유효성을 고려한 XML 데이타 암호화 시스템의 설계 및 구현

(Design and Implementation of on XML Data Encryption System considering Validation)

남 궁 영 환 † 박 대 하 †† 허 승 호 ††† 백 두 권 ††††

(Young-Hwan Namkoong) (Dae-Ha Park) (Seung-Ho Hurh) (Doo-Kwon Baik)

요 약 XML은 정보공유 및 검색에 있어서 매우 효과적인 장점을 지닌 마크업 언어이지만 보안에 취약한 단점을 가지고 있다. 한편, 이를 보완하는 XML 전자 서명, XML 데이타 암호화, XML 접근 제어 등의 연구는 XML 문서의 유효성을 배제하고 있다. 그러나 XML 기반 정보 교환 및 공유 환경에서 XML 문서의 유효성은 필수적으로 요구된다. 본 논문에서는 XML 문서의 보안성과 유효성을 동시에 지원하는 XML 보안 시스템을 설계하고 구현하였다. 제안된 시스템은 XML 데이타의 암호화 과정에서 유효성 유지를 위해 XML 스키마 정보를 갱신하여 이의 참조를 통해 XML 문서의 유효성 지원한다. 또한 XML 스키마 전자 서명과 같은 XML 스키마 관련 보안 기능을 지원하며 이 과정에서 빠른 정규화 XML 스키마 해석을 위해 DOMHash 기법을 사용하였다. 제안된 시스템을 통해 유효성을 유지하는 XML 문서는 유연성 및 확장성과 신뢰성에 있어 기존 시스템보다 좋은 기능을 갖는다.

키워드 : XML, XML 보안, XML 유효성, XML 스키마, DOMHash

Abstract XML(eXtensible Markup Language) is effective to information retrieval and sharing but has defects related to the data security. And, as a solution of this problem, the current XML security researches such as XML digital signature, XML data encryption, and XML access control exclude the validation property of XML document. The validation of XML should be considered for the secure information sharing in the XML-based environment. In this paper, we design and implement the system to support both security and validation to XML document. Our system performs data encryption and maintenance of valid status of XML document by referencing new XML schema namespace. In addition, it also provides the XML schema security function through the XML schema digital signature. During generating XML schema digital signature, DOMHash method which has the advantage of the faster speed than canonical XML method is applied to XML schema. In conclusion, our system shows the improved functions in flexibility, scalability, and reliability compared with the existing XML security researches.

Key words : XML, XML security, XML validation, XML Schema, DOMHash

1. 서 론

XML(eXtensible Markup Language)[1]은 문서의 표현과 내용의 분리를 통해 네트워크 상에서 정보 표현 및 공유에 효과적인 언어이다[2]. 최근 XML/EDI에 이어 ebXML이 전자 비즈니스(e-business)를 위한 표준 모델이 되는 등 전자 비즈니스의 모든 표준들은 XML기반으로 재구축되고 있다. 표현의 편의성, 구조적 계층성, 데이타의 의미부여 등 다양한 장점을 지닌 XML은 현대 인터넷 환경에서 필수 불가결한 요소로 자리잡았다.

한편, 전자 비즈니스나 전자 상거래(e-commerce) 시

† 비 회 원 : University of Southern California
circinus@dreamwiz.com
†† 정 회 원 : (주)시큐리티테크놀로지스 책임연구원
dhpark@stitec.com
††† 비 회 원 : (주)시큐리티테크놀로지스 선임연구원
deuxists@stitec.com
†††† 종신회원 : 고려대학교 컴퓨터학과 교수
baik@software.korea.ac.kr
논문접수 : 2001년 5월 31일
심사완료 : 2002년 9월 16일

