인터넷/인트라넷 보안을 위한 공증 사무소의 설계와 구현

정주원*, 김종우+, 박성주‡

*한국과학기술원 정보시스템연구소
+충남대학교 통계학과
‡한국과학기술원 테크노경영대학원

Design and Implementation of Certification Authority for Internet/Intranet

Jung Joo-won*, Kim Jong-woo+, Park Sung-joo‡

*Center for Advanced Information System
Korea Advanced Institute of Science and Technology
email: jwjung@camis.kaist.ac.kr

+Department of Statistics
Chungnam National University
email: jwkim@cais.kaist.ac.kr

‡Graduate School of Management
Korea Advanced Institute of Science and Technology
email: sjpark@cais.kaist.ac.kr

요 약 (Abstract)

인터넷의 기술은 인트라넷(Intranet), 익스트라넷(Extranet)이라는 이름으로 인터넷 기술을 활용한 기업 내 또는 기업간 네트워크 구성에 관한 연구들이 활발하게 진행되고 있다. 인터넷의 개방성으로 인해서 인터넷은 보안성이 떨어지는 문제점을 가진다. 이를 극복하기 위해서 최근 많은 암호기반 보안 기술들이 개발되고 있다. 대부분의 암호기반 보안 기술들은 전체네트워크 객체의 증명을 책임지는 공증 기관을 필요로 하는데, 이를 공증 사무소(Certification Authority)라 한다. 본 논문에서는 인터넷과 인트라넷의 취약한 부분을 보완하기 위한 최근의 암호기반 보안 기술과 그들 간의 상관관계를 파악한다. 보안의 기초인 공증사무소의 구조 및 역할을 조사하고, 실제로 공증 사무소를 운영하기 위한 점검 사항 및 운영 규칙 등을 제시한다. 또한, 공증 사무소에 필요한 소프트웨어를 구현한다.

I. 서론

인터넷 기술의 발전과 상업화는 인터넷 광고, 홈 쇼핑, 웹 예약, 전자상거래 등의 형태로 우리생활 깊숙이 침투하기 시작했다. 그러나 대부분의 인터넷 기술들은 보안이 빈약하여, 언제 무너질지 모르는 삼풍백화점 안을 쇼핑하는 것과 같은 불안감을 느끼게 한다. 최근 들어서는 인터넷 상의 크래킹 문제뿐만 아니라 전자 메일에 대한 보안의 필요성이 중요시되고 있다. 전자문서, 계약서 등과 같이 중요한 내용들이 전자메일을 통해서 전달되는 경우, 송신자의 증명, 변조의 방지가 필수적이다.

암호 기술은 크게 단일열쇠 암호, 공개열쇠 암호, 보안용 단 방향 함수의 세가지 기술로 나눌 수 있으며, 이를 이용하여 안전한 메시지 전송, 전자 지문, 디지털 서명 등이 가능하다.

그러나, 암호기술을 보안 시스템에 적용하려면 몇 가지 기본사항을 지켜야만 한다. 첫째, 단일열쇠 암호화 알고리즘의 열쇠와 공개열쇠 암호화 알고리즘의 비밀 열쇠는 누출되어서는 안 된다. 마치 열쇠를 도난 당하면 아무리 자물쇠가 튼튼하더라도 무용지물이 되는 것과 같은 이치이다. 둘째, 공개열쇠 암호를 사용할 경우 공개 열쇠의 소유주가 틀림없는지 검사할 수 있어야 한다. 이것은 공격자가 임의의 열쇠 쌍을 생성하여 공개 열쇠를 위조할 수 있기 때문이다. 따라서, 공개열쇠의 소유주를 확인하고 그 내용에 디지털 서명을 함으로서 공개열쇠의 사실 여부를 증명해주는 기관을 둘 수 있다. 이 기관을 공증사무소 (Certification Authority, 줄여서 CA)라 하며, 공개열쇠와 그 주인을 적은 내용에 이 기관이 서명함으로써 기관의 이름으로 공개열쇠의 주인을 증명해 주는 것을 증명서(Certificate)라 한다.

본 논문에서는 공증 사무소의 설계 이슈들을 한국과학기술원의 사례를 중심으로 소개한다. 2 장에서는 공증 사무소와 관련된 암호기반 보안 기술들을 간략하게 소개한다. 3 장에서는 공증사무소 설계 이슈들을 공증사무소 자체보안 이슈, 공증대상과 범위 결정 이슈, 공증 절차이슈로 구분하여 제시한다. 4 장에서는 한국과학기술원의 공증 사무소 설계 사례를 소개한다. 5 장에서는 KAIST 공증사무소 서버 구현에 대하여 소개한다. 6 장에서는 결론과 추후연구 과제들을 제시한다.

II. 암호기반 보안기술

본 장에서는 암호 기술을 통신 보안에 응용한 암호기반 보안 기술에 대해 간단히 살펴본다. 먼저, 인터넷에 적용하기 위해 제정중인 실제적인 인터넷 응용 보안 기술에 대해 살펴본다. 실제로 이 표준들이 어떤 식으로 적용되는지 알아본다.

1. 인터넷 암호기술 표준

본 절에서는 암호를 기반으로 한 인터넷 응용 보안 기술에 대하여 알아본다. 본 절에서 다룬 것 이외에도 많은 보안 기술이 있지만, 여기서는 본 공증사무소 구현에서 목표한 응용과 연관된 기술들만 나열하기로 한다.

