특 집

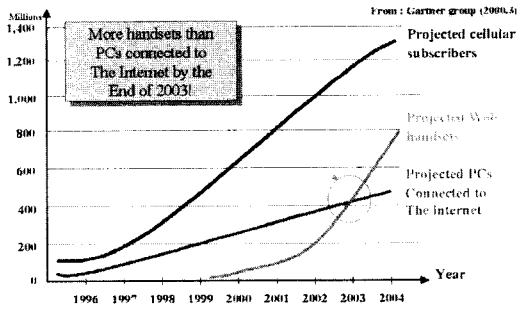
무선 PKI를 중심으로 한 무선 인터넷 보안 분석

박 기 준*

● 목 차 ●

1. 서 론

세계적으로 유명한 컨설팅 회사인 가트너 그룹의 2000년 3월 보고자료(그림 1)를 보면 2003년이 되면 인터넷을 사용하기 위해 인터넷이 가용한 Device를 구매하는 사람들은 PCs보다는 Handsets (휴대폰, PDA 등)을 구입하는 사람이 훨씬 많아지리라고 한다.



(그림 1) PCs VS Handsets

이는 점점 현실로 다가와서 핀란드의 대표적인 이동통신사인 SONERA 社의 최근 기사를 보면 핀란드에서는 이미 인터넷 사용자중에는 Handsets를 구입하는 고객이 PCS를 구매하는 고객을 능가하고 있다고 한다.

점점 인터넷 사용 고객들은 새로운 환경에 대한 갈망이 커져 나갔으며 따라서 다양한 Handsets을 위한 서비스를 요구 하게 되었다.

처음에는 게임, 캐릭터, 벨 소리, 이모티콘 등의 엔터테인먼트가 주류를 이루던 서비스 영역에서 증권, 은행, 지불, 예약 및 매매, 경매 등 다양한 서비스의 영역으로 확대되고 있다.

예를 들어 필자가 생각하는 2005년의 우리의 생활 모습은 아마도 이러할 것이다.

철수는 영희와의 데이트에 무척 기대를 가지고 있다. 철수는 영희에게 신촌역 앞에서 7시에 만나기로 하고는 자신의 휴대폰으로 전화를 걸고는 약속 확인을 하였다. 영희가 스파게티가 먹고 싶다고 하자 역시 자신의 휴대폰 중 LBS (Location Based Service)를 통해 신촌역 반경 2km 이내에 있는 스파게티 전문점을 검색하기 시작하였다. 휴대폰 단말기 창에는 각종 메뉴가 포함되어 있는 스파게티 전문점의 아이콘이 번쩍거리기 시작했으며 철수는 마침내 마음에 드는 전문점에서 자리 예약을 하고 메뉴를 시간에 맞추어 주문하고 지불을 한다.

감미로운 저녁식사를 마친 후 미리 지불한 휴대

* (주) 드림시큐리티 무선보안 사업본부 부장

폰을 신용카드 조회기에 스친 후 스파게티 전문점을 나왔다. 과연 지불은 어떻게 했을까? 미리 휴대폰 결제를 통해 지불을 했으며 최종 신용카드 조회기에서 영수증 만을 받을 뿐이었다. 일련의 모든 과정에는 당연히 본인임을 증명하는 인증서를 통한 인증 및 주요 DATA에 대한 보안이 이루어 졌음은 물론이다.

이와 같이 무선 인터넷은 우리의 생활에 큰 변화를 가지고 올 것이며 아울러 다양한 서비스를 촉진시키는데 무선 보안 즉 WPKI는 매우 중요한 요소라 아니할 수 없는 것이다.

2. 무선 보안의 현실

현재 국내에 보급되어 있는 휴대폰은 대략 2천7백만대로 추산된다. 이는 PC보급대수의 배가 넘는 수량이다. 이를 토대로 국내 이동통신 3사는 각각의 서비스에 부합되는 보안 및 인증 메커니즘을 보유했거나 하고 있는 중이다.

먼저 각 이동통신사의 보안 현황을 살펴보면 아래 <표 1>과 같다.