스템 관련 애플리케이션의 수준을 판단하는 중요한 척도는 '보안성'이다. 즉, 불특정 다수가 이용하는 인터넷 환경에서 개인 사생활 정보와 같은 높은 보안 수준을 요구하는 데이타들의 전송 및 처리 용량이 급증함에 따라 이에 대한 적절한 대책이 요구되고 있다. 이러한 측면에서 XML은 데이타에 대한 의미를 구현하고 전달하는 장점을 지닌 반면, 보안성에 대해 매우 취약한 단점을 드러내고 있다. XML은 데이타의 구조화에 초점을 두고 개발되었기 때문에 보안 측면은 상대적으로 매우 간과되어 있다. 이러한 구조적 문제점을 해결하기 위해 XML에 대한 여러 보안 기법이 제안되었다. 이러한 보안 기법으로는 XML 전자 서명[3], XML 데이타 암호화 기법[4,5], XML 접근 제어[6,7,8]가 있다. XML 전자 서명은 XML 문서 작성자에 대한 정보 및 XML 데이타를 전자 서명 방식을 이용하여 XML 문서에 첨부한 형태로 변환시켜 교환하는 방식이다. 이 방법은 정보 수신시 전송자에 대한 서명 데이타를 기반으로 내용의 무결성 뿐만 아니라 전송자에 대한 인증을 보장하는 장점이 있으나, 전자 서명 기능만으로는 문서내의 데이타에 대한 기밀성 유지가 허술하다는 단점을 가지고 있다. XML 데이타 암호화 기법은 대칭키 암호화에 기반을 둔 방법으로, XML 문서 내의 데이타를 다양한 암호화 알고리즘을 이용하여 암호화된 형태로 송신하고 수신자가 복호화 함으로써 데이타의 기밀성을 유지한다. [4]는 보안에 요구되는 엘리먼트를 자체적으로 정의하고 이에 기반한 엘리먼트 및 데이타를 암호화하는 방법을 사용하고 있으며, [5]는 스타일시트 형식의 시큐리티 시트(Security Sheet) 방법을 제안하였다. 이러한 방법들은 XML 보안에 효과적이지만 엘리먼트 값이 변조될 경우 애플리케이션에서 이를 식별하지 못하여 주요 정보가 그대로 유출될 수 있는 단점을 안고 있다. 특히 [5]는 보안 데이타의 효과적인 교환 및 공유를 위해 작성된 마크업 언어로 실질적인 데이타 암호화 기능을 제공하지 않는다. XML 접근 제어 기법은 불특정 다수에게 개방되어 있는 XML 문서에 대한 승인(authorization)을 통하여 사용자의 접근 수준에 따라 문서를 가공한 결과를 제공하는 방법으로 사용자 제어에 매우 효과적이나 계정 접근 공격을 통해 사용자 접근 수준을 변조할 경우 역시 데이타 노출 가능성이 있다.

기존의 XML 보안 기법의 특징은 XML 데이타에 중점을 두고 제안된 내용들이다. 따라서 XML 문서 내의 데이타에 대해서는 효과적인 보안 장치라고 볼 수 있다. 그러나 전자 비즈니스 및 전자 상거래 환경에서 요구하는 XML은 XML DTD(Data Type Definition)와 XML 스키마(Schema)등을 기반으로 작성된 XML 문서를 요구하고 있다. 왜냐하면 이들을 기반으로 궁극적인 정보의 공유 및 교환이 이루어지기 때문이다[13]. 따라서, 이들에 대한 적절한 보안 장치가 요구되고 있다. XML DTD를 보호하기 위해 DTD를 파싱하여 먼저 전자 서명을 생성한 후 이를 XML 문서에 첨부하여 XML 문서의 구조에 대해서도 신뢰성을 보장하는 방법도 고려해볼 수 있으나 현재 XML 표준이 지속적으로 바뀌고 있고 DTD에 대한 많은 제약으로 인해 XML DTD보다는 XML 스키마에 대한 보안 장치가 더욱 중요하게 부각되고 있다.

본 논문에서는 XML 스키마에 대한 보안 장치를 기반으로 한 XML 데이타 암호화 시스템을 설계하고 구현하였다. XML 스키마에 보안성을 부여하기 위해 XML 문서의 기반이 되는 XML 스키마에 대한 전자 서명의 생성, 유지 및 검증을 위한 모듈과 XML 데이타의 암호화를 위해 웹 브라우저에서 원하는 데이타에 대한 암호화를 수행할 수 있는 모듈을 지원한다. 그리고 이 모듈들 이외에 암호화된 XML 문서가 유효성을 갖지 못하는 단점을 XML 스키마를 이용하여 보안성과 유효성을 동시에 유지하도록 하였다. 이상의 제안된 방법을 통해 XML 기반의 정보 교환 및 공유시 안전도와 보안성의 수준을 높일 수 있다.

본 논문의 구성은 다음과 같다. 2 절에서는 XML 보안을 위한 다양한 관련 연구에 대해 알아보고, 3 절에서는 제안된 XML 보안 기법, 4 절에서는 제안된 시스템의 설계 및 구현 결과를 살펴본다. 5 절에서 기존 연구와의 비교 평가를 논하고 마지막으로 6 절에서 결론을 맺는다.