SSL 은 Secure Socket Layer 의 약자로 HTTP 및 많은 인터넷 프로토콜이 쉽게 도청 당할 수 있다는 문제를 해결하기 위하여 Netscape 의 Frier 가 고안하였다. 현재 IETF TLS WG 에서 작업 중이며, 현재 version 3 protocol (SSLv3)을 제정중이다.[3]

SSL 은 공개 열쇠를 X.509 증명서 형태로 전달한다. 증명서는 서버 증명서와 클라이언트 증명서가 있는데, 서버 증명서는 필수이며, 서버의 서명을 검사함으로써 서버를 확실하게 확인(strong authentication) 할 수 있다. 클라이언트 증명서는 클라이언트의 확실한 확인을 위해 사용되며 필요에 따라 서버 측에서 요구할 수 있다. 확실한 확인은 login/password 형태의 단순 확인보다 훨씬 더 강력하다.

인터넷 메일의 보안은 1980 년대 초 Philip Zimmermann 의 PGP(Pretty Good Privacy:RFC-1991, 2015)를 시작으로 PEM(Privacy Enhanced Mail:RFC-1421~1424), MOSS(MIME Object Security Services: RFC-1848)등 여러 가지 해결책이 제시되어 왔다. 이들의 보안 기능은 대부분 비슷하나, 내부 데이터 표현 방식에서 차이가 난다.

S/MIME 은 Secure/Multipurpose Internet Mail Extensions 의 약자로 안전한 인터넷 메일의 교환을 위해 RSA Data Security 사에서 제안한 것이다. 현재 Internet Draft 를 제정중이다.[2] S/MIME 은 암호화된 메시지와 전자 서명을 기능을 가지고 있으며, 첨부파일(attachment file)이 포함된 전자 메일에도 적용할 수 있다는 장점이 있다. S/MIME 은 PKCS #7 과 PKCS #10 을 그대로 MIME type 에 추가한 것으로, 8 bit 전송을 기본

으로 한다. PEM 과 MOSS 는 7bit 전송을 기본으로 하고 암호정보를 Text 로 포함시킨다는 면에서 S/MIME 과 다르다.

자체 제작한 클라이언트나 Java, ActiveX 와 같은 mobile code 의 배포는 트로이 목마 공격의 목표가 되기 쉽다. 따라서, 배포되는 코드가 변형되지 않음을 증명하기 위해 프로그램에 디지털 서명을 추가하는 것을 코드 서명이라 한다.

Signed Java Applet 은 applet 에 프로그램 작성자의 디지털 서명을 추가함으로써, 프로그램 작성자를 명시하는 방식이다. 프로그램 제작자에 따라 하드디스크 등 기존의 Sandbox model 에서는 제한되었던 자원들을 사용할 수 있게 한다는 장점도 가지고있다. JDK 1.1 부터 지원되며, Applet 자체에 디지털 서명이 붙는 것이 아니라 jar 라는 저장형태로 추가된다. 따라서, Java applet 뿐 아니라 JavaScript, HTML 등의 디지털 서명으로 응용하기도 한다. 제작자의 공개열쇠 증명 형태는 X.509v3 증명서를 사용한다.

MS Authenticode 는 Active X control 에 디지털 서명을 추가하거나 서명의 진위 여부를 검사하는 기초 보안 구조이다. DLL, OCX, EXE, CAB 등의 파일형식을 지원한다. 이것은 Java 와는 달리 환경을 제한하지 않으며, 디지털 서명의 진위 여부만을 검사한다. 즉, 기본적으로 모든 책임을 사용하는 사람의 판단에 맡긴다. 마찬가지로 X.509 증명서로 공개열쇠를 증명한다.

2. 암호관련 소프트웨어

본 절에서는 암호기반 보안기술이 실제로 구현된 소프트웨어들에 대해 조사한다. 인터넷과 인트라넷에 관련된 web server 와 browser, mobile code 개발 도구 등을 살펴본다.

① Web Servers

Netscape 은 SSL 을 처음 개발한 곳인 만큼 다양한 종류의 Web server 군을 가지고 있다. 최초의 SSL web server 인 Commerce server 를 비롯하여, 초보자를 위한 Fast Track server, 그리고 Netscape SuiteSpot 에 포함되어 있는 Enterprise Server 등이 있는데 모두 SSL 을 지원한다. 단, Enterprise server 만이 사용자 증명서를 지원한다.

Internet Information Server(IIS)는 Microsoft 가 개발한 web server 로 Windows NT 4.0 Server edition 에 내장하여 판매하고 있다. 현재 3.0 까지 개발되어 있으며, Netscape server 군에 대항하기 위한 IIS 4.0 이 개발중이다. SSL 사용자 증명서를 지원하나 프로그램 자체의 보안성에 의문이 제기되고 있다.

Apache-SSL 서버는 현재 전세계 web server 의 40%이상을 차지하고 있는 Apache 서버와 SSL 공개 라이브러리인 SSLeay 를 합한 것이다. SSLv3 와 사용자 증명서를 지원한다. Apache-SSL 은 그 구성 요소들이 미국 밖에서 제작되었으므로 ITAR(무기수출 금지규정)를 위반하지 않는다. 따라서, 그 나라에서 제한하지 않는 한 미국 이외의 국가에서도 1024 bit RSA, 128 bit RC4 를 적법하게 사용할 수 있다.

Apache SSL 은 세 개의 구성요소를 모아 compile 해야 하기 때문에 컴퓨터에 익숙하지 않은 사람들이 사용하기 불편하다. 따라서, Stronghold 와 Sioux 같은 Apache-SSL 의 변형들도 판매되고 있다.