<표 1> 각 이동통신사별 보안,인증 현황

구 분	KTF (016, 018)	SKTelecom (011, 017)	LGTELECOM (019)
CA 구축여부	· 공인인증기관과의 연동	· 자체 CA 구축 및 위탁운영 · 공인인증기관과의 연동	· 공인인증기관과의 연동
RA 구축여부	· RA Center 구축	· RA Center 구축	· RA Center 구축
서비스 내용	· Mobile Commerce · USIM 연동추진 (PlugIN) · VM based SERVICE	· Mobile Commerce · USIM 연동추진 (PlugIN, Slot type) · VM Based SERVICE	· Mobile Commerce · JAVA Based SERVICE
단대단 보안	· SSL (Extension)	· SK E2E (WALS)	· WALS
적용단말기	· ME 1.2 · BREW Phone · IC Chip Based Phone	· Wallet Phone · Wi_TOP Phone · IC Chip Based Phone	· UP Phone · JAVA Phone
서비스 개시 시기 (예정)	2002. 하반기	2001. 하반기	2002. 하반기

위의 표를 살펴보면 먼저 인증 기관의 경우에는 KTF와 LGTelecom은 무선공인인증기관(가칭)으로 준비중인 한국증권전산㈜, 한국전자인증㈜, 한국정보인증㈜(이상 가나다순) 등의 인증서비스를 근간으로 서비스를 준비하고 있으며 SKTelecom의 경우에는 공인인증기관 연동 및 자체 CA(Certificate Authority)에 의한 서비스를 병행해 준비하고 있는 상태이다.

또한 계속 중요한 화제로 이야기 되고 있는 단대단 보안(일명 E2E) 의 경우에는 각각의 환경에 맞추어서 KTF는 SSL Protocol을, SKTelecom과 LG Telecom 경우에는 응용계층에서의 암호화 복호화를 수행하고 있다. 이 서비스는 이미 시장에 뿌려진 거의 모든 단말기에 포팅되어 무선 인터넷 은행 및 증권 서비스 등에 널리 사용되고 있다.

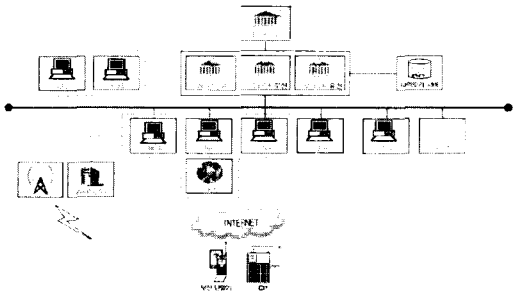
정리해보면 무선 인터넷의 기밀성 무결성 및 사용자인증의 경우에는 이미 서비스를 수행하고 있다고 해도 과언은 아니다.

이제는 전자서명을 통한 거래사실에 대한 부인방지 및 인증서 기반의 사용자 인증에 대한 인프라 확산이 주요 이슈화 내용이다. 물론 모든 이동통신사가 준비를 끝내놓은 상태이기는 하지만 사용자 편리성에 입각하면서도 완벽한 보안이 이루어 지는 그런 서비스를 내놓는 것이 가장 중요하다 할 수 있다.

본 특집에서는 무선 보안 인증 서비스를 위한 서비스 내용을 살펴 보도록 하며 예제 프로그램을 같이 따라 해봄으로써 무선 보안 인증에 대한 이해를 돕고자 하는 것이다.

3. 무선 PKI란?

