코드 서명 기술과 인증서 프로파일 연구

이 래, 이동훈

고려대학교 정보보호대학원

Study of Code Signing Technology and Certificate Profile

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 다운로드 되는 소프트웨어에 대한 무결성과 배포자에 대한 신원 확인을 해주는 코드 서명 기술이 필요하게 되었다. 본 논문에서는 Microsoft사의 Authenticode와 Netscape의 Object Signing기술에 대하여 분석하고 이러한 코드 서명 기술을 국내 공개키 기반구조(PKI)에 적용하기 위한 인증서 프로파일을 제안한다.

I. 서론

점차 인터넷 기술이 발전하고 사회적 문화의 중요한 수단으로 자리 잡아 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. ActiveX Control이나 Java Applet등 의 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 되었다. 상점에서 판매되는 소프트웨어와는 달리 바이너리 코드 상태로 전송되는 인터넷에서는 프로그램을 실제로 누가 제작해서 배포하는지에 대해 신뢰할 수가 없으며, 다운로드 중에 프로그램이 악의적인 공격자에 의해 위조 또는 변조되거나 기타 네트워크 문제로 인해 에러가 발생하지 않았다는 보장이 확실치 않기 때문이다.

이렇게 웹 페이지를 통해 실행코드가 담긴 소프트웨어를 다운받는 과정에서 소프트웨어에 대한 무결성(Integrity) 검증과 배포자에 대한 인증(Authentication)을 제공해주는 것이 코드 서명 기술(Code Signing Technology)이다. 앞으로 이어질 내용에서는 코드 서명 기술의 기본적인 메커니즘을 설명하고, 현재 사용되고 있는 코드 서명 기술 중 대표적으로 Microsoft사의 Authenticode 기술과 Netscape사의 Object Signing 기술에 대해 비교 분석한다. 그리고 국내의 PKI에 활용 가능한 코드 서명용 인증서의 프로파일을 제안한다.

II. 코드 서명 기술

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 있

도록 해준다는 의미이다.

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 인증기관(CA)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)라는 메커니즘을 기반으로 이뤄진다.

2. 코드 서명 및 검증 과정

1) 서명

세부적인 코드 서명 기술에는 약간씩 차이가 있겠지만, 대략적인 코드 서명 과정은 그림 1과 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 RSA등이 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 만들어진다.

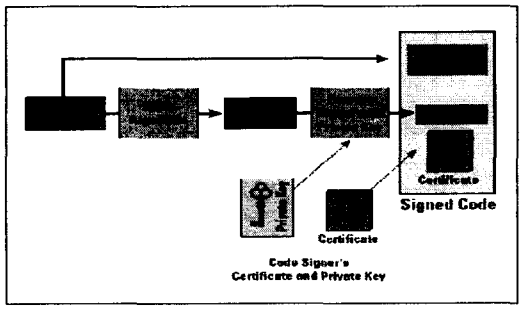

그림 1. 코드 서명 과정

2) 검증

서명된 코드는 소프트웨어 사용자의 컴퓨터에서 코드 서명 검증 프로그램을 통해서 소프트웨어 사용자에게 확인되어진다.

- Step 1. 인증서의 검사 : 인증서를 서명 블록으로부터 읽어온 후 정해진 형태의 유효한 코드 서명용 인증서인지 검사한다. 인증서의 검증에 관한 사항은 일반적으로 인증서 발급기관의 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.

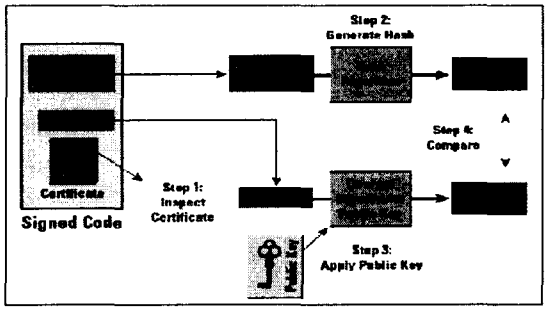

그림 2. 코드 서명 검증 과정

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.

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 것인지

의 여부를 가지고 인증서 검증과 서명 검증을 처리한다. 현재 코드 서명에 사용되는 타임스탬프 서비스는 Verisign CA[1]를 통해서만 받을 수 있으며 무료로 제공된다.

