情報保護學會論文誌
第14券 第3號, 2004. 6

코드 서명 기술의 국내 PKI 적용 방안 비교 연구

이 래†, 이 동 훈‡
고려대학교 정보보호대학원

Research on Applying Code Signing Technology to National PKI

Rae Lee†, Dong Hoon Lee‡
Center for Information Security Technologies (CIST)

요 약

오늘날 많은 웹 페이지들이 제한적인 정보의 제공에서 벗어나 ActiveX Control이나 Java Applet과 같은 응용프로그램을 사용자 컴퓨터에 다운로드하게 하여 다양한 서비스를 제공하고 있다. 이러한 과정에서 인터넷을 통해 다운로드되는 소프트웨어에 대한 무결성과 배포자에 대한 신원 확인을 해주는 코드 서명 기술이 필요하게 되었다. 본 논문에서는 대표적인 코드 서명 기술인 Microsoft사의 Authenticode기술에 대하여 분석하고 국내 공개키 기반구조(PKI)에 적합한 코드 서명용 인증서 프로파일을 제안하며, 코드 서명 기술을 국내 PKI에 적용하기 위한 방안과 문제점을 제시한다.

ABSTRACT

Nowadays most web pages provide various services by downloading the applications program such as ActiveX Control or Java Aapplet. To provide code integrity and publisher authentication of downloaded software in internet, we need code signing technology. In this paper, Authenticode technology of Microsoft is first analyzed. Based on the analysis, we propose code signing certificate profile and applying method for National Public Key Infrastructure.

Keywords : *Code Signing, Authenticode, PKI, Code Security*

Ⅰ. 서 론

1990년대부터 인터넷 산업이 발전하면서 많은 웹 사이트들이 생겨났고 사용자들은 WWW(World Wide Web) 서비스를 통해 이러한 웹 사이트에 접속하여 유용한 정보를 얻게 되었다. 점차 인터넷 기술이 발전하고 사회적 문화의 중요한 수단으로 자리잡아 가면서, 인터넷은 우리 사회의 필수적인 의사소통의 도구이며, 새로운 대중 매체가 되었다. 과거의 웹 페이지(웹 서버)와 사용자의 컴퓨터가 주종관계였다면, 이제는 웹 페이지와 사용자 컴퓨터가 서로 상호 동작을 하면서 사용자들은 매우 다양한 서비스를 접하게 되었다. 이러한 과정 속에서 사용자의 컴퓨터로 다운되는 실행코드(소프트웨어)를 얼마나 신뢰할 수 있는가의 문제가 매우 중요한 문제로 떠올랐다. ActiveX Control이나 Java Applet등의 기술을 통해 웹 서버는 자신과 상호 작동하기 위해 필요한 클라이언트 응용프로그램을 사용자 컴퓨터에 설치하고 실행하도록 하고 있지만, 이러한 과정에서 사용자는 자신의 컴퓨터에 트로이 목마와 같은 유해한 코드가 사용자 모르게 설치되지 않을까 하는 의심을 갖게

접수일 : 2003년 10월 28일 ; 채택일 : 2004년 3월 31일
*본 연구는 정보통신부 대학 IT연구센터 육성·지원사업의 연구결과로 수행 되었습니다.
† 주저자, raedit@korea.ac.kr
‡ 교신저자, donghlee@korea.ac.kr

되었다. 소프트웨어 배포자의 입장에서도 정당한 배포자를 가장하여 어떤 악의의 공격자가 자신의 이름과 상표를 도용하고 유해한 코드가 담긴 소프트웨어를 배포할 수 있지 않을까 하는 우려도 생겨나기 시작했다. 이렇게 웹 페이지를 통해 실행코드가 담긴 소프트웨어를 다운받는 과정에서 소프트웨어에 대한 무결성(Integrity) 검증과 배포자에 대한 인증(Authentication)을 제공해주는 것이 코드 서명 기술이다.

앞으로 이어질 2장에서는 코드 서명 기술의 기본적인 메커니즘과 그 제반 기술들에 대해 설명하고, 3장에서는 현재 사용되고 있는 코드 서명 기술 중 대표적인 Microsoft사의 Authenticode 기술을 분석한다. 그리고 4장에서는 PKI 관련 표준안을 토대로 국내의 PKI에 활용 가능한 코드 서명용 인증서의 프로파일을 제안하며, 5장에서는 코드 서명 기술을 국내 PKI에 적용하기 위하여 고려해야 할 사항과 그 방안을 모색한다.

II. 코드 서명 기술

2.1 제공되는 암호학적 서비스와 기술

코드 서명 기술이 제공하는 암호학적 서비스로는 무결성과 인증이 있다. 무결성이란 데이터 무결성을 말하는 것으로 서명된 코드가 사용자에게 무해하다는 의미가 아니며, 배포자가 자신이 서명한 이후 사용자까지의 전달과정에서 어떠한 위조나 변조가 이뤄지지 않았다는 확신을 가질 수 있도록 해준다는 의미이다. 즉, 무결성은 서명하고자 의도된 메시지가 수신자에게 전송된 이후에도 변형되지 않음을 보장해 주는 성질이다. 이것은 x를 가지고 $f(x)$를 구하기는 쉬워도, $f(x)$를 가지고 x를 구하기는 어려운 일 방향 해쉬(One-way hash) 함수와 전자 서명을 통해 보장된다. 코드 서명 기술이 제공하는 인증이란, 배포자 인증을 말하는 것으로 배포되는 소프트웨어의 내용에 대한 인증을 의미하는 것이 아니라, 소프트웨어 배포자의 신원에 대한 인증을 뜻한다. 이것은 제 3의 공인 인증 기관(Certification Authority)이 먼저 배포자의 신원과 신뢰정도를 판단하여 공인 인증서를 발급하고, 최종 사용자들은 공인 인증기관이 발급한 인증서를 통해 소프트웨어 배포자에 대한 신원을 확인하고, 신뢰 여부를 판단 할 수 있다. 이러한 총체적인 과정들은 공개키 기반 구조(PKI : Public Key Infrastructure)라는 메커니즘을 기반으로 이뤄진다.

2.2 코드 서명 및 검증 과정

2.2.1 서명

대략적인 코드 서명 과정은 다음과 같다. 먼저, 배포자는 일 방향 해쉬 함수를 사용하여 원본 코드에 대한 해쉬 값을 계산한다. 해쉬 함수는 임의의 길이의 코드를 고정된 길이의 해쉬 값으로 축약한다. 이러한 해쉬 함수로는 SHA-1이나 MD5등이 사용된다. 계산된 해쉬 값을 배포자의 개인키를 사용하여 암호화(서명)한다. 이때 사용되는 암호화 알고리즘으로는 RSA 등이 사용된다. 이렇게 생성된 서명 값은 배포자의 개인키에 상응하는 공개키가 담겨진 코드 서명용 인증서와 함께 서명 블록(Signature Block)이란 특별한 구조로 캡슐화(Encapsulation) 된다. 그리고 배포하고자 했던 원본 코드와 함께 묶여져 서명된 코드로 만들어 진다.

2.2.2 검증

서명된 코드는 소프트웨어 사용자의 컴퓨터에서 코드 서명 검증 프로그램을 통해서 소프트웨어 사용자에게 확인되어진다. 코드 서명 검증 프로그램은 다운 로드된 코드의 서명을 검증하기 위해서 다음과 같이 서명 블록(Signature Block)을 검증한다.

- Step 1. 인증서의 검사 : 인증서를 서명 블록으로부터 읽어온 후 정해진 형태의 유효한 코드 서명용 인증서인지 검사한다. 인증서의 검증에 관한 사항은 일반적으로 인증서 발급 기관의 인증업무 준칙과 인증서 검증에 관한 정책 등에 의해 이뤄진다.
- Step 2. 해쉬값 생성 : 다운 로드된 코드의 원본 코드 부분에 대한 해쉬 값을 계산한다.
- Step 3. 공개키의 적용 및 검증 : 코드 서명된 값을 서명 블록으로부터 가져와 인증서에 포함된 배포자의 공개키를 적용하여 서명을 검증한다. 검증과정에서 원래 코드에 대한 해쉬 값을 얻을 수 있으며, 이를 Step 2에서 얻은 해쉬 값과 비교한다. 비교 값이 일치하지 않을 경우, 위조된 서명이거나 코드 서명 검증 과정이 실패한 것이다.

