Linux 기반 공개키 인증시스템 설계 및 구현

반용호*·홍주형*·김종훈*
동아대학교 컴퓨터공학과

yongho@spring.donga.ac.kr

The Design and Implementation of Linux Based Public-Key Certification System

Yong-Ho Ban*·Joo-Hyung Hong*·Jong-Hoon Kim*
*Dept of Computer Engineering, Dong-A University

요약

PKI(Public Key Infrastructure)는 공개키 암호 어플리케이션에 사용되어지는 공개키의 유효성을 보장하기 위해 공개키에 대한 전자인증서의 발행과 획득, 조회, 검증을 위해 사용되는 인증서 관리 기반구조를 말한다. 본 논문에서는 인트라넷 환경에서의 PKI 응용 서비스에 적용 가능한 공개키 인증 시스템을 Linux를 기반으로 설계하고 그 prototype을 구현하였다. 인증 시스템은 Root CA와 하위 CA 및 Client로 구성되어 있고, 인증서 발행, 갱신, 폐지 등의 기능을 수행한다. 제안된 시스템의 구현 환경은 Linux를 기반으로 하는 Apache Web-Server와 암호 라이브러리는 SSL을 기반으로 하는 Openssl을 사용하였다.

1. 서론

최근 인터넷의 발전속도는 엄청난 가속력을 가지고 빠르게 진행되고 있다. 그 결과 과거에는 상상조차 할 수 없었던 정보서비스의 편리함과 효율성 그리고 신속성을 제공받고 있는 것이 사실이다. 그러나 인터넷의 출현 및 발전과정의 개방적인 성격으로 인해, 정보보호보다는 정보공유에 치중함으로써 네트워크 상에서 운용되고 있는 많은 정보들에 대한 보안상의 문제점들이 최근 중요한 사회문제로 인식되고 있으며, 더욱이 인터넷이 단순한 정보교환에서 가치를 상호 교환하는 상거래에 확대 적용됨으로써 가치 있는 정보나 프라이버시 보호와 같은 요구사항이 점차 증대되고 있다.

전자상거래 등과 같은 인터넷 응용 기술의 안전성과 신뢰성을 확보하기 위해 인증, 무결성, 부인봉쇄 등의 서비스는 전자서명 기술을 활용함으로써 해결 가능하다. 전자서명 기술을 효과적으로 이용하기 위해서는 공개키 암호 방식이 필요하며, 공개키 암호 방식을 이용한 인증 방법을 구현하기 위해서는 기술적, 제도적 기반이 요구되는데 이를 공개키 기반구조(PKI : Public Key Infrastructure)라고 한다. PKI를 구축함으로써 암호키 갱신, 복구, 위탁 등과 같은 키 관리, 인증서 생성 및 취소 관리, 그리고 인증 정책 관리와 같은 서비스의 제공이 가능하다. 또한 PKI는 전자상거래 뿐만 아니라 전자우편, FTP, Telnet 등과 같이 네트워크 상에서 운용되는 모든 데이터의 보호를 위하여 필요한 기반체계로서, 특히 인터넷과 같은 개방 환경에서는 암호키 관리 체계 없이는 데이터의 안전한 유통을 보장할 수 없다. PKI에 의해서 구현되는 계층적 인증 구조나 상호 인증 방식에 의하여 조직간의 안전한 데이터 통신을 위한 신뢰 관계를 형성할 수 있게 된다.[1)]

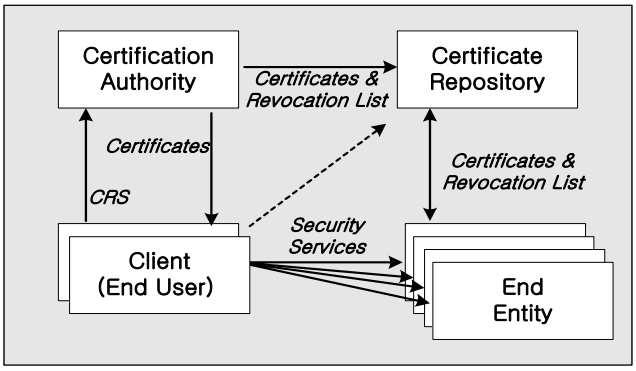
본 논문에서는 PKI를 구축하는데 있어 중추적인 역할을 담당하게 되는 공개키 인증 서버를 Linux