앞에서 언급한 바와 같이 무선 PKI란 휴대폰을



(그림 2) 무선 PKI 시스템 구성도

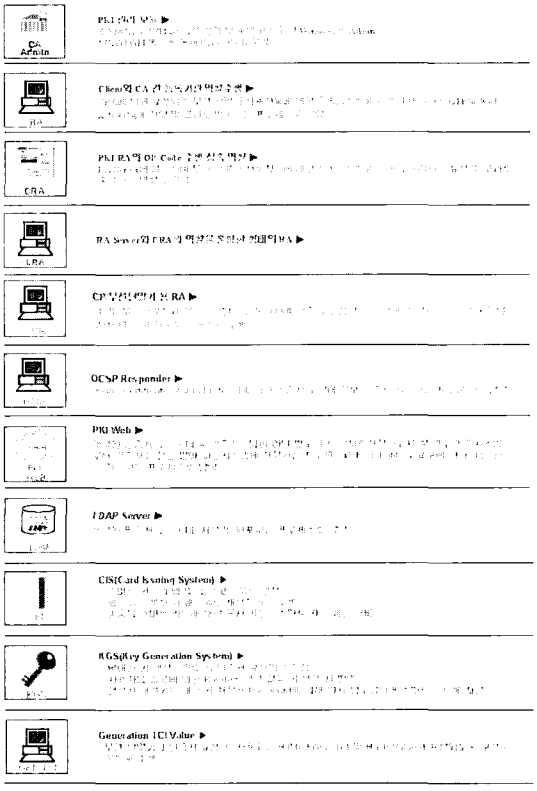
통해 안전한 거래를 할 수 있도록 해주는 보안 기술을 뜻하며 구성요소는 크게 인증기관과 등록기관 그리고 단말기 및 CP/SP server 용 프로그램으로 구성된다(그림 2).

주요 구성요소에 대해 구체적으로 살펴보면

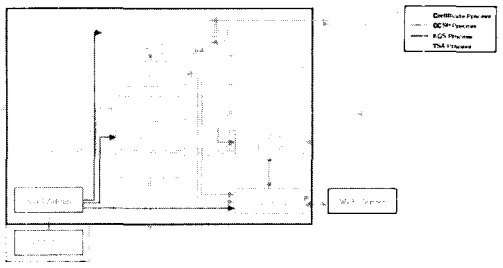
CA(Certificate Authority: 인증기관)란 인증서를 발급해주고 인증서의 유효성을 항상 게시해 주는 역할을 수행하는 기관을 말하며 RA(Registration Authority : 등록대행기관)이란 인증기관을 대신하여 인증서를 발급받을 수 있도록 일련의 등록 절차를 대행해주는 기관을 뜻한다.

이밖에 TSA(시점 확인 기관)과 OCSP server (실시간 인증서 상태 검증 서버) 등으로 구성된다.

다음은 각 구성요소별 주요 특징 및 설명이다.



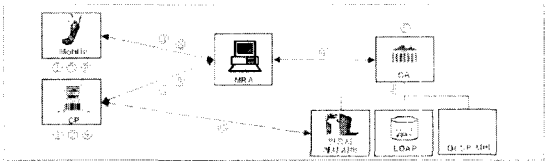
(그림 3)은 무선 PKI 시스템의 동작 FLOW 를 설명하는 내용이다.



(그림 3) 무선 PKI 시스템의 동작 FLOW

언뜻 보기에는 매우 복잡해 보이지만 결국은 인증서의 발급 절차와 인증서 검증 절차, 그리고 시점확인 절차를 그려놓은 그림이므로 화살표 대로만 확인한다면 쉽게 알 수 있으리라 생각된다.

이제는 실제 인증서를 발급 받는 절차를 살펴 보도록 하자.



① 참조번호와 인가코드 입력 후 키 생성
② 생성된 키로 인증서 요청 양식 생성
③ MRA와 SSL Class2를 통해 인증서 요청 양식 전송
④ 전송 받은 인증서 요청 양식을 CA에 OPP 거래를 통해 전송
⑤ 전송된 데이터의 이상 유무 및 인증성 발급 가능 여부 등을 검증 및 인증서 발급
⑥ 발급된 인증서를 LDAP 및 인증서 게시 서버에 저장
⑦ 인증서 게시 후 해당 URL을 MRA 재전송
⑧ CA가 되돌려 준 URL을 무선 단말기 CP에 재전송
⑨ 전송 받은 URL을 저장
⑩ 매일 인증서 게시 서버로부터 갱신된 인증서 수신

지금까지 인증서를 발급 받는 과정에 대해 살펴보았다. 아마 쉽게 서비스의 내용이 다가오지 않는 독자들을 위해 단말기 에뮬레이터를 통한 서비스 내용을 알아보기로 하자.

여기서는 KTF 의 메뉴를 중심으로 설명하면 아래와 같다.

