가 발생하지 않는다. 그러나, 다중 응용프로그램이 적재되는 스마트카드에 대한 이런 제반 문제에 대한 문제 해결책은 아직 제시되지 않고 있는 실정이다. 논문에서는 스마트카드의 산업계에서 표준이 되고 있는 자바카드를 위주로 자바카드에서의 다중 응용프로그램(multi-application) 실행 환경에서의 문제점과 여러 가지 보안 사항 관점에서 그 해결 방법을 제안한다. 2절에서는 스마트카드와 관련된 연구에 대하여 알아보고 3절과 4절에서는 다중 응용프로그램에서 필요한 여러 가지 키관리 기법과 보안 모델에 대하여 제안하고 5절에서는 결론을 맺는다.

2. 관련 연구

2절에서는 스마트카드(자바카드)의 특성과 그에 따른 보안적인 측면에서 특징을 살펴본다.

2.1 자바카드 기술

스마트카드(자바카드)[1, 2, 3, 4, 5]는 오늘날 가장 작은 플랫폼으로 인정되고 있다. 스마트카드의 메모리는 보통 1K의 RAM과 16K의 EEPROM, 그리고 24K의 ROM으로 구성되는데 최근에는 플래쉬 메모리도 사용되는 추세이다. 자바카드는 이와 같이 자원의 한계 때문에 자바카드 플랫폼을 위해 적절히 선택된 자바 언어의 부분집합이 사용된다. 자바카드는 자바 언어의 long, double, float 자료형, 다차원 배열, 스레드(thread), 쓰레기 수집(garbage collection)등의 기능을 지원하지 않으며 이 부분집합은 자바가 가지는 객체지향적 특성을 유지하면서 스마트카드와 같은 장치에서 응용프로그램이 동작할 수 있도록 구성되어 있다.

자바카드가상기계 (Java Card Virtual Machine)는 자바 기술에서의 자바가상기계(JVM)에 해당되는 부분으로서 자바 언어의 특징을 부분적으로 지원하고 이를 실행시킬 수 있는 가상 기계에 해당된다. 자바카드가상기계는 off-card에서 동작하는 부분과 on-card에서 동작하는 부분으로 나누어 질 수 있다. 즉, 실행 시점에 처리되는 부분 중에서, 클래스 로딩(class loading), 바이트 코드 검증(byte code verification), 클래스 링크(linking)과 해결(resolution) 부분은 자원에 제약을 받지 않는 off-card에서 처리된다. 자바카드 기술은 스마트카드의 메모리, 통신, 보안, 그리고 응용프로그램의 실행 모델을 지원하는 실행환경(Runtime Environment)에 대해 정의하고 있다. 보통 자바카드 플랫폼에서 동작하는 응용프로그램은 애플릿(*applet*)이라고 하는데 이는 브라우저에서 동작하는 기존의 애플릿(applet)과는 다른 것이다.

2.2 자바카드와 다중 프로그램

자바카드 애플릿은 JCRE 내에서 동작할 수 있도록

여러 가지 규칙을 따르는 Java 프로그램의 일종이다 [12, 13, 14]. 최초 카드가 제조될 때 카드내의 ROM에 애플릿이 저장되어 지는 경우도 있으나 대부분 카드가 제조된 이후에 동적으로 적재(다운로드)되어 사용되는 경우도 있다.

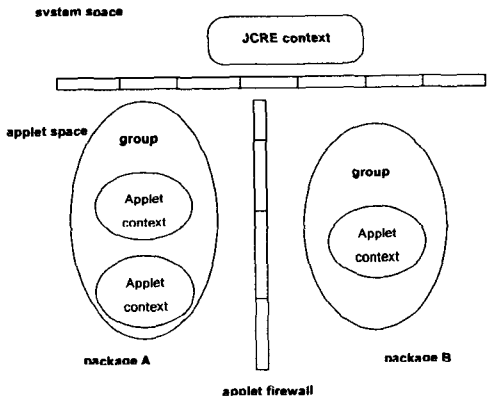
JCRE는 다중 응용프로그램(multi-application) 환경을 지원한다. 즉, 여러 개의 애플릿이 하나의 자바카드에 함께 존재할 수 있으며, 애플릿은 여러 개의 인스턴스를 가질 수 있다. 예를 들어, wallet 애플릿이 U.S. 달러와 영국 파운드에 대해 각각 생성될 수 있다. 애플릿 설치(Applet Installation)는 자바 카드가 제조될 때 적절한 카드 시스템과 JCRE가 ROM에 적재되며 ROM 애플릿은 JCRE와 함께 애플릿이 ROM에 마스크되는 것으며 카드 제조사에 의해 제공된다. ROM 애플릿은 네이티브 메소드(native method)를 선언하고 사용할 수 있으며 보안성은 JCVM에 의해 체크될 수 없는 단점을 가지고 있다.

