IMT-2000 단말기용 USIM상에서의 오프라인 지불 솔루션 탑재에 관한 연구

백장미[†] · 하남수[††] · 홍인식[†††]

요 약

모바일 기기의 사용이 증가하면서 E-Commerce에서 M-Commerce로 변화하고 있다. 특히 IMT-2000 (International Mobile Telecommunication 2000) 서비스가 준비중에 있으며, USIM(Universal Subscriber Identity Module) 기술을 내장함으로써 서비스의 폭이 넓어질 전망이다. 그러나 아직 USIM을 내장하는 서비스는 미흡한 단계로써 기본적인 애플리케이션의 개발조차 미비한 상태이다. 따라서 본 논문에서는 USIM의 전반적인 구조와 프로토콜의 분석을 통하여, 모바일 기기와 USIM 상에서 안전하게 지불할 수 있는 지불 솔루션을 제안한다. 특히 모바일 기기의 유동성과 USIM의 안전성 및 계산능력, 저장능력을 이용하여 오프라인 상에서 안전하게 거래할 수 있는 시스템을 제안한다. 또한 제안한 시스템의 유효성을 입증하기 위하여 Java Card 상에서의 지불 시스템을 개발하여 시뮬레이션 한다.

Implementation of Offline Payment Solution using USIM in IMT-2000

Jang-Mi Baek[†], Nam-Su Ha[††] and In-Sik Hong[†††]

ABSTRACT

As mobile device is becoming more popular, E-Commerce changes into M-Commerce. Especially, IMT- 2000 (International Mobile Telecommunication 2000) service is prepared for M-Commerce and this has USIM (Universal Subscriber Identity Module) as a core of certification of individuality and transactions. As a result, the area of mobile service going to expand by USIM. But, mobile services using USIM leave much to be desired, and developed application don't variety. In this paper, for the efficient design of USIM, the structure of USIM and protocol is analyzed, and secure payment solution in USIM is proposed. Specially, offline payment system is proposed for the verification of proposed protocols including security, saving, and calculation of balance. Finally, the simulation of proposed payment system on USIM is performed using Java Card.

Key words: USIM, IMT-2000 서비스, M-Commerce, 오프라인 지불 솔루션

1. 서 론

모바일 기기의 사용이 급증하면서, 전자 상거래의 물결은 급속도로 모바일 상거래(M-Commerce)로 바뀌고 있다. 특히, 제 3세대 이동 통신 서비스인 IMT-2000 서비스가 준비중에 있으며, IMT-2000 서비스의 핵심으로 USIM을 내장함으로써 모바일 상거래의 중심은 휴대폰 단말기 안에 내장되는 USIM이 될 것으로 전망한다. USIM을 기존의 GSM(Global System for Mobile communications System)방식에서 제공하는 SIM(Subscriber Identification Module)보다 발전된 형식으로써 기본적으로 사용자 인증 기능을 제

본 논문은 2001년도 순천향대학교 교수연구년제에 의하여 연구되었음.
본 논문은 정보통신부의 지원을 받아 연구되었음.
접수일 : 2002년 3월 25일, 완료일 : 2003년 2월 3일
[†] 준회원, 순천향대학교 대학원 전산학과 박사과정
[††] 준회원, 비자캐쉬코리아 연구원
[†††] 정회원, 순천향대학교 공과대학 정보기술학부 부교수

공하며, 메모리와 CPU(Central Processing Unit)의 향상으로 인하여 다양한 애플리케이션의 탑재가 가능하다. 특히 USIM을 위한 COS(Chip Operating System)가 개발되면서 USIM상에서의 애플리케이션의 개발이 용이해졌다. 그러나 아직 USIM을 위한 애플리케이션의 개발은 초기 단계로서, 실질적인 다양한 애플리케이션의 개발이 필요하다. 따라서 본 논문에서는 USIM을 위한 COS로써 Java Card를 채택하고, Java Card상에서 수행할 수 있는 전자 지불 솔루션을 제안하였다. 본 논문의 2장에서는 관련 기술로써 USIM의 구조와 단말기 사이의 프로토콜을 설명하고, 3장은 모바일 기기와 USIM 상에서 안전하게 지불할 수 있는 오프라인 지불 프로토콜을 제안하였다. 본 시스템의 유효성을 입증하기 위하여 기존의 모바일 기기와 신용 카드 기반의 지불 솔루션과의 차이점을 비교 분석하였으며, 시스템의 일부를 시뮬레이션 하였다. 시뮬레이션을 위하여 ISO(International Standardization Organization) 7816 스펙과 3GPP(3rd Generation Partnership Project) 문서 등을 참조하였으며, 지불 솔루션에 사용되는 전자 지갑 애플릿을 개발하였다. 개발된 애플릿은 Java Card에 탑재되며 Gemplus사에서 제공하는 GemXpresso RAD 211 툴킷과 JBuilder를 통하여 시뮬레이션하였다[1-3].

2. USIM

본 장에서는 USIM과 단말기 사이의 전송을 위한 프로토콜 및 애플리케이션의 구조를 설명한다.

2.1 전송 프로토콜(APDU)과 전송계층

USIM은 각 터미널, 즉 모바일 단말기와의 데이터 전송을 위하여 비동기식 반이중 통신 프로토콜을 사용한다. 그림 1에서 보는 바와 같이, USIM과 터미널 사이의 전송계층은 4계층으로 이루어져 있다. 물리적 계층은 비트 정보데이터를 기계적, 기능적, 절차적 특성을 갖는 물리적 매체를 통해 전달하는 역할을 하며, 데이터 링크 계층은 전송단위순서를 제어하거나 오류검출, 오류복구, 흐름제어 기능 등을 담당한다. 전송 계층은 에러 검출과 검정, 흐름제어, 연결분리, 분류합류, 다중화 및 역 다중화 기능을 담당하며, 애플리케이션 계층을 통해 각각의 애플리케이션과의 명령어 교환이 이루어진다[4,5].