기반에서 설계하고 prototype을 구현하였다. 본 논문의 구성은 다음과 같다. 2절에서는 PKI를 구성하는 각 객체들의 요구사항 및 객체의 특징을 분석하고, 3절에서는 PKI 구축에서 중추적인 역할을 담당하는 공개키 인증시스템의 전체적인 구조를 소개한다. 4절에서는 제안된 시스템의 동작 환경 및 기능을 검증하고, 마지막으로 결론 및 향후 연구방향을 제시하였다.

2. 관련 연구

2.1 PKI 구성요소 및 요구 사항 분석

본 절에서는 PKI 구성객체들의 운용을 위해 요구되는 사항들을 기술한다. PKI는 인증서의 발급, 사용 및 취소와 관련된 서비스를 제공하며, PKI 환경을 구축하는 주요 객체는 인증기관(CA - Certification Authority), 인증서 저장소(Certificate Repository), 최종 사용자(End-User or Client)[9] 그리고 서비스 제공 주체(e-commerce services 제공자)로 구분된다. [그림 1]은 PKI 구성객체의 기본적인 트랜젝션을 보여준다.



[그림 1] PKI 기본 구성 객체

인증 기관(CA: Certification Authority) : 사용자에게 인증서를 발급하거나 취소하고 공개키 인증서와 인증서 취소 목록(CRL:Certificate Revocation List)을 공개된 장소에 공고하는 역할을 담당한다. CA에서 요구되는 기능을 요약하면 다음과 같다.[1,2,3,11]

- 전자서명의 생성기능
- 인증서의 생성 및 확인
- 인증서의 발급 및 재발급
- 인증서 소유자에 대한 정보 관리
- 사용자의 이름 및 공개키의 유일성 확인
- 디렉토리 서버 관리
- 인증서 취소 목록(CRL) 관리
- 자료의 백업

CA의 기능 중 발행된 인증서 및 CRL을 실시간적으로 인증서 저장소에 공고 할 수 있는 기능이 요구되며. 인증서 발행은 사용자가 직접 공개키를 생성하여 사용자의 정보와 해당 공개키에 대한 인증을 요구하는 SPKAC 형태 및 PKCS#10 형태의 인증서를 모두 발급 가능해야 한다. CA는 항상 CA 인증서와 상호 인증서 쌍과 CRL을 공개 할 수 있어야 한다. 또한 CA들은 계층적 신뢰 모델에 기반을 둔 PKI를 지원하기 위해 상위 CA로부터 인증서를 요구할 수 있어야 한다.[10]

인증서 저장소(Certificate Repository) :디렉토리 서비스(Directory Service)라고도 하며, 인증서와 인증경로를 찾기 위해서 사용된다. CA에 의해 발행된 공개키 인증서와 CRL을 저장하고 열람할 수 있도록 검색 기능을 제공해야 한다. 보통 X.500 표준에 따른 고유 이름이 이용되며, 디렉토리에 대한 접근에는 LDAP[4](Lightweight Directory Access Protocol)가 사용된다. 최상위 CA를 포함한 모든 CA는 디렉토리를 직접 관리하거나 어느 한 디렉토리에 연계되어야 한다. 각각의 CA는 자신의 인증서를 Directory에 게시하며, 동일한 디렉토리를 사용하는 CA들은 취소된 인증서에 대한 CRL을 실시간으로 공고해야 한다.[1,3]

사용자(Client ,End-User): PKI 구성 객체중에서 Client가 요구하는 기능은 전자서명을 검증하고, 인증서 저장소에서 인증서와 CRL을 획득 할 수 있어야 한다. 최종 사용자측에서 요구되는 기능은 다음과 같이 요약 할 수 있다.