카드가 제조된 후에 애플릿 클래스와 관련되는 클래스들이 EEPROM과 같은 불변성을 가지는 메모리에 적재될 수 있는데 이런 애플릿들은 발행전(Preissuance) 혹은 발행후(Postissuance) 애플릿으로 분류될 수 있다. 발행전 애플릿은 ROM 애플릿과 같은 방법으로 처리되며 발행후 애플릿은 네이티브 메소드를 사용할 수 없다. JCRE는 네이티브 메소드를 가지는 애플릿을 통제할 수 있는 방법이 없기 때문이다. 실제 다운로드된 애플릿이 네이티브 코드를 가지고 있다면 보안 문제가 발생 할 수 있기 때문이다

자바카드에서의 다중 응용프로그램 구동은 하나의 카드에 여러 개의 애플릿을 적재함으로써 가능하며 카드 발행 후에도 추가로 애플릿들을 다운로드시킬 수 있다. 자바카드 플랫폼은 다중 응용 프로그램 환경을 지원하기 때문에 중요한 데이터를 서로 공유하는 경우가 발생한다. 다중 응용 프로그램들 간에 서로 협업할 수 있기 위해 자바카드에서는 객체 고립화(Object Isolation) 메커니즘을 사용한다.

애플릿 고립은 애플릿 파이어월을 통해서 이루어진다. 이것은 하나의 애플릿을 정해진 공간에 고립시킴으로써 다른 애플릿에 의해 중요 데이터가 유출되는 것을 방지하며 해킹에 대한 보호를 제공한다. 서로 다른 패키지에 존재하는 애플릿은 직접적으로 객체에 대한 레퍼런스를 가질 수 없다. 그러므로, 애플릿의 오동작 혹은 악의를 가진 애플릿은 다른 애플릿이나 JCRE의 동작에 영향을 미칠 수가 없게 된다. 애플릿 파이어월은 자바카드 객체 시스템(object system)을 컨텍스트라고 불리는 분리되어 보호되는 객체 공간으로 나누게 된다. 파이어월은 서로 다른 컨텍스트 사이의 경계이다.

애플릿의 인스턴스가 생성되면 JCRE는 그 애플릿에게 컨텍스트를 할당한다. 이 컨텍스트는 필수적으로 그룹 컨텍스트에 해당된다. 하나의 패키지내에 포함되는 모든 애플릿 인스턴스들은 같은 그룹 컨텍스트를 공유한다. 동일한 그룹 컨텍스트내의 애플릿 간에는 파이어월이 존재하지 않는다. 서로 다른 그룹 컨텍스트에 속한 객체는 파이어월에 의해서 접근이 차단된다. 다음 그림은 자바카드 객체 시스템의 분할된 구조를 보여준다. 같은 컨텍스트 내의 애플릿 사이의 객체 접근은 허용되고 그렇지 않은 경우에는 파이어월에 의해 거부가 된다.



<그림 1> 자바카드 시스템에서의 객체 분할

JCRE는 특히 자신만의 JCRE 컨텍스트를 가진다. JCRE 컨텍스트는 특별한 권한을 가지는 시스템 컨텍스트이다. JCRE 컨텍스트에서는 어떠한 애플릿의 컨텍스트에 대한 접근도 허용된다. 그러나, 애플릿에서 JCRE 컨텍스트에 대한 접근은 파이어월에 의해 거부된다.

자바카드 플랫폼에서 기본적으로 제공하는 보안 기능은 다음과 같다. 응용프로그램의 작성 후 컴파일할 때는 기존의 자바 컴파일러가 사용되며 클래스 파일을 생성하면서 보안 관련 부분이 검사된다. 생성된 클래스 파일은 자바의 부분 집합인 자바카드 클래스이어야 하기 때문에 자바카드에서 지원되는 범위를 벗

어나는지에 대한 체크가 필요하다. 이어서 자바카드 애플릿 포맷(CAP 파일과 엑스포트 파일)에 부합한지를 검사하는 과정과 애플릿 파이어월을 통한 애플릿 고립화(isolation)가 정당한지를 검사하게 된다.