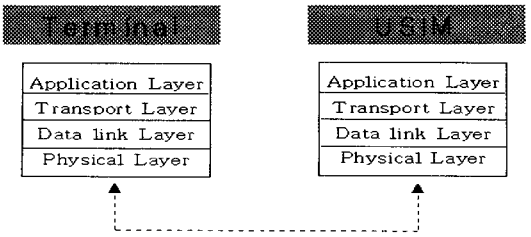

그림 1. 전송계층

2.2 애플리케이션의 구조

USIM상에서 애플리케이션의 구조는 그림 2와 같은 구조로 형성된다. MF(Master File)는 파일 시스템의 루트에 해당하고, DF(Dedicated File)는 디렉토리 파일이며 EF(Elementary File)는 데이터 파일에 해당한다. 모든 애플리케이션은 EF DIR에 포함되어 있는 각각의 애플리케이션 식별자에 의해 구별된다. 이 애플리케이션 식별자는 애플리케이션을 선택하기 위해 사용된다. EF DIR, EF PL, EF IC CID는 MF 아래의 디렉토리로서 MF의 명령과 지시를 받는다. DF TELECOM 은 MF 아래 거주하며 각각의 독립적인 정보를 지닌 애플리케이션을 포함한다. 파일에 대한 조작이 이루어지기 전에 먼저 파일을 선택하여야 한다. 파일에 대한 조작은 접근 조건에 의해 통제되고, 보안을 위한 파일 접근 레벨도 존재한다.

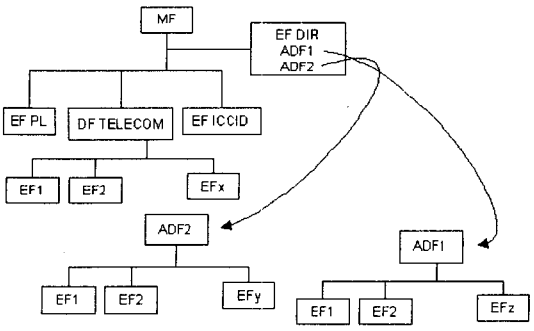

그림 2. USIM에서의 애플리케이션의 구조

3. 오프라인 지불 시스템

본 논문에서는 IMT-2000 단말기상에서 USIM을 이용하여 개발할 수 있는 애플리케이션들 중에서 가장 기본이 되는 지불 솔루션을 제안한다. 특히 모바일 기기의 유동성과 USIM의 안전성 및 보안성을 바탕으로 하여 기존의 시스템보다 안전한 오프라인 지

불 시스템을 구현한다. 또한, 기존의 모바일 단말기 시스템 및 신용카드 기반의 시스템과의 차이점을 분석하여 본 제안한 시스템의 유효성을 입증하고자 한다. USIM상에서 애플리케이션 개발을 위하여 각각의 프로토콜과 인터페이스를 정의하고 오프라인 지불 애플리케이션을 개발하여 시뮬레이션 한다[6-11].

3.1 오프라인 지불 시스템의 전체 구조도

오프라인 지불은 지불 서버나 Wallet 서버 등의 개입 없이 오프라인 상에서 지불할 수 있는 것을 의미한다. 본 연구에서 오프라인 지불 구조가 가능한 것은 모바일 단말기와 USIM을 사용함으로써 USIM상에서 지불 서버의 필요 없이 지불에 관련된 정보를 저장하고 관리할 수 있기 때문이다. 특히 블루투스나 적외선 통신과 같은 무선 네트워크 망의 개발이 활발히 이루어지고 있기 때문에 오프라인 상에서의 USIM을 통한 지불 시스템의 연구는 필수적인 요소이다. USIM은 개인 인증을 위한 보안적 요소와 안전성을 확보하고 있으며, 거래를 통한 데이터를 저장하기 충분한 메모리를 지니고 있으므로 오프라인 상에서 편리하고 안전한 거래를 할 수 있다. 그림 3은 오프라인 지불 시스템의 전체 구조도이다.

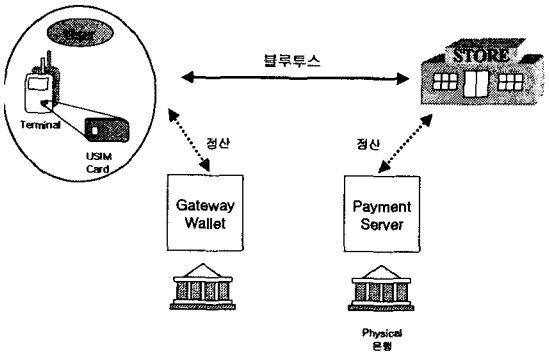

그림 3. 오프라인 지불 시스템의 전체 구조도

3.2 오프라인 지불 프로토콜

오프라인 지불 프로토콜은 그림 4와 같다. 인증서의 교환 및 저장후 상품의 리스트를 전송하고, 구매요청과정이 일어나면 지불 요구가 발생한다. 사용자는 지불요구에 대한 지불 정보를 전송하고 지불 영수증을 받게된다. 각각의 자세한 과정은 그림 4의 하단에 설명하였다.