2.3 코드 서명과 타임스탬프

코드 서명용 인증서는 특정 기간의 유효기간을 가지고 있다. CA에서 발급하는 코드 서명용 인증서는 보통 1년에서 2년의 유효기간을 표시하고 있다. 그러므로 유효기간이 경과한 인증서는 더 이상 코드 서명 작업에 사용 할 수 없고 해당 CA로부터 갱신을 하거나 재발급 받아야 계속적인 코드 서명 작업을 수행 할 수 있다. 그런데, 이미 서명된 코드에 담겨있는 유효기간이 지난 인증서에 대한 처리 문제는 그리 간단하지 않다. 유효기간이 지난 인증서로 서명된 코드를 무조건 모두 신뢰할 수 없는 것으로 처리할 수는 없기 때문이다. 배포를 위해 서명된 코드는 인증서의 유효기간보다 훨씬 더 오랫동안 사용되어져야 할 경우가 있다. 이러한 문제를 해결하기 위해 코드 서명 과정에서 타임스탬프를 사용한다.[1]

유효기간이 지난 코드 서명용 인증서를 통해 서명을 하여 배포를 할 수는 없지만, 유효기간이 지난 인증서가 포함된 서명된 코드에 대해서는 코드 서명 당시에 유효한 인증서를 통해 서명된 것인지의 여부를 가지고 인증서 검증과 서명 검증을 처리한다. 소프트웨어의 배포자는 코드 서명 작업 시에 타임스탬프를 추가하여 서명하고 이를 통해 인증서의 유효기간이 지났을 때에도 현재 서명한 코드를 사용자들이 믿고 다운로드 가능하도록 할 수 있다. 즉, 타임스탬프를 통해 개발자는 인증서가 유효한 기간 중에 서명된 코드임을 나타내고, 최종 사용자들은 서명된 인증서를 검증할 때 유효기간이 경과한 인증서로 서명된 코드라 하더라도, 코드 서명의 시점을 타임스탬프를 통해 확인하고, 해당 시점이 인증서의 유효기간 내에 속하는지 여부를 검사하여, 만약 정당한 인증서의 유효기간 내에 만들어진 서명된 코드임을 확인할 수 있다면 서명된 코드는 신뢰할 수 있는 것으로 판단한다. 현재 코드 서명에 사용되는 타임스탬프 서비스는 Verisign CA를 통해서만 받을 수 있으며 무료로 제공된다.

2.4 주요 코드 서명 기술과 인증서

주요 코드 서명 시스템을 유선 인터넷 환경과 무선 모바일 환경으로 나누어 생각해보면 유선 인터넷 환경의 경우 Microsoft사와 Sun사에 의해 개발된 기술이 주를 이루어 사용되고 있고 무선 모바일 환경의 경우는 BREW나 Smartphone처럼 모바일 환경의 플랫폼(운영체제)별로 독자적인 시스템이 구축되어 사용되고 있다. 현재 사용되는 코드 서명 시스템들이 유사한 구조의 메커니즘과 알고리즘을 사용하고 있지만, 서로 호환되지는 않는다. 그렇기 때문에 코드 서명용 인증서라 하더라도 종류가 여러 가지가 있다.

Ⅲ. Microsoft사의 Authenticode 기술

3.1 개 요

Microsoft사(이하 MS)의 Authenticode 기술은 자사의 인터넷 브라우저인 Internet Explorer[1]와 MS-OFFICE등의 응용 프로그램에서 동작하는 코드 서명 기술이다. 이 기술을 통하여 소프트웨어 개발자는 cab, ocx, class, exe, stl, dll 파일을 웹을 통해 배포할 수 있다. Authenticode 기술은 PKCS#7[1], PKCS#9[3], X.509[4]등의 표준과 SHA, MD5등의 해쉬 알고리즘을 바탕으로 이루어져있다.

인터넷 익스플로러를 통하여 소프트웨어를 배포하고자 원하는 배포자는 MS에 의해 인정된 몇몇의 인증서 발급기관(CA)을 통해 코드 서명용 인증서를 발급받고 MS에서 다운로드받은 코드 서명 툴을 이용하여 자신이 만든 소프트웨어에 서명을 한 후 배포한다. 사용자들은 해당 웹 사이트를 통해 배포자의 소프트웨어를 다운로드 받게 되는데, 이 때 익스플로러에서는 다운되는 소프트웨어에 담긴 배포자의 서명이 올바른지를 먼저 검사하고 검증 결과를 사용자에게 확인하도록 하고 있다.

3.2 MS의 코드 서명용 인증서

3.2.1 코드 서명용 인증서의 발급

소프트웨어의 배포를 원하는 배포자는 먼저 MS에 의해 인정된 몇몇의 특정 인증기관(CA)을 통하여 코드 서명용 인증서를 발급 받아야 한다. MS 응용프로그램들과 연동 가능한 코드 서명용 인증서를 발급해주는 곳은 MS에서 인정해 주는 몇몇의 인증기관들(Microsoft Root Certificate Program Members[2])뿐이다.

1) 인터넷 익스플로러 버전 4.0과 그 이후 버전에 해당함.
2) http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsecure/html/rootcertprog.asp

현재는 Baltimore, Entrust, Verisign 등 9개 국외 인증기관을 통해 발급받은 MS 코드 서명용 인증서만이 인터넷 익스플로러에서 동작하는 올바른 코드 서명을 수행하고 검증하는데 사용 될 수 있다. 국내 소프트웨어 개발업체에서는 주로 Thawte CA를 이용하고 있는 실정이며 Verisign CA도 이용하고 있다. MS의 자료[3]에 의하면 코드 서명용 인증서는 크게 상업용과 개인용으로 나누어 발급하도록 하고 있으나, 실질적으로 Verisign을 포함한 인증기관에서는 상업용으로 발급하고 있는 상황이다. 그 이유는 개인용 인증서의 발급에 필요한 개발자의 신원 검증 비용이 수지 타산에 맞지 않기 때문이다. 그런데, 문제는 고가의 상업용 코드 서명용 인증서를 대부분의 소규모 개발자들은 이용하지 않을 것이라는데 있다. 그 결과 서명되지 않은 코드 배포가 급격히 증가되고 있으며, 사용자들도 서명되지 않은 컨트롤을 다운받는 것에 익숙해지면서 경고 메시지들에 신경을 쓰지 않고 있다. 결국 전체적인 보안 모델은 약화되고 있는 실정이다.

3.2.2 MS 코드 서명용 인증서

MS로부터 인정된 CA들에게서 발급 받게 되는 코드 서명용 인증서는 X.509, RFC2459[5]등의 표준에 준하는 인증서이다. MS의 코드 서명용 인증서를 발급하고 있는 코드 서명용 인증서 확장필드의 내용을 정리하면 표 1과 같다. MS에서는 인증서의 확장필드 중 확장키 사용(Extended Key Usage) 필드를 통해 코드 서명용 인증서를 나타내고 있는데, critical일 경우 해당 용도로만 사용토록 하고, non-critical일 경우는 다른 용도로도 사용할 수 있게 한다.