2. 관련 연구

2.1 XML과 XML 스키마

2.1.1 XML

XML[1]은 문서에 대한 향상된 구조화를 위해 제안된 마크업 언어로, 문서의 표현에만 중점을 두었던 HTML과 이상적인 구조화 언어임에도 불구하고 지나친 일반화를 위해 높은 복잡도의 단점을 지닌 SGML의 장점만을 취하여 개발되었다[2]. 따라서 인터넷 환경에서 데이타 교환 및 공유와 검색에 있어 매우 효과적이다. XML은 XML 문서 및 그 기반이 되는 XML DTD 또는 XML 스키마로 구성되며, 웹 브라우저상의 표현을 위해 스타일시트 언어가 따로 개발되어 있다. 또한, XML로 표현된 데이타는 SQL과 같은 형식의 일반 질의어로 데이타 검색, 갱신, 삭제를 처리하는데 많은 제약이 있다. 따라서 이를 보완하는 효과적으로 처리하기

위한 질의언어로 XML-QL과 XQuery등이 제안되었다. XML은 웹 환경에서 데이타 처리에 있어 매우 효과적인 언어이다. 그러나 계층적 구조를 유지하기 위해 대량의 데이타를 표현할 경우에 동일한 엘리먼트를 반복해서 명시해주어야만 하기 때문에 공간의 낭비를 초래하는 단점도 지적되고 있다. 이 경우에 XML과 관계형 데이타베이스를 연계하거나 eXcelon의 XIS와 같은 XML 전용 데이타베이스를 이용하면 효과적인 데이타 관리가 가능하다.

2.1.2 XML 스키마

XML 스키마[9]는 기존의 XML DTD에서 드러난 많은 문제점을 보완한 새로운 문서타입 정의 방식이다. 다양한 방식의 데이타 타입을 지원하며, 정규 표현을 이용한 사용자 정의 타입도 가능하다. 그리고 XML 스키마는 XML 구문법을 따르므로 문서 작성에 있어 일관성이 있다는 점이 장점이다. XML 스키마는 모든 XML 처리 도구에서 구현될 수 있도록 설계되었기 때문에, 애플리케이션에 대한 제약 사항이 없다[2]. 한 가지 주의할 점은 엘리먼트 선언에 있어 DTD와는 다르게 그림 1과 같이 다양한 형태의 선언이 가능하다는 점이다. 따라서 이러한 다양한 형태의 엘리먼트 선언에 대하여 공통적인 형태로 정규화하는 방법이 요구된다.

```
i) <xsd:element name="name" type="type"
    minOccurs="int" maxOccurs="int"/>
ii) <xsd:element name="name" minOccurs="int"
   maxOccurs="int">
        <xsd:complexType>
          ...
        </xsd:complexType>
      </xsd:element>
ii) <xsd:element name="name" minOccurs="int"
   maxOccurs="int">
        <xsd:simpleType>
          <xsd:restriction base="type">
            ...
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
```

그림 1 다양한 XML 스키마 내의 엘리먼트 선언 방법

2.2 XML 보안의 개요

XML은 인터넷 환경에서 원활한 데이타 교환을 위해 요구되는 다양한 애플리케이션에 적합하다. 인터넷은 불특정 다수를 위한 네트워크로 보안상의 취약점을 지니고 있으며, XML은 문서의 데이타 구조화에 중점을 두고 만들어진 마크업 언어이기 때문에 문서 변조 및 데이타 삭제 등의 공격에 노출되어 있다. 최근 XML의 비중이 높아짐에 따라 XML 보안은 과거 웹(Web) 보안의 일부분에서 독립하여 새로운 분야로 분류되었다. XML 보안은 XML 전자 서명, XML 데이타 암호화, XML 접근 제어로 분류된다. 또한 보안 관련 데이타의 공유 및 교환을 위하여 XML 스키마에 기반하여 S2ML 또는 SAML 등의 마크업 언어도 제안되었다.

