韓國 OA學會 論文誌
第7卷 第4號, 2002. 12.

2002-7-4-1-10

디지털 서명을 위한 XML 구조 설계

표 성 배*

A Design of XML Structure for Digital Signature

Sung-bae Pyo*

요 약

웹 기반 하에서의 문서유통의 경우 문제가 되는 것은 작성된 문서가 표준화되어 있지 못하다는 것이고, 또 다른 한가지는 기밀 자료의 유출이며 의도적이고 악의적으로 유통되는 문서를 변조하려는 공격에 대한 대처가 미흡하다는 것이다.

최근에는 마그업 언어인 SGML로부터 HTML을 만들어내고 이를 웹개발에 이용하였으나 많은 사용자들이 웹에서 사용하는 문서나 메시지나 데이터의 정의를 위한 방법들을 표준하기 미흡하였다. 그러나 XML이 등장하면서 이러한 문제를 해결하고 있으나 문서 유통의 안전을 보장하기 어려웠다.

본 연구는 웹환경 하에서 업무처리에 필수적인 문서유통에서 전자서명을 이용한 정보보호를 위하여 XML을 기반으로 하여 전자서명의 절차를 규정하고 컴포넌트들을 설계하려 한다.

Abstract

Circulation of documents and data has two serious problems in this web-based system. The first of the problems is standardization problem which the flowing documents and data are not always standardized, the other is security problem which secret data can be hacked or cracked, and non-authenticated person can access and change the contents with malicious intent.

Therefore two issues come out on the web-based system, the first is how we can standardize the documents and data and the second is how we can assure safe flowing of them. One way of the standardizing methods is Mark-Up language which uses XML which is extended from SGML to redefine and use documents. And to ensure security of flowing documents we adopt digital signature system using public key and private key.

The focus of this study is to specify procedures for digital signature on the XML base, and to design XML components for the digital signature to assure information security in circulation of documents and data.

* 인덕대학 소프트웨어개발과

논문접수 : 2002. 10. 4
심사완료 : 2002. 12. 4

Ⅰ. 서론

인터넷을 이용한 생활환경이 나날이 확대되어 가고 기업의 업무나 개인 생활의 환경에서도 점차 원격처리가 가능한 웹 기반을 기본으로 급격하게 전환되고 있다. 따라서 컴퓨터 환경에서 이러한 웹 환경 하에서의 업무나 문서, 자료 등의 표준화 확대를 위하여 엄청난 노력을 기울이고 있다. 기업과 기업 간의 문서교환, 자료 유통, 기업과 고객간의 의사전달 등 그 필요성은 점차 증대하고 있으며 이를 위하여 마크업 언어인 SGML[1]을 제안하였고 그 중 일부의 set을 추출하여 웹페이지를 제작할 수 있는 HTML[2]로 개편하였다.

그러나 이러한 HTML만으로는 완벽하게 자신만이 사용하는 독특한 방식의 문서나 의사를 전달하고 전달받기에 어려움을 겪으면서 표준화의 필요성을 느끼게되었다. 이에 따라 XML[3]이라는 마크업 언어를 제안하게 되었고 이제는 웹 기반의 모든 프로그래밍에서 이를 받아들이고 사용하게 되었다.

한편으로 웹 환경에 의해 많은 중요한 문서나 자료들이 흘러 다니게 됨에 따라 이를 공격하고 중요한 자료를 절취하는 사례도 빈번하여져서 정보보호를 소홀히 할 수 없는 상황에 이르게 되었다. 우리가 보유하고있는 중요한 자료들에 대하여 이를 소중하게 보호하고 이를 공공의 통신망을 통하여 안전하게 외부로 전달하기 위한 노력을 가중하게 되었다. 이 노력의 일환으로 자료를 암호화[4]하는 연구와 전자서명[5]에 사용에 관한 연구들이 활발히 진행되고 있으며 네트워크의 보안으로 방화벽[6]이나 바이러스에 관한 연구 등도 매우 활발히 연구들이 진행되고 있다.

본 논문은 많은 기업들이 관심을 표명하고 있는 XML과 웹기반 업무 시스템에 관하여 웹 환경 하에서의 취약부분인 문서 및 자료 표준화를 위한 부분과 웹 문서 유통의 안전성 보장을 위한 방안으로 사용중인 XML언어를 기반으로 하는 전자서명 관련 알고리즘을 개발하고 이를 사용하기 위한 절차와 검증에 관한 컴포넌트들을 제시하고 이러한 컴포넌트들을 효율화하기 위한 연구를 하려는 것이다. 본 논문에서 사용하는 알고리즘의 일부는 W3C (World Wide Web Consortium)의 웹사이트인 Http://www.w3.org의 RSA와 SHA-1등의 알고리즘들을 원용하였다.