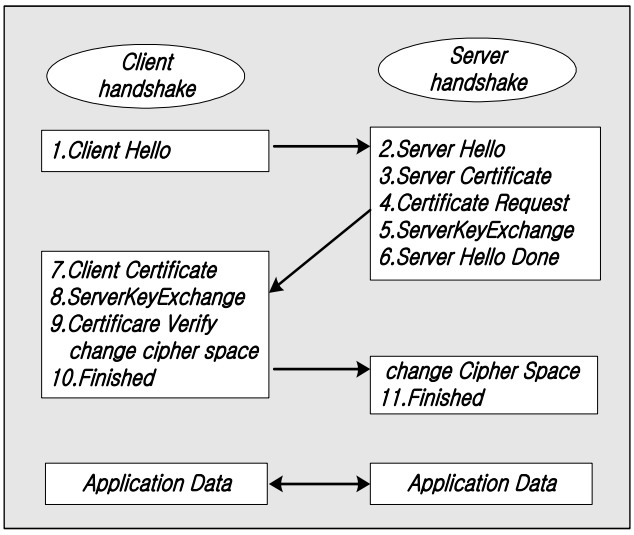
- 전자 서명키 생성기능
- 인증서 설치 기능
- CRL 설치 기능
- 인증서 검증 기능

인증서(Certificate) : 인증서는 공개키의 합법성을 보증하는데 이용된다. 서명을 확인 하는 사람은 인증서의 서명을 확인하여 서명에 위조나 변조가 없다는 사실을 확인한다. 현재 공개키 인증 시스템에서 사용되는 표준은 ITU-T X.509[5] 표준에 의해서 정의 된다. 본 논문에서 구현된 인증 시스템의 인증서 양식은 ITU-T X.509v3[6] 형식을 따르고 있다.

인증서 폐지목록(Certificate Revocation List) : 예정된 유효 기간의 만기일이 도래하기 전에 취소된 인증서에 대한 정보를 인증서 취소 목록이라 한다.[5)]

2.2 공개키 인증 시스템 구현 관련 기술

SSL : SSL[7,8)](Secure Scoket Layer)은 응용 프로토콜과 TCP/IP사이에 위치하며 데이터의 암호화, 서버의 인증, 메시지의 무결성을 제공하며, 서버에 대한 인증은 반드시 수행되지만 클라이언트에 대한 인증은 선택적으로 수행 할 수 있도록 해준다. SSL은 서버와 클라이언트 양쪽의 TCP/IP 연결을 위해서 핸드쉐이크(handshake) 프로토콜을 수행하여 양쪽은 암호화 통신에 합의하고, 암호화 통신과 인증에 필요한 값들을 초기화한다. 초기화 후, SSL은 응용 프로토콜에서 생성해 낸 바이트 스트림의 암호화와 복호화만 수행하게 된다. 즉 HTTP 요청과 HTTP 응답에 포함되는 모든 정보들이 암호화되어 전송됨을 의미한다. [그림 2]는 핸드쉐이크 프로토콜의 수행과정을 보여준다.