2.2.1 XML 전자 서명

XML 전자 서명은 문서의 부인봉쇄(non-repudiation)를 지원해주는 전자 서명 방법을 XML에 적용한 것으로 XML 문서의 해시값을 계산하고 이것을 서명자의 개인키로 암호화한 결과를 서명값으로 활용한다. XML 전자 서명에서 주요 고려사항은 공백 문자 처리, 속성 기본 값, 문자 인코딩이 다른 XML 문서와 비교하여 논리적으로 내용이 동일하다면 같은 전자 서명값을 생성해야 한다는 점이다. 이를 위해 정규화 XML(canonical XML)과 DOMHash 기법이 제안되었다[10, 11]. 정규화 XML은 XML 문서를 논리적으로 동일한 형태로 인식할 수 있는 규칙을 제정한 것으로, XML 문서에서 발생 가능한 예외 부분을 특정 규칙에 따라 처리하여 논리적으로 동일한 형태의 XML 문서로 변환시키는 방법이다. 이 방법은 XML 문서를 논리적으로 동일하게 변환함으로써 전자 서명의 신뢰성을 향상시켰지만, 기본적으로 ASCII 코드를 지원하고 있어 Unicode 표준을 따르는 문자는 인코딩을 수행하기 어렵다. DOMHash를 이용한 서명 생성 방법은 XML 파싱에 이용되는 구조 중 하나인 DOM(Document Object Model)[12] 구조에 기반하여 해시값을 생성하는 방법으로, 기본적으로 유니코드의 일종인 UTF-16으로 인코딩을 지원하여 정규화 XML의 문자 인코딩 단점을 극복하였고, 정규화 XML 생성 과정의 필요 없이 표준 DOM API를 이용하여 해시값을 계산하는 방법을 정확하게 정의할 수 있는 장점을 지니고 있다. DOMHash를 이용하여 계산된 DOM 노드의 해시값은 해당 노드의 서브 트리를 모두 포함한 것으로, 해시값이 다를 경우 서브 트리 상의 다른 부분이 존재함을 파악할 수 있게 해준다.

2.2.2 XML 암호화 기법

[4,5,14]은 기존의 암호화 기법을 XML에 적용시켰다. XML 문서 전체의 암호화 기법으로 시작하였으며, 속도 문제의 단점이 지적되어 현재는 엘리먼트 단위로 암호화를 수행하는 향상된 기법이 제안되었다. [4]는 자체적으로 정의한 엘리먼트에 기반하여 XML 문서의 엘리먼트 암호화를 수행한다. 이 방법은 부분적인 데이타 암호화 연산을 하므로 속도 및 비용 면에서 효과적인 반면, 암호화 후 갱신된 DTD를 지원해주지 못하여 DTD선언이 포함된 유효한 XML 문서(valid XML document)를

처리해주지 못하는 단점이 있다. [5]는 세 가지 방식의 XML 데이타 보안 기법을 제안하였다. 암호화 엘리먼트의 <secure> 엘리먼트 대체 방법, 데이타 값만 암호화하는 방법, XML 스타일시트(XSL)를 이용하는 방법이 그것이다. <secure> 엘리먼트 대체 방법은 암호화 후 XML 문서의 구조가 깨질 수 있는 단점이 있으며, XML 데이타 값 암호화 기법은 엘리먼트 선언 내에 불리언(boolean) 타입의 보안 속성을 설정 후 이를 이용하는 방법인데 데이타만 암호화가 가능하지만 DTD 속성에 불리언 타입이 정의되어 있지 않아 DTD 문법을 위반한다. XML 스타일시트를 이용하는 방법은 암호화를 위한 스타일 문서를 정의하여 강력한 보안 기능을 지원 가능한 장점이 있으나 암호화를 위한 특정 컨텐츠를 분리시키는 컴포넌트 없이는 암호화가 불가능하며, 암호화 후 갱신된 DTD를 지원하지 못한다. 한편, DTD의 전자 서명을 XML 문서에 추가하고 기존의 DTD에 기반한 XML 문서의 데이타를 암호화하며 이에 따른 XML 문서를 위한 XML 스키마를 생성하여 문서의 무결성을 지원하며 DTD를 기반으로 작성된 XML 문서에 대해서도 XML 스키마를 지원하는 방법도 고려해볼 수 있는데 이 방법은 XML 문서에 대한 유연성 및 확장성이 좋은 장점을 지니고 있는 반면 현재 국제 표준화 동향으로는 XML 구문법과 비교하여 일관성이 없고 비확장적인 DTD를 배제하고 XML 스키마를 XML 문서를 위한 DTD로 대체하고 있어 시기적으로 필요성이 배제되는 단점이 있다.