Thawte CA의 경우 현재 표준인 RFC2459에는 나와 있지 않으나, 과거 X.509 표준 draft에 제시되었던 키 사용 제한(Primary Key Usage Restriction)

표 1. MS 코드 서명용 인증서의 확장 필드

CA	확장 필드 항목	내용(값)
Verisign	Extended Key Usage (NC)	코드 서명 (1.3.6.1.5.5.7.3.3)
	Key Usage (C)	Digital Signature (80)
Thawte	Extended Key Usage (NC)	코드 서명 (1.3.6.1.5.5.7.3.3) MS 상업용 코드 서명용 (1.3.6.1.4.1.311.2.1.22)
	Primary Key Usage Restriction (NC)	[1]Cert PolicyId =1.3.6.1.4.1.311.2.1.22 Restricted Key Usage =Digital Signature (80)

C : Critical, NC : Non-Critical

이라는 확장 필드를 통해 인증서와 키의 사용 용도에 제한하고 있다.[4] 특히 MS에서 등록하여 사용하고 있는 MS만의 OID를 사용하여 코드 서명용 인증서를 발급하고 있다. 현 X.509v3 표준에는 제외된 키 사용 제한 확장필드를 사용하고 있는 이유는 MS의 Authenticode기술이 만들어질 당시에 draft중인 표준안에 의존해서 개인용과 상업용을 구분하던 방식을 계속 유지하고있기 때문이다.

3.2.3 소프트웨어 배포용 인증서(SPC)

MS에서는 코드 서명용 인증서를 특별히 소프트웨어 배포용 인증서(Software Publishing Certificate, 이하 SPC)라는 형태로 변환하여 코드 서명에 사용한다. SPC는 X.509v3 형태의 인증서들을 하나 또는 그 이상의 인증서들과 관련 CRL등을 포함 할 수 있는 형태의 패키지 형식이다. SPC는 인증서의 배포를 위한 PKCS#7 signed data 형식을 따르고 있다. CA에게서 발급 받는 코드 서명용 인증서는 SPC 형태로 발급된다. 실제 배포자의 코드

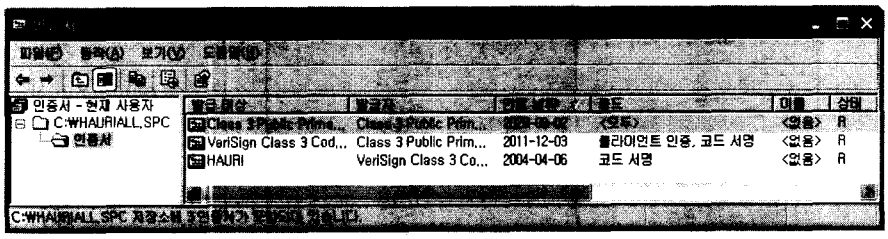


그림 1. SPC에 포함되어 있는 인증서들

3) MSDN (http://msdn.microsoft.com/library/default.asp)

4) http://www.alvestrand.no/objectid/2.5.29.4.html

서명용 인증서뿐만 아니라, 이 인증서의 검증에 필요한 인증 경로의 CA의 인증서들도 포함된 패키지 형식의 인증서 모음이다. 소프트웨어 배포자의 코드 서명용 인증서와 검증에 필요한 관련 CA들의 인증서를 모아 패키지 형식의 SPC로 만드는 이유는 인증서 검증에 필요한 CA의 인증서를 함께 전달하여 사용자의 응용 프로그램에서 코드 서명 검증을 빠르고 간편하게 하기 위함이다. 즉, 인증서 검증을 위해 다시 인증 경로의 CA들의 인증서를 요청하고 받아오는 과정 등을 생략할 수 있기 때문에 단일 컴퓨터에서도 빠르고 간편하게 인증서를 검증할 수 있다.

3.3 MS의 코드 서명 과정

코드 서명용 인증서를 CA로부터 발급 받은 후 MS의 웹 사이트를 통해 제공되는 코드 서명 프로그램을 이용해 소프트웨어에 서명을 할 수 있다.[5] 코드 서명을 하여 배포가 가능한 파일들의 형식은 .cab .cat .ctl .dll .exe .ocx이며, SignCode 프로그램이 실제 배포용 소프트웨어의 서명에 사용된다.

3.3.1 코드 서명 과정 (SignCode의 내부 작동 원리)

① 서명할 코드(m)를 MD5 또는 SHA1의 해쉬 함수를 통해 고정길이의 해쉬 값 (hash(m) 생성

② 해쉬 값을 배포자의 개인키로 서명, 서명 알고리즘은 RSA 사용 (sign(hash(m)생성)

③ 서명된 내용과 배포자의 인증서를 PKCS#7 signed-data object형태의 signature block으로 조합

④ 원본 코드 뒷부분에 signature block 첨가

> o 서명된 코드 m' = m || signature block
> o signature block
> = sign(hash(m))과 PKCS#7 signed-data object

3.3.2 코드 서명의 검증

사용자가 인터넷으로부터 실행 소프트웨어(코드)를 다운로드 할 때, 브라우저 혹은 클라이언트의 특정 어플리케이션은 WinVerifyTrust()라 불리는 WIN32함수를 사용한다. 이 함수는 서명된 값을 뽑아내고, 인증서의 유효성을 검증하며 배포자의 공개키를 사용하여 signature block에서 해쉬값, 즉 hash(m)을 획득한다. 이어서 서명으로부터 복구된 해쉬 값과 원래 코드를 이용해 자신이 만든 해쉬 값이 같은지를 검사한다. 만약 해쉬 값이 다르다면 키 값이 틀린 것이거나 다운로드된 코드가 변조된 것이다. 사용자들에 의해 선택된 인터넷 보안 설정 옵션에 따라서 인터넷 익스플로러는 배포자에 의한 서명 여부나 혹은 다운로드 하는 동안 변조되었는지 여부를 사용자에게 알리게 되어있다.

3.3.3 서명된 코드에 대한 분석

자체적으로 제작한 테스트용 .exe 실행 파일에 테스트용 인증서를 이용해 서명을 한 후 서명 전의 실행 파일과 서명 된 실행파일을 WinHex라는 에디터를 통해 비교해 보았다. 그 결과 그림 2처럼 서명된 실행파일에는 서명 전 실행파일의 원본 코드가 앞부분에 그대로 있고 뒷부분에 코드 서명과 관련된 내용들이 그대로 추가 된 것으로 분석되었다. 결국 서명된 파일은 원래 파일의 뒷부분에만 그대로 디지털 서명 관련 내용이 추가되는 것이며, 원본 코드는 변형되지 않는다는 것을 알 수 있다. 특히 이 과정은 .exe파일뿐만 아니라 .cab나 .ocx등 다른 확장자를 가진 파일들에 대해서도 비슷한 결과를 얻을 수 있었다. 웹을 통해서 서명된 코드가 배포 될 때 인터넷 익스플로러에서 서명된 파일을 검증하고 올바르면 서명된 파일 중 앞부분의 원래 코드만 떼어내어 사용자 컴퓨터에서 실행되도록 저장하거나 설치하는 방식으로 이뤄져 있다는 것을 알 수 있다.

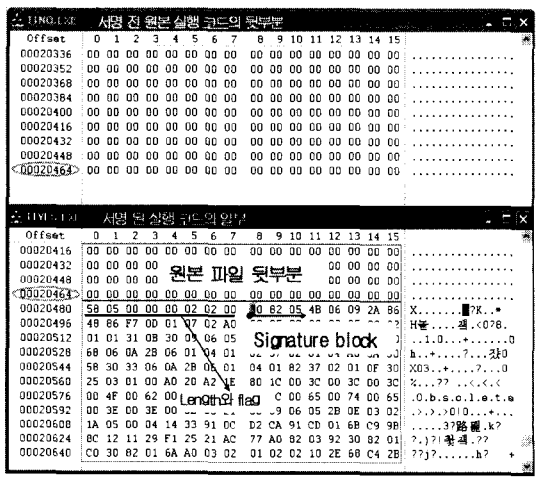


그림 2. 서명 전 파일과 서명된 파일의 코드 분석

5) http://www.microsoft.com/downloads/details.aspx?FamilyID=2B742795-D0F0-4A66-B27F-22A95FCD3425&displaylang=en