Ⅱ. 관련연구

1. One-way 함수

일방향 또는 단방향 함수라고 부른다. 함수 y=f(x)가 일방향 함수라고 하면 x를 알고 y를 계산하는 것은 매우 쉽지만 y를 알고 x를 알기는 매우 어렵다. 이러한 종류의 함수의 예는 "y = x mod n"이다. 여기서 n=10이라고 하고 x를 11이라고 하면 y는 1이 된다. 하지만 y가 1이라는 사실을 알면 x는 1, 11, 21, 31, ...이 되어 결국 하나의 x 값을 알기는 거의 불가능하다.

2. 해쉬(hash) 함수

one-way 함수의 특성을 갖는 함수이다. y=f(x)를 해쉬 함수라고 하면 이 함수가 갖추어 야하는 조건은 다음과 같이 요약될 수 있다. 단 여기서 x,y는 이진수로 표시된 임의의 수이다.

① x의 길이에는 제약이 없다.

② y의 길이는 정해져 있다.

③ one-way 함수의 특성을 갖는다.

④ Collision(synonym)이 없어야 한다.

즉, 서로 다른 모든 x에 대하여 계산된 모든 y 값은 서로 같지 않아야 한다.

one-way 함수의 예로 제시한 "y = x mod n"이라는 함수는 흔히 사용되는 해쉬 함수이며 일반적으로 y = H(x)로 표기한다.

3. Trap door

Trap door는 일종의 "알려지지 않은 비밀문"을 말한다. 예를 들어 8자리의 숫자와 문자를 패스워드로 입력하여 시스템에 로그인하는 과정에서 그 패스워드 인증시스템을 만든 사람이 "12345678"라는 패스워드를 입력하면 무조건 로그인이 가능하도록 만들었다면 바로 이

"12345678"라는 패스워드가 일종의 그 인증 시스템의 trap door인 것이다. 이와 함께 trap door를 갖는 one-way 함수라 함은 어떤 특정한 y 값에 대해서는 x 값을 계산하는 것이 쉬운 특성을 갖는 one-way 함수를 뜻한다.

4. 비밀키(Secret Key) 암호 기법

예로부터 전해 내려오는 거의 모든 암호기법이 비밀키 암호기법에 속한다. 이것은 단순히 데이터를 하나의 비밀키로 암호화와 복호화를 모두 하는 기법이다. 이 비밀키는 대칭키(Symmetric Key) 암호기법이라고도 불린다.

여기에 속하는 대부분의 암호기법들은 간단한 비트 연산을 반복하는 작업을 통해 암호화가 이루어지기 때문에 단위시간당 암호화할 수 있는 데이터 양이 비교적 많다. 따라서 대용량의 데이터를 취급하는데 적합하다. 그러나 하나의 비밀키에 대한 의존도가 너무 크기 때문에 그 응용성에 있어서 제약이 많이 따른다. 비밀키 암호기법에서는 생성된 비밀키를 메시지의 송/수신자에게 안전하게 전달할 수 있는 채널(secure channel)이 필요하다. (그림 3)은 비밀키 암호기법의 기본적인 mechanism을 설명하고 있다.

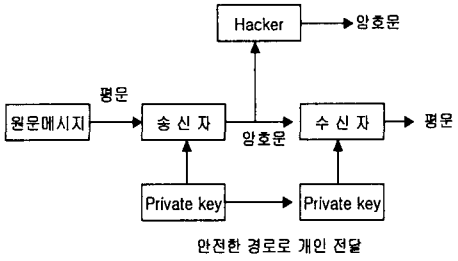

그림 2 비밀키 암호기법
fig 1. private key cryptography

그림 1에서 표시된 "Hackers"는 정확히는 암호분석가(Cryptanalyst)를 의미한다. 암호분석가는 암호화된 메시지가 돌아다니는 통신채널에서 암호화된 메시지를 얻어서(wire tapping) 이것을 복호화하기 위한 키값을 얻어내는데 주력한다. 암호 분석가가 사용하는 공격방법들에는 알려진 것들이 많이 있다. 실제로 이들의 응용에서는 확률이론에 근거한 여러 가지 방법들이 사용되며 이러한 암호기법들 중에 대표적인 것으로는 DES와 IDEA, CLIPPER, SKIPJACK 등이 있으나 Skipjack에 관하여는 추측만 가능할 뿐 알려진 사항이 별로 없다.