2.2.3 XML 보안 마크업 언어

다양한 운영체제와 애플리케이션 언어가 혼합되어 사용되는 특성을 가진 인터넷 환경에서 XML은 효율적인 정보 교환을 위한 포맷을 제시하였고 이를 기반으로 웹 기반의 전자 비즈니스 환경의 핵심 기술로 성장하였다. 특히, 인터넷상에서 특정 비즈니스 업무를 수행하기 위해서 다른 어플리케이션에 데이타와 서비스를 제공하는 웹 기반의 어플리케이션 컴포넌트로 '웹 서비스(Web Service)'가 등장하였다. 이러한 웹 서비스의 핵심으로 SOAP(Simple Object Access Protocol)이라는 프로토콜이 사용되고 있기는 하지만, *B2B 환경에서 SOAP이* 모든 걸 해결주지는 않는다. 즉, SOAP은 단순히 교환 메시지 포맷이며 프로토콜일 뿐 보안 특히 인증 서비스를 제공하지 못하고 있다. 이에 대한 보완책으로 SAML (Security Assertion Markup Language) 및 PKI에 기반을 둔 인증 프로토콜인 *XML-DSig(XML Digital* Signature)을 웹 서비스에서 통합하는 방안을 모색하고 있다. 여기서 SAML 또는 S2ML(Security Sheet Markup Language)은 데이타 보안에 관련된 메시지 포맷을 지원하는 sXML에서 파생된 마크업 언어이다. 즉, 이러한 XML 보안 마크업 언어는 XML 스키마를 기반으로 작성되었다. 따라서 보안 정보 교환에 있어 요구되는 메타 데이타만을 가지며 데이타 보안 또는 전자 서명, 인증 작업과는 다른 영역으로 취급된다.

3. 유효성을 유지하는 XML 보안 기법

XML 보안에서 동시에 고려해야할 문제는 데이타에 대한 안전한 보호와 정형화된 문서의 구조 유지이다. 이를 위해 XML 데이타 암호화 기법은 엘리먼트 단위의 데이타 암호화 기법[4]이 현재까지 최적의 방법으로 인식되고 있다. 그러나 보안 측면만 강조된 기존의 연구는 문서 교환의 핵심 속성인 유효성을 배제하고 있는 문제점을 안고 있다. 본 논문에서는 XML 스키마를 기반으로 작성된 XML 문서에 대하여 엘리먼트 단위의 데이타 암호화 기법을 적용시키고 원본 XML 스키마와 변환된 XML 스키마를 동시에 지원하는 방법을 제안한다. 이는 XML 문서 내에서 XML 스키마의 네임스페이스(namespace)를 이용하여 다중 XML 스키마의 정의가 가능하기 때문에 하나의 XML 문서에 변환 XML 스키마와 원본 XML 스키마, XML 스키마를 위한 메타 스키마 등 여러 가지 XML 스키마를 참조함으로써 효과적인 문서 처리가 가능하다.

3.1 XML 데이타 암호화 기법

1 절에서 기술한 바와 같이 전자 비즈니스 및 전자상거래 환경에서 유효성은 필수적으로 요구된다. 따라서 유효성을 잃은 경우 XML 문서는 정보 교환 및 공유에서 많은 문제점을 나타낼 수 있다. 그러므로 XML 데이타 암호화 작업은 필수적인데, 이러한 XML 데이타 암호화는 엘리먼트 단위로 이루어진다. 본 논문에서 제안하는 방법은 기본적으로는 [4]에서 제안한 엘리먼트 기반의 XML 데이타 암호화 기법을 따른다. 다만 [4]에서 제시된 방법과는 약간의 차이를 보이는데, [4]는 XML 데이타 보안을 위해 XML 엘리먼트를 자체 정의하여 이를 기반으로 데이타 암호화를 수행하였으나 제안하는 방법은 엘리먼트를 보존하는 상태에서 데이타 암호화만을 수행하는 방법을 이용한다. 이는 유효성을 유지하는데 있어 가장 단순하면서도 효과적인 방법으로 데이타 암호화 이후에도 XML 문서의 구조를 그대로 유지하는 장점이 있다. 그림 2는 데이타가 암호화되기 이전과 이후의 상태를 나타내는 XML 문서의 예이다. 그림에서 사용자의 신용카드 정보만 암호화 이후 스트링 형태로 변환되었을 뿐 엘리먼트는 변환되지 않았다. 이렇게 하

```
<?xml version="1.0"?>
<BookStore xmlns ="http://www.books.org"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://www.ecampus.com
           BookStore.xsd">
  <Checkout>
    <Userid>Hermes77</Userid>
    <ShippingAddress>3311W.3rd st.#2-202 LA, CA 90023</ShippingAddress>
    <Date>July. 30th. 2001</Date>
    <TotalPrice>$147.22</TotalPrice>
    <CardNumber>9430-120215-43892</CardNumber>
    <ExpirationDate>11/04</ExpirationDate>
  </Checkout>
</BookStore>
```