조작순서	Phone Display	비고
1. 공인 인증기관 메뉴 선택	[증권과 금융] 1.증권 2.뱅킹 3.카드/신용정보 4.보험/대출 5.재테크 ▶6.공인인증기관 확인 상위	◇ 각 서비스 중 공인인증기관 메뉴를 선택한다. ◆KTF 메뉴
2. 공인 인증기관 선택	[공인인증기관] ▶1.AA CA 2.BB CA 3.CC CA 4.DD CA 확인 상위	◇ 제공되는 공인인증 기관 중 거래를 원하는 공인인증기관을 선택한다. ◆KTF 메뉴
3. 고객의 인증서 점검	[AA CA] 고객의 인증서를 점검합니다. 확인 취소	◇ 고객이 소유하고 있는 인증서의 종류를 요구하는 HTML을 수신하여 단말기에서는 CA 인증서를 RA에게 전송한다. ◆RA 메뉴
4. 인증서 관리 메뉴 선택	[AA CA] ▶1.인증서 신청 2.인증서 폐지 3.인증서 효력정지 4.인증서 재발급 5.인증서 갱신 6.개인키 비밀번호 변경 7.인증서 보기 확인 상위	◇ AA CA 인증서가 없는 경우 ◇ 인증서 신청 및 관리 메뉴 중 인증서 신청을 선택한다. ◆RA 메뉴
5. 참조번호/인가코드 소유 확인	[인증서 신청] 인증서 신청에 필요한 참조번호와 인가코드를 가지고 계십니까? 예 아니오	◇ 대면 확인 인증서 신청과 비대면 확인 인증서 신청을 나눈다. ◆RA 메뉴

조작순서	Phone Display	비고
6. 참조번호 입력	[인증서 신청] 참조번호를 입력하세요 _ 확인 취소	◇ 참조번호/인가코드가 있는 경우[사용자 등록을 마친 경우] ◇ 참조번호를 입력한다. ◆ RA 메뉴
7. 인가코드 입력	[인증서 신청] 인가코드를 입력하세요 _ 확인 취소	◇ 인가코드를 입력한다. ◆ RA 메뉴
8. 개인키 비밀번호 입력	[인증서 신청] 개인키 비밀번호를 입력하세요 _ 확인 취소	◇ 개인키 비밀번호를 입력한다. ◆ RA 메뉴
9. 개인키 비밀번호 재입력	[인증서 신청] 개인키 비밀번호를 다시 한번 입력하세요 _ 확인 취소	◇ 개인키 비밀번호를 재 입력한다. ◆ RA 메뉴
9. 확인 메뉴 선택	[인증서 신청] 확인 확인 취소	◇ 확인 버턴을 누른다. ◆ RA 메뉴

조작순서	Phone Display	비고
10. 키 쌍 생성 및 인증서 신청 양식 생성	[인증서 신청] 키 쌍과 인증서 신청 양식을 생성을 생성합니다. 약 10초 정도 소요됩니다. 확인 취소	◇ 전자 서명용 및 키 분배용 키쌍을 각각 생성 후 인증서 신청 양식을 생성한다. ◆ RA 메뉴
11. 인증서 신청 양식 전송	[인증서 신청] 보안을 유지하여 페이지를 봅니다. ■■■	◇ 인증서 신청 양식을 전송한다. ◆ 단말기 내부 메시지
11. 인증서 URL 수신	사용자 인증서 설치가 완료 되었습니다. 확인	◇ 사용자 인증서 수신 및 설치 ◆ RA 메뉴
12. 초기 메뉴	[증권과 금융] ▶ 1.증권 2.뱅킹 3.카드/신용정보 4.보험/대출 5.재테크 6.공인인증기관 확인 상위	◇ 초기 메뉴로 이동. ◆ KTF 메뉴

지금까지 개략적인 무선 PKI 구축 내용 및 세부 서비스 시나리오에 대해 살펴 보았다.