5. 공개키(Public Key) 암호 기법

비밀키 암호기법에 있어서 모든 키들은 그것을 사용하는 두 사람이외에는 아무도 알 수 없도록 비밀리에 보관되어야 하므로 만약 수많은 사람들이 동시에 상호 메시지를 주고받고자 하는 경우에는 이러한 비밀키의 관리가 매우 어려워진다.

이와 같이 암호화를 위한 키의 생성과 전달 그리고 보관하는 문제 즉, 키의 관리 문제는 암호를 이용한 정보보안에 있어서 중요한 이슈가 되는 문제이다.

비밀키 암호기법이 갖는 위와 같은 키관리의 문제를 해결하기 위해서 제안된 것이 바로 공개키 암호 기법이다. 이것은 1976년 Whitfield Diffie와 Martin Hellman에 의해 제안되었다. 이것을 이해하기 위해 다음의 그림 2를 보자. 먼저 메세지의 송신자와 수신자는 각각 두개의 키를 할당받는다. 그 두개의 키중에 하나는 비밀키(Ks)이고 나머지 하나의 공용키(Kp)이다. (그림 6)의 Krp는 수신자(Receiver)의 공개키, Ksp는 송신자(Sender)의 공개키를 의미하고 Kss는 송신자의 비밀키를 그리고 Krs는 수신자의 비밀키를 의미한다.

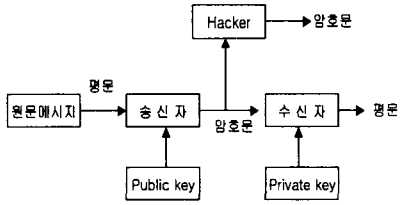

그림 3 공개키 암호 기법
fig 2 public key cryptography

비밀키는 메세지의 송/수신에 참여하는 모든 사람들이 개인적으로 비밀리에 보관하고 있어야 하고 공개키는 공개키 디렉토리에 보관되어 모든 사람들이 자신이 원하는 공개키를 얻을 수 있어야 한다. 그리하여 송신자가 메세지를 보내기 위해 먼저 공개키 디렉토리에서 메세지 수신자의 공개키를 획득한다. 그리고 송신하려는 메세지를 수신자의 공개키를 이용하여 암호화한 후 발송한다. 암호화된 메세지를 받은 수신자는 자신만이 갖고 있는 비밀키를 이용하여 그 메세지를 복호화하여 메세지를 얻는다.

이러한 공개키 알고리즘의 복잡도는 다음의 표1과 같다.

분 류		복잡도	연산횟수
poli-nomial	constant	O(1)	1
	linear	O(n)	10^6
	quadratic	O(n^2)	10^{12}
	cubic	O(n^3)	10^{18}
exponential		O(2^n)	10^{301030}

표 1. 공개키 알고리즘 복잡도
table 1. public key algorithm computational complexity

Ⅲ. 디지털서명을 위한 고려사항

1. 디지털 서명 개요

문서의 전자적인 교환이 일반화되면서 문서의 인위적인 변조나 여러 가지 결함에 의한 망실에 의한 문제점들이 생겨나게 되었다. 이러한 문제들은 그 문서가 갖는 특성상 법적인 문제가 발생할 때는 더욱더 복잡해진다. 따라서 일반적인 종이문서에 상호 서명이나 도장을 찍는 것과 같이 전자문서의 송신자와 수신자간에 문서내용에 서명을 하는 방식이 필요하게 되었다. 이러한 서명 방식들 중 지금까지 설명한 암호학적 기법을 응용하여 만든 방식을 디지털 서명이라 부른다.

디지털 서명이 전자서명의 범주에 포함되는 개념이기는 하지만 일반적으로 말하는 전자서명은 서명자의 지문이나 기타 이미지 데이터를 만들어서 이것을 데이터베이스에 저장한 후 필요에 따라 꺼내어 복사하는 방식의 mechanism을 갖는다. 이 경우에는 저장되어 있는 이미지 데이터를 서명자이외의 사람이 그 데이터를 얻을 수 있다면 무제한 복사가 가능해서 마치 남의 도장을 훔쳐서 마음대로 서명하는 것과 같은 결과를 초래할 수도 있다.