2.3 스마트카드의 보안 기능

스마트카드 출현의 배경은 카드가 가지는 보안 제공 능력 때문이라고 할 수 있다. 스마트카드가 보유한 보안 특성은 크게 다음 3가지 형태로 구분할 수 있다 [10, 11].

2.3.1 보안 메커니즘 보안블록과 개인식별번호

스마트카드는 특정 메모리 부분에 대한 접근을 차별화할 수 있도록 하기 위해 스마트카드 제조자, 카드 발행자, 서비스 제공업자, 카드 소지자가 서로 독립된 비밀키를 가질 수 있도록 보안 블록(security block) 기능을 제공한다. 보안 블록은 스마트카드의 메모리 내에 위치하며 카드운영체제(Card Operating System, COS)에 의해서만 접근이 가능하며 각 디렉토리에 하나의 보안 블록을 두어 디렉토리내의 파일 접근 제어를 하고 있다.

2.3.2 파일 접근 제어

메모리에 저장된 각 파일들은 고유의 식별자와 파일 형식, 보안변수, 파일에 대한 접근 조건에 대한 내용을 포함한 각종 정보를 가지고 있다. 각각의 파일들은 저장하고 있는 정보의 보안정도에 따라 접근 조건이 차별화 되며, 보안변수는 파일이 요구하는 전자서명의 사용여부, 검증을 위한 보안모드 값 등을 갖는다.

2.3.3 보안 메커니즘

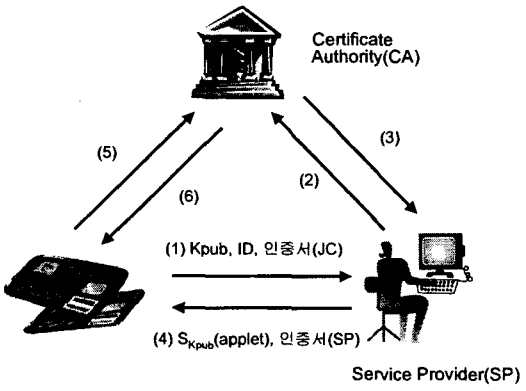
기밀성(Confidentiality), 인증(authentication), 무결성(integrity), 부인봉쇄(non-repudiation)등의 기능을 제공하고 있다.

3. 스마트카드에서의 키 관리 기법

본 절에서는 자바카드에 적용될 수 있는 키의 관리, 응용프로그램의 다운로드 기법, 공유 기법 및 보안 모델에 대하여 제안한다.

3.1 키의 관리 및 응용프로그램 다운로드

자바카드에서 사용될 키의 관리는 다음 절차에 의해 행한다. 먼저 카드사용자가 자바카드(Java Card : JC)를 인증기관(Certificate Authority : CA)에 넘겨주면 CA는 자바카드의 ID, 유효 기간 등의 정보와 새롭게 생성된 자바카드의 공용키를 사용하여 인증서를 생성한 다음 자바카드내에 저장한다. 이 공용키는 차후 시그니쳐 생성등의 용도로 사용이 된다. 자바카드에 다운로드되는 모든 응용 프로그램은 CA로부터 인증을 받아야 한다. 자바카드내에는 자바카드 가상기계(JavaCard Virtual Machine)와 같은 기본적인 기능을 내장하고 있지만 하드웨어 면에서 여러 가지 제한적 요소로 인하여 바이러스, 카드의 오동작등에 대하여 안정성을 보장받을 수 있는 기능은 없다. 이러한 이유로 인해서 서비스 제공자(Service Provider : SP)가 판매하는 모든 응용 프로그램은 인증기관 혹은 대리 인증기관(CA)으로부터 인증을 받아야만 한다