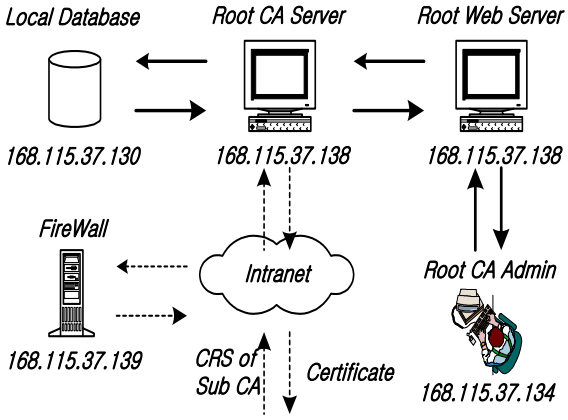


[그림 2] SSL 핸드쉐이크 프로토콜

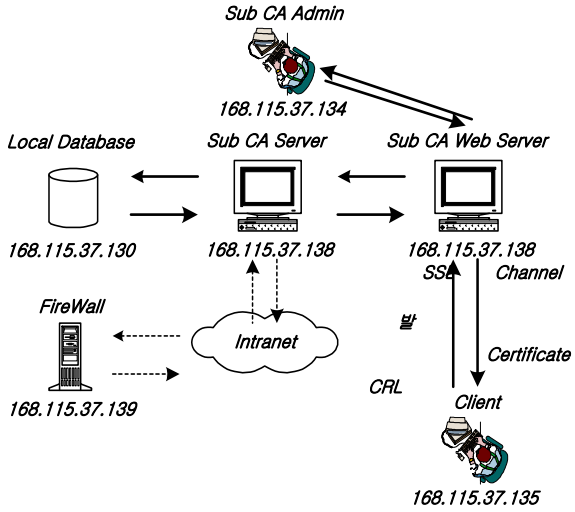
3. Linux 기반 공개키 인증시스템

3.1 시스템 구성 및 CA의 초기화

본 논문에서 제안된 인증 시스템의 전체적인 구성은 [그림 3]과 [그림 4]와 같다. 본 논문에서 제안된 인증시스템은 Root CA, 하위 CA 시스템, Client로 구성된다. Root CA는 Self-sign 형태로 초기화한다.



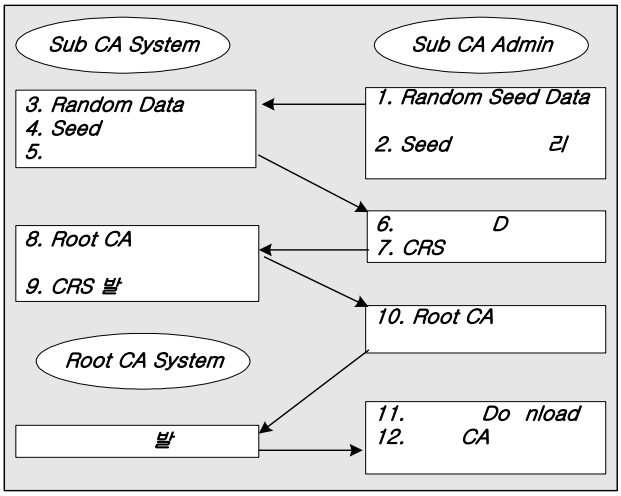
[그림 3] Root CA 시스템 구성



[그림 4] 하위 CA 시스템 구성

하위 CA는 Root CA로부터 공개키에 대한 인증서를 발급 받는 계층구조 형태의 인증서 체인을 갖게 된다. 인증 시스템의 초기화를 위한 과정은 다음과 같다. 먼저 임의의 난수를 선택하여 키를 생성하고, Root CA로부터 인증서를 발급받기 위한 CRS(Certification Request Syntax Standard)를 요청하여 CRS를 발급 받는다. 발급 받은 CRS를 Root-CA로 보내 인증서 신청을 요청하고, 인증서 생성을 요청받은 Root CA는 인증서 발급 여부를 결정한다. 하위 CA관리자는 인증서를 발급 받아 하위 CA에 설치 한다. 인증서 발급이 완료되면 발급된 인증서의 상태정보 및 유효성을 검증하는 수단인 CRL(Certification Reoked List) 발행한다. Root CA의 초기화는 자신의 공개키를 자신의 비밀키로 서명하는 형태를 가지게 된다. [그림 5]는 인증기관의 초기화 과정

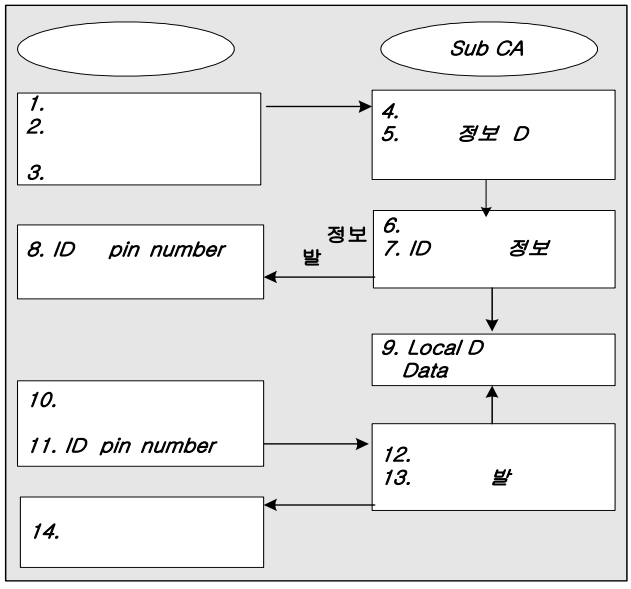
을 보여준다.