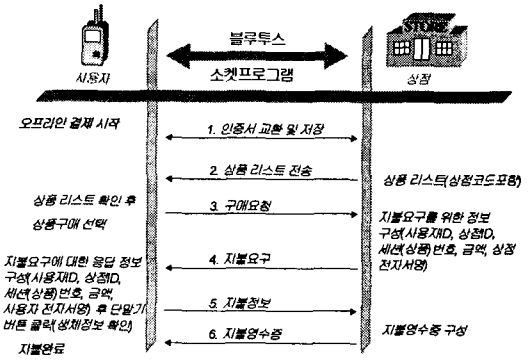

그림 4. 오프라인 지불 프로토콜

Step 1: 물품 구매 요청

사용자가 물품 구매를 요청한다. 사용자는 자신들이 관심 있는 물건이나 서비스를 선택하고 상점에게 견적을 요구한다. 블루투스 등의 무선 통신망을 사용하여 데이터의 견적을 요구하며, 고객은 전화번호 등의 특정 ID를 상점에 전송한다. 상점에서는 이 정보를 이용하여 고객의 전화기를 식별한다.

Step 2: 상호 인증

상점과 고객은 서로의 인증과정이 필요하다. 인증과정에서 필요한 데이터는 USIM에 저장되어 있기 때문에 오프라인 상에서의 거래를 위한 인증과정이 가능하다. USIM의 인증서는 WPKI(Wireless Public Key Infrastructure) 기반으로 USIM은 인증을 위해 필요한 인증서 및 비밀키 등의 데이터를 저장한다.

Step 3: 물품 견적서 전송

상점은 사용자가 선택한 물품의 리스트와 총 금액 등을 포함하는 견적서를 만들어서 고객에게 전송한다. 견적서는 표 1과 같은 항목으로 이루어진다. 견적서에 포함되는 내용은 응용 프로그램 레벨에서 암호화되어 전송된다.

표 1. 물품 견적서의 데이터

물품 견적서에 필요한 데이터
- 고객의 단말기에 대한 정보
- 상점의 정보
- 전체 금액
- 통화의 종류
- 상점의 국가 코드와 ZIP 코드
- 거래일시
- 견적서의 유효기간
- 트랜잭션 ID

Step 4: 전자 화폐 전송

지불을 위한 데이터는 표 2와 같다. 고객은 상점이 보낸 지불 견적서를 검토한 다음, 맞으면 서명 PIN을 입력하여 전자 지갑에 내장된 전자 화폐를 상점에 전송한다.

표 2. 지불을 위한 데이터

지불을 위한 데이터
- 지불 ID - 지불 금액 - 통화의 종류 - 상점의 정보(상점의 ID,...) - PAN(Primary Account Number)

Step 5: 확인 정보 전송

영수증에 필요한 데이터는 표 3과 같다. 영수증에 필요한 상점은 전송된 전자 화폐를 검증한다. 검증이 확인되면 구매자는 상점으로부터 물품을 받고, 상점을 결제의 완료와 구매에 대한 영수증을 전송한다.

표 3. 영수증에 필요한 데이터

영수증에 필요한 데이터
- 트랜잭션 ID - 상점의 ID - 물품의 스펙 - 전체 금액 - 거래 일시

3.3 오프라인 지불을 위한 지불 트랜잭션

오프라인 지불을 위한 지불 트랜잭션을 위한 Key 설정과 Key에 대한 설명은 표 4와 같다. 각각 정의 내린 Key는 인증을 위하여 필요한 Key이다.

오프라인 지불 트랜잭션 상에서의 Secret Key에 대한 정의와 설명은 표 5와 같다.

표 4와 표 5에서 정의한 Key값을 이용하여, USIM과 터미널의 상호인증, 지불, 잔액확인을 위한 프로토콜은 생성한다. 표 6은 USIM과 터미널의 상호인증을 위한 프로토콜이다. 터미널에 의해 USIM의 인증 확인 후 구매 프로세스 시작한다. CSN는 MKauth로부터 Kauth를 얻기 위해 터미널에 의해 읽혀지고 사용된다. 터미널은 USIM에 의해 암호화되기 위한 랜덤 수를 생성하고, USIM은 Kauth로부터 계산되는 임시 키를 사용하여 랜덤수를 암호화한다. USIM에 의해 계산된 암호문은 터미널에 리턴 되며, 인증이 실패되면 프로세스는 종료되고 USIM은 리젝트된다.

표 7은 현재 지갑에 남아 있는 잔액을 확인하기 위한 프로토콜이다. USIM은 자식 키 Kbal 키를 사용하여 CBC(Card Balance Certificate) 생성한다. 생성된 CBC는 터미널에 의해 검증된다.

지갑에서 지불을 위한 트랜잭션은 표 8, 표 9와 같다. 지갑 잔액 지불이 수행되면, CDC(Card Debit Certificate)는 Kdt를 사용하여 인증과정을 거친다. 인증서는 터미널에 의해 체크되며 검증은 Kdt가

표 4. 오프라인 지불 트랜잭션을 위한 Key

Key name	위치	설명
K_{auth}	USIM	터미널에 의한 USIM 인증키
K_{bal}	USIM	잔액 인증서를 계산하기 위하여 사용 되는 키
K_{dt}	USIM	지불인증서를 계산하기 위해 사용 되는 키 인증서는 터미널에 의해 로컬적으로 검증
$K_{sign-dt}$	USIM	지불서명을 계산하기위해 USIM에 의해 사용 되는 키 인증서는 트랜잭션 콜렉션 후에 발행자에 의해 검증
K_{cr}	USIM	충전 프로세스 동안 충전 인증서 계산을 위해 사용되는 키 인증서는 USIM에 의해 해석
K_{ccr}	USIM	USIM에 충전 인증서 전송을 암호화하기 위해 사용 되는 키 USIM은 충전 프로세스 동안 로딩 디바이스로부터 충전 암호문을 복호화
K_{mac}	터미널	각각의 트랜잭션 및 트랜잭션 배치에 대한 MAC값 계산을 위해 검증
K_{adm}	USIM	보안 메시징 메커니즘을 위한 개인화 단계에서 사용