이제 부터는 실제 독자 여러분이 무선 PKI를 어떻게 활용하고 서비스에 적용 할 것인가를 살펴 볼 차례다. 이미 언급한 바와 같이 무선 PKI에 대한 인프라는 이동통신사에서 적용하고 있는 중이므로 단말기 부분에 있어서는 독자여러분이 고려할 내용이 전혀 없다고 해도 과언이 아니다. 다만 CP/SP 측에서 보안 인증 서비스를 적용하기 위한 솔루션을 구입 또는 제공 받아서 서비스를 적용하면 되는 것이다. 하지만 여기서는 여러분들의 이해를 돕기 위해 KTF MultiPack(BREW)용 WPKI module 사용법 예를 살펴 보기로 하자.

먼저 WPKI CLIENT 를 구성하는 요소는 <표 2>와 같다.

<표 2> WPKI CLIENT 를 구성 요소

모듈명	기 능	비 고
인증서 관리 모듈	BREW PHONE에서 사용하는 인증서 관리 모듈로서 전자 서명 인증서와 키 분배 인증서를 관리 한다.	전자서명 인증서와 키 분배 인증서는Pair로 관리됨
SSL 모듈	데이터 송/수신을 암호화 및 복호화를 통하여 수행하며 추가적으로 서버 인증 및 사용자 인증 기능을 포함한다.	
전자 서명 모듈	송신 부인방지를 위하여 전자 서명기능을 수행한다.	

이중에서 먼저 인증서 관리 모듈 (Certificates Manager : Certmgr) Applet 을 구동하기 위한 예제를 살펴보면 아래와 같다. 여기서는 무선 인터넷 뱅킹을 멀티팩으로 구축할 때의 예제 프로그램이다.

```
boolean BankApp_HandleEvent(IApplet * pi, AEEEvent eCode,
uint16 wParam, uint32 dwParam)
{
        CBankApp * pMe = (CBankApp *)pi;

ISHELL_GetDeviceInfo(pMe->a.m_pIShell,&pMe->di);

        switch (eCode)
        {
                case EVT_APP_START:
중략...
                        return TRUE;
                case EVT_APP_STOP:
                        return TRUE;
                case EVT_NOTIFY:
                {
                        AEENotify *pnot =
(AEENotify*)dwParam;
                        return(TRUE);
                }
                case EVT_KEY:
                {
                        switch(wParam)
{
case AVK_SELECT: //확인 버튼을 눌렀을 때
        // Certmgr Applet를 구동시킨다.
        // AEECLSID_TRUSTM는 Certmgr Applet의
ClassID 이다
        ISHELL_StartApplet(m_pIShell,
AEECLSID_TRUSTM);
        Break;
}
                                return TRUE;
                        }
                        return TRUE;
                        case EVT_APP_RESUME:
                                return TRUE;
                        case EVT_APP_SUSPEND:
                                return TRUE;
        }
        return FALSE;
}
```

따라서 ISHELL_StartApplet을 호출하면 EVT_APP_SUSPEND Event가 발생하고, ISHELL_StartApplet로 구동한 Applet이 종료되면 EVT_APP_RESUME Event가 발생한다.