② Browsers

Netscape 은 Navigator 1.0 부터 SSL 을 설계, 구현, 추가하였다. 처음에는 단지 secure channel 만 생각했던 Netscape 은 Netscape Navigator 2.0 까지 사용자 증명서에 대한 개념이 없었으나, Netscape Navigator 3.0 부터 사용자 증명서를 추가하는 한편, hard coded 되어 있던 CA 증명서도 추가 삭제할 수 있도록 변경하였다. 가장 최근 버전인 Netscape Communicator 4.x 의 경우 사용자가 생성한 비밀열쇠와 사용자 증명서를 PKCS#12 형식으로 읽고 쓸 수 있는 기능을 추가 시켰다. 이 기능 덕분에, 증명된 열쇠 쌍을 backup 할

수 있으며, Netscape Communicator 를 포함한 PKCS #12 형식을 지원하는 다른 응용프로그램에서도 같은 열쇠 쌍을 사용할 수 있게 되었다. 또한, 전자우편 부분인 messenger 에 S/MIME 이 추가됨으로써 메시지의 암호화 및 디지털 서명 등이 가능해졌다. Java virtual machine 도 JDK 1.1 을 지원함에 따라 Java signed applet 을 사용할 수 있게 되었다.

인터넷 분야에서는 후발 주자였던 Microsoft 의 Internet Explorer 3.0 은 Netscape 과 마찬가지로 SSL v3 와 사용자 증명서를 지원한다. 그러나 Netscape 과는 달리 CryptoAPI 라는 별도의 API 를 제정하여 ActiveX 객체를 통하여 열쇠 쌍의 생성, 증명서 신청, 관리 등을 수행하도록 하였다. 또한 Active X object 에 서명을 확인하는 Authenticode 1.0 을 포함하고 있다. IE 4.0 은 Netscape Communicator 4.0 과 유사하게 PKCS #12 형태로 열쇠 쌍을 보관할 수 있으며, Java Signed Applet 을 지원한다. timestamp 기능이 추가된 Authenticode 2.0 으로 upgrade 되었다. Internet Explorer 의 Mail 과 News 부분은 Outlook express 라는 형태로 통합되었고, 이것은 S/MIME 을 지원한다.

③ 프로그램 개발 환경

JDK 1.1 에는 javakey 라는 코드 서명 도구가 포함되어 있다. 순수하게 Java 로 짜여진 java application 으로 열쇠 쌍의 생성 및 X.509 증명서 발급, jar 파일 서명 등의 기능을 갖고있다.

Active X SDK 에는 열쇠 쌍 및 증명서를 생성하는 MAKECERT.EXE, X.509 증명서 및 비밀 열쇠를 PKCS#7 형식으로 변환하는 CERT2SPC.EXE 그리고 DLL,OCX, EXE,CAB 파일을 서명하는 SIGNKEY.EXE 등의 프로그램들이 포함되어 있다.

III. 공증사무소의 설계

증명서는 객체를 확인하는 수단이며, 공증사무소는 그 증명서를 발급하므로, 공증사무소가 모든 객체를 증명하는 역할을 한다고 볼 수 있다. 따라서, 체계적인 절차 없이 마구잡이로 증명서를 발급해 준다거나, 증명서의 관리가 허술하게 이루어진다면, 전체적인 보안은 혼란에 빠지게 된다.

본 장에서는 공증사무소 설계 고려사항에 대하여 논의한다. 크게 세가지로 나누어, 공증사무소 자체 보안 이슈와, 공증사무소를 처음 설치할 때 고려하여야 할 공증대상과 범위, 그리고 공증사무소의 절차에 관한 절차 이슈 등을 살펴본다.

1. 공증사무소 자체 보안 이슈

공증사무소에서 가장 보안에 신경 써야 할 것은 공증사무소의 비밀열쇠다. 만약 비밀열쇠가 유출되면, 마음대로 증명서를 발급할 수 있으므로 다른 악의적인 목적을 가진 객체들을 증명하거나, 위조 증명서를 만들 수 있기 때문이다. 따라서, 공증사무소의 비밀열쇠는 추측할 수 없는 크기여야 하며, 물리적, 논리적으로 보호되어야 한다.

또한, 증명서의 서명은 비밀열쇠를 통하여 이루어지므로 증명서의 서명 또한 물리적, 논리적으로 보호된 공간에서만 하여야 한다. 즉, 외부로 연결되어 있는 기기나, 여러 사람이 사용할 수 있는 위치에 설치된 기기에서 하지 않도록 하여야 한다. 또한, DOS 의 unerase 와 같이 지워진 파일도 복구될 수 있으므로, 증명서 서명에 관련되었던 기기 및 media 를 폐기처분 할 경우에는 자료를 깨끗이 삭제하여야 한다.

GTE 의 CyberTrust 의 경우 비밀 열쇠를 아예 Hardware 로 구현함으로써 비밀 열쇠의 유출을 방지하였다. 이 Hardware 는 전자파 차단까지 되어 있어서 Hardware 에서 나오는 전자파로 비밀 열쇠를 유추할 수도 없다고 한다. 또한, GTE 의 공증사무소는 2 명의 관리자가 합의하에 들어갈 수 있는 공간에 설치되어 철

저한 물리적 보안을 유지한다고 한다.