디지털 서명을 이해하기 위해서 먼저 일반적인 서명이 갖는 특징에 대해 정리를 할 필요가 있다. 흔히 우리가 말하는 서명(손으로 하는 서명)의 특징은 다음과 같이 요약될 수 있을 것이다. 즉 위조불가성(Unforgability) 서명자 이외의 다른 사람이 그 어떤 방법으로도 그 서명을 위조할 수 없어야 한다. 즉, 하나의 서명은 반드시 한사람만이 생성할 수 있어야 한다.

2. 디지털 서명의 제약 조건

위절의 설명과 같은 이유로 전자서명은 다음과 같은 제약 사항들을 고려하여 검증할 수 있어야 한다.

- 인증성(authenticity)

서명된 메시지가 반드시 메시지의 송신자에 의해 서명된 것인지를 확인할 수 있어야 한다. 이것은 특히 법적인 문제가 발생하였을 경우 중재가 가능해야 한다.

- 부인 봉쇄 (non-repudiation)

서명자는 자신의 서명에 대하여 서명한 사실을 부정할 수 없어야 한다. 즉, 실제로 서명자 자신이 서명을 한 문서에 대해서 서명을 한 적이 없다고 말할 수 없게 증거를 제시할 수 있는 기능이 있어야 한다는 것이다.

- 재사용 불가성(non-reusability)

한번 사용한 서명은 그것을 다시 사용할 수 없어야 한다. 즉, 복사가 불가능해야 한다.

- 문서의 변경불가성(message integrity)

서명된 문서는 그 내용이 변경되어서는 안된다. 즉, 문서의 내용이 바뀌었는데도 그 전의 서명이 유효하면 안된다.

위에서 정리한 내용은 손으로 하는 서명의 기능적인 특징이기도 하거니와 디지탈 서명이 갖추어 야할 요구조건이기도 하다. 그러나 디지털 서명은 '0'과 '1'로 이루어진 비트열이므로 손으로 하는 서명이외의 몇 가지 추가적인 특성을 갖는다. 즉 손으로 하는 서명은 그 사용자에 따라 항상 일정하지만 디지털 서명은 서명하려는 메시지에 따라 생성되는 결과가 다르다. 이러한 특성을 메시지 의존성이라 부른다(이 말은 한 사람이 여러 개의 서명을 갖는다는 의미는 결코 아니다). 바로 이러한 특성으로 말미암아 메시지의 송수신시 메시지의 내용이 변경이 되면 수신자가 메시지 내용이 변경되었음을 확인할 수 있게 된다.

디지털 서명은 서명과정과 서명확인과정에 참여하는 사람들이 누구인가에 따라서 크게 직접서명방식과 간접서명방식으로 나누어진다. 직접서명방식은 메시지의 송신자가 서명을 하고 수신자가 서명을 직접 확인하는 방식이다. 간접서명방식은 메시지의 송신자가 서명을 하고 그 메시지와 서명을 중재자에게 보내면 중재자는 서명을 확인한 후 그 결과를 수신자에게 보낸다. 물론 직접서명방식에서도 문제가 생기면 중재자를 필요로 하지만 메시지

를 주고 받을 때마다 중재자가 개입을 하는 간접서명방식에 비해 과정이 간단하고 통신량이 적은 장점이 있다.

또한 디지털 서명은 사용하는 암호기법의 종류에 따라서 비밀키 암호기법에 기반을 둔 방식과 공개키 암호기법에 기반을 둔 방식의 두 가지로 나누어진다. 디지털 서명을 위해서는 앞서 설명한 암호화 기법에 대한 기술과 더불어 메시지의 압축에 관한 기술들을 이용하여 보다 효율적인 처리를 할 수 있다.

3. 기본적인 서명 메카니즘

앞서 말한 바와 같이 디지털 서명은 적용하는 암호기법의 종류에 따라서 비밀키 암호기법을 이용하는 것과 공개키 암호기법을 이용한 것 것이 있다. 전자의 경우는 송신자가 수행하는 서명과정과 수신자가 행하는 서명확인 과정에서 동일한 비밀키를 이용한다. 따라서 Diffie-Hellman 키 분배 방식과 같은 방법을 이용하여 송신자와 수신자가 공유하는 비밀키를 생성하는 과정이 선행되어야 한다. 후자의 경우에는 송신자는 자신의 비밀키를 이용하여 서명을 하고 수신자는 송신자의 공개키를 이용하여 서명을 확인한다.

디지털 서명의 기본적인 메카니즘을 다음의 그림 3에 나타내었다. 여기서는 공개키 암호 기법을 사용한다는 것을 전재로 그림을 설명해보기로 한다.