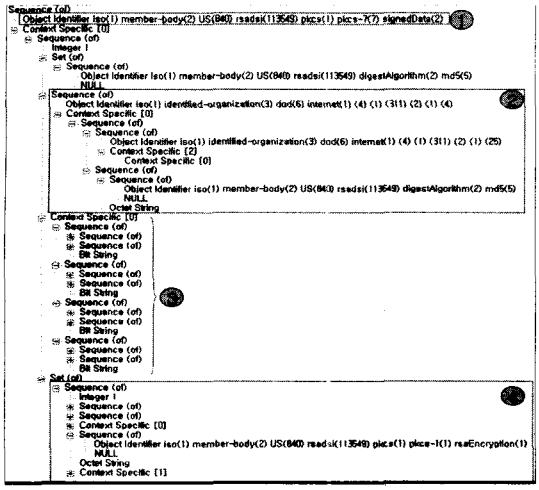


그림 3. 서명된 파일의 signature block

다음으로는 서명된 파일의 signature block이 어떤 형식으로 이뤄져 있는지 분석해 보았다. Verisign CA로부터 발급된 인증서로 서명된 파일을 샘플로 그림 2처럼 분석하여 signature block만을 따로 떼어낸 후 이 부분을 DER[6]로 분석하여 ASN.1형식으로 나타내었다. 그 결과는 그림 3과 같다. 서명된 코드에는 Length와 flag가 붙고 이어서 PKCS#7 SignedData 형식의 서명 블록이 이어진다. Length필드는 보통 32bit로 표현되며(예: 0xD8030000), 배포하고자하는 소프트웨어 바로 다음에 이어지며 length 및 flag를 포함해(64bit) PKCS#7 SignedData 형식으로 표현된 서명 블록까지의 전체 길이를 알려준다. length 필드의 형식은 최하위 비트가 먼저 오고, 최상위 비트가 맨 끝에 오도록 표현된다. Flag는 항상 0x00020200으로 이루어진다.

그림 3의 ①번 부분을 통해 전체 블록의 형식이 PKCS#7 signed data 형식을 따르고 있다는 것을 알 수 있다.②번 부분은 서명된 컨텐츠에 대한 정보를 나타내는 contentInfo 부분에 해당한다. 이 부분에서는 MS에서 등록하여 자체적으로 사용하는 OID를 발견할 수 있다. ③번 부분은 서명 검증에 필요한 X.509 인증서들이 들어있는 부분이다. 관련 CA의 인증서와 소프트웨어 배포자의 코드 서명용 인증서가 포함되어있다. ④번 부분은 SignerInfo 부분으로 컨텐츠에 대하여 서명한 서명자의 정보와 서명 알고리즘, 서명값, 타임스탬프 등이 포함되어 있다. MS에서 등록하여 자체적으로 사용하고 있는 OID들에 대한 정보는 MS SDK[6]를 통해 습득할 수 있다. MS SDK에 포함되어있는 WinTrust.h 파일을 통해 Microsoft사에서 사용하는 OID의 정의 내용을 볼 수 있다.

IV. 우리나라 PKI를 위한 코드 서명용 인증서 프로파일

4.1 국내 공인 인증 체계에 활용 가능한 코드 서명 인증서 프로파일

코드 서명용 인증서의 프로파일은 전자서명 인증서 프로파일에 기초하여 구성할 수 있으며, 키 사용 목적과 확장키 사용 목적, 인증서 정책이라는 확장필드를 통해 코드 서명용 인증서 프로파일을 만들 수 있다. 국내 공인인증체계에서 활용 가능한 코드 서명용 인증서 프로파일을 정리하면 표 2와 같다. 기본필드는 전자 서명 인증서 프로파일과 동일하기 때문에, 확장 필드에 대해서만 나타냈다. 코드 서명용 인증서 프로파일에서 가장 중요한 확장 필드는 키 사용 목적, 인증서 정책, 확장키 사용 목적 필드이다. 키 사용 목적, 인증서 정책, 확장키 사용 목적 필드들에 대해서는 반드시 코드 서명용 인증서에 포함되어야 하고 인증기관과 응용 프로그램은 이 확장필드를 처리할 수 있어야 한다. 그리고 인증서 정책 필드의 내용 중 policyIdentifier는 인증기관의 코드 서명용 인증서의 정책을 나타내는 OID 값을 가져야 한다.

4.1.1 키 사용목적 (Key Usage)

키 사용목적 확장필드는 암호화용이나 키 동의용 등에 사용되지 않고 코드 서명에 대한 검증용으로 사용되어야 한다는 것을 명시하기 위해 digital-Signature와 nonRepudiation 값을 가져야 한다. 이 확장필드는 critical로 설정되어야 하며 지정된 사용목적 이외로 사용되지 말아야 한다.

4.1.2 인증서 정책 (Certificate Policies)

인증서 정책 확장필드의 policyIdentifier 하위필드가 반드시 사용되어야 하며 인증서 정책에 대한 OID를 값으로 가져야 한다. 그런데, 현재 국내에는

6) http://www.microsoft.com/msdownload/platformsdk/sdkupdate/ default.htm

표 2. 국내 코드 서명용 인증서 프로파일 확장 필드 (기본 필드는 기존 전자 서명 인증서 프로파일 표준과 동일)

확장 필드 항목	구분		
	critical 여부	선택여부	
		생성	처리
Authority Key Identifier	non-critical	mandatory	mandatory
Subject Key Identifier	non-critical	mandatory	mandatory
Key Usage	**critical**	**mandatory**	**mandatory**
	digitalSignature, nonRepudation (C0)		
Private Key Usage Period	non-critical	not recommended	not recommended
Certificate Policies	**critical or non-critical**	**mandatory**	**mandatory**
Policy Mapping	not defined	not defined	not defined
Subject Alternative Name	non-critical	mandatory	mandatory
Issuer Alternative Name	non-critical	optional	mandatory
Subject Directory Attributes	non-critical	not recommended	not recommended
Basic Constraints	critical	not recommended	not recommended
Name Constraints	not defined	not defined	not defined
Policy Constraints	not defined	not defined	not defined
Extended Key Usage	**critical**	**mandatory**	**mandatory**
	코드 서명 (1.3.6.1.5.5.7.3.3)		
CRL Distribution Points	non-critical	mandatory	mandatory
Authority Information Access	non-critical	optional	optional
Procuration	not defined	not defined	not defined

코드 서명용 인증서의 정책을 나타낼 수 있는 OID가 존재하지 않는다. 코드 서명용 인증서에 대한 서비스가 가능하려면 코드 서명용 인증서의 정책을 나타내는 OID를 인증서를 발급하는 인증기관이 할당받아 사용해야 하며, 또한 응용 프로그램들이 이를 처리할 수 있도록 해야 한다.

4.1.3 확장 키 사용목적 (Extended Key Usage)

확장 키 사용목적 확장필드가 시점확인용 인증서에 대해서는 반드시 포함되어야 하듯이 코드 서명 인증서에도 반드시 포함되어야 한다. 또한 코드 서명용 인증서의 확장키 사용 목적 확장필드는 critical로 설정되어야 한다. 물론, 국내 최상위 인증기관에서 코드 서명을 위한 확장키 사용 목적 OID를 할당하여 사용할 수도 있지만, 이러한 경우에는 국제적인 호환성에 문제가 발생 할 수도 있다. 그러므로 코드 서명용 인증서에 대한 확장 키 사용목적 OID는 국제적으로 통용되는 OID를 사용해서 "id-kp-codeSigning OBJECT IDENTIFIER ::={ 1 3 6 1 5 5 7 3 3 }"으로 한다.