표 5. 오프라인 지불 트랜잭션을 위한 Secret Key

Secret Code	위치	설명
MSC_{auth}	터미널	Secret Codes(SC_{auth})를 deversify하기 위해 사용
PIN	User(사용자)	PIN코드는 카드 소지자에 의해 카드에 표시되는 Secret code 검증
Secret Code	USIM	터미널에 의해 카드에 표시되는 Secret code를 검증

표 6. USIM과 터미널의 상호인증을 위한 프로토콜

스텝 1 USIM과 터미널의 상호인증		
USIM(3DES)	Termianl 애플리케이션	터미널
리턴 ATR CSN	Power On Read CSN Get Resp	
Function SelPk CTC1=CTCinit +1 TK1=3DES(CTC1, K_{auth}) Cryto = 3DES(RND, TK1) Crypto, CTC1	**인증** get RND 인증요구 SelPk(K_{auth} indetx, Pf, RND) Get Resp Cryptogram verification Verify Crypto (MK_{auth} index, CSN, CTC1, Crypto) ESC = Reject Card	RND TTC = TTC +1 Key diversification K'auth = 3DES(CSN, MK_{auth}) TK1' = 3DES(CTC1, K'_{auth}) Crypto' = 3DES(RND, TK1') Checks Crypto = Crypto' If yes returns ACK, If not returns ESC
Check Secret Code Function ChkCod SC' = 3DES(,SC'>, TK1) 비교 SC = SC' Returns 90 00 if ok, If not returns error	**터미널 인증** Get Encrypted secret code (MSC index, CSN) or (PIN, CSN) Sends ciphered Secret code ChkCod(Code index, <SC'>)	Secret Code Diversification Previous TK2 needed (SC' = 3DES(CSN, MSC) <SC'> = DES $^{-1}$(SC', TK1) or <SC'> = DES $^{-1}$(PIN, TK1) or <SC'>

MKdt와 CSN로 부터 diversify 된 후에 행해진다. 터미널은 새로운 잔액을 표시한다.

지불시 사용자의 서명을 위한 트랜잭션은 표 9와 같다. Teminal에서 생성되는 false 트랜잭션을 발견하기 위하여, 각 트랜잭션은 USIM 자체에 의해 서명되어야 한다. 서명은 트랜잭션 컴포넌트와 Ksign-dt로 부터 고객의 USIM에 의해 계산되며, 새로운 임시키는 각각의 구매 프로세스 단계 전에 설립되어야만 한다.

3.4 USIM에 내장되는 데이터

USIM상에서 생성되는 모든 데이터는 USIM내의 EEPROM에 저장된다. 현재 시뮬레이션을 위하여 사용된 Java Card의 메모리는 32K이며, 16K는 Java Card Virtual Machine에 할당되고 나머지 부분에 실질적인 데이터가 저장된다. 개인 인증 정보와 오프라인 거래를 위하여 저장되는 USIM의 필수 데이터는 표 10과 같다.

표 7. 잔액 확인 트랜잭션 프로토콜

스텝 2 잔액 확인		
USIM(3DES)	Termianl 애플리케이션	터미널
	Temporary Ket Generation	
	SelPK(K_{bal} index, Pf, RND)	
CTC2 = CTC1 +1		
TK2 = 3DES(CTC2, K_{bal})	CTC2 = CTC + 1	
Checks and selects the purse		
	ReadBalance(0, Pf, TTC)	
Checks access condition(만족하면)		
Reads the purse balance value		
Computers the card balance certificate		
CBC = 3DES(00 || Pf || Bv || TTC, TK2)		
	GetResp	
Bv, CBC		
	Optional	
	Certificate Verificatoin	
	verrify Certificate	
	(MK_{bal} index, CSN, CTC2, CBC)	Key deversification
		K'bal = 3DES(CSN, MK_{bal})
		TK'2 = 3DES(CTC'2, K'_{bal})
		CBC' = 3DES(00 || Pf || Bv || TTC,TK'2)
		비교 CBC' = CBC
	ACK = Display balance, ESC = Reject card	If yes returns ACK,If not returns ESC

표 8. 지불 트랜잭션 프로토콜(1)

스텝 3 지불		
USIM(3DES)	Termianl 애플리케이션	터미널
	Temporary Ket Generation	
	SelPk(K_{dt} index, Pf, RND)	
CTC3 = CTC2 +1		
TK3 = 3DES(CTC3, K_{dt})	CTC3 = CTC2 + 1	
Checks and selects the purse		
	Debit(Dv, TTC)	
Checks Balance >= Dv(만족하면 잔액 계산)		
CBC = 3DES(0, Pf, Dv, TTC, TK3)	GetResp	
	인증서검증	
	(MK_{dt} index, CSN, CTC'3, CDC)	Key diversification
		K' dt = 3DES(CSN, MK_{dt})
		TK'3 = 3DES(CTC'3, K'_{dt})
		CDC' = 3DES(0, pf, Dv, TTC, TK'3)
		비교 CDC' = CDC
	ESC = Deny Transaction, Update Logfile	If yes returns ACK,If not returns ESC

표 9. 지불 트랜잭션 프로토콜(2)

스텝 4 지불 트랜잭션 서명		
USIM(3DES)	Termianl	지갑 제공자
	Sign(Kn, Ko)	
TK4 = 3DES(CTC3, $K_{sign\ -\ dt}$)		
Signature = 3DES(0, Pf, Dv, TK4)		
	GetResp	
Signature		
	지불 트랜잭션	
	CTC3, Signature, Dv, Pf, ...)	
	트랜잭션 Collection	Key diversification
		Ksign' = 3DES(CSN, MKsign)
		TK'4 = 3DES(CTC3, Ksign')
		Signature' = 3DES(0, Pf, Dv, TK'4)
		비교 Signature = Signature'
		Error = Trasaction *Repudiation*