Ⅲ. Microsoft Authenticode 기술

Microsoft사(이하 MS)의 Authenticode 기술은 자사의 인터넷 브라우저인 Internet Explorer와 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, PKCS#9, X.509등의 표준과 SHA, MD5 등의 해쉬 알고리즘을 바탕으로 이루어져있다.[1]

1. MS의 코드 서명용 인증서

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)뿐이다. 9개의 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. 인증서 발급은 CA마다 차이가 있지만, 보통 업무일 3-6일정도의 기간이 소요되며, 가격은 VeriSign CA의 경우 상업용 소프트웨어 배포 인증서(Class 3)가 1년에 400$, Thawte CA의 경우 상업용 소프트웨어 배포 인증서가 1년에 199$, 2년에 399$이다.

MS는 코드 서명용 인증서는 크게 상업용과 개인용으로 나누어 발급하도록 하고 있다. 인증서의 신청과 발급 절차는 다른 종류의 공인 인증서와 유사하며 해당 CA의 인증업무준칙과 PKI 관련 표준에 의해 이뤄진다. 하지만, MS의 코드 서명용 인증서를 상업용과 개인용으로 나누어 발급하고 관리하도록 하는 MS의 기본 정책과는 달리 실질적으로 Verisign을 포함한 인증기관에서는 인증서를 대부분 상업용으로 발급하고 있는 상황이다. 개인용 인증서는 유명무실하게 된 실정이며 현재 사용되는 대부분의 코드 서명용 인증서는 상업용 인증서라 생각하면 된다. 그 이유는 개인용 인증서의 발급에 필요한 개발자의 신원 검증 비용이 수지 타산에 맞지 않기 때문이다.

2) MS 코드 서명용 인증서

MS로부터 인정된 CA들에게서 발급 받게 되는 코드 서명용 인증서는 X.509, RFC2459등의 표준에 준하는 인증서로, 다른 인증서들과 크게 차이가 나지는 않지만, 인증서의 확장필드의 내용 중 일부에 차이가 있으며 확장필드의 내용 중 표 1과 같은 항목들이 사용된다. 특히 MS에서는 확장키 사용을 critical로 하면 해당 용도로만 사용토록 하고, non-critical로 하면 다른 용도로도 사용할 수 있게 한다. 대신 키 사용 제한(Key Usage Restriction) 필드를 통해 MS는 인증서와 키의 사용 용도에 제한을 둘 수 있도록 하고 있다.[2][3]

표 1. MS 코드 서명용 인증서의 확장 필드

CA	확장 필드 항목	내용(값)
베리사인	Extended Key Usage(non-critical)	코드 서명 (1.3.6.1.5.5.7.3.3)
	Key Usage(critical)	digitalSignature(80)
Thawte	Extended Key Usage(non-critical)	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 (non-critical)	[1]Cert PolicyId =1.3.6.1.4.1.311.2.1.22 Restricted Key Usage =digitalSignature(80)

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 CRL 등을 포함 할 수 있는 형태의 패키지 형식이라 생각하면 된다. SPC는 전체적으로 PKCS#7 형식을 따르고 있으며 인증서의 배포를 위한 PKCS#7 signed data 형식을 따르고 있다. CA에게서 발급받는 코드 서명용 인증서는 SPC 형태로 발급된다. 실제 배포자의 코드 서명용 인증서뿐만 아니라, 이 인증서의 검증에 필요한 인증 경로의 CA의 인증서들도 포함된 패키지 형식의 인증서 모음이다. SPC는 배포자의 서명된 배포용 소프트웨어 내에 탑재되어 배포자가 누구인지, 그리고 그의 공개키가 무엇인지를 알려준다. 만약 소프트웨어

1) Verisign CA의 timestamp URL = http://timestamp.verisign.com/scripts/timstamp.dll

배포자가 X.509v3형식의 단일인증서로 발급 받게 된다면 Cert2SPC라는 툴을 통해 인증서를 SPC형식으로 변환할 수 있다. 또한 SPC형식 파일에서도 원하는 인증서만 X.509v3형태의 파일로 만들 수 있다.

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.