어떠한 실수나 사고에 의해서 비밀열쇠가 유출되는 사태를 방지하기 위하여, X.509 의 11 장 "Management of keys and certificates"에서는 서명을 off-line 으로 할 것을 권고한다. 즉, 모든 서명 절차는 네트워크에 연결되지 않은 기계에서 사람의 수작업으로 이루어져야 한다는 것이다. 그러나, 보안 요구 정도가 비교적 낮은, 많은 수의 객체를 증명하는 경우에도 수작업으로 하기는 곤란하다.[4]

Verisign 의 경우 실제 서명을 하는 부분은 방화벽(firewall) 안쪽에 설치함으로써 비밀 열쇠를 보호하는 방식을 도입하였다. 그림 1 에 그 구성을 보인다. 증명서의 신청은 Web server 인 Lifecycle server 들이 받고, 실제 서명은 Signing Manager 와 Digital ID database 에 의해서 이루어진다. Digital ID database 등 주요 정보를 가지고 있는 기기와의 연결은 Firewall 이 차단하며, Connection manager 가 적법한 사용자인지 판단한다.

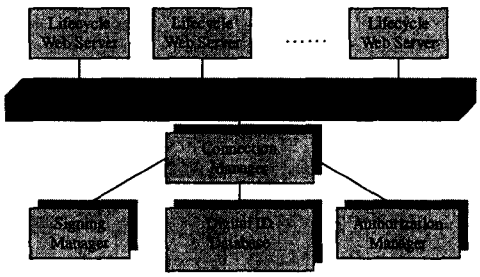

그림 1. Verisign 의 경우

공증사무소의 공개열쇠는, 증명서의 형태로 되어 있다면 위조가 불가능하므로, 공개하여도 무방하다. 도리어, 제 3 자가 임의로 열쇠 쌍을 생성하여 (이름은 같지만 공개열쇠가 다른) 가짜 증명서를 발급하여 혼란을 야기할 수도 있으므로 공증사무소의 증명서는 응용프로그램에 포함시키는 등의 방법으로, 안전하게 사용자에게 전달되어야 한다. 만약 그렇지 못할 경우, 증명서를 받아가는 방법과 간단한 확인 절차를 위해 지문(fingerprint) 값 등을 공개하는 것이 좋다.

기술적으로는 증명서의 서명이나 메시지의 서명이나 거의 다를 바가 없으므로, 단순히 공개열쇠만을 증명한다면 신청자가 자신의 서명을 남용할 우려가 있다. 따라서, 각 증명서는 다른 공증사무소의 증명인지, 사용자의 증명인지 그 용도가 확실히 증명서에 명시되어야 한다. 증명서에 명시되어 있으면 공개 열쇠 사용자, 즉, 신청자와 같이 통신하는 객체들이 서명의 책임한도를 알고 있으므로 적법하지 않은 용도로 사용한 증명서를 걸러낼 수 있다.

증명서의 용도를 표시하는 방법은 다음과 같이 3 가지 방법을 고려해 볼 수 있다. 참고로 이 3 가지 방법은 독립적이므로 3 가지 방법을 다 취할 수도 있다.

(1) 용도별로 공증 사무소를 따로 만든다.

열쇠 쌍의 용도별로 공증 사무소 증명서를 따로 두는 경우이다. 예를 들어, 서버 증명을 위한 공증 사무소와 사용자 증명을 위한 공증사무소를 다르게 두는 것이다. 이렇게 하면 증명을 확인하는 쪽에서는 용도별로 다른 공증 사무소의 증명서로 확인하므로 다른 용도로 사용된 것을 잡아낼 수 있다.

(2) X.509v3 확장을 사용하여 증명서에 명기한다.

1 번의 경우 증명대상자는 같은 열쇠 쌍에 대하여 필요한 용도만큼의 증명서를 가지고 있어야 하므로 관리가 불편하다. X.509v3 의 경우 확장 증명서 형식을 정의하고 있는데, 열쇠 쌍의 용도를 표현하기 위한 keyUsage 와 CA 증명서인가 아닌가를 표현하는 basicConstraints 라는 속성으로 증명서의 용도를 표현할 수 있다.

(3) 별도의 X.509v3 extension 을 만들어서 사용한다

X.509v3 의 표준 확장 형식은 일반적인 용도만을 제한할 뿐이다. 따라서, 어떤 응용프로그램을 중심으로 용도를 제한하고자 한다면, 별도로 X.509v3 확장을 정의하여 사용하는 것이 바람직하다. 이런 경우

ISO 로부터 OID(object identifier)를 부분적으로 부여할 수 있는 권한을 얻어야 한다.

많은 공증사무소의 경우 용도별로 공증사무소의 증명서를 분리한다. Verisign 의 경우 Class 1, 2, 3 별로 각각 주 공증사무소(Primary CA)를 따로 두었으며 그 밑에 용도별로 다시 공증사무소를 나누고, 다시 그 밑에 지역별로 공증사무소를 두는 형태를 취하였다.[11]

Netscape 의 경우 netscape-cert-type 이라는 별도의 X.509v3 확장형식을 지원한다. Netscape-cert-type 은 SSL client, SSL server, S/MIME user, Object Signing, SSL CA, S/MIME CA, Object Signing CA 의 7 개의 용도로 분류하여 제한한다.[6]

Thawte 의 경우 이 두 가지를 같이 지원한다. 용도와 심각도에 따라 5 가지로 공증사무소를 나누고 각 공증사무소에서 발급되는 증명서는 열쇠 쌍의 용도가 제한된다.

2. 공증대상과 범위

공증사무소를 운영하기로 결정했다면 실제 운영에 들어가기 전에 고려해야 할 것들이 있다. 먼저 무엇을 증명할 것인가, 즉, 증명대상의 범위를 명확히 구분함으로써 애매한 신용관계를 가지지 않도록 하여야 한다. 예를 들어, 서버를 증명할 것인지 E-mail 사용자를 증명할 것인지, 또는, 기관내의 정식 직원들만 증명해 줄 것인지, 어떤 사람이라도 신청만 하면 증명해 줄 것인지 확실히 결정하여야 한다.