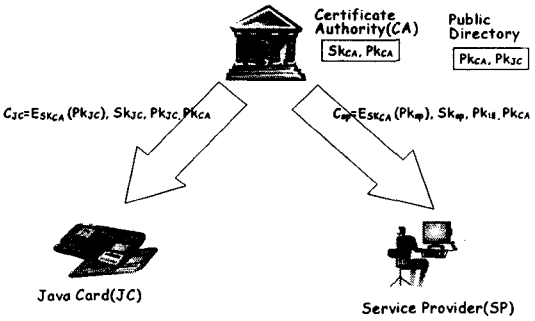


<그림 2> 자바카드에서의 다운로드 체계

(즉, CA_{SP}로 표기). 응용 프로그램을 안전하게 다운로드받는 절차는 그림 2에 있으며 그 과정은 다음과 같다. 자바카드 쪽에서는 SP에게 공용키, ID 및 인증서를 제공하여 응용프로그램(애플릿)을 요구한다. 이는 그림 2의 (1)에 해당한다. SP(Service Provider : 서비스 공급자)는 해당 자바카드가 실제 존재하는 카드이며 카드에 내장된 공용키가 인증된 것임을 확인함으로써 공인된 사용자만 사용할 수 있다는 것을 인지한다. SP는 단말기 등을 통하여 공용키, ID, 인증서등과 함께 응용 프로그램 인증을 CA에게 요청한다. 이 단계를 위하여 SP는 CA에게 자바카드의 인증을 위해 CA_{JC}(즉, 자바카드 *JC*의 인증기관 *CA*)의 공용키를

사용한다. 이는 그림 2의 (2)에 해당한다. 이 작업은 안전 채널인 경우 CA_{SP}(즉, 서비스공급자 *SP*의 인증기관 *CA*) 공용키를 사용하며 비안전 채널인 경우 시그니쳐를 사용할 수 있다. 다음 단계는 그림 2의 (3)에서와 같이 CA_{SP}는 SP에게 CA_{JC}의 공용키를 보낸다. 그림 2의 (4)의 단계에서 JC는 자바카드의 공용키로서 암호화된 응용 프로그램(애플릿)과 애플릿의 시그니쳐에 해당되는 인증서를 수신한다. 다음 단계는 그림 2의 (5)에서와 같이 JC는 다운로드된 응용 프로그램의 시그니쳐를 검증하기 위해 CA의 CA_{JC}와 접촉한다. 다음 단계는 그림 2의(6)에서와 같이 단계 (3)의 비안전 채널 경우와 동일하다.

3.2 공용키를 사용한 자바카드용 응용프로그램의 다운로드



<그림 3> 초기화 과정

자바카드에 공용키를 사용하여 응용프로그램(즉, 자바카드용 애플릿)을 적재하기 위한 과정은 초기화(개인화), 키의 분배 및 실제 애플릿을 적재하는 과정으로 이루어져 있다.

3.2.1 초기화(Initialization)

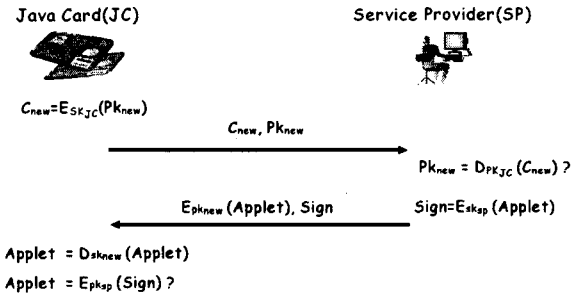
CA는 자바 카드의 초기 인증 생성을 위해 Sk_{CA}(즉, CA의 Secret Key 비밀키)와 Sk_{CA}(즉, CA의 Public Key 공용키)를 생성 및 관리하고 있다. 카드를 제조하거나 그 이후에는 CA는 카드상에 인증을 득한 마스터 키(Sk_{JC} 와 Sk_{JC}의 쌍)와 인증서 C_{JC}를 저장한다. 마찬가지로 SP에게도 같은 방법으로 인증을 득한 마스터 키(Sk_{SP} 와 Sk_{SP}의 쌍)와 인증서 C_{SP}를 저장한다. 초기화 과정이 종료되면 CA의 공용디렉토리에는 CA와 JC의 공용키가 보관되고 SP와 JC에는 각각 공용키와 비밀키가 보관이 된다.

3.2.2 키의 분배(Key Distribution)

자바 카드가 새로운 애플릿을 위해서 새로운 키 Sk_{new}와 Pk_{new}를 생성하면 자바 카드는 SP에게 Sk_{JC}를 사용하여 공용키의 시그니쳐에 해당되는 인증서 C_{new}와 Pk_{new}를 전송한다. 이 과정은 그림 4의 오른쪽 방향 화살표에 나타나 있다. SP는 C_{new}를 자바카드의 공용키를 사용하여 해독한 뒤 Pk_{new}를 검증한다.