[그림 5] CA의 초기화 과정

3.2 인증서 발급신청 및 발급

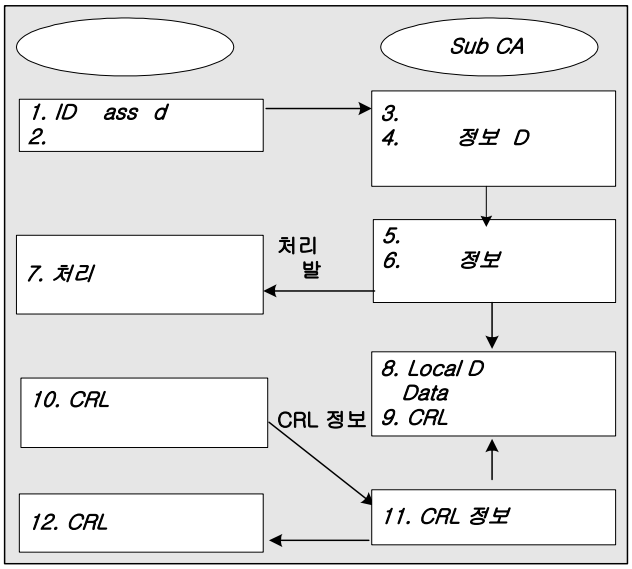
인증서 발행 신청은 사용자가 직접 공개키를 생성하여 사용자의 정보와 해당 공개키에 대한 인증을 요구하는 SPKAC 형태 및 PKCS#10 형태의 인증서를 모두 발급 가능하다. 사용자가 인증서 신청을 발급하는 절차는 다음과 같은 트랜젝션을 거친다. 사용자가 작성한 인증서 요청양식과 공개키를 SSL-channel을 통해 CA에 전달한다. CA에서는 신청서 내용을 검증하고 인증서 발급 여부를 결정하고 발행 여부 결과를 사용자에게 메일로 전송한다. 인증서 생성 관련 정보는 database에 저장된다. 사용자는 CA에서 전송한 ID를 pin_number를 이용하여 CA에 접속하여 인증서 다운로드를 요구하여 인증서를 발급받아 사용자의 시스템에 설치한다. [그림 6]은 사용자의 인증서 발급 신청과정을 보여준다.



[그림 6] 인증서 발급 과정

3.3 인증서 폐기 과정

사용자에게 발급된 인증서를 폐기하기 위한 과정은 다음과 같다. 폐기하고자 하는 인증서를 선택하고, CA에 인증서 폐기를 요청한다. CA는 인증서 폐기 요청을 검증하고, 해당 정보를 database에 저장한다. 사용자의 정보와 인증서의 정보를 검증하여 폐기 신청이 유효하다면 인증서를 폐기하고, 해당 정보를 CRL에 등록하고, 결과를 사용자에게 전달한다. 사용자는 최종 갱신된 CRL을 CA로부터 전송받아 설치하고 CRL을 확인한다. [그림 7]은 인증서 폐기과정을 보여준다.



[그림 7] 인증서 폐지 신청 및 CRL 생성

3.4 인증서의 재발급 및 갱신

인증서의 갱신은 다음과 같은 경로를 거친다. 먼저 사용자는 갱신을 원하는 인증서를 선택하고, CA에게 인증서 재발급 또는 갱신을 요청한다. CA는 해당 요청에 대한 유효성 및 인증서 재발급 · 갱신 내역을 검토하고, 인증서의 재발급 · 갱신 여부를 결정하고, 재발급 · 갱신이 확정되면 새로운 인증서 발급을 위하여 사용자에게 발급에 필요한 데이터를 요구한다. 사용자는 새로 작성된 인증서 등록 양식과 공개키를 CA에게 전송한다. CA측 인증 시스템은 해당 사용자의 기존 인증서를 폐기하고, 신청된 결과를 처리하여 그 결과를 메일을 통해 전달한다.