또한, 증명 대상의 이름이 모호하지 않도록 하여야 한다. 모호한 이름은 언제든지 다른 객체로 오인될 수 있다. 예를 들어, A 라는 이름을 가진 동명이인의 B 씨와 C 씨가 있다고 하자. B 씨는 증명서 신청을 하지 않았는데, C 씨는 증명서 신청을 했다고 가정하면, 증명서 목록에는 C 씨의 증명서만 있을 것이다. 외부에 있는 D 씨가 B 씨에게 암호화 된 메시지를 보내고자 증명서 목록에서 A 라는 이름의 증명서를 찾으면 C 씨의 증명서가 검색된다. 이름 이외에 확인할 정보가 없는 D 씨는 그 증명서가 B 씨의 증명서라고 생각하고 C 씨의 공개 열쇠로 메시지를 암호화 해서 보낸다. 이렇게 되면, B 씨에게 갈 기밀 정보가 C 씨에게 유출된다.

공증사무소가 증명해야 할 증명대상의 범위는 다음과 같이 분류할 수 있다.

- 종류 : 공증사무소, SSL 서버, SSL 클라이언트, E-mail 사용자, Program 개발자
- 심각도 : 테스트용, 공개 증명용, 기밀용
- 범위 : 같은 부서 내, 같은 기관 내, 같은 지역 내

증명서의 심각도에 따라 CA 열쇠 쌍의 bit 수를 조절할 필요가 있다. 그러나 bit 수를 늘리면 늘릴수록 증명서 검사 시간이 길어지므로 적당히 조절하는 것이 필요하다. 일단 증명 대상의 범위가 결정되면, 해당 범위를 증명할 공증사무소를 만들고, 증명서의 종류에 따라 열쇠의 용도를 제한한다.

X.509 증명서는 증명 대상의 이름에 모호함을 주지 않기 위해서 subject 와 issuer 이름을 X.500 DN(distinguished name)을 사용한다. X.500 DN 을 정하기 위해서는 먼저 DIT(directory information tree) 구조를 확정하여야 하는데, 이것은 객체의 이름을 표기할 때 어느 수준까지 어떤 형식의 이름으로 표현할 것인지를 정하는 것이다. 예를 들어, 기관의 구성원을 표현할 때 기관의 이름까지만 표시할 것인지 아니면 소속 부서 level 까지 표시할 것인지, 기관이나 부서의 이름을 표시 할 경우 어떤 이름을 공식 명칭으로 쓸 것인지를 결정하는 것이다. 이 작업이 이루어지지 않으면, 증명 신청자가 DN 을 적을 때 혼동할 뿐 아니라, 유사한 이름으로 신청함으로써 한 객체에 2 개 이상의 증명서가 발급될 수 있다.

같은 부서에 소속된 동명이인과 같이 DIT 와 같이 계층적인 이름을 사용한 경우에도 해결할 수 없는 경우도 있는데, 이 경우에는 사원번호나 학번, E-mail address 와 같이 그 객체들을 구별할 수 있는 ID 를

DN에 포함시킴으로써 해결할 수 있다. 가장 이상적인 방법은 주민등록증 발급과 비슷하게, 증명대상이 될 수 있는 객체들의 목록을 체계적으로 관리하면서 그 목록에 있는 객체들에게만 증명서를 발급해 주는 것이다.

DN에 영문으로 표기하지 않고 한글을 직접 입력하는 경우도 고려할 수 있다. 이 경우 KSC-5601 code를 그대로 입력하는 것은 국제 표준에 어긋난다. 현재 구현되어 잇는 DN 부호화는 IA5String, 즉, ASCII code만 지원하는 것이 보통이기 때문이다. 원칙대로라면 Unicode, 즉, ISO-10646 BMP를 사용하여 BMPString으로 부호화 하거나 UCS-4를 사용하여 UniversalString으로 부호화 하여야 한다. 현재로서는 Unicode를 제대로 지원하는 소프트웨어가 그리 많지 않으므로 한동안은 영어로 표기하는 것이 좋다고 생각된다. 영어로 표기하면 전 세계적으로 증명서의 주인이 누구인지 알 수 있다는 장점도 있다.

3. 공증 절차 이슈

공증 사무소의 원칙을 정해졌다면, 다음은 실제 업무 수행 과정의 정의가 필요하다. 원칙만으로는 실제 업무를 수행하기에는 어려움이 있을 뿐 아니라, 수행과정이 명확하지 않다면 목적하는 바를 이루지 못하기 쉽고 문제가 심각한 경우 원칙도 무너질 수 있기 때문이다.

공증 사무소의 업무는 증명서 발급, 취소, 갱신, 조회의 4가지로 생각할 수 있다. 증명서 발급은 증명되지 않은 객체의 증명서를 발급하는 절차로서, 실체 증명을 필요로 한다. 증명서 취소는 비밀 열쇠의 분실, 도난이나, DN의 변경등에 따라서 증명서의 효력이 없어질 경우에 밟는 절차다. 증명서 갱신은 증명서의 유효기간이 지나서 새로운 증명서를 다시 발급 받는 경우이다. 증명서 조회는 공증 사무소가 증명해 준 객체의 증명서 목록을 조회하는 것으로, 암호화 된 메시지를 보내거나, 전자서명을 확인하기 위해 증명서가 필요할 경우 사용할 수 있도록 제공해 주는 것이다.

각각의 업무 절차를 결정하는데 다음과 같은 선택사항이 있을 수 있다.