4.3 Internet Explorer와 Netscape Navigator에서 호환 가능한 국내 코드 서명용 인증서 프로파일

현재 코드 서명은 웹 브라우저를 통해 검증되고 있기 때문에 주요 웹 브라우저인 Internet Explorer와 Netscape Navigator에서 무리 없이 사용 가능한 인증서 프로파일을 구성한다. 앞서 알아본 우리나라의 전자서명 인증서 프로파일의 표준에 의거해 기본적인 프로파일을 구성하고 MS의 Authenticode 기술과 Netscpae의 Object Signing 기술에서 필

표 3. Internet Explorer와 Netscape Navigator에서 호환 가능한 국내 코드 서명용 인증서 프로파일 (확장필드)

확장 필드 항목	critical 여부	내용(값)
Authority Key Identifier	non-critical	KeyID = 발급기관 공개키 해쉬 값
Subject Key Identifier	non-critical	KeyID = 소유자 공개키 해쉬 값
Key Usage	critical	digitalSignature, nonRepudation (C0)
Certificate Policies	critical or non-critical	policyIdentifier = 인증기관 코드 서명 인증서 정책 OID
Subject Alternative Name	non-critical	소유자(주체) 대체 이름
Extended Key Usage	critical	코드 서명 (1.3.6.1.5.5.7.3.3)
Netscape Cert Type	non-critical	Object Signing(10) 또는 Signature(10)
CRL Distribution Points	non-critical	인증기관의 CRL 배포 지점 URL

요로 하는 확장필드 내용을 추가하여 수정하였다. MS에 필요한 확장필드로는 Verisign CA에서 발급하는 코드 서명용 인증서와 같이 확장 키 사용, 키 사용 필드가 필요한데, 이러한 필드는 앞서 제시한 국내 코드 서명용 인증서 프로파일에 모두 들어있는 내용이다. 그러므로 Netscape에 필요한 Netscape Cert Type이라는 확장 필드만 추가하여 구성한 Internet Explorer와 Netscape Navigator에서 호환 가능한 국내 코드 서명용 인증서의 확장필드 프로파일은 표 3과 같다. MS의 코드 서명용 인증서에 필요한 확장필드는 Verisign CA에서 발급하는 인증서의 형태를 따라 구성하였으므로 문제가 없다. 확장 필드 중 인증서 정책 필드에는 인증서 발급 CA에서 규정하는 인증서의 정책을 나타내는 OID를 기재하도록 되어있으므로 국내 CA에서 코드 서명용 인증서를 발급하기 위해서는 먼저 코드 서명용 인증서의 정책 OID를 정해야 할 것이다.

V. 코드 서명 기술을 국내 PKI에 적용하기 위한 방안 연구

본 장에서는 인터넷 익스플로러(Internet Explorer)로 대표되는 유선 인터넷 브라우저에서 국내 PKI기반의 코드 서명 기술을 적용하는데 따른 문제점을 알아본다. 즉, 국내의 공인 인증기관에서 발행하는 코드 서명 인증서를 사용하여 코드 서명을 시행하고 사용자들은 인터넷 익스플로러를 통해 다운로드할 수 있도록 하기 위해 어떠한 문제를 해결해야 하는지 알아보겠다. 코드 서명 기술을 국내 PKI에 적용하기 위한 방법은 크게 두 가지로 나눠 볼 수 있다. 서명된 코드에 대한 서명 검증을 웹 브라우저가 담당하도록 하는 방법과 현재 인터넷 뱅킹에서 이뤄지고 있는 방법처럼 다른 코드 서명 검증 프로그램을 통해 검증하도록 하는 방법이 그것이다. 각 방법에 따른 장단점을 분석하고 이러한 방법들을 통해 실제로 국내 공인 인증기관이 발급한 코드 서명용 인증서를 사용하여 코드 서명을 시행하고 검증하기 위해서는 어떤 문제점들이 있는지 알아보겠다.

5.1 웹 브라우저에서 코드 서명 검증하는 방법과 그에 따른 문제점

첫 번째의 경우는 현재 많은 사용자들의 컴퓨터에 이미 탑재되어있는 인터넷 익스플로러와 같은 웹 브라우저를 통해 코드 서명 검증을 하는 방법이다. 소프트웨어의 배포자는 국내 공인인증기관에 코드 서명용 인증서의 발급을 요청하여 발급 받고, 이 인증서에 합당한 개인키와 코드 서명용 인증서, MS의 코드 서명 프로그램(Signtool.exe)을 통해 코드 서명을 시행한다. 그리고 인터넷 익스플로러를 통해 서명된 코드를 배포하게 된다.

일반 사용자들은 해당 배포자의 웹 사이트에 접속하여 서명된 프로그램을 다운로드 하게 되고 이때 서명에 대한 검증은 인터넷 익스플로러가 자동적으로 코드 서명 검증 API (Winverifytrust()함수)를 호출하여 검증하게 되며, 그 결과를 사용자에게 보여주게 된다.

이 방식은 코드 서명 도구는 MS의 서명 프로그램을 그대로 사용하며, 검증도 인터넷 익스플로러에서 담당하는 방식이다. 단지 코드 서명에 사용되는 인증서를 국내 공인 인증기관이 발행한 코드 서명용 인증서로 사용하는 방식이다. 이러한 방식을 취할 경우 코드 서명을 위한 별도의 알고리즘이나 프로그램 개발이 필요하지 않고, 사용자의 입장에서도 이미 친

숙해져 있는 인터넷 익스플로러만으로 모든 과정이 이뤄지므로 피부로 느껴지는 다운로드 속도가 현재와 다르지 않으며 조작하기에 간편하다는 장점이 있다. 그렇지만, 이렇게 서명 검증을 웹 브라우저인 인터넷 익스플로러에게 전담시킬 경우의 국내 공인 인증 체계에서 필요로 하는 여러 가지 요구사항을 만족시키지 못하고 MS의 인증 정책에 의거해 이뤄지게 된다는 단점이 있다. 이러한 기술적인 문제나 정책적인 견지에서 발생하는 문제점으로는 어떠한 것들이 있는지 자세히 알아보겠다.

표 4. 인터넷 익스플로러를 통해 서명 검증하는 방안

o 코드 서명 인증서의 발급 : 국내 공인 인증기관 o 코드 서명 : MS의 코드 서명 툴 사용 o 서명된 코드의 배포 및 접근 (웹 브라우저) : MS의 인터넷 익스플로러 o 서명 검증 : 사용자 컴퓨터의 인터넷 익스플로러의 코드 서명 검증 API

5.1.1 국내의 기존 공인 인증 체계 표준과의 차이

우선 코드 서명의 생성과 검증을 모두 MS의 프로그램에 의존할 경우 현재 국내 전자서명 인증서 관련 표준에 의거해 이뤄지지 못하는 부분들이 발생한다. 인증서의 저장 위치의 경우 국내 표준에서 정하고 있는 것은 특정 폴더에 저장하도록 하고 있으나 MS의 경우는 특정 레지스트리에 저장하도록 하고 있어 차이가 있다. 또한 개인키의 저장위치와 방법도 MS에서는 자체적으로 마련한 방법에 의해 이뤄지고 있어 우리나라 전자서명 표준과 차이가 있다. 또한 국내 표준 전자 서명 KCDSA나 표준 해쉬 알고리즘인 HAS-160을 사용할 수 없다. 결국, 우리나라가 코드서명과 관련된 표준안에 대한 수정하거나 보완 시 MS의 인증 체계를 그대로 받아들여야 한다. 다행히 인증서 프로파일의 경우는 Verisign CA에서 발급하는 인증서 형태가 우리나라의 전자서명 인증서 표준 프로파일과 많이 유사하므로 앞 장의 표 3과 같이 확장필드를 구성하도록 국내 코드 서명용 인증서 프로파일을 구성하면 된다. CRL검증 문제도 국내에서 사용하고 있는 LDAP 방식을 MS에서 지원하고 있기 때문에 큰 문제가 생기지는 않는다.