다음으로는 보안 및 인증 프로토콜로 널리 사용되는 SSL module을 call하는 예제를 살펴 보도록 한다.

```
static void Handshake(SSLApp *pme)
{
        AECHAR szBuf[200];
        char fmt[100];
        int nRtn;

        ////////////////////////////////////////////////////
        // handshake
        nRtn = IWPKI_SSL_Handshake(pme->pIWPKI,
pme->pBuffer, &(pme->nLen));
        // 정상 종료
        if (nRtn == SSL_NO_ERR)
        {
                //SSL Handshake를 정상적으로 마쳤
으면 SendDingDong로 이동한다
        ISOCKET_Writeable(pme->m_piSock,
(PFNNOTIFY)SendDingDong, (void *)pme);
                ISOCKET_Readable(pme->m_piSock,
NULL, (void *)pme);
                //SSL로 보내고자 하는 데이터를 암
호화 한다. "GET /r\n\r\n"
```

```
                pme->m_SendTextLen = IWPKI_SSL_Encrypt(pme->pIWPKI, "GET /\r\n\r\n", 9, pme->m_SendText);
                pme->m_SentTextLen = 0; // 송신된 cipher text 길이

                pme->m_RecvTextLen = 400; // cipher text body 길이
                pme->m_RecvedTextLen = 0; // 수신된 cipher text 길이(헤더 길이 포함)
        }
        // send
        else if (nRtn == SSL_HANDSHAKE_SEND)
        {
                ISOCKET_Writeable(pme->m_piSock, (PFNNOTIFY)HandshakeSend, (void *)pme);
                ISOCKET_Readable(pme->m_piSock, NULL, (void *)pme);
        }
        // send 직후 recv
        else if (nRtn == SSL_HANDSHAKE_SEND_RECV)
        {
                ISOCKET_Writeable(pme->m_piSock, (PFNNOTIFY)HandshakeSendRecv, (void *)pme);
                ISOCKET_Readable(pme->m_piSock, NULL, (void *)pme);
        }
        // recv
        else if (nRtn == SSL_HANDSHAKE_RECV)
        {
                ISOCKET_Writeable(pme->m_piSock, NULL, (void *)pme);
                ISOCKET_Readable(pme->m_piSock, (PFNNOTIFY)HandshakeRecv, (void *)pme);
        }
        // continue
        else if (nRtn == SSL_HANDSHAKE_CONTINUE)
        {
                Handshake(pme); // 재귀호출이후에 수행하는 내용은 없도록 한다.
        }
        // 인증서선택 및 password를 받는다.
        else if (nRtn == SSL_HANDSHAKE_GET_PASSWORD)
        {
                return;
        }
        // error
        else // < 0
        {
                SPRINTF(fmt, "hs:%d", nRtn);
                STR_TO_WSTR(fmt, szBuf, sizeof(szBuf));
        IDISPLAY_DrawText(pme->a.m_pIDisplay, AEE_FONT_NORMAL, szBuf, -1, 0, 20, NULL, 0);
                IDISPLAY_Update(pme->a.m_pIDisplay);
                return;
        }
}

static void HandshakeSend(void *pme)
{
        SSLApp *pMe = (SSLApp *)pme;
        int nRtn;

        nRtn = ISOCKET_Write(pMe->m_piSock, pMe->pBuffer, pMe->nLen);
        if (nRtn <= 0) {
                int err = ISOCKET_GetLastError(pMe->m_piSock);
                return ;
        }
        Handshake(pMe);
}

static void HandshakeSendRecv(void *pme)
{
        SSLApp *pMe = (SSLApp *)pme;
        int nRtn;

        nRtn = ISOCKET_Write(pMe->m_piSock, pMe->pBuffer, pMe->nLen);
        if (nRtn <= 0) {
                int err = ISOCKET_GetLastError(pMe->m_piSock);
                return ;
        }
        ISOCKET_Writeable(pMe->m_piSock, NULL, (void *)pMe);
        ISOCKET_Readable(pMe->m_piSock, (PFNNOTIFY)HandshakeRecv, (void *)pMe);
}

static void HandshakeRecv(void *pme)
{
        SSLApp *pMe = (SSLApp *)pme;
        int nRtn;

        nRtn = ISOCKET_Read(pMe->m_piSock, pMe->pBuffer, 5);
        if (nRtn <= 0) {
                int err = ISOCKET_GetLastError(pMe->m_piSock);
                return ;
        }
        pMe->nLen = pMe->pBuffer[3]*256 + pMe->pBuffer[4];
        nRtn = ISOCKET_Read(pMe->m_piSock, (char *)(pMe->pBuffer+5), pMe->nLen);
        if (nRtn <= 0) {
```

```
                int err = ISOCKET_GetLastError
(pMe->m_piSock);
                return ;
        }
        pMe->nl.en += 5;
        Handshake(pMe);
}
```

지금까지 간단하게 WPKI applet client 를 구동하는 예제를 살펴 보았다.