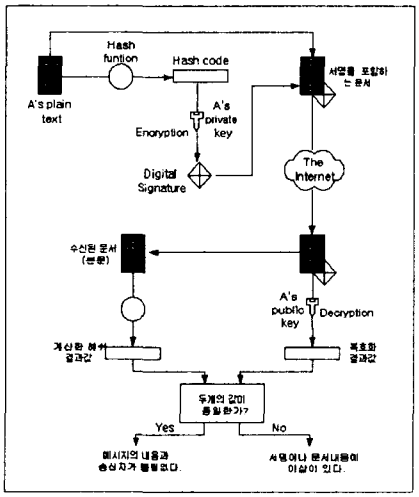

그림 4 디지털서명 처리흐름도
fig 3. Digital Signature Processing Flow Diagram

4. 디지털서명의 검증

송신자 A는 해쉬 알고리즘을 통해 문서의 해쉬 코드를 구하고 이것을 자신의 비밀키로 암호화한다. 암호화된 해쉬코드가 문서에 대한 A의 서명 값이다. A의 비밀키로 암호화를 하였으므로 A의 공개키를 가지고 있는 모든 사람이 검증할 수는 있으나 A의 비밀키를 알지 못하므로 위조는 할 수 없다. 해쉬코드에 의해서 무결성이 제공된다. 송신자는 문서에 서명 값을 첨가하여 전자서명이 포함된 문서를 수신자에게 보낸다.

수신자는 문서와 서명 값을 분리한 후, 해쉬알고리즘을 통하여 수신된 문서의 해쉬코드를 계산한다. 수신자는 수신된 서명 값을 A의 공개키를 이용하여 복호화한다. 복호화된 해쉬코드와 수신자가 계산한 해쉬코드를 비교하여 이상이 없으면 A의 서명이 포함된 문서임을 확신한다.

앞에서 서명이 갖추어 야할 요구조건들이 위의 서명/서명확인 과정에서 만족되는지를 알아보자.

- 위조 불가성

공개키 암호기법의 키생성 과정에서 생성된 비밀키는 서명자 자신 외에는 아무도 알 수 없는(알 수 없어야하는) 값이므로 이 조건은 만족된다.

- 인증성

이것은 서명 확인 과정의 "비교" 기능에서 만족된다.

- 부인 봉쇄

수신자는 자신이 수신된 문서를 따로 보관하고 있다가 나중에 송신자가 자신이 직접 서명한 사실을 부인할 경우 증거물로 사용할 수 있다.

- 재사용 불가성

수신된 문서는 문서에 첨부된 서명을 사용한 해시함수를 이용한다. 따라서 다른 메시지에 대하여 서로 다른 결과를 출력하므로 이것을 다른 메시지의 서명으로 사용하는 것은 불가능하다.

- 문서의 변경 불가성

서명 확인 과정에서 알 수 있듯이 원문의 내용이 서명 과정이 끝난 후에 암호화된 후 변경된다면 서명 확인과정을 거쳐 복호화되고 나면 "비교" 결과가 같지 않게 된다.

Ⅳ. XML 디지털서명을 위한 컴포넌트 구성

1. XML signature structure

XML Signature란 XML을 기반으로 하는 Digital signature을 위한 정의된 구문을 말하는 것으로 이것은 이 구문들 자체가 하나의 완전한 XML프로그램이라기보다 XML tree 구조의 단지 특정한 부분을 서명하는 능력을 갖는 것을 말한다. 이 서명은 문서 내에 존재하며 이 서명으로 인하여 언제라도 안전하게 복원하게 될 것이다. 그러나 서명에 의해 사용자에게 안전하게 전달되어야 하지만 만약 기존의 서명 값들을 사용자가 훼손하거나 변경하는 경우 그 기존의 원본 서명은 무효가 될 것이다.

XML 서명은 자원의 하나 이상의 문서 유형에 서명이 가능하다. 예를 들면, 하나의 XML 서명은 문자자료뿐인 HTML 문서, 2진 데이터형식의 JPG 자료, XML로 변환된 자료 등 그 다양한 종류의 문서에 각각 서명을 할 수 있다. XML 문서에서의 서명을 위해서는 다음과 같은 구조를 가져야 한다.

```
〈Signature〉
  〈SignedInfo〉
    ((CanonicalizationMethod))
      ((SignatureMethod))
        ((Reference (URI = " "))
          ((Transforms))
          ((DigestMethod))
          ((DigestValue))
        〈/Reference〉)
        〈SignedInfo〉
      ((SignatureValue))
      ((KeyInfo))
      ((Object))
〈/Signature〉
```