(가) 암호화 전의 XML 문서

```
<?xml version="1.0"?>
<BookStore xmlns ="http://www.books.org"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://www.ecampus.com
           BookStore.xsd">
  <Checkout>
    <Userid>Hermes77</Userid>
    <ShippingAddress>3311W.3rd st.#2-202 LA, CA 90023</ShippingAddress>
    <Date>July. 30th. 2001</Date>
    <TotalPrice>$147.22</TotalPrice>
    <CardNumber>AthI4n+GOoo6p/WV8dNJyZY4rjD4Tt6QJfrbBjS7FaKjouxojyBHw==</CardNumber>
    <ExpirationDate>11/04</ExpirationDate>
  </Checkout>
</BookStore>
```

DES 알고리즘으로 생성된 암호화된 데이터

(나) 데이터 암호화 후의 XML 문서

그림 2 원본 XML 문서와 데이타 암호화 후의 XML 문서

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.ecampus.com"
            xmlns="http://www.ecampus.com"
            elementFormDefault="qualified">
  <xsd:simpleType name="CreditCardType">
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="4-4-4-4"/>
      <xsd:pattern value="4-6-5"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:element name="BookStore">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Checkout" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="Userid" type="xsd:string"/>
              <xsd:element name="ShippingAddress" type="xsd:string"/>
              <xsd:element name="Date" type="xsd:gYear"/>
              <xsd:element name="TotalPrice" type="xsd:decimal"/>
              <xsd:element name="CardNumber" type="CreditCardType"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

```
<xsd:simpleType name="CreditCardType">
  <xsd:restriction base="xsd:string">
  </xsd:restriction>
</xsd:simpleType>
```

그림 3 원본 XML 스키마와 데이타 암호화 후 수정되어야 할 XML 스키마의 일부

는 이유는 데이타 암호화에 관련된 정보는 XML 문서와는 별개의 내용이기 때문에 [4]와 같이 XML 엘리먼트에 정보를 표현할 필요가 없기 때문이다. 즉, 정보 교환 과정에서 애플리케이션간의 암호화 키 및 알고리즘 정보를 XML에서 분리하여 전송하는 것이 데이타 보호에 있어 안전도를 높일 수 있기 때문이다.

3.2 XML 스키마 갱신

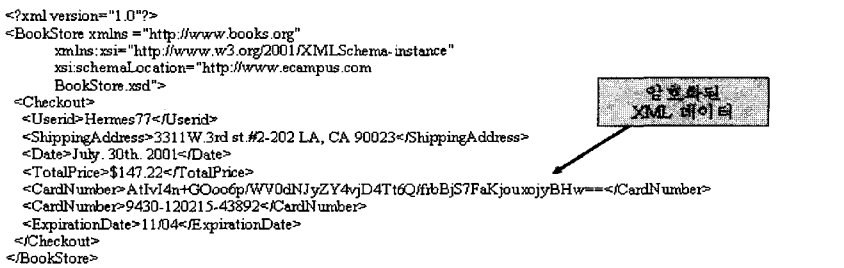
한 가지 고려사항은 XML 스키마 내의 엘리먼트 선언이다. 암호화 이후 엘리먼트 선언 정보에서 특히 데이타 타입의 변경이 일어날 수 있는데 이 때 원본 XML 스키마를 기반으로 엘리먼트 선언을 파싱하여 데이타 타입 선언을 변경시킨 갱신된 XML 스키마를 생성한다. 예를 들어, 전자 상거래에서 전자 결제를 이용할 경우 신용카드 번호를 입력받는 XML 스키마의 엘리먼트 선언은 그림 3의 <xsd:pattern value=…/>와 같이 정규 표현식으로 표현될 수 있다. 여기서 나타나는 데이타를 암호화할 경우 데이타는 정규 표현식을 따르지 않는 스트링 형태로 변환을 시켜야 한다. 암호화 수행 후 변환된 XML 문서에 대한 XML 스키마 정보를 수정함과 동시에 기존의 XML 스키마를 기반으로 갱신된 XML 스키마를 생성하도록 한다. 그리고, 갱신된 XML 스키마를 변환된 XML 문서가 참조할 수 있도록 네임스페이스를 변환시킨다. XML 문서에 대한 XML 스키마 참조 정보는 XML 스키마는 다중 참조가 가능하다는 성질을 이용한 것으로, XML 네임스페이스를 이용하여 갱신된 XML 스키마를 추가참조 시킴으로써 문서의 유효성은 유지 가능하게 한다.