5.1.2 안전성 문제

현재까지 MS에서 보고하고 있는 코드 서명 관련 취약점에 대한 기술 자료로는 "MS02-050 인증서 확인 결함으로 인해 신분을 속일 수 있다(기술문서 KB329115)", "MS03-041 Authenticode 확인의 취약점으로 인한 원격 코드실행 문제(기술문서 KB823182)"등이 있다. 첫번째 취약점은 2002년 9월 5일 최초 발표되었으며, 주요 원인은 인증 체인을 구성하고 유효성을 검사하는 CryptoAPI의 일부 함수들이 기본 제약 필드를 검사하지 않는 점이었다. 사용자가 신뢰할 수 있는 회사에 발급되었다고 주장하는 Authenticode 인증을 사용해 악성 소프트웨어의 디지털 서명을 생성하여 배포할 수 있기 때문에, 경우에 따라서는 침입자가 사용자 컴퓨터에 대한 제어권을 얻도록 허용할 수도 있는 매우 위험한 취약점이다. 두 번째 취약점은 2003년 10월 16일에 최초 발표되었으며 주요 내용은 Authenticode의 취약점으로 인해 메모리가 적은 특정 조건에서 사용자의 승인을 받기 위한 대화 상자를 표시하지 않고 ActiveX 컨트롤이 Authenticode에 대한 검증없이 다운 로드되고 설치될 수 있다는 것이다. Authenticode의 취약점으로 인해 확인 창을 표시하지 않은 상태에서 허가되지 않은 ActiveX 컨트롤이 사용자와 동일한 권한을 이용하여 시스템에 설치되고 실행될 수 있기 때문에 공격자가 원하는 대로 사용자의 컴퓨터를 원격에서 조정하거나 정보를 누출시킬 수도 있는 심각한 위험에 빠질 수 있다. 또한 중요한 점은 이 취약점이 Microsoft Windows의 Authenticode 기술 자체에 결함이 있기 때문에 Authenticode 기술을 사용하는 모든 응용 프로그램이 이 결함에 취약할 수 있다는 점이다.

5.1.3 사용자 웹 브라우저 보안 설정문제

기본적으로 MS의 코드 서명 검증이 사용자 측 컴퓨터에서 올바르게 이뤄지려면 인터넷 익스플로러의 보안 설정을 그림 4와 같이 해야 한다. 인터넷 익스플로러의 인터넷 옵션 중 보안 탭에서 이러한 설정을 할 수 있는데, 이때 Authenticode에 대한 검증 여부와 ActiveX에 대한 서명 검증 여부를 설정할 수 있다. 만약 사용자의 인터넷 익스플로러에 대한 보안설정이 올바르지 않다면 서명되지 않은 프로그램이 검증없이 설치될 수 있다.

그리고 Authenticode, ActiveX관련 보안 설정뿐만 아니라, 인증서 검증에 필수적인 CRL검증에 대한 설정도 그림 5와 같이 해야 한다. 인터넷 익스플로러의 인터넷 옵션의 고급 탭에서 인증서 검증과 관련된 설정을 할 수 있다. 필요한 설정을 미리 하지

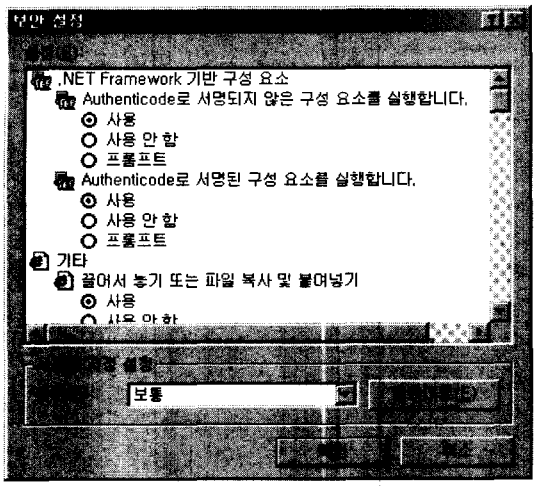


그림 4. 인터넷 보안 설정

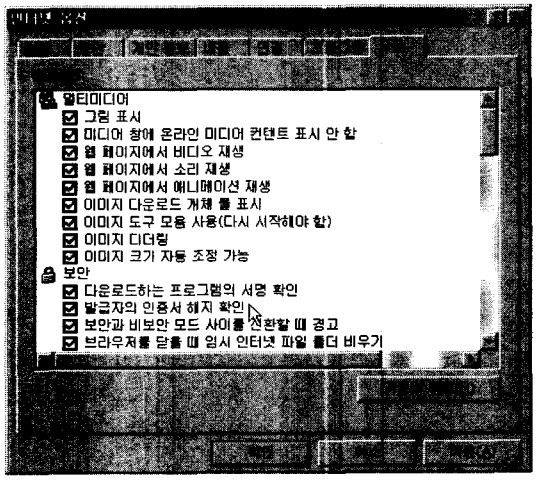


그림 5. 인증서 검증 관련 설정

않을 경우에는 유효하지 않은 인증서로 서명된 코드를 유효한 코드서명으로 확인하게 될 수가 있다. 이처럼 MS의 인터넷 익스플로러에 코드 서명에 대한 검증을 맡길 경우 일반 사용자들의 인터넷 익스플로러의 보안설정이 필수적으로 이뤄져야만 코드 서명 검증을 올바르게 수행할 수 있다.

5.1.4 코드 서명용 타임스탬프

기본적으로 코드 서명에는 코드 서명 시점을 기록해야하기 때문에 타임스탬프가 필요하다. Netscape는 코드 서명을 수행한 컴퓨터에서 자체적으로 시간정보를 삽입하지만, MS에서는 외부 Verisign CA의 코드 서명용 타임스탬프 서비스에 의존한다. 만약 코드 서명의 생성과 검증을 위해 MS의 프로그램을 그대로 사용하게 된다면 타임스탬프를 Verisign CA의 코드 서명용 타임스탬프 서비스에 의존해야 한다는 문제가 발생한다. MS의 코드 서명이나 검증 프로그램이 사용하는 타임스탬프는 PKCS#9의 Countersignature(연대서명) attribute를 PKCS#7의 unauthenticated attribute에 포함시키는 방식을 사용한다. Verisign 타임스탬프 서비스가 PKCS#7의 서명 값에 대하여 시간정보를 추가하여 연대서명해주는 형태로 이뤄지기 때문에 기본적으로 타임스탬프 표준(RFC 3161)과 다르고, 그 결과 국내 공인 인증체계에서 사용하는 타임스탬프 서비스로 연동시키거나 대체하여 사용할 수가 없다. 또한 MS의 코드 서명 및 검증 프로그램의 내부 동작원리가 자세하게 공개되지 않고 있기 때문에 우리나라에서 자체적으로 코드 서명용 타임스탬프 서비스를 개발하기도 어렵다. 결국 코드 서명 시 타임스탬프를 삽입하기 위해서는 Verisign CA의 코드 서명용 타임스탬프 서비스에 의존해야 하고, 만약 타임스탬프 서비스를 Verisign CA에 의존하지 않으려면 코드 서명 시 타임스탬프를 사용하지 않는 수밖에 없다.

5.1.5 최상위 루트 인증서의 탑재 문제

MS의 인터넷 익스플로러에게 코드 서명에 대한 검증을 맡길 경우 가장 문제되는 것이 바로 인터넷 익스플로러 자체에 있는 인증서 저장소의 "신뢰된 루트 인증기관"에 국내 공인 인증기관이 존재하지 않는다는 것이다. MS에 필요한 코드 서명용 인증서 형식에 맞춰 만들어 코드 서명 후 배포한다 해도 사용

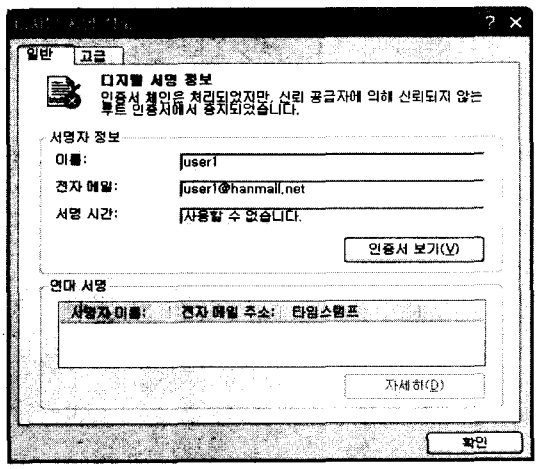


그림 6. 신뢰되지 않은 루트 인증기관 에러

자에게는 "신뢰된 루트인증기관에 인증서 발급 CA가 존재하지 않는다."라는 경고 메시지가 그림 6처럼 전달된다. 그 이유는 국내 공인 인증기관이 Internet Explorer의 "신뢰된 루트 인증기관" 인증서 저장소에 포함되어있지 않기 때문이다. 이러한 경고 메시지를 없애고 올바르게 검증되도록 하려면 MS의 "Microsoft Root Certificate Program Members"에 국내 공인인증기관이 속해야 한다. 그렇지 않으면 일반 사용자들로 하여금 자신의 컴퓨터에 있는 인증서 저장소에 국내 공인인증기관의 인증서를 "신뢰된 루트 인증기관"에 삽입하도록 유도해야한다. 국내 인증 체계와 관련된 표준에 벗어나지만, 코드 서명 기술의 개발이나 검증 프로그램의 배포 없이 Internet Explorer에서 코드 서명을 수행하도록 하는데 가장 큰 문제는 바로 이것이다.