2. MS의 코드 서명 과정

코드 서명용 인증서를 CA로부터 발급 받은 후 MS의 웹 사이트를 통해 제공되는 코드 서명 프로그램 SignCode를 이용해 소프트웨어에 서명을 할 수 있다.

1) 코드 서명 과정 (SignCode 동작원리)

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

② 해쉬 값을 배포자의 개인키로 서명 생성, 서명 알고리즘은 RSA사용

③ 서명된 내용과 배포자의 인증서를 PKCS#7 signed-data object형태의 signature block으로 조합한 후 원래 서명할 원본 코드의 뒷부분에 첨가

2) 코드 서명의 검증

사용자가 인터넷으로부터 실행 소프트웨어(코드)를 다운로드 할 때, 브라우저 혹은 클라이언트의 특정 어플리케이션은 WinVerifyTrust()라 불리는 WIN32 함수를 사용한다. 이 함수는 서명된 값을 뽑아내고, 인증서의 유효성을 검증하며 배포자의 공개키를 사용하여 signature block에서 해쉬 값, 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.

3. 서명된 코드에 대한 분석

다음으로는 서명된 파일의 signature block이 어떤 형식으로 이뤄져 있는지 분석해 보았다. Verisign CA로부터 발급된 인증서로 서명된 파일을 샘플로 signature block만을 따로 떼어낸 후 이 부분을 DER로 분석하여 ASN.1형식으로 나타내었다. 그 결과는 그림 3과 같다. 그림 3의 ①번 부분을 통해 전체 블록의 형식이 PKCS#7 signed data 형식을 따르고 있다는 것을 알 수 있다. ②번 부분은 서명된 컨텐츠에 대한 정보가 들어가는 부분으로 contentInfo에 해당하는 부분이다. 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, timestamp등이 포함되어 있다. MS에서 등록하여 자체적으로 사용하고 있는 OID들에 대한 정보는 Microsoft SDK를 통해 습득할 수 있다. MS SDK에 포함되어있는 WinTrust.h 파일을 통해 Microsoft사에서 사용하는 OID의 정의 내용을 볼 수 있다.

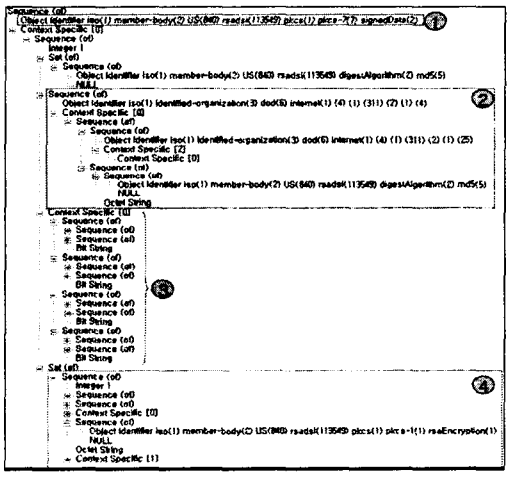

그림 3. 서명된 파일의 signature block

결론적으로 Authenticode 기술은 PKCS#7, PKCS#9, RFC2459, X509v3 등의 업계 표준에 의거하여 이루어진 기술임을 알 수 있다. 하지만, MS의 코드 서명 기술의 전체적이고 세부적인 알고리즘에 대한 정보가 있어야 좀더 정확한 안전성과 효율성의 분석이 이루어질 수 있을 것이다.

IV. Netscape Object Signing 기술

1. 개요

Netscape사의 Authenticode기술도 앞에서 살펴

봤던 Microsoft사의 기술과 유사하다. 새롭게 개발된 코드를 웹을 통하여 배포할 경우, 배포과정에서 소프트웨어가 손상되지 않았음을 공인된 인증기관에 의해서 검증받는다는 것은 MS사의 경우와 같다. 그러나, Netscape사의 경우는 SHA1, MD5의 알고리즘을 사용하여 서명된 값으로 JAR 파일을 만들고, JAR 파일을 가지고 서명된 값이 정당한 값인지 확인하여 코드의 무결성을 확인한다.

2. Netcsape의 코드 서명용 인증서

Netscape의 코드 서명용 인증서의 발급 절차나 가격은 MS의 코드 서명 인증서와 유사하다. 소프트웨어의 배포를 원하는 배포자는 먼저 Netscape 네비게이터 인증서 관리소에 포함된 Verisign CA, Thwate CA등 특정 인증기관(CA)를 통하여 코드 서명용 인증서를 발급 받을 수 있다. Netscape이 인증서 발급 절차나 가격은 MS의 인증서 관련 내용에서 설명한 내용과 유사하므로 생략한다.