리스트1. 디지털 서명을 위한 XML 구조
list 1. XML structure for Digital Signature

위 리스트에서 〈Signature〉element는 XML Signature의 parent element이며 주어진 문맥 속에서 XML Signature가 완전함을 입증한다. 이 〈Signature〉element에는 자식 엘리먼트로 〈SignedInfo〉 〈SignatureValue〉, 〈KeyInfo〉, 〈Object〉등이 이 순서대로 포함될 수 있으며 이 〈Signature〉 element는 다음의 두가지 점에서 중요하다. 첫째는 만약 한 문서 내에 여러 개의 〈Signature〉 구조가 필요할 때 이러한 optional element들이 중요하다는 것이고 둘째는 스키마 검증을 시도하는데 최대의 노력의 검증형태라는 점에서 중요하다.

이러한 엘리먼트들 중에서 알고리즘이나 데이터객체가 필요한 엘리먼트의 경우에는 참조할 URI를 명기함으로 이를 이용할 수 있도록 구조화하고 참조할 모듈들을 컴포넌트화 함으로 구조를 단순하도록 만들었다.

1) 〈SignedInfo〉 엘리먼트

이 element는 가장 복잡한 element로 〈Signature〉 element의 가장 먼저 나타나는 자식 엘리먼트이다. 이 엘리먼트에는 서명의 안에 포함되어있는 각 데이터객체를 참조할 것들을 포함하고 있다. 이곳에는 〈CanonicalizationMethod〉, 〈SignatureMethod〉, 〈Reference〉 엘리먼트가 순차적으로 포함되며 〈CanonicalizationMethod〉, 〈SignatureMethod〉 이 두 엘리먼트는 〈SignatureValue〉를 생성하는 Canonicalization 알고리즘과 Signature 알고리즘의 내용을 포함하고 있다.

```
〈Signature〉
  〈SignedInfo〉
    〈CanonicalizationMethod algorithm=
"http://www.w3.org/TR/2001/
REC-xml-c14n20010315"〉
      〈SignatureMethod algorithm =
"http://www.w3.org/2000/09/xmldsig#rsa-sha1"〉
        〈Reference URI =
          "http://www.pyosb.pe.kr"〉
          〈/Reference〉
        〈SignedInfo〉
            :
〈/Signature〉
```

리스트 2. SignedInfo element 작성예
list 2. example of SignedInfo element

2) 〈CanonicalizationMethod〉엘리먼트

이것은 empty 엘리먼트로 last tag를 사용하지 않는다. 단지 Canonicalization 알고리즘이 존재하는 URI를 나타내는 속성을 가질 뿐이다. 물론 이 알고리즘도 W3C 그룹에서 이미 만들어진 내용을 사용할 수 있다.

3) 〈KeyInfo〉 엘리먼트

이 엘리먼트에는 XML Signature를 검증하기 위해 사용하는 특정 정보들이 포함되어있으며 이 엘리먼트를 이용하여 공개키나 x.509인증 등과 같은 법적으로 공인된 것뿐만 아니라 원격지에 존재하는 공개키를 이용할 수 있도록 만들어준다. 다음그림4에 보는 것과 같은 절차에 따라 〈KeyInfo〉 엘리먼트가 사용된다.

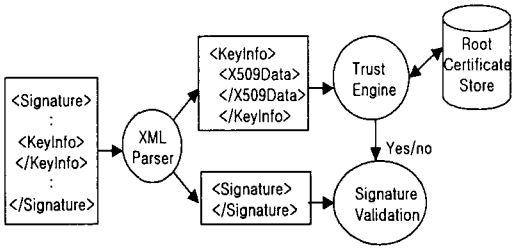

그림 5. 〈KeyInfo〉엘리먼트 활용
fig 4. usage of 〈keyInfo〉element

이 엘리먼트의 자식엘리먼트로는 〈Keyname〉, 〈KeyValue〉, 〈RetrivalMethod〉, 〈X509Data〉, 〈PGPData〉, 〈SPKIData〉, 〈Mgmt Data〉들이 있으며 각각의 내용은 다음 테이블 2와 같다.