표 10. USIM에 내장되는 필수 데이터

이 름	기 술	길이(바이트)
Application Identifier (AID)	전자 지갑 애플리케이션의 식별자	5-16
전자 지갑 ID	전자 지갑의 ID	4
Application Expiration Date	애플리케이션 유효 날짜	3
Application Effective Date	애플리케이션 발효 날짜	3
Application File Locator (AFL)	Indicates the location (SFI, range of records) of the AEFs related to a given application	var. up to 252
Application Interchange Profile	Indicates the capabilities of the card to support specific functions in the application	2
Application Label	Mnemonic associated with the AID according to ISO/ IEC 7816- 5	1- 16
Application Primary Account Number (PAN)	Valid cardholder account number	var. up to 10
Application Primary Account Number (PAN) Sequence Number	Identifies and differentiates cards with the same PAN	1
Application Currency Code	Indicates the currency in which the account is managed according to ISO 4217	2
Application Transaction Counter	Counter maintained by the application in the ICC	2
Current Balance	전자 지갑의 현재 잔액	4
Maximum Balance	최대 잔액	4
Maximum Transaction Amount	1회 최대 거래액	4
Certification Authority Public Key Index	Identifies the certification authority's public key in conjunction with the RID	1
Enciphered Master Personal Identification Number (PIN) Data	Transaction PIN enciphered at the PIN pad for online verification or for offline verification	8
Enciphered User Personal Identification Number (PIN) Data	Transaction PIN enciphered at the PIN pad for online verification or for offline verification	8
Issuer Public Key Certificate	Issuer public key certified by a certification authority	NCA
Customer Public Key Certificate	Customer public key certified by a certification authority	NCA
Customer Private Key	USIM의 비밀키	32
Lower Consecutive Offline Limit	Issuer- specified preference for the maximum number of consecutive offline transactions for this ICC application allowed in a 터미널 with online capability	1
Personal Identification Number (PIN)	Try Counter Number of PIN tries remaining	1
Upper Consecutive Offline Limit	Issuer specified preference for the maximum number of consecutive offline transactions for this ICC application allowed in a 터미널 without online capability	1
Transaction Log File	지불 관련 거래 기록 저장	var.
Remote Wallet Server IP Address	전자 지불 서버의 주소	4

3.5 USIM에서의 상호인증과 암호화

상점과 사용자간의 상호인증은 일반적인 카드 상에서의 인증구조를 기본으로 하여 구현한다. 상호인증을 위하여 인증서를 교환하고 검증함으로써 상대방을 인증한다. 공개키 기법에 기반을 둔 디지털서명을 이용하여 USIM의 개인화후의 부정한 데이터의 변조를 감지한다. 본 논문에서는 기존의 SSL에 기반을 두고 지불 프로토콜을 설계하였다. 데이터의 기밀성 유지를 위하여 Session Key를 랜덤하게 생성하여 데이터를 암호화한다. Session Key는 USIM안에 저장되어 있는 16바이트의 Master Key와 2바이트의 ATC(트랙잭션 카운터)를 사용하여 수학적으로 생성되며, 이 Session Key를 이용하여 암호화 알고리즘을 통하여 데이터를 암호화한다. 사용자의 단말기와 상점의 컴퓨터와 연결되면 암호화 방법에 대하여

협상을 시작한다. 즉 암호화 방법, 트랜잭션 카운터, 난수 등의 정보를 교환한 다음, 서로 인증서를 교환하고, USIM 애플리케이션이 자신의 Master Key를 공개키 알고리즘으로 암호화하여 상점의 터미널로 보낸다. 상점의 터미널은 USIM의 공개키로 이를 해독하여 Master Key를 알 수 있고 이 Master Key를 이용하여 미리 정해진 알고리즘에 따라 USIM Write Key, 터미널 Write Key, USIM MAC Write Key, 터미널 MAC Write Key를 생성하게 된다. 표 11는 3DES 알고리즘을 사용하여 USIM과 터미널 사이의 실제적인 암호화 과정 프로토콜을 보여준다.

3.6 기존 모바일 기기 및 신용카드 기반의 전자화폐 시스템과의 비교 분석

본 논문에서 제안한 모바일 단말기와 USIM의 연계를 통한 오프라인 지불 프로토콜은 기존에 제공되고 있는 모바일 기기 및 신용카드 기반의 전자화폐 시스템과는 차이점이 있다. 모바일 기기 상에서 제공하는 전자화폐 시스템은 보통 핸드폰 요금을 통한 결제나 신용카드 번호를 저장하고, 저장되어 있는 번호를 통하여 지불을 하는 시스템이 일반적이다. 또한 신용카드를 이용한 인터넷 지불 시스템은 번호 유출의 위험성을 지니고 있으며, 안전성이 미비한 상태이다. 그러나 본 논문에서 제안한 모바일 기기와 USIM의 연계를 통한 지불 프로토콜은 모바일 기기에서 제공하는 저용량의 메모리와 연산 처리 능력을 USIM이 대신함으로서, 많은 양의 데이터를 저장할 수 있고, 지불 서버와의 연계 없이 오프라인 상에서 연산 기능을 수행할 수 있다. 특히, USIM상에는 개인 인증 정보와 비밀키 등의 데이터를 암호화적으로 저장하고 있으므로, 오프라인 상에서의 인증을 위한 기술을 갖추고 있다. 즉, 본 논문에서 제안한 시스템은 기존의 모바일 기기와 신용카드 기반의 전자화폐 시스템보다 발전된 형태로서 모바일 기기와 USIM의 연계를 통하여, 오프라인 상에서 안전하게 거래할 수 있는 지불 시스템이다.