5.1.6 이미 발행된 기존 전자서명용 인증서와의 문제

만약, 위 5.1.5절에 나온 문제를 해결하여 신뢰된 루트인증기관에 국내 공인 인증기관이 포함된다하더라도 또 하나의 중요한 문제가 발생한다. 바로 이미 국내에 발급되어 사용되는 기존 전자서명 인증서로도 코드서명이 가능해진다는 문제이다. 신뢰된 루트 인증기관에 국내 공인 인증기관이 포함된다면 기존 전자서명 인증서를 사용하여 코드 서명이 가능할 뿐 아니라 그림 7처럼 검증도 올바르게 이루어지기 때문에 코드 서명 인증서를 별도로 발급 받지 않고 코드 서명에 사용하는 경우가 발생할 수 있다. 이렇게 이미 발급되어 사용하고 있는 전자서명 인증서를 코드 서명에 사용하지 못하게 하려면 서명 프로그램이나

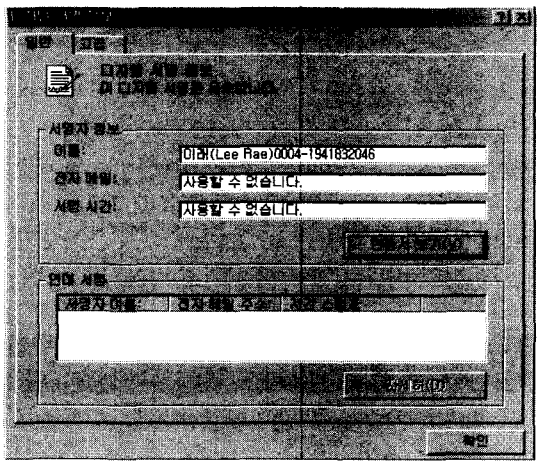


그림 7. 기존 인터넷 뱅킹용 인증서로 서명한 코드

표 5. 별도의 검증 프로그램을 통해 서명 검증하는 방안

o 코드 서명 인증서의 발급 : 국내 공인 인증기관 o 코드 서명 : 국내 자체 개발한 코드 서명 프로그램 o 서명된 코드의 배포 및 접근 (웹 브라우저) : MS의 인터넷 익스플로러 o 서명 검증 : 국내 자체 개발한 코드 서명 검증 프로그램

검증 프로그램에서 막아야 하지만, MS의 서명 및 검증 프로그램을 그대로 사용하기 때문에 프로그램에 대한 수정이 불가능하다. 결국 기존 전자서명 인증서로 코드 서명을 수행하는 것을 막으려면 공인인증체계의 인증기관 구조를 수정해야한다. 예를 들면, 코드 서명을 위한 새로운 루트 인증기관을 만든 후 기존의 국내 PKI 최상위 루트 인증기관인 KISA의 CertRSA01은 인터넷 익스플로러의 신뢰된 루트 인증기관에 포함되지 않도록 하고, 새로운 루트 인증기관만 신뢰된 루트 인증기관에 포함되도록 해야 할 것이다.

5.2 독립적인 서명 검증 프로그램을 사용하는 방법과 그에 따른 문제점

코드 서명 기술을 국내 공인 인증 체계에 적용하기 위한 두 번째 방안은 현재 국내에서 이뤄지고 있는 인터넷 뱅킹 서비스와 같이 웹 브라우저와는 별도의 프로그램에서 서명 검증을 담당하는 방식이다. 이 방안을 사용하기 위해서는 국내에서 자체적인 코드 서명 기술을 설계하여 서명 프로그램과 서명 검증 프로그램을 만들어 배포하여야 한다. 소프트웨어 배포자는 국내 공인인증기관에 코드 서명용 인증서발급을 요청하여 발급 받고, 이 인증서에 합당한 비밀 키와 코드 서명용 인증서, 그리고 국내 코드 서명 프로그램을 통해 코드 서명을 시행한다. 그리고 인터넷 익스플로러를 통해 서명된 코드를 배포하게 된다. 일반 사용자들은 해당 배포자의 웹 사이트에 접속하여 서명된 프로그램을 다운로드 하게 되고 이때 서명에 대한 검증은 사용자 컴퓨터에 설치된 국내 코드 서명 검증 프로그램이 검증하게 되며, 그 결과를 사용자에게 보여주게 된다. 코드 서명 프로그램과 검증 프로그램을 국내에서 자체적으로 개발하여 사용하며, 코드 서명에 사용되는 인증서도 국내 공인 인증기관이 발행한 코드 서명용 인증서를 사용하게 된다.

이 방식은 앞에서 알아본 첫 번째 방법과는 다르게 현재 우리나라의 공인인증체계의 표준에 의해 마

표 6. 코드 서명 기술의 국내 PKI 적용을 위한 모델 비교

구 분	인터넷 익스플로어 기반 모델	독립 서명검증 프로그램 모델
Root CA인증서 등록	특정 웹 사이트로 접속하여 설치하도록 유도	클라이언트 모듈이 설치 될 때 자동 설치
서명 검증 프로그램 최초신뢰 문제	인터넷 익스플로러에 포함되어있음	오프라인으로 전달하거나, 기존의 MS Authenticode 기술 이용해서 배포
키 생성 및 인증서 요청 프로그램 배포	MS Authenticode 기술을 이용해 서명 배포	이미 설치되어 있는 독립 서명검증 모듈을 이용해서 코드 서명 검증
코드서명인증서 요청	인터넷 익스플로러에서 인증서 요청 모듈의 코드 서명 검증 후 인증서 요청	독립 서명검증 모듈에서 인증서 요청 모듈의 코드 서명 검증 후 인증서 요청
인증서 프로파일	표 4의 국내 코드 서명용 인증서 프로파일 적용	
코드 서명	MS의 코드 서명 툴 Signcode.exe 이용	국내 자체 개발한 코드서명 프로그램을 이용
코드 서명 검증 및 소프트웨어설치	인터넷 익스플로어에서 WinVerifyTrust() 함수에서 서명 검증 후 설치	자체 개발한 코드서명 검증 모듈에서 서명 검증 후 설치

련된 환경을 그대로 사용할 수 있다. 인증서의 신뢰 검증 부분도 외부 브라우저와의 호환성이나 보안 설정을 염려하지 않아도 될 것이고, KCDSA, HAS-160과 같은 국내 서명 및 해쉬 알고리즘도 사용 할 수 있다. 기타 나머지 서명 검증과 관련된 사항들을 국내 공인 인증 체계의 여러 가지 표준안에 따라 대부분 구성 가능할 것이다. 또한 외국의 코드 서명 기술에 의존하지 않고 자체적으로 코드 서명 기술을 개발하여 사용하게 되므로 안전성에 대한 증명과 보완이 가능하여 전체적인 코드 서명과 검증의 메커니즘을 국내 공인인증 체계 표준에 어긋나지 않게 할 수 있다는 장점이 있다. 그리고 첫 번째 방식의 문제점으로 제기된 현재 발급되어 사용되고 있는 전자 서명용 인증서로 코드 서명 및 검증이 가능한 문제도 인증기관 구조를 그대로 유지하면서 해결할 수 있다. 서명 프로그램과 검증 프로그램을 자체 제작하는 것이므로 이러한 프로그램에서 인증서 정책 확장 필드의 OID를 검사하여 전자서명 인증서로는 코드 서명 및 검증이 불가능하도록 하면 된다. 그리고 코드 서명 프로그램을 자체 제작하는 경우이므로 기존의 국내 타임스탬프 서비스를 사용하여 코드서명에 사용하도록 할 수도 있고, 별도의 코드 서명용 타임스탬프 서비스를 만들어 사용할 수도 있다.