3.3 XML 스키마 전자 서명

XML 스키마는 XML 문서의 기반이 되는 중요한 메타 데이타들로 구성된다. XML 데이타 암호화 및 복호화 작업은 적절한 XML 스키마 구조를 바탕으로 이루어진다. 따라서 XML 스키마에 대한 무결성을 XML 문서 교환시 증명할 수 있는 장치가 필요하다. 이를 위해 본 논문에서는 XML 스키마에 대한 전자 서명 방법을 제안하였다. XML 스키마 전자 서명은 일반적인 전자 서명 방식을 XML 스키마 문서에 적용하는 방법이다. XML 스키마 전자 서명은 XML 문서에 대한 전자 서명과는 처리방식에 있어 차이가 있는데, XML 스키마는 기본적으로 XML 구문법을 따르지만 형태가 매우 다양하게 나타날 수 있다는 특징을 가지고 있다. 따라서 이러한 다양한 형태에 대하여 표준화된 형태로 변환할 필요가 있다. 본 논문에서는 DOMHash[11] 방법을 이용하여 정규화 형태로 변환하는 방법을 사용하였다. DOMHash는 정규화 XML 스키마 형태를 구축하는데 있어 엘리먼트 선언들을 메모리에 저장과 동시에 전자 서명 생성 작업이 가능하므로 불필요한 파일 입출력 연산을 줄이게 되어 속도에 있어 매우 효율적이다. 따라서 트리를 형성하고 순회를 거쳐서 XML 스키마를 재구성하는 정규화 방식과 비교하여 간단하고 빠른 장점을 보인다. 그림 4는 DOMHash에 기반한 XML 스키마 전자 서명

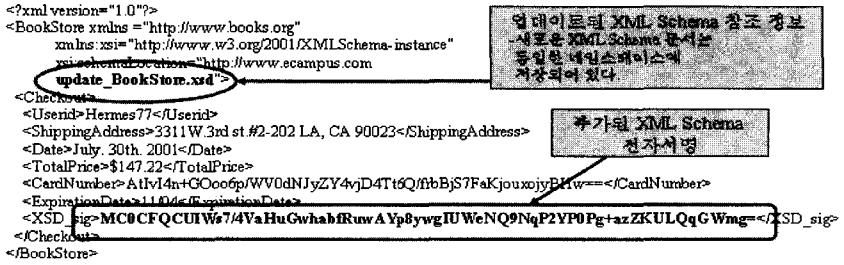
```
Input : XML Schema 파일, XML 문서 파일, 서명자의 개인키
Output : XML Schema 파일의 전자 서명 값

Procedure XMLSchema_D_Signature(XMLSchema_file, sig_key)
     Hashtable hash_table;                    /* Hashtable */
     MessageDigest MessageDigest_XS;          /* MessageDigest */
     XML_Schema XS;       /* XML 스키마 객체 */
     Signature D_sig;       /* 전자서명 객체 */
 begin
      Load("XMLSchema_file");   /* 서명을 생성할 원본 XMLSchema 파일 */
      hash_table = new Hashtable();       /* 해시 테이블 객체 생성 및 메모리 할당 */
      while(Endof_XMLSchema_file)
      begin
           Parser parse = new Parser();
           /* XML 스키마 파일을 읽어들여 파싱 */
           XS = parse.readXMLSchemaStream(XMLSchema_file);
           /* Hashtable에 XML 스키마 엘리먼트 및 키 값 및 속성 값 저장 */
           hash_table.addElement(XMLSchema_elelements);
           hash_table.addElement(XMLSchema_attributes);
           hash_table.addElement(XMLSchema_entities);
     end while;
     StreamBuffer str_buf = StreamBuffer.read(hash_table);   /* Hashtable을 읽음 */
     while(str_buf !=Endof_hash_table)
     begin
           /* Hashtable의 메시지 다이제스트 */
           MessageDigest_XS = MessageDigest(str_buf);
     end while;
     /* 전자서명 값의 생성 */
     D_sig = Signature(MessageDigest_XS, Private_key);
     return D_sig;
 end;
```