4. 시뮬레이션

본 장에서는 제안한 오프라인 지불 시스템의 효율성을 입증하기 위하여 각 부분을 소프트웨어로 시뮬레이션 하였다. 아직까지 단말기와 USIM과의 연계를 통한 지불 솔루션은 실용화되어 있지 않기 때문에, 표준화조차 미비한 상태이다. 따라서 시뮬레이션을 위한 애플리케이션은 EMV(Europay Master Visa) 스펙과 3GPP 문서의 표준을 바탕으로 개발하였다. 시뮬레이션을 위하여 8비트 프로세서와 32K의 EEPROM을 탑재한 Gemplus사의 Java Card를 사용하였고, 사용자와 상점간의 데이터 교환을 위하여 TCP/IP 통신을 이용하였으며, 애플리케이션 도구로는 JBuilder을 사용하였다. 3장에서 설계한 내용을 바탕으로 결제 시스템을 작성하여 시뮬레이션 하였

표 11 . 3DES를 이용한 USIM과 터미널의 암호화 과정

USIM		터미널
키(인증용, 지불용, Secret Code 용, 충전용키, 서명키)	<- Command -> SW(USIM시리얼정보)	마스터키(인증용, 지불용, Secret Code 용)
USIM 시리얼정보	<- Command(파일, 랜덤수)	Teminal 랜덤수 생성
랜덤수 생성, 세션키생성(카드랜덤수와 인증용 키를 3DES를 돌려 나온값), 검증용암호문생성(세션키와 터미널랜덤수를 3DES를 돌려 나온값)	->SW(암호문)	이미 저장되어있는 USIM시리얼정보를 이용하여 USIM의 개별키를 계산하고, USIM과 같은 로직을 사용하여 암호문을 생성하여 전송받은 값과 같으면 세션활성, 값이 틀리면 에러

다[12-15].

4.1 오프라인 지불 시뮬레이션 시나리오

상점의 컴퓨터는 전화번호나 블루투스를 이용하여 단말기에 결제를 요청한다고 가정하였다. 실제 시뮬레이션은 TCP/IP상에서 데이터를 교환하는 시스템으로 구성하였다. 단말기는 결제에 필요한 초기화를 시작하며, 상점의 컴퓨터와 인증서를 교환한다. 인증서의 공개키를 이용하여 암호화된 Session Key를 발생시킬 수 있는 Master Key를 서로 교환한다. Session Key에 대하여 일치가 되면 이 Session Key를 이용하여 모든 데이터를 암호화한 후 지불을 수행한다. 표 12은 Java Card상에서 Session Key를 발생시키는 과정을 보여준다. Session Key는 단말기와의 접속시 매번 발생하며, 생성된 Session Key를 암호화 알고리즘에 적용하여 데이터를 암호화한다. 인증과정이 완료되면 Java Card내의 애플리케이션을 선택한다.

4.2 시뮬레이션

기본적으로 Java Card내에 저장되는 애플리케이션은 각각의 AID(Application Identifier)를 갖게 된다. 또한 각각의 애플리케이션에 맞는 APDU를 생성해야 한다. 본 연구에서 필요한 기본적인 APDU는 전자화폐 시스템에서 필요한 메소드로서, 결제 메소드, 잔액확인 메소드, 충전 메소드, 핀 검증 메소드를 기본적으로 정의하였다. 표 13은 전자지갑 애플릿에 필요한 메소드를 보여주고 있다.

그림 5는 Java Card 상에서 애플릿을 select 하는 과정을 보여주는 그림이다. 각각의 메소드의 사용은

표 12. 인증과정과 초기화 과정

```
ATR returned by the Card...
<--- Card : 3B 6E 00 00 80 31 80 65 B0 03 02 01 5E 83 00 00 90 00
cardRandom: 86 DB BF 9D A6 67 28 18
hostRandom: 00 00 00 00 00 00 00 00
derivationInputData: A6 67 28 18 00 00 00 00 86 DB BF 9D 00 00 00 00
Encryption staticKey: CA CA CA CA CA CA CA CA 2D 2D 2D 2D 2D 2D 2D 2D CA CA CA CA CA CA CA CA
Encryption sessionKey: A4 7E 16 D7 43 49 69 EA FA EC 5F 2D 86 CF B7 67 A4 7E 16 D7 43 49 69 EA
Macing staticKey: 2D 2D 2D 2D 2D 2D 2D 2D CA CA CA CA CA CA CA CA 2D 2D 2D 2D 2D 2D 2D 2D
Macing sessionKey: 87 7D D3 6C 3D A8 0F 70 0D 75 F3 47 E3 18 E7 9F 87 7D D3 6C 3D A8 0F 70
KEK staticKey: CA 2D CA 2D CA 2D CA 2D 2D CA 2D CA 2D CA 2D CA CA 2D CA 2D CA 2D CA 2D
KEK sessionKey: CA 2D CA 2D CA 2D CA 2D 2D CA 2D CA 2D CA 2D CA CA 2D CA 2D CA 2D CA 2D
hostAndCardRandom: 00 00 00 00 00 00 00 00 86 DB BF 9D A6 67 28 18 80 00 00 00 00 00 00 00
  calculatedCardCryptogram: B8 C7 3C B9 97 4A BD D4
cryptoFromCard: B8 C7 3C B9 97 4A BD D4
cardAndHostRandom: 86 DB BF 9D A6 67 28 18 00 00 00 00 00 00 00 00 80 00 00 00 00 00 00 00
hostCryptogram: 9E 6B 01 44 5C 23 43 11
Authentication OK
Initialize OP global PIN
PIN initialisation OK
Select application...
Select application null OK
```

표 13. 전자지갑 애플릿에 저장되는 데이터

설계내역(전자지갑 애플릿 내장 데이터)		
AID	A0 00 00 00 18 FF 00 00 00 00 00 00 00 00 01 02	
지불을 위한 메소드	**메소드**	**command**
결제	debit()	00 31 00 00 02 00 05 02
잔액확인	getBalance()	00 30 00 00 02 10 00 02
충전	credit()	00 32 00 00 02 00 10 00
핀검증	verifyPin()	00 33 00 00 04 01 02 03 04 00