3.5 사용자측 소프트웨어

본 논문에서 구현한 시스템의 사용자측 소프트웨어는 Netscape사에서 개발된 Communicator 4.71을 사용하였다. Communicator 4.71은 전자 서명키 생성, 인증서 설치, CRL 설치, 인증서 검증 기능, 서명기능을 모두 지원한다. Communicator 4.71에서 전자서명키를 생성하는 태그는 다음과 같이 구성된다.

```
<html>
<head> </head>
<body>
<form name="test"
action="https://168.115.37.138/cgi-bin"
method="GET">
<KEYGEN TYPE="hidden" NAME="public_key"
VALUE="">
<input type=submit>
</form> </body> </html>
```

4. 시스템 구현 환경 및 실행 화면

본 논문에서 제안된 시스템의 구현 환경은 다음과 같다. 본 논문에서 제안된 CA측 시스템은 Linux를 기반으로 apache web-server를 사용하였으며 암호 라이브러리는 SSL을 기반으로 하는 openssl을 사용하였다. 사용자측 응용프로그램은 Netscape사의 Netscape Communicator 4.71을 사용하였다. [표 1]은 시스템의 구현 환경에 사용된 하드웨어 환경 및 라이브러리를 보여준다.[12,13,14)]

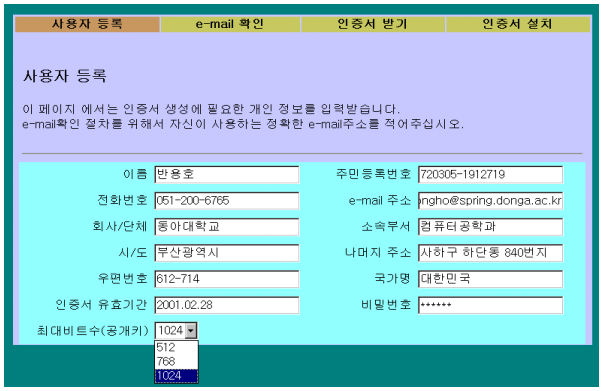
	Root CA	하위 CA	Client
운영체제	Linux 2.77	Linux 2.77	window95/98
하드웨어	Pentium II 300	Pentium II 300	Pentium II 450
컴파일러	gcc 2.8.1	gcc 2.8.1	-
라이브러리	Openssl 0.9.4 Open LDAP	Openssl 0.9.4 Open LDAP	-
database	mysql 3.22	mysql 3.22	-
Application	Apache Web Server 1.3.11	Apache Web Server 1.3.11	Netscape Communicator 4.71

[표 1] 개발 환경

[그림 8]은 본 논문에서 구현한 시스템의 인증서 발행을 위한 초기화면이다. 인증서를 발급 받기 원하는 사용자는 화면 중앙에 위치한 인증서 발급 버튼을 눌러 사용자가 직접 생성한 키 또는 인증 서측에서 생성한 공개키 모두에 대해 인증서를 발급받을 수 있다.

[그림 8] 인증서 발급 신청 초기화면

사용자는 [그림 9]와 같은 양식에 인증서 발급에 요구되는 사항들을 입력하여 인증서 발급 신청을 한다. [그림 9]는 브라우저용 인증서를 발급받기 위해 사용하는 양식이다. 데이터 입력이 완료되면 화면 하단에 위치한 개인키 생성 버튼을 클릭하면 입력한 정보가 CA 서버 측으로 전달된다. 사용자는 개인키 길이를 512, 768, 1024bit중 하나를 선택한다.