3.2.3 응용 프로그램(애플릿) 다운로드(Applet Download)

서비스공급자가 자바카드에 필요한 응용프로그램(애플릿)을 다운로드하는 과정은 그림 4의 왼쪽 화살방향에 해당된다. 먼저 자바카드쪽에서는 새로운 공용키 C_{new}를 비밀키로서 암호화시켜 인증서를 생성한다. 이 인증서와 자바카드의 새 공용키(Pk_{new})를 SP에게 전달한다. SP에서는 수신한 Cnew가 정당한지 자바카드의 공용키를 사용, 해독하여 수신된 공용키와 동일한지 검사한다. 이 검사를 통과하면 애플릿을 SP의 비밀키로서 시그니쳐를 생성하여 이 시그니쳐와 암호화된 애플릿과 함께 자바카드쪽으로 송신한다. 자바카드쪽에서는 이 수신된 시그니쳐와 해독한 애플릿이 동일한지를 검사한다.



<그림 4> 자바카드에서의 응용프로그램 다운로드 체계

4. 스마트카드에서의 보안 모델

OS 수준에서 지켜져야 할 보안 정책은 여러 응용프로그램들 사이에서 각종 정보들의 공유 혹은 흐름을 통제할 수 있어야 한다. 본 절에서는 스마트카드 환경에서 제안 모델에 필요한 신뢰 관계와 정보 공유 기법에 대하여 제안한다.

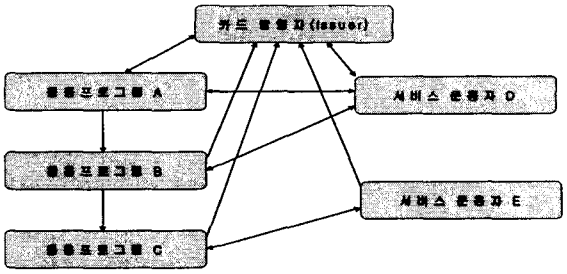
4.1 신뢰 관계(Trust Relationship)

가장 간단한 형태의 신뢰 관계는 카드 발행자가 모든 응용 프로그램(혹은 객체)를 신뢰할 수 있는 상태이나 일반적으로는 이러한 관계는 존재하지 않는다고 할 수 있다. 카드 발행자는 운영체제를 마음대로 통제할 수 있으며 운영체제 수준에서는 모든 응용프로그램과 관련된 비밀키를 읽고 수정하고 생성하는등의 모든 작업을 할 수가 있다.

자바카드에서 다중응용프로그램을 적용한 경우는 전자 지갑의 예를 사용하여 설명할 수 있다. 이 전자지갑에는 2가지의 응용 프로그램, 즉, 항공사에서 서비스하는 A라는 마일리지프로그램(애플릿 A)과 렌트카업체에서 자동차 대여등을 서비스해 주는 B라는 프로그램(애플릿 B)을 적재하고 있다. 전자지갑 자체(애플릿 C)도 하나의 애플릿으로 생각할 수 있기 때문에 이 자바카드에는 3가지의 애플릿이 적재되어 있다. 전자 지갑, 즉 애플릿 C는 자바카드 소유자의 모든 입출금 내역을 관리하고 있으며 항공사에서 티켓을 구매할 때 전자 지갑에서 해당 요금이 출금되며 항공사는 A 애플릿을 적용시켜 마일리지를 제공하게 된다. 렌터카 애플릿 B에 대해서도 같은 방법으로 렌터카를 대여할 때 전자지갑, 즉 애플릿 C에서 출금이 되며 애플릿 B에서는 마일리지가 추가되도록 자바카드를 작동시킨다. 항공사와 렌터카업체가 서로 제휴를 맺고 있을 때는 이 2가지 마일리지를 합산하여 애플릿 A를 통하여 항공사에서 비행기 표를 무료로 제공받거나 애플릿 B를 적용시켜 무료로 렌터카를 대여할 수 있다.

그러나, 이러한 경우는 항공사와 렌터카 업체가 우호적인 관계를 고려할 때 가능한 구상이며 만약 적대적인 관계에 있다면 쌍방간의 마일리지를 서로 공유할 수가 없게 된다. 즉, 상호간에 마일리지에 대한 정보, 즉 마일리지 객체를 공유할 수 없도록 자바카드가 관리를 하여야 한다. 이러한 문제를 해결하기 위해 객체들 간에 신뢰 관계를 고려해야만 한다.