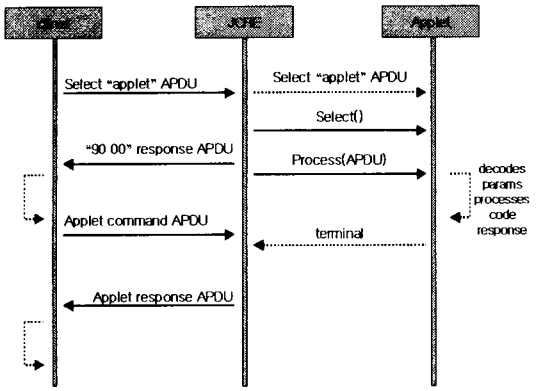

그림 5. Java Card 상에서의 애플릿 select() 단계

command와 response의 교환을 통하여 이루어진다.

다음의 그림들은 전자지갑 애플리케이션이 선택되었을 때의 결제 과정을 보여준다. 단말기의 USIM은 기본적으로 여러 개의 애플리케이션을 저장할 수 있다. 본 논문에서는 전자 지불 시스템을 중점적으로 구현하였으며, 실제 상점과의 통신을 위해서 TCP/IP상에서 시뮬레이션 하였다. 그림 6은 전자지갑을 선택하는 과정이며, 그림 7과 그림 8은 전자지갑이 구동되어 잔액을 확인하는 그림이다. 잔액확인을 위하여 USIM 내의 데이터를 호출해야 한다. 따라서 생성한 잔액확인 APDU 명령을 통하여, USIM 내의 잔액을 확인한다. 호출된 명령어가 정상적이면 남아있는 금액을 binary 코드로 표현하고 '90 00' 이라는

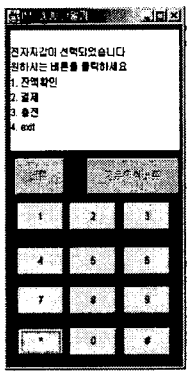

그림 6. 서비스선택

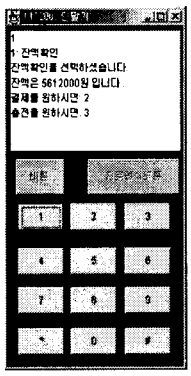

그림 7. 잔액확인

```
Performing a getBalance()...
--> Term : 00 30 00 00 02
<-- Card : 15 EC 90 00
```

그림 8. 잔액 APDU

메시지를 보낸다.

잔액확인을 위하여 getBalance()메소드를 구현하였으며, 표 14는 getBalance()메소드의 구현 소스를 보여준다. getBalance()의 역할은 카드의 잔액을 확인하기 위한 메소드로써, 이 메소드를 통하여 현재 카드에 남아있는 금액을 확인할 수 있다.

표 14. getBalance() 수행 과정

```
private void getBalance( APDU apdu ){
if(OPSystem.getCardContentState()==
    OPSystem.APPLET_BLOCKED)
                ISOException.throwIt

(ISO7816.SW_COMMAND_NOT_ALLOWED);
byte[] apduBuffer = apdu.getBuffer();
    apduBuffer[5] = (byte)(balance >> 8) ;
    apduBuffer[6] = (byte)balance ;
    apdu.setOutgoing() ;
    apdu.setOutgoingLength((short)2) ;
    apdu.sendBytes((short)5, (short)2) ; }
```

그림 9과 그림 10은 상점과 접속하여 인증서를 교환하는 것을 보여준다.

그림 11는 결제를 위한 정보를 확인하는 단계이며, 그림 12에서와 같이 PIN을 입력하게 된다. PIN을 입력한 후의 command와 response 상태는 그림 13와 같다. 결제를 하기 위하여 PIN을 부여하는 것은 적법한 사용자인지의 여부를 판단하기 위한 과정이다.

그림 14는 최종 결제가 완료되었을 경우의 완료 그림이며, 그림 15은 결제 APDU의 상태를 보여준다.

표 15은 위에서 설명한 결제를 위한 debit()메소드의 소스를 보여준다.