- 본인 확인 방법
  - e-mail로 확인
  - 전화로 확인
  - 다른 증명서(예: 주민등록증, 신용카드)로 확인
  - 본인 직접 방문 확인
- 취소된 증명서의 처리
  - CRL을 정기적으로 download
  - 사용할 때 마다 증명서의 validity 검사.
- 갱신 조건
  1. 갱신 시기에 대해서: 기간 만료 전, 기간 만료 후
  2. 갱신 방법에 대해서: 자동으로, 사용자의 요청에 의해
  3. 갱신된 열쇠 쌍에 대해서: 새로운 열쇠 쌍으로만, 기존의 열쇠 쌍 허용

IV. KAIST 공증사무소의 설계 사례

본 장에서는 KAIST의 공증 사무소를 설치하는데 있어서 업무 절차를 설계한 사례에 대하여 논한다. 앞 절에서 논의한 고려사항을 중심으로 설명한다.

1. 주요의사 결정

증명대상은 다음 표와 같이 4 종류로 나누었다.

증명서 종류	응용 프로그램	Netscape-cert-type
서버 증명서	SSL web server 및 그 밖의 SSL 응용	SSL server
사용자 증명서	Netscape Communicator, Internet Explorer	SSL client, S/MIME
Java 개발자 증명서	Java applet 개발 도구	Object Signing
ActiveX 개발자 증명서	Active X control 개발 도구	Object Signing

Netscape Communicator 4.x 와 Internet Explorer 4.x 와는 서로 열쇠 쌍을 주고 받을 수 있으므로 같은 종류로 분류하였다. 그러나 Java 와 ActiveX 는 증명 대상인 프로그램이 기본적으로 다른 모양일 뿐더러 응용분야가 다르다고 판단되었기 때문에 다른 종류로 분류하였다. 또한, 현재까지 알려진 코드 서명도구끼리는 비밀열쇠를 전송할 방법이 없다는 것도 이 결정에 영향을 끼쳤다. 비밀열쇠를 전송하지 못하면, 한 사람 혹은 기관에서 양쪽 다 증명서가 필요할 경우 하나의 공증 사무소에서 같은 이름으로 두개의 증명서를 나누어 주어야 한다는 점이 문제가 되기 때문이다.

사용자 증명서와 개발자 증명서를 하나로 하는 문제도 고려하였다. 그러나, 사용자 증명서의 경우 증명대상이 자연인이라는 것이 확실한 데 비해, 개발자 증명서는 자연인일수도, 소프트 웨어 개발 단체일 수도 있기 때문에 분리하기로 하였다.

X.509v3 의 표준 확장을 추가하는 것도 고려하였으나, X.509v3 가 아직 확정되지 않은 상태인데다가 OID 가 자꾸 바뀌는 등 실제 사용에는 도움을 주지 않는다는 결론을 내라고 포함하지 않기로 하였다.

증명서의 보안 단계는 다음과 같이 3 단계로 나누었다.

보안단계	용도	실체 확인 방법	CA 증명서 bit 수	증명서 유효기간
Level 0	시험용	e-mail 확인, 자동	512 bit	3 개월
Level 1	공개용	전화 확인, 서약서	1024 bit	2 년
Level 2	기밀용	본인 방문 확인	1024 bit	1 년

또한, 증명서의 오용을 막고 용도별 증명서의 관리를 따로 하기 위하여 그림 2 와 같이 2 단계의 공증 사무소 구조를 만들었다. 단, Java 와 ActiveX 의 코드 서명 도구는 심각도가 높으므로 level 0 를 제외하였다.

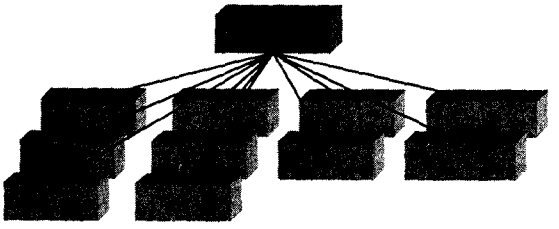

그림 2. KAIST CA 의 2 단계 증명 구조

DIT Structure 는 다음과 같이 정하였다. DN 의 내용은 ASCII code 범위 내에서 모두 영문으로 기록하도록 하였다.

종류	RDN 구조
CA 증명서	C="KR",O="KAIST",CN=CA name
사용자 증명서	C="KR",O="KAIST",OU=부서이름,CN=사용자명, Email=mailid@cais.kaist.ac.kr
서버 증명서	C="KR",O="KAIST",OU=부서이름,CN=hostname.kaist.ac.kr
개발자 증명서	C="KR",O="KAIST",OU=부서이름,CN=개발자명,Email=mailid@cais.kaist.ac.kr

사용자와 개발자 증명서의 경우 동명이인을 막기 위하여 Email address 를 추가하였다. 서버의 경우 hostname 이 unique 하므로 별도의 ID 를 추가할 필요가 없다고 판단하였다.

2. 주요 절차 분석 및 설계

공증사무소의 업무는 증명서의 발급, 갱신, 취소, 조회로 나눌 수 있다. 여기서 우리는 업무 분석을 위하여 UML(Unified Model Language)를 사용하였다. [7] 그림 3에 전반적인 업무에 관한 Use case diagram을 보였다.