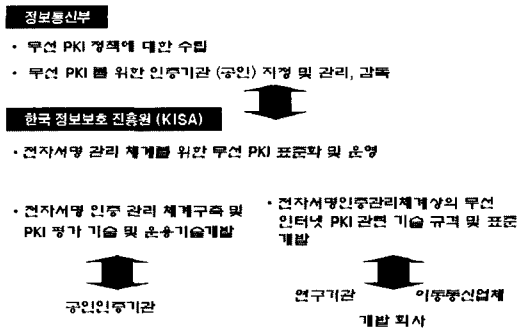
4. 무선 보안의 표준화 동향

국내에서는 현재 한국정보보호진흥원(KISA)(www.kisa.or.kr)을 중심으로 표준화 활동이 활발하게 추진되어 무선 PKI 기술 규격 V 1.21 이 탄생되었다. 이는 공인인증기관(가칭)과 이동통신사 그리고 개발 업체 등이 참여해 국내 무선 인터넷의 활성화 및 기반 확대를 위해 필요한 무선 보안 및 인증 분야에 대한 표준을 완성 함으로써 국가 정책에 입각한 보안 인프라를 확장 시키는 것을 목적으로 하고 있으며 이는 전세계적으로도 가장 앞서 있는 표준 체계라 할 수 있다.

다음의 <표 3> 및 (그림 5)는 무선 PKI 표준화 동향 및 무선 PKI 구축 추진 체계를 설명하는 것이다.

<표 3> 무선 PKI 표준화 동향

일자	세부 내용	참가 기관
2001/1~ 2001/3	- WPKI 국내외 동향 및 WAP 진영 표준검토	KISA, 유선 공인 인증기관
2001/3~ 2001/6	- 인증서 및 CRL 프로파일, DN규격, 알고리즘(전자서명/키분배) - 인증서 OID 등 일반적인 표준화 방향은 결론 확정	KISA, ETRI, 이동통신사, 유선 공인인증 기관 개발 업체
2001/6~ 2001/7	- 이동통신 3사와의 호환성 및 공인인증기관과의 호환성 검토	상동
2001/8	- 최종 공인인증기관 실질심사 기준안(표준 v1.21) 발표	상동



(그림 5) 무선 PKI 구축 추진 체계

5. 결 론

올 하반기가 되면 독자 여러분의 단말기에서는 인증서비스를 통한 무선 인터넷 뱅킹이나 지불 그리고 증권, 보험 등의 서비스를 하게 될 것이다. 그동안 보안의 취약성을 막연히 알고 있어 사용하지 않으셨다면 그 때부터는 마음껏 사용해도 괜찮을 거라고 감히 말할 수 있다. 또는 보안 인프라가 확산되지 못해 망설였던 무선 인터넷 서비스를 다양하게 펼쳐 나갈 수도 있을 것이라 생각된다.

특집기사를 의뢰 받은 후 어렵다고 생각되는 보안 인증 분야를 어떻게 쉽게 설명할 수 있을까 많이 고민 하였다. 가능한 한 많은 사람들이 보안 및 인증 분야에 대해 관심을 가지고 꾸준히 준비해 나간다면 무선 보안 인증 분야는 참으로 많은 영역에서 그 역할을 충실히 이행할 수 있으리라 생각된다.

특집기사를 읽어주신 여러분께 감사하다는 말을 끝으로 특집을 마치기로 한다. 궁금한 점이나 미진한 부분은 이메일(kjpark@dreamsecurity.com)을 통해 문의를 바란다.

참고문헌

[1] WAP Forum Proposed Version 9-Mar-2000, WAP-211-X.509 : WAP Certificate and CRL Profile.

[2] WAP Forum Proposed Version 3-Mar-2000, WAP-217-WPKI : Wireless Application Protocal Public Key Infrastructure Definition.

[3] IETF RFC 2510(1999.3), Internet X.509 Public Key Infrastructure Certificate Management Protocol.

[4] ITU-T Recommendation X.509(1997), Information technology — Open System Interconnection — The Directory : Authentication Framework.

[5] IETF RFC 2459(1999), Internet X.509 Public Key Infrastructure Certificate and CRL Profile.

저자소개

박 기 준

1989년 경희대학교 공과 대학 졸업
1998년 미국 Montgomery college IT MD 과정 수료
2000년~현재 (주) 드림시큐리티 무선보안 사업본부 부장
e-mail : kjpark@dreamsecurity.com