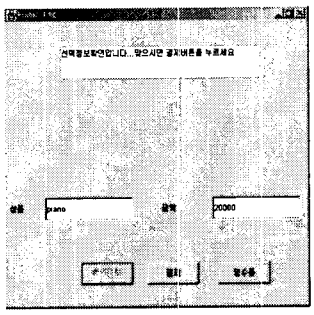

그림 9. 상점화면

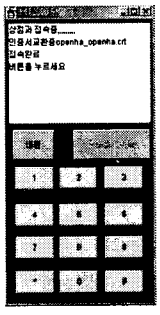

그림 10. 상점과 접속

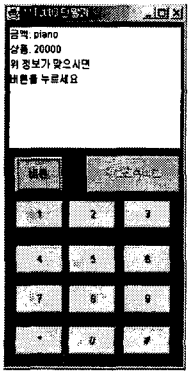

그림 11. 결제1

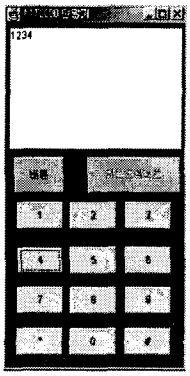

그림 12. 결제2

--> Term : 00 33 00 00 04 01 02 03 04
<-- Card : 90 00

그림 13. PIN 체크 APDU

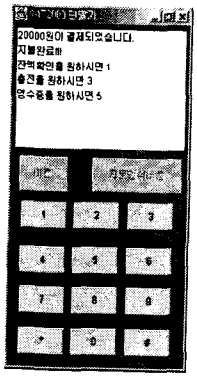

그림 14. 결제3

--> Term : 00 31 00 00 02 00 14 02
<-- Card : 15 D8 90 00

그림 15. 결제 APDU

5. 결 론

본 논문에서는 IMT-2000 상에서의 USIM을 이용한 안전하고 효율적인 오프라인 지불 프로토콜을 제안하고 시뮬레이션 하였다. 본 시스템은 모바일 기기와 USIM의 연계를 통하여 기존의 모바일 기기의 부족한 메모리와 연산기능을 해결하였으며, USIM의 개인인증 정보 모듈을 통해 오프라인 지불이 가능한 시스템을 설계하였다. USIM상에서의 오프라인 지불 시스템은 기존의 모바일 기기의 지불 시스템 및 신용카드 기반의 지불 시스템을 동시에 확장한 것으로써, 안전성 및 보안성을 높였으며, USIM내에 전자지불 애플리케이션을 탑재하여 오프라인 상에서 언제든지 거래를 할 수 있다. 제안한 시스템의 유효성을 입증하기 위하여 기존의 시스템과 비교분석 하였으며 차세대 COS로 주목받고 있는 Java Card를 채택하여 USIM상에서 오프라인 지불이 가능한 전자지불 애플리케이션을 개발하였다. 시뮬레이션을 위하여 3GPP 문서와 ISO 7816 표준을 기준으로 잡았으며, Gemplus사에서 제공하는 GemXpresso RAD 211 툴킷과 JBuilder를 이용하였다.

현재 모바일 기기와 USIM의 연계를 통한 애플리케이션의 개발은 미비한 실정으로, 다양한 형태의 지불 솔루션의 개발이 필요하며, 로열티 등의 지불과 연계하여 사용될 수 있는 애플리케이션의 개발이 뒷받침되어야 할 것이다. 또한 좀 더 강력한 보안성을 위하여 모바일 기기에 지문 인식 버튼 등의 생체 정

표 15. 결제를 위한 debit() 메소드

```
private void debit(APDU apdu) throws ISOException, UserException{
if(OPSystem.getCardContentState() == OPSystem.APPLET_BLOCKED)
ISOException.throwIt(ISO7816.SW_COMMAND_NOT_ALLOWED);
if(!validPIN) ISOException.throwIt(ISO7816.SW_SECURITY_STATUS_NOT_SATISFIED);
        byte[] apduBuffer = apdu.getBuffer();
if(apduBuffer[4] != 2 || apdu.setIncomingAndReceive() != 2){
          ISOException.throwIt(ISO7816.SW_WRONG_LENGTH) ;}
short amount = (short)(((apduBuffer[6]) & (short) 0x000000FF) | ((apduBuffer[5 ]<< 8) & (short)0x0000FF00));
if((balance >= amount) && (amount > 0)){
        balance -= amount ;
        apduBuffer[7] = (byte)(balance >> 8) ;
        apduBuffer[8] = (byte)balance ;
        apdu.setOutgoing() ;
        apdu.setOutgoingLength((short)2) ;
        apdu.sendBytes((short)7, (short)2) ; }
else    throw new UserException(ILLEGAL_AMOUNT) ; }
```

보를 저장하는 기술과의 접목을 통하여 모바일 기기와 USIM의 활용성을 높이도록 해야 할 것이다.

참 고 문 헌

[1] ftp://ftp.3gpp.org/specs
[2] http://www.sun.com
[3] http://www.gsmworld.com
[4] Wolfgang Effing and Wolfgang Rankl, "Smart Card Handbook", John Wiley & Sons, 2000.
[5] Zhiqun Chen, "Java Card Technology for Smart Cards", Addison-Wesley, 2000.
[6] S.Brands, "Off-Line Cash Transfer by Smart Cards", CWI', 1994.
[7] J.Bos and D.Chaum, "Smart Cash: a practical electronic payment system", Technical Report CS-R9035, CWI,'1990.
[8] 하남수, 홍인식 "IMT-2000에서의 USIM을 위한 구조 설계 및 응용 프로그램 구축에 관한 연구", 정보처리학회 춘계학술대회, 제 8권 제1호, pp. 627-630, 2001.
[9] C.Schnorr, "Efficient Signature Generation by Smart Cards", Journal of Cryptology, Vol. 4, No 3, pp 161-174, 1991.
[10] Eric Vetillard, "Java Card 2.1 general presentation", Gemplus Developer Conference, 1999.
[11] Chen, "Java Card Technology for Smart Cards", Addison Wesley, 2000.
[12] Ivor Horton, "Beginning Java2", WROX, 2000.
[13] Gemplus사의 GemXpresso RAD 211 툴깃 스펙
[14] http://www.gemplus.com
[15] http://www.emvco.com

백 장 미

2001년 순천향대학교 컴퓨터학부 (학사)
2003년 순천향대학교 대학원 전산학과(석사)
2003년 순천향대학교 대학원 전산학과 박사과정

관심분야 : M-Commerce, 스마트 카드, 모바일 통신

하 남 수

1999년 순천향대학교 컴퓨터학부 (학사)
2002년 순천향대학교 대학원 전산학과(석사)
2002년 2월~현재 비자캐쉬코리아 연구원

관심분야 : 스마트 카드, USIM, 모바일 통신

홍 인 식

1981년 한양대학교 전자공학과 (학사)
1986년 한양대학교 대학원 전자공학과(석사)
1988년 한양대학교 대학원 전자공학과(박사)
1991년~1995년 순천향대학교 공과대학 전산학과 전임강사
1995년~1999년 순천향대학교 공과대학 컴퓨터학부 조교수
1999년~순천향대학교 공과대학 정보기술학부 부교수
관심분야 : 임베디드 시스템, 스마트 카드, 모바일 통신

교 신 저 자

백 장 미 336-745 충남 아산시 신창면 읍내리 646 순천향대학교 정보기술공학부 멀티미디어관 m604