엘리먼트명	내 용
〈KeyName〉	키의 이름
〈KeyValue〉	RSA 또는 DSA공개키
〈RetrivalMethod〉	원격지에 있는 공개키정보
〈X509Data〉	X.590인증/이름/관련된데이터
〈PGPData〉	PGP 관계된 키
〈SPKIData〉	SPKI 키와 인증 관련데이터
〈Mgmt Data〉	키 어그리먼트 파라메타

표 2. KeyInfo 엘리먼트 child 엘리먼트
table 2. child element of 〈KeyInfo〉 element

4) 〈Object〉 엘리먼트

이 엘리먼트는 XML signature를 처리하기 위해 중요한 속성들을 나타내게 되는데 이 속성에는 Id, MimeType, Encoding 이 있다. Id속성은 참조해야할 객체의 이름을 나타내는 것으로 〈SignedInfo〉엘리먼트의 〈Reference〉엘리먼트에서 Object에 관한 사항을 정의하고 있어야 한다. MimeType은 적용해야할 데이터의 형식을 정의하는 속성이다. Encoding 속성은 실제 Encoding에 적용해야할 알고리즘을 참조할 주소를 나타내기 위해 사용한다.

```
〈Object Id="TestPicture"
  MimeType="image/gif"
  Encoding="../xmldsig#base64〉
```

리스트 3. 〈Object〉엘리먼트의 예
list 3. example of 〈Object〉 element

5) 〈Menifest〉 엘리먼트

이것은 〈Object〉 엘리먼트의 자식 엘리먼트로 잘 정의된 엘리먼트이다. 이것은 다양한 서명절차와 복잡한 패키지로 된 서명을 위하여 유연한 해결책을 제시해 주는 강력하고도 유용한 엘리먼트이다. 이것은 〈Reference〉 엘리먼트를 단순히 수집하는 정도의 개념으로 사용되며 이것도 단순히 그 자체가 자원으로 사용된다. 이것은 결론적으로 검증절차 도중에 서명전환을 거쳐 검증하는데 사용하는 엘리먼트이다. 다음의 예제 리스트와 같이 작성한다.

```
〈Object〉
 〈manifest Id="docuList"
  〈Referece
    URI="http://www.pyosb.pe.kr/Docu.pdf"〉
      :
  〈/Reference〉
  〈Referece
    URI="http://www.pyosb.pe.kr/Docu.hwp"〉
      :
  〈/Reference〉
 〈/Manifest〉
〈/Object〉
```

리스트 4. 〈Manifest〉 엘리먼트 작성 예
list 4. example of 〈Manifest〉 element

6) 기타 엘리먼트

〈Reference〉엘리먼트는 가장 중요한 URI 속성을 가지고 있으며 digest될 메시지들을 참조할 URI를 알려주는 역할을 한다. 이 엘리먼트는 〈Transforms〉, 〈DigestMethod〉, 〈DigestValue〉라는 자식 엘리먼트를 가질 수 있으며 이 속성들은 메시지를 틔 signature 기능을 이용하여 메시지를 Digest로 변환하도록 만드는 속성들의 기능들을 갖도록 한다.

V. XML 디지털서명과 검증을 위한 절차

1. XML 디지털서명을 위한 절차

XML서명을 위한 절차로는 다음의 두가지 중요한 수행 절차로 나타낼 수 있다. 이 중 먼저 수행해야할 수행절차는 〈Reference〉 엘리먼트를 만들어내는 것이다. 이 Reference를 통하여 각 메시지 자원에 서명을 해야하는데 다음과 같은 것들을 고려해야한다.

(1) 서명해야할 자원을 선정
(2) 각 메시지 자원을 Digest화하는 작업 (변환 작업 적용)
(3) 〈Reference〉엘리먼트를 만들거나 기존에 만들어진 엘리먼트를 수집.

두 번째 수행절차는 서명(Signature)을 만드는 것으로 다음과 같은 절차를 수행하면 된다.

(1) 앞 단계에서 만든 〈Reference〉 엘리먼트를 사용하여 〈SignedInfo〉 엘리먼트를 작성하고 이 때 SignatureMethod와 CanonicalizationMethod를 함께 작성
(2) 〈SignedInfo〉엘리먼트에 〈CanonicalizationMethod〉와 〈SignedMethod〉를 적용
(3) 적시에 생성된 〈SignatureValue〉를 사용한 부모인 〈Signature〉요소와 기존에 생성되었던 〈SignedInfo〉 엘리먼트에 모든 선택적 엘리먼트들과 속성들을 작성

이상과 같은 절차를 차례로 수행하면 디지털 서명이 첨부된 메시지를 만들게 되며 이 메시지를 원하는 곳에 안전하게 전송할 수 있게 되는 것이다.