신뢰 관계는 대칭적(symmetric)이거나 이행적(transitive)인 관계는 가지고 있지 않다. 즉, A라는 객체가 B라는 객체에 대해 신뢰를 하고 있고 B는 C라는 객체를 신뢰한다고 하더라도 A 객체는 C 객체를 신뢰하지는 않는다. 이러한 문제는 응용 프로그램(즉, 서비스)을 제공하는 업체들간의 이해 관계가 더 복잡하게 얽혀있기 때문에 이러한 대칭적 혹은 이행적인 관계가 쉽게 바뀌어지지 않는다. 따라서, 스마트카드 서비스 업계에서는 이러한 다중 응용프로그램에서의 정보 공유에 관한 문제점을 해결할 수 있어야 한다.



<그림 5> 신뢰 관계의 예

그림 4는 3가지의 응용 프로그램과 2개의 사업자들간의 신뢰 관계를 표현한 예이다. 이 예에서는 모든 응용프로그램과 서비스 운용자들은 카드 발행자를 신뢰하고 있고 서비스 운용자 D는 응용프로그램 A와 B를, 서비스 운용자 E는 응용프로그램 C를 신뢰하고 있다. 또한 응용 프로그램 A는 B를, B는 C를 신뢰하고 있지만 응용 프로그램 A는 응용프로그램 C를 신뢰하지는 않는다. 그러므로, 응용프로그램 C가 가지는 여러 정보는 응용 프로그램 A에서 접근이 불가능하다.

본 논문에서는 다중 응용프로그램에서 객체들간의 정보 공유 가능을 표현할 수 있는 그리드(grid) 기법을 적용하였다. 이 기법은 표 형식으로 가로방향, 세로방향 각각에 응용프로그램과 서비스 운용자등의 필요한 객체들을 나열하고 세로 방향의 객체에서 가로방향의 객체가 공유가 가능하면 ○ 표시를 하고 그렇지 않으면 ×표시를 한다. ×표시가 된 경우 객체 접근을 시도할 때 자바카드가상기계(JCVM)에서는 파이어월이 적용되어 공유 에러를 발생시킴으로써 객체

<표 1> 공유 관계의 예

	A	B	C	D	E
응용프로그램A	○	○	×	○	×
응용프로그램B	×	○	○	○	×
응용프로그램C	×	×	○	×	○
서비스운용자D	○	○	×	○	×
서비스운용자E	×	×	○	×	○

공유를 금지시키게 된다. 이 기법은 기존의 격자(lattice)를 사용한 방법[7]보다 공유 관계의 표현과 구현면에서 훨씬 간단하다는 것을 알 수 있다. 표 1은 그림 4의 신뢰 관계를 바탕으로 작성된 공유 관계를 나타낸 것이다.

5. 결 론

본 논문에서는 자바 카드를 중심으로 다중응용프로그램 스마트카드의 작동 원리를 고찰하고 다중 응용 프로그램의 구현 및 적재될 스마트카드에서의 키관리 기법과 보안 및 공유 모델에 대하여 제안하였다. 또한, 기존 격자를 사용한 공유 관계의 표현을 그리드 기법을 사용함으로써 구현이 쉽게 할 수 있도록 하였다.
스마트 카드는 최근 여러 가지 장점들로 인해서 많은 관심을 끌고 있으며 일반 사용자들은 지갑에 가지고 다니는 카드의 수를 줄일 수 있고 카드 발행자는 카드의 발행 이 후 새로운 응용 프로그램을 추가하거나 기존의 응용 프로그램을 손쉽게 업그레이드시킬 수 있으며 개발자들은 개발 기간 단축 가능성등으로 인해 다중 응용 프로그램을 지원하는 스마트카드의 사용이 활성화되는 추세이다. 또한, 다중응용프로그램 스마트카드 서비스는 사업 파트너 상호간의 비즈니스 상승효과 측면에서 볼 때도 아주 많은 장점을 가지고 있는 실정이다.

본 논문에서 제안한 키관리 기법과 보안 모델을 토대로 하여 스마트카드용 응용프로그램 개발 기법과 다중 응용프로그램에서의 실제 구현 기법(예, 객체지향 기법 적용)[15, 18] 및 제안 기법의 효과를 실제 적용시켜 운용할 때의 문제점에 대하여 더 연구가 필요하다.