Netscape로부터 인정된 CA들에게서 발급 받게 되는 인증서도 기본적으로 X.509v3 표준에 의거한 인증서이다. Netscape의 코드 서명과 관련된 확장필드는 Netscape Cert Type으로 이 필드를 통해 인증서를 활용하는 응용 프로그램의 제한을 부여한다. 만약 인증서에 Netscape Cert Type이라는 필드가 있다면, 그것이 critical이든, non-critical이든 상관없이 Netscape에서는 Netscape Cert Type에 명시된 목적으로만 인증서를 사용하도록 제한하고 있다. 만약 Netscape Cert Type 필드가 존재하지 않는다면 코드 서명을 제외한 다른 어떤 용도로든 응용 프로그램에서 사용할 수 있다.

표 3. Netscape 코드 서명 인증서 확장 필드

확장 필드 항목	내용(값)
Netscape Cert Type (critical or non-critical)	Object Signing(10) 또는 Signature(10)
Key Usage (critical)	digitalSignature (80)

Netscape 코드 서명용 인증서는 Netscape Cert Type라는 확장필드를 갖도록 하고 있으며, 코드 서명용 인증서는 Netscape Cert Type이 Object Signing(10), 코드 서명용 인증서 발급 CA의 인증서는 Object Signing CA(01)을 갖도록 정하고 있다. 또한 Netscape만의 확장필드 이외에도 X.509v3, RFC2459에 명시된 키용도(Key Usage)라는 확장 필드가 있는데, 이 필드도 용도에 따라 필요한 값을 갖게 되어있다. RFC2459에서는 이 키 용도라는 확장필드에 대해 "인증서에 존재하면 critical이어야한다"라고 명시하고 있다. 하지만, Netscape에서는 키용도(Key Usage)가 critical로 되어있고 인증서에 존재한다면 그 인증서와 해당 공개키는 키용도(Key Usage)에 명시된 용도로만 사용하게 되고, 반면 키용도(Key Usage) 확장필드가 없거나, non-critical로 되어있다면 그 인증서와 해당 공개키는 어떤 용도로든 사용 할 수 있다고 명시하고 있다.[4]

3. Netcsape의 코드 서명 과정 및 결과 분석

Netscape 코드 서명을 위해서는 MS와 유사한 방법으로 먼저 인증서를 CA에게서 구입한 후 인증서를 Netscape Communicator에 저장하고 signtool이라는 프로그램을 이용하여 코드 서명을 실행한다.

1) 코드 서명 과정

코드 서명을 하기 위해서 배포를 원하는 파일들을 새로운 폴더를 만든 후 그곳으로 옮기고 signtool 프로그램을 이용해 서명을 실행한다. signtool 명령을 수행하면, 비밀 키에 접근하기 위한 패스워드를 입력한 후 코드 서명 작업이 완료된다. 그리고 코드 서명의 결과로 jar파일이 생겨난다. 본 테스트에서는 test라는 폴더 안에 T1NO.EXE와 readme.txt를 넣어 코드 서명을 수행했고 그 결과 test.jar이라는 파일을 얻을 수 있었다. 코드 서명의 결과로 만들어진 test.jar 파일에 관해 분석하겠다.

2) 코드 서명 결과 분석

코드 서명 결과 만들어진 test.jar 파일의 압축을 풀어보면 META-INF라는 폴더가 T1NO.EXE와 readme.txt의 원본 파일에 추가된 것을 알 수 있다. 특히 T1NO.EXE와 readme.txt의 원본 파일은 확인결과 아무런 변화가 없었으며 단지 META-INF라는 폴더만 추가된 것임을 알 수 있었다. 새롭게 추가된 META-INF 폴더에는 manifest.mf, zigbert.rsa, zigbert.sf의 총 세 개의 파일이 담겨 있었다.