2. XML 디지털서명을 검증하기 위한 절차

이 검증 절차는 서명을 생성하기 위한 절차보다 조금 더 복잡하다. 먼저 reference 검증을 수행하고 난 뒤에 signature 검증하는 절차를 수행한다. reference 검증을 위해 제일 먼저 할 일은 〈SignedInfo〉엘리먼트 반드시 규범화시키는 것이다. 그리고 각 〈reference〉 엘리먼트들을 검증하기 위해 다음과 같이 수행하도록 한다.

(1) Reference Validation 수행절차

단계1. Digest화된 데이터 스트림은 각 〈Reference〉 엘리먼트의 URI속성을 참조하여 검증을 완수한다. 만약 URI가 존재하지 않는다면 이 경우에는 반드시 데이터 소스의 주소를 알고 있어야 한다.

단계2. 단계1에서 작성된 데이터 스트림은 현재의 처리되는 〈Reference〉엘리먼트를 위하여 〈DigestMethod〉 엘리먼트에서 정의된 해시함수를 사용하여 반드시 Digest화 한다.

단계3. 단계2에서 계산된 digest value는 현재의 처리되는 〈Reference〉엘리먼트를 위하여 〈DigestValue〉 엘리먼트를 가지고 서로 비교한다. 만약 이 값들이 서로 같지 않다면 이 reference validation은 실패한 것이다.

(2) Signature Validation 수행절차

단계1. 〈KeyInfo〉엘리멘트나 특정한 키가 존재하는 곳으로부터 검증 키를 검색한다.

단계2. 〈SignatureMethod〉의 규범양식을 사용하여 사용된 서명 알고리즘을 선택하고 〈SignedInfo〉엘리먼트의 규범양식 위에 서명 값을 계산한다.

단계3. 단계2에서 계산된 서명값과 〈SignatureValue〉 엘리멘트에 있는 값을 비교하여 만약 두 개의 값이 서로 같지 않으면 Signature Validation은 실패한 것이다.

VI. 결론

본 연구는 XML 문서에서의 각 메시지 그룹들 간에 자료의 기밀성 보장을 위한 디지털 서명을 적용하는 방법을 제안한 것이다. 이 연구로 XML 로 디지털 서명을 하기 위한 XML psrser tree 구성 방법을 제안하였으며

공개키와 개인키를 이용한 디지털 서명을 나타내기 위한 속성들을 제안하였으며 이를 구현하기 위한 방법들을 제시하고 있다.

또한 디지털 서명을 위한 서명의 절차와 검증의 절차를 제시함으로 이를 통하여 디지털서명을 이용하여 보안에 취약한 구조일 수밖에 없는 웹기반 업무처리 및 문서유통에 있어 이를 XML로 표현할 경우 정보보호의 개념을 적용할 수 있는 방안을 제시하였다.

향후 이 Encrypting 부분과 Decrypting 부분을 SAX를 통하여 API형 컴포넌트로 제작함으로서 이를 필요로 하는 많은 웹기반의 시스템들에게 이를 참조하게 하여 도움을 줄 수 있을 것으로 생각된다.

참고문헌

[1] Burnett, Steve and Stephen Paine. RSA Security's Official Guide to Cryptography. McGraw-Hill, 2001

[2] Eastlake, D,J.Reagle, and D.Solo. "XML-Signiture syntax and Processing" "Http://www.w3.org" 2001

[3] Boyer, J, D. Eastlake, and J. Reagle. "Exclusive XML Canonicalization" W3C Working Draft. Oct. 2001.

[4] Boyer, J. Canonical XML Recommendation. "http://www.w3.org/tr/2001/rec-xml-c14n-20010315"

[5] Beech, D., M. Maloney, N.Mendelsohn and H. Thompson. "XML Schema Part I: Structures" W3C Recommendation may. 2001

[6] Stinson, Douglas R. "Cryptography : Theory and Practice" Boca Raton, FL: CRC Press, 1995

[7] Allen Brown, Barbara Fox, Satoshi Hada, Brian LaMacchia, Hiroshi Maruyama "SOAP Security Extensions : Digital Signature" W3C note Feb. 2001 "http://w3.org/TR/2001/NOTE-SOAP-dsig-20010206

저 자 소 개

표 성 배

1979년 숭실대학교 전산학과 졸

1984 - 1990 국방품질연구소 선임연구원

1992 - 현재 인덕대학 소프트웨어개발과 부교수

1997 - 현재 숭실대학교 전자계산전공 박사 수료

관심분야 :

Network security,

Web solution development,

Image processing,,

Multi-media 모델링