참 고 문 헌

[1] Zhiqun Chen. How to write a Java Card applet : A Developer's Guide. http://www.javaworld.com/javaworld/jw-07-1999/jw-07-javacard_p.html

[2] Sun Microsystems. Java Card 2.2 Platform APISpecifications. http://java.sun.com/products/javacard/htmldoc/index.html

[3] Sun Microsystems. Java Card 2.2 Runtime Environment(JCRE) Specifications. http://java.sun.com/products/javacard/JCRESpec.pdf

[4] Sun Microsystems. Java Card 2.2 Virtual Machine Specification. http://java.sun.com/products/javacard/htmldoc/javacard21.html

[5] Sun Microsystems. Java Card Applet Developer's Guide. http://java.sun.com/products/javacard/AppletDevelopersGuide.html

[6] Michael Montgomery and K. Krishna. "Secure Object Sharing in Java Card," In Proceedings of the USENIX Workshop on Smartcard Technology, pages 119-128, 1999

[7] P. Girard. "Which security policy for multi-application smart cards," In Proceedings of the USENIX Workshop on Smartcard Technology, pages 21-28, 1999

[8] D. Caromel, L. Henrio and B. Serpette. "Context inference for Static Analysis of Java Card Object Sharing," In Int'l Conference on Research in Smart Cards E-Smart 2001, France, 2001

[9] P. Bieber, J. Cazin, J. Lanet, V. Wiels, and G. Zanon. "Checking secure interactions of smart card applets," In ESORICS, 2000

[10] Zhiqun Chen. "Java Card Technology for Smart Cards: Architecture and Programmer's Guide," Addison-Wesley, 2000

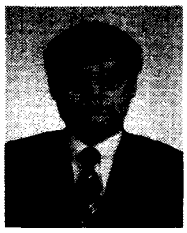
[11] P. Muller and A. Hefter. "Universe : A type system for alias and depending control," Technical Report 279, Fern Universitat Hage, 2001

[12] M. Eluard, T. Jensen and E. Denney. "An Operational Semantics of the Java Card Firewall," Lecture Notes in Computer Science, Vol. 2140, 2001

[13] P. Bieber, J. Cazin, A. Marouani, P. Girard, J. Lanet, V. Wiels and G. Zanon. "The PACAP Prototype: A Tool for Detecting Java Card Illegal Flow," Lecture Notes in Computer Science, Vol. 2041, 2001

[14] Gilles Grimaud, Jean-Louis Lanet, Jean-Jacques Vandewalle, "FACADE: A Typed Intermediate

Language Dedicated to Smart Cards," In Proceedings ESEC/FSE '99, pages 476-493, 1999
[15] I. Attali, D. Caromel, C. Courbis, L. Henrio and H. Nilsson. "Smart tools for Java Cards," Smart Card Research and Advanced Application Conf. (CARDIS). Kluwer Academic Publishers, September 2003.
[16] Anup K. Ghosh. "Security Risks of Java Cards," CardTech/SecurTech '98, Volume I: Technology. pages 465-470, 1998.
[17] O. Fodor and V. Hassler. "JavaCard and Opencard Framework: A Tutorial," Technical University of Vienna, 2003.
[18] 김시관, 여동규, 이주완, "자바카드 기반 다중서비스에 있어서 데이터 및 공유기법 연구에 관한 연구," 연구보고서, 한국전자통신연구원, 2002.



김 시 관 (Si-Gwan Kim)

1982년 경북대학교 전자공학과 (학사)
1984년 한국과학기술원 전산학과 (석사)
1984년~ 1995년 삼성전자, LG정보통신
2000년 한국과학기술원 전산학과(박사)
2002년 ~ 현재 금오공과대학교 컴퓨터공학부 교수
(관심분야 : 컴퓨터구조, 병렬처리, 이동컴퓨팅등)

임 은 기 (Eun-Ki Lim)

1977년 서울대학교 수학과(학사)
1988년 한국과학기술원 전산학과 (석사)
1993년 한국과학기술원 전산학과 (박사 수료)
1998년 ~ 현재 금오공과대학교 컴퓨터공학부 교수
(관심분야 : 데이터베이스 설계, 소프트웨어 개발 프로세스, 컴퓨터 교육)