먼저 manifest.mf 파일의 내용을 알아보면, 그림 4에서 ①번 부분은 사용 툴에 관한 정보와 코멘트가 나와있는 부분이며, ②번과 ③번 부분은 각 원본 파일에 대한 해쉬 알고리즘별 해쉬값이 Base64 코딩법에 의해 기재 되어있는 부분이다. 같은 내용의 원본 파일을 이름만 달리해 코드 서명을 해 본 결과, 같은 해쉬 값이 동일하게 기재되는 것을 추가적인 테스트로 확인할 수 있었다.

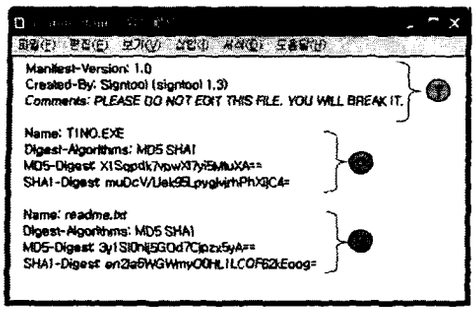

그림 4. manifest.mf의 내용

다음으로 그림 5의 zigbert.sf 파일 내용을 보겠다. zigbert.sf의 경우 ①번 부분에도 해쉬 값이 기재되어있는 것을 볼 수 있다. ①번 부분의 해쉬값은 manifest.mf 파일의 전체 내용을 ASCII 코드로 읽어 들인 후 해쉬한 값이다. zigbert.sf의 ②번, ③번 부분은 manifest.mf의 ②번과 ③번 부분을 각각 ASCII 코드로 읽어 들인 후 해쉬한 값이다. 이렇게 Netscape의 코드 서명 과정에서 manifest 파일(manifest.mf)과 signature 파일(zigbert.sf)을 따로 두는 것은 코드의 생산자와 코드의 서명자가 다를 수도 있고, 또 개발자 나름대로의 관련 정보를 추가하여 배포자에게 전달할 수 있으며, 배포자 또한 배포나 서명과 관련된 정보나 정책을 포함시킬 수 있기 때문이다.

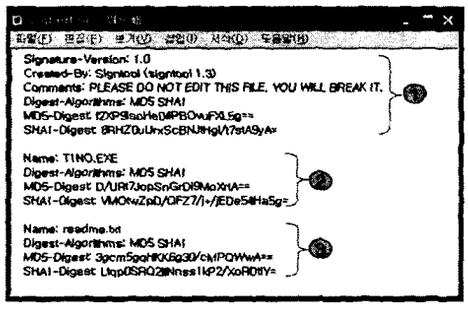

그림 5. zigbert.sf의 내용

이제 마지막으로 zigbert.rsa를 살펴보겠다. 이 파일은 이미 설명한 manifest.mf, zigbert.sf와 같이 직접 읽을 수 있는 형태의 파일이 아니며, PKCS#7의 Signed Data형식에 따라 DER 인코딩 법에 의해 만들어진 파일이므로 MS의 코드 서명 분석에서와 같이 ASN.1 형식으로 분석하였다.

그림 6 에서 ①번 부분을 통해 전체 구조가 PKCS #7 signed data형식을 따르고 있음을 알 수 있다. 위 결과는 테스트용 인증서를 사용해서 서명을 한 결과인데 이때 쓰인 테스트용 인증서가 코드 서명용 인증서를 발급할 수 있는 CA의 역할도 할 수 있는 CA 인증서이기 때문에 중간에 Certificates, Crls 부분의 내용이 없고 Content Info 다음에 Signer Info가 곧장 이어진다. ②번 부분이 모두 Signer Info 부분으로 서명자의 인증서 정보와 서명내용이 저장되는 부분이다. ③번 부분은 서명을 실시한 사람의 인증서가 발급된 CA의 정보와 코드 서명에 사용된 인증서의 일련번호가 들어있는 부분이다. ④번 부분은 서명되어진 내용에 관한 정보가 들어있는 부분으로, 서명시간과 원래 코드의 해쉬 값이 저장되어있다. ④번 부분의 아랫부분에 있는 Octet String은 바로 서명되어진 내용의 해쉬 값으로, zigbert.sf 파일의 전체 내용에 대한 해쉬 값이 기재되어있는 부분이다. 즉 서명의 대상으로 zigbert.sf 파일의 전체 내용에 대한 해쉬 값이 사용된다는 것이다. ⑤번은 이 해쉬 값을 배포를 원하는 서명자의 개인키로 암호화한 서명 값과 서명 알고리즘이 함께 저장되어 있는 부분이다.