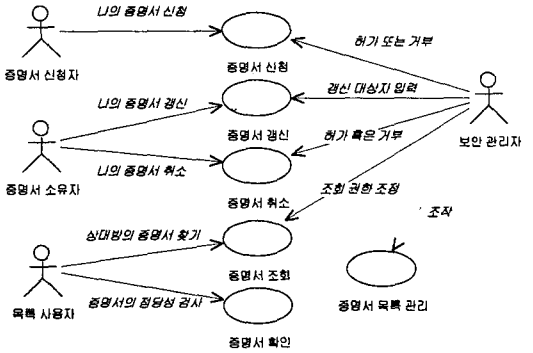

그림 3. 공증 사무소 업무의 Use case diagram

공증사무소 업무에 참여하는 actor로는 신규 증명서를 발급 받고자 하는 증명서 신청자, 이미 증명서를 발급 받는 증명서 소유자, 증명된 사용자와 통신하면서 증명서의 정당성 여부를 확인하는 목록 사용자 그리고 전체 시스템을 관리하는 보안 관리자로 나누었다.

Use case는 공증 사무소의 업무인 증명서의 발급, 갱신, 취소, 조회를 기준으로 나누었다. 조회의 경우 정당성 검사와 목록 조회를 다른 기능으로 보고 Use case를 분리하였다. 전체 증명서의 목록을 관리하는 use case를 관리자를 위하여 추가하였다.

각 업무에 대한 절차는 UML의 Activity diagram으로 표현할 수 있다. 그림 4에 증명서 발급과 갱신 과정을 swim lane 형태로 나타내었다.

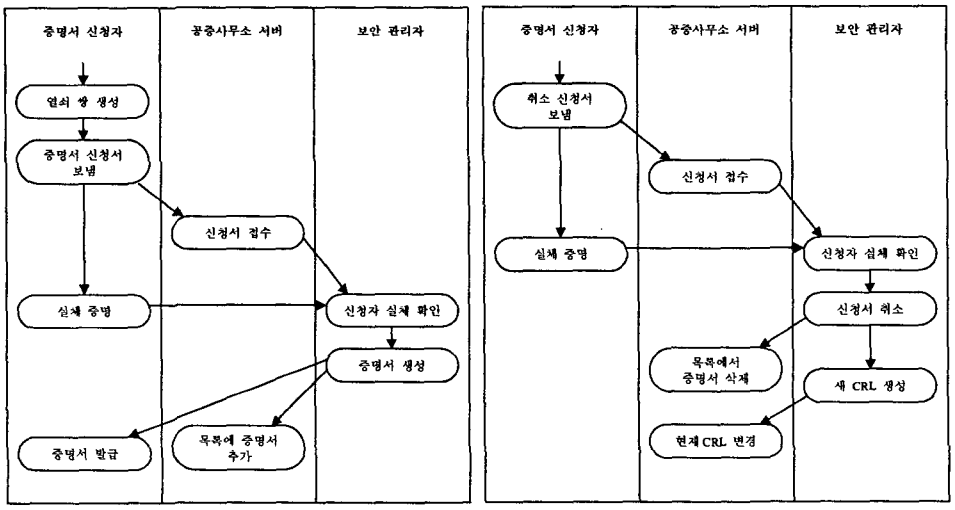

그림 4. 증명서 신청과 취소의 activity diagram

증명서 발급과정을 살펴보면, 사용자가 열쇠 쌍을 생성한 후 공개열쇠와 함께 공증 사무소 서버에 보내면, 관리자가 실체확인을 한 후 증명서를 발급하는 순서를 따르고 있다. 증명서 발급 시 목록에도 추가함으로써 목록 사용자가 조회할 수 있도록 배려한다. 실체확인은 증명서의 심각도에 따라 다른 방법을 사용하지만 전체적인 업무 절차는 동일하다. 증명서 취소과정에서도 실체 확인을 하도록 하였다. 이것은 신원 사취를 위해 이미 존재하는 증명서를 취소한 후 위조 증명서를 사용할 경우에 대비한 것이다. 또한, 취소 즉시 목록에서 삭제하고 CRL을 갱신하여 해당 증명서가 취소되었음을 알린다.

V. 공증사무소 서버의 구현

공증사무소를 운영할 정책과 절차와 함께 그 절차를 수행할 적절한 컴퓨터 시스템이 필요하다. KAIST 의 경우 공증사무소 서버를 별도로 구현하였다.본 장에서는 KAIST 공증사무소 서버의 구현에 대해 언급한다. 앞장에서 언급한 공증 사무소 서비스를 제공하기 위해 어떠한 기능에 중점을 두었는지 언급한다.

1. 전체적 구조

대부분의 증명서 서비스가 web 상에서 이루어지므로 관리자의 interface 를 제외하고는 전부 web page 로 구현하였다. 그림 5 에 전체적인 구성을 나타내었다. 증명서의 신청과 갱신, 취소, 조회 등의 업무를 받는 부분은 web server 이지만, 실제적인 증명서의 발급은 네트워크에 연결되지 않은 별도의 컴퓨터에서 이루어진다. 각종 신청서는 디스켓이나 serial line 과 같이 접수 서버에서 관리자의 컴퓨터로 연결할 수 없는 방법으로 전송하며, 발급된 신청서 역시 마찬가지이다.

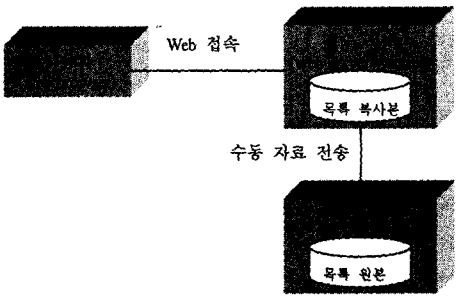

그림 5. KAIST 공증사무소 서버의 구성

개발 환경은 Unix 운영체제 상에서 PHP/FI 라는 server side script 로 1 차 개발을 하였다. 2 차 개발은 C 언어로 CGI 를 구현하는 것으로 현재 추진중이다. 접수 서버로는 Apache-SSL 을 사용하였다. Netscape 이나 Microsoft 의 서버로는 512 bit 이상의 RSA 공개열쇠를 사용할 수 없기 때문에 Apache-SSL 을 선택하였다. 같은 이유로 암호화 소프트웨어는 SSLeay 를 사용하였다. 현재 구현된 내용으로는 SSL server 증명서 및 Netscape Navigator 3.0, Communicator 4.0, Internet Explorer 3.0 의 증명서 발급이 가능한 상태이며, Java 와 Active X code signing 부분이 구현중이다.