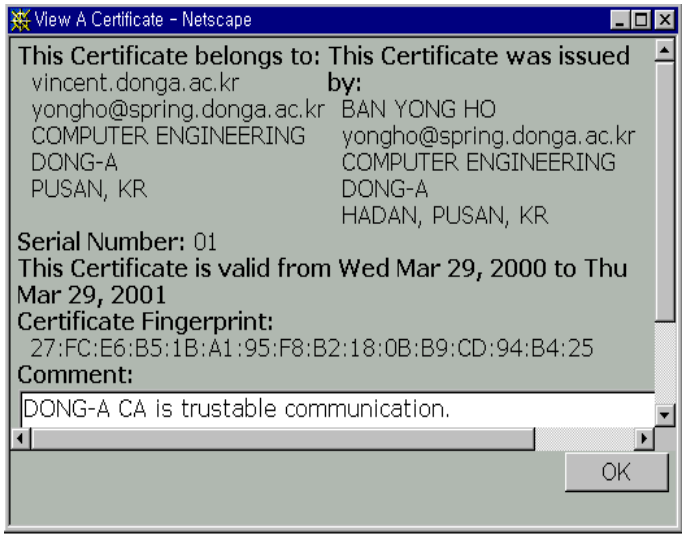


[그림 9] 사용자 등록화면

[그림 10]은 Root CA의 초기화를 위해 Root CA 자신의 자체 인증서를 발급한 것을 보여준다. 인증서 발급 내용을 살펴보면 인증서 발급자와 발행 대상 객체가 동일함을 알 수 있는데, 이는 Root CA의 공개키를 CA 자신의 전자서명 생성키로 자체 서명(Self-Sign)한 것임을 알 수 있다.

[그림 10] Root CA의 Self-Sign된 인증서

[그림 11]은 web-server 초기화를 위해 Root CA가 해당 web server에 대해 발급한 인증서를 보여준다. 인증서 발급 내용을 살펴보면 인증서 발급자와 발행 대상 객체가 서로 상이함을 알 수 있다.



[그림 11] web server에 대한 인증서

5 결론 및 향후 연구방향

본 논문에서는 PKI를 구축하는데 있어 중추적인 역할을 담당하게 되는 공개키 인증 서버를 Linux 기반에서 설계하고 prototype을 구현하였다. 본 논문에서 설계한 시스템은 기본 인터페이스를 웹 환경을 그 기반으로 하고 있기 때문에 인증서를 신청하는 사용자와 관리자에게 운용상의 장점을 제공할 수 있다.

향후 연구 과제로는 사용자측 응용 소프트웨어를 기존에 개발된 브라우저를 사용함으로서 특정 사용자에게 운용상 제약을 제공할 수 있으므로, 범용으로 사용할 수 있는 사용자측 소프트웨어의 개발이 요구되며, Root CA와 하위 인증 시스템을 하나의 시스템에 구성하였으나 각 CA를 서로 분리할 필요성이 있으므로, 이에 대한 연구가 필요하다.

참고문헌

1. “전자상거래를 위한 보안 기술 체계 및 요소기술에 대한 이해”, 한국전산원 차세대 서비스부, 1999.6
2. “PKI 구성 객체의 상호연동을 위한 명세서 분석”, 한국정보보센터 기술연구팀“ 1998.7
3. www.id2tech.com/whitepaper/largescalepki.asp
4. http://www.openldap.org/
5. http://www.itu.int/itudoc/itu-t/rec/obsolete/x/x-500up/x509.html
6. http://www.itu.int//itudoc/itu-t/rec/obsolete/x/x-500up/index.html
7. SSL 3.0 SPECIFICATION, Http://home.netscape.com/eng/ssl3/3-SPEC.htm
8. http://www.openssl.org
9. “CA-Browsing System - A Supporting Application for Global Security Services”, ISOC Symposium on Network and Distributed System Security, San Diego, Feb. 94, pp. 123-128
10. http://www.rsa.com/rsalabs/pkcs/
11. Recommendation for a Unique Identifier for X.500 Distinguished Names. Dated 4 March 98
12. http://www.apache.org/library/
13. http://www.php.net/docs.php3
14. http://www.openldap.org/foundation/