그러나 이 경우는 코드 서명을 생성하고 검증하는 프로그램을 국내 PKI 실정에 맞게 개발하여 배포해야 한다는 문제가 있다. 현재 인터넷 뱅킹에서 사용되는 클라이언트 프로그램처럼 검증 프로그램이 웹 브라우저를 통해 서명된 파일이 다운로드 되는 것을 자동으로 감지하고 검증을 수행하여 결과를 사용자에게 보여주어야 한다. 그러나 웹 브라우저와는 다른 프로그램을 통해 이러한 과정이 이뤄지므로 사용자 입장에서는 번거롭고 불편할 수 있다. 뿐만 아니라 코드 서명 프로그램과 검증 프로그램을 사전에 제작 배포해야 하며, 검증 프로그램이 일반 사용자 컴퓨터에 먼저 설치되도록 해야 한다. 특히 검증 프로그램을 사용자 컴퓨터에 먼저 배포하여 설치되도록 할 때, 이 프로그램에 대한 신뢰 검증은 어떤 방식으로 취할 것인가가 가장 큰 문제이다. 현재 이 문제에 대한 국내 기술만으로 해결할 수 있는 방법은 없으며, 코드 서명 검증용 독립 프로그램의 최초 배포에는 외국 Verisign에게서 발급받은 코드 서명 인증서와 MS의 Authenticode 기술에 의존해야 하는 수밖에 없다.

5.3 코드 서명 기술의 국내 PKI 적용 방안 비교

인터넷 익스플로러 기반 모델에서는 코드 서명된 파일의 검증을 위해, 최상위 인증기관의 인증서가 사용자의 신뢰된 인증기관 인증서 목록에 반드시 있어야 한다. 그러므로 최상위 인증기관(KISA)의 인증서 탑재 문제의 해결이 모델 개발에 가장 중요한 문제이다. 인증서의 탑재는 지정된 웹 사이트로 사용자를 유도하여, 온라인상으로 설치하게 하는 방법이 가장 좋은 방안이다. 물론, 사용자 측면에서는 약간의 번거로움이 있지만, 한번의 인증서 설치로 인해, 이후 코드 서명된 모든 파일의 검증이 가능하므로 현실적인 방안이 된다.

독립 서명검증 프로그램 모델에서는 이러한 인증

서 설치 문제를 고려하지 않아도 된다. MS의 지정 디렉토리에 인증서를 설치할 필요 없이, 독립 모듈이 설치될 때 자동적으로 루트 인증서가 정해진 디렉토리에 저장하도록 설계하면 된다. 여기에서는 이러한 독립적인 서명검증 모듈을 어떻게 안전하게 배포하느냐가 가장 큰 문제이다. 독립 서명검증 모듈을 배포할 때 발생하는 무결성 검사는 오프라인으로 전달하는 방법과 기존의 인터넷 익스플로러 기반의 코드 서명검증 방법이 있을 수 있다. 기존의 인터넷 익스플로러와 Verisgin에서 발급받은 인증서를 통해 독립 서명검증 모듈을 서명하여 내려 주는 방법이 코드 서명 모듈의 최초신뢰를 확보할 수 있는 현실적인 방안이다.

VI. 결 론

코드 서명 기술은 웹을 통해 배포되는 코드의 무결성과 배포자에 대한 인증을 제공해 주어 사용자들이 소프트웨어를 믿고 설치하여 사용할 수 있도록 하는 기술이다. 본 논문에서는 MS사의 코드 서명 기술인 Authenticode 기술을 분석하고, 코드 서명 기술을 국내 PKI에 적용하기 위한 코드 서명 인증서 프로파일을 제시하였다. 또한 인터넷 익스플로러를 통해 국내 인증기관이 발급한 인증서로 코드서명을 수행하기 위한 방안과 문제점들을 제시했다. 코드 서명기술은 유선인터넷뿐만 아니라 다양한 플랫폼과 응용프로그램에 적용될 수 있다. 웹을 비롯해 ftp, 전자 우편, 각종 패치 프로그램 배포 등의 소프트웨어 분야 뿐 아니라 중요한 뉴스나 공공기관의 공지사항 배포, 저작자가 명확해야할 각종 컨텐츠의 배포와 같이 내용의 무결성과 배포자에 대한 신원 확인이 보장되어야 할 여러 분야에 널리 활용될 수 있다. 하지만, 국내에서는 이에 대한 연구와 국내 기술 적용이 부족했으며, 코드 서명 기술을 사용하여 소프트웨어를 배포하기 위해서는 국외의 인증서를 비싼 가격에 구입하여 사용해야하기 때문에 재정적 손실이 큰 상황이다. 결국 이러한 이유로 배포자에 대한 인증이 이뤄지지 않은 테스트용 인증서를 사용하여 코드 서명 후 배포하거나, 심지어는 코드 서명 없이 사용자 컴퓨터로 소프트웨어 혹은 악성코드를 배포하여 설치토록 유도하는 경우가 급증하고 있다. 코드 서명 기술은 여러 유형의 통신과정에서 다운로드 되는 모든 유무선 컨텐츠의 무결성 검사와 배포자에 대한 인증을 위해 사용 되어질 수 있으므로 국내 공인 인증 체계를 바탕으로 한 코드 서명 기술의 개발이 필요하다. 이는 국내 정보보호시장의 활성화를 위해서도 필수적인 과제이다.

참 고 문 헌

[1] C. Adams, "Internet X.509 Public KeyInfrastructure Time-stamp Protocol", RFC 3161, 2001.

[2] RSA Laboratories, "PKCS #7 : Cryptographic Message Syntax Standard", 1997.

[3] RSA Laboratories, "PKCS #9:Selected Object Classes and Attribute type", 2000.

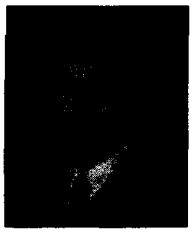
[4] Public Key Infrastructure X.509, http://www.ietf.org/html.charters/pkix-charter.html.

[5] R. Housley, "Internet X.509 Public Key Infrastructure Certificate and CRL profile" , IETF RFC 2459, 1999.

[6] "Information technology-ASN.1 Encoding Rules", ISO/IEC 8825-1, 1999.

[7] 한국정보보호진흥원, "전자서명 인증서 프로파일 표준", 2000

〈著者紹介〉



이 래 (Rae Lee) 정회원
2002년 2월 : 고려대학교 전산학 학사
2004년 2월 : 고려대학교 정보보호대학원 공학석사
〈관심분야〉 정보보호, 프로토콜, PKI, 코드보안, 전자정부, 디지털 방송, 게임보안

이 동 훈 (Dong Hoon Lee) 종신회원
1983년 8월 : 고려대학교 경제학사
1987년 12월 : Oklahoma University 전산학 석사
1992년 5월 : Oklahoma University 전산학 박사
1992년 8월~1993년 2월 : 단국대학교 전자계산학과 전임강사
1993년 3월~1997년 2월 : 고려대학교 전산학과 조교수
1997년 3월~2001년 2월 : 고려대학교 전산학과 부교수
2001년 3월~현재 : 고려대학교 정보보호대학원 교수
〈관심분야〉 정보보호, 암호이론, 프로토콜, 정보이론