2. 특징

본 절에서는 구현한 공증 사무소 서버의 특징을 설명하겠다.

(1) 조직 데이터베이스와의 연계를 통한 보안성 향상

대부분의 CA server 는 DN 을 신청자가 직접 입력하게 되어 있으나 KAIST 의 경우는 교수, 학생, 직원의 목록이 이미 인사, 학사 데이터베이스로 구축해 놓았으므로, 사용자가 일일이 DN 의 내용을 기입할 필요 없이, 고유번호만 입력하면 DN 을 채워주게끔 하였다. 이것은, DIT 를 조직 데이터베이스와 일관성 있게 유지하면서도 사용자에게 편리함을 제공하는 일석이조의 효과를 낳았다.

(2) 512 bit 이상의 증명서 발급

미국에서 만들어진 소프트웨어들은 ITAR(무기 수출 금지 규정)에 의해서 RSA 512 bit, RC2, RC4 40bit, DES 56 bit 초과하는 암호화가 가능한 소프트웨어를 수출할 수 없다. 따라서, Netscape Certificate server 나 Internet Information Server 4.0 역시 512 bit 이상의 증명서를 발급할 수 없다. 그러나 KAIST 공증 사무소 서버는 SSLeay 를 사용하여 그 이상의 bit 수로 만들어진 증명서도 발급 할 수 있도록 하였다.

(3) Netscape certificate extension support

Netscape 은 Navigator 3.0 부터 자사의 X.509 v3 extension 을 지원한다. KAIST 공증사무소 서버는 Netscape 의 확장 증명서 방식을 이용하여 증명서에 필요한 정보들을 추가하였다. Netscape-comment 에 증명서의 보안 수준에 관한 안내문을 추가하는 한편, netscape-ca-policy-url 에 접수서버의 안내 URL 을 추가함으로써 사용자의 편의를 도왔다. netscape-cert-type 확장을 지원함으로써 증명서의 오용을 막았다. 또한, 서버

증명서의 경우 netscape-ssl-server-name 에 증명될 서버의 이름을 입력함으로써, 다른 서버의 증명서로 둔갑하는 경우를 막았다.

(4) 취소 확인

netscape-revocation-url 과 netscape-ca-revocation-url 에 증명서의 정당성을 검사하는 CGI program 의 URL 을 줌으로서 취소된 증명서인지를 항상 확인 할 수 있게 하였다.

VI. 결론 및 추후연구

본 논문에서는 공증사무소의 운영에 관련된 전반적인 사항을 살펴보았다. 먼저, 암호기반 보안의 기초기술과 및 응용기술을 살펴보았다. 또한, 암호기반 보안기술을 실제로 응용한 소프트웨어들을 살펴보았다. 공증사무소를 운영함에 있어서 필요한 고려 사항들을 살펴보고, KAIST 공증사무소의 설계 및 구현에 대해서 소개했다. 공증사무소 서버를 구현하고 그 특징에 대해 이야기했다. 특히, 한글을 사용하는 문제에 대해서도 다루었다.

현재는 1 차적인 구현만 되어 있지만 교내 전체 증명 업무를 수행함으로써 실제적인 문제점을 찾아보고자 한다. X.509 증명서를 지원하는 다른 응용프로그램에 대해서도 서비스를 확장할 생각이며, PGP 와 같은 다른 암호화 프로그램과의 연계도 고려중이다. 또한, 요즘 각광 받고 있는 LDAP 을 사용하여 목록 서비스를 추가함으로써 인터넷과의 전체적인 연계를 계획중이다.

참고 문헌

[1] Dean, D., Felten, E. W., Wallach, D. S., "Java Security: From HotJava to Netscape and Beyond" In Proceedings of 1996 IEEE Symposium on Security and Privacy, May 1996.

[2] Dusse, S., Hoffman, P., Ramsdell, B., Lundblade, L., Repka, L., "S/MIME Message Specification", Internet Draft, Oct. 1997.

[3] Freier, A. O., Karlton, P., Kocher, P. C., "The SSL Protocol – Version 3.0", Internet Draft, Nov. 1996.

[4] ITU-T Recommendation X.509: "The Directory - Authentication Framework", 1993.

[5] ITU-T Recommendation X.520: "The Directory - Selected attribute types", 1993.

[6] Jeff Weinstein, "Netscape Certificate Extensions Communicator 4.0 Version", Netscape Draft, Aug. 13, 1997.

[7] Rational Software Co., "UML Notation Guide – Version 1.1", Rational Software, Sep. 1, 1997.

[8] RSA Laboratories, "PKCS #7: Cryptographic Message Syntax Standard, Version 1.5", November 1993.

[9] RSA Laboratories, "PKCS #10: Certification Request Syntax Standard, Version1.0", November 1993.

[10] RSA Laboratories, "PKCS #12: Personal Information Exchange Syntax Standard, Version 1.0", DRAFT, April 1997.

[11] Verisign, Inc., "Verisign Certification Practice Statement – Version 1.2", Verisign, May. 30, 1997.