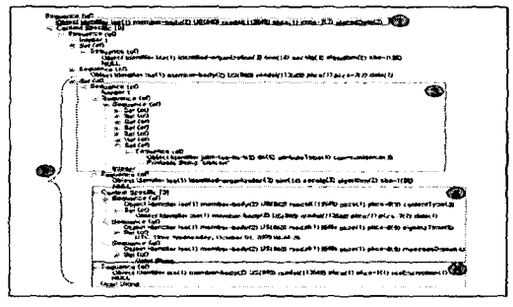

그림 6. zigbert.rsa 파일 분석

결론적으로, Netscape 코드 서명 기술에서는 JAR파일의 전자서명 추가기능을 바탕으로 PKCS#7 Signed data 형식의 zigbert.rsa에 서명값을 저장하게 된다.

특히 코드 서명 시 활용할 수 있는 타임스탬프 측면에서 MS와 Netscape의 코드 서명 기술을 비교해본다면, MS는 타임스탬프를 추가할 경우 외부 타임스탬프 서비스를 받을 수 있는 URL을 입력하도록 요구하고 있지만, Netscape의 경우에는 타임스탬프를 외부 공인인증기관을 통해 제공받도록 URL형식의 입력을 지원하지 않는다. Netscape의 코드 서명 기술에서도 타임스탬프가 사용되지만, 서명자의 컴퓨터에서 발생하는 시간 정보를 사용하여 타임스탬프의 역할을 하도록 하고 있다. 이것은 소프트웨어의 배포자가 시간정보를 조작하여 코드 서명을 할 수 있기 때문에 보안상 취약점이 될 수 있다.

V. 국내 PKI에 적합한 코드 서명용 인증서 프로파일

1. MS와 Netscape 코드 서명 인증서

MS와 Netscape의 코드 서명 기술에서 필수로 필요로 하는 인증서의 확장 필드는 매우 상이하다. 그 이유는 RFC2459에 명시된 인증서 확장 필드 내용 중 확장 키 사용(Extended Key Usage) 필드에 대한 MS와 Netscape의 사용법이 다르기 때문이다. MS의 경우는 확장 키 사용 필드를 통해 코드 서명용 인증서를 나타내고 있지만, Netscape에서는 Netscape Cert Type라는 필드를 통해 인증서 용도를 제한하고 있다. 그러므로 코드 서명 인증서는 서로 호환될 수가 없다.

2. MS와 Netscape에 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. 우리나라의 전자서명 인증서 프로파일의 표준에 의거해 기본적인 프로파일을 구성하고 MS의 Authenticode 기술과 Netscpae의 Object Signing 기술에서 필요로 하는 확장필드 내용을 추가하여 수정하였다. 반드시 포함되어야 할 주요 확장 필드(mandatory extension field)는 표 4와 같다. 기본 필드와 나머지 확장필드는 국내 전자서명 인증서 프로파일을 따르면 된다. 특히 MS의 코드 서명용 인증서에 필요한 확장필드는 Verisign CA에서 발급하는 인증서의 형태를 따라 구성하였다. 확장 필드중 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. 또한 현재 국내 전자서명용 인증서로도 코드 서명이 가능하기 때문에 코드 서명용 인증서만이 코드 서명 가능하도록 인증서 정책 필드 등을 이용하여 문제를 해결해야 할 것이다. [5]

VIII. 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 MS와 Netscape사의 코드 서명 기술을 분석하고, 코드 서명 기술을 국내 PKI에 적용하기 위한 코드 서명 인증서 프로파일을 제시하였다. 코드 서명 기술은 여러 유형의 컨텐츠 무결성 검사를 위해 널리 사용될 수 있으므로 국내 공인 인증 체계를 바탕으로 한 코드 서명 기술의 개발이 필요하다.

표 4. MS와 Netscape에 호환 가능한 국내 코드 서명용 인증서 프로파일

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

C : Critical NC : Non-critical

참고문헌

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

[2] "Internet X.509 PKI - Qualified Certificates Profile", IETF work in progress, 2000.

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

[4] http://wp.netscape.com/eng/security/certs.html

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