그림 4 XML 스키마 전자 서명 알고리즘




```
<?xml version="1.0"?>
<BookStore xmlns ="http://www.books.org"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.ecampus.com
        BookStore.xsd">
 <Checkout>
  <Userid>Hermes77</Userid>
  <ShippingAddress>3311W.3rd st.#2-202 LA, CA 90023</ShippingAddress>
  <Date>July. 30th. 2001</Date>
  <TotalPrice>$147.22</TotalPrice>
  <CardNumber>AtIvI4n+GOoo6p/WV0dNJyZY4vjD4Tt6Q/fibBjS7FaKjouxojyBHw==</CardNumber>
  <CardNumber>9430-120215-43892</CardNumber>
  <ExpirationDate>11/04</ExpirationDate>
 </Checkout>
</BookStore>
```

(가) 유효성이 고려되지 않은 보안 XML 문서의 예




```
<?xml version="1.0"?>
<BookStore xmlns ="http://www.books.org"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.ecampus.com
        update_BookStore.xsd">
 <Checkout>
  <Userid>Hermes77</Userid>
  <ShippingAddress>3311W.3rd st.#2-202 LA, CA 90023</ShippingAddress>
  <Date>July. 30th. 2001</Date>
  <TotalPrice>$147.22</TotalPrice>
  <CardNumber>AtIvI4n+GOoo6p/WV0dNJyZY4vjD4Tt6Q/fibBjS7FaKjouxojyBHw==</CardNumber>
  <ExpirationDate>11/04</ExpirationDate>
  <XSD_sig>MC0CFQCUIWs7/4VaHuGwhabfRuwAYpBywgIUWeNQ9NqP2YP0Pg+azZKULQqGWmg=</XSD_sig>
 </Checkout>
</BookStore>
```

(나) 유효성을 고려하여 변환된 보안 XML 문서의 예

그림 5 유효성 유지 기법이 적용된 XML 문서의 예

생성 및 검증 알고리즘이다. 서명값을 생성할 XML 스키마를 읽어들인 후 해시 테이블 내에 엘리먼트 선언 및 속성 값들을 모두 저장한다. 해시 테이블은 XML 스키마 정보를 저장할 만큼 충분한 메모리를 요구한다. 본 논문에서 많은 메모리를 요구하는 DOM 기법을 적용한 것은 XML 문서와는 다르게 XML 스키마는 메타 데이타만을 보유하고 있기 때문에 상대적으로 데이타 양이 매우 적기 때문이다. 다음 과정으로 해시 테이블에 저장된 XML 스키마 정보를 다시 읽어들여서 해시 연산을 수행하여 XML 스키마에 대한 해시값을 생성한다. 그리고, 서명자의 키 값과 함께 서명자의 전자 서명을 생성한다. 서명 검증 과정 역시 서명 생성 과정과 동일하며, 차이점은 생성한 서명값과 수신된 서명값의 비교 과정이 추가된다는 점이다.

3.4 XML 스키마에 기반한 XML 보안성과 유효성 유지

XML 데이타 암호화 및 XML 스키마 전자 서명이 생성된 후, 유효성 유지를 위하여 XML 스키마 전자 서명과 갱신된 XML 스키마를 추가 참조할 수 있는 네임스페이스의 XML 문서에 대한 이식 작업이 요구된다. 반면 복호화 작업에서는 삽입된 XML 스키마 전자 서명값을 추출하여 원본 XML 스키마를 통해 생성된 서명값과 일치여부를 판별하고, 추가된 XML 스키마 네임스페이스를 삭제하는 과정이 요구된다. 이것은 파일의 특정 위치에 대한 데이타 삽입, 검색, 삭제 개념을 적용한다. 데이타 이식 작업에서는 XML 문서의 루트 엘리먼트 바로 밑의 자식 엘리먼트로 '서명' 엘리먼트인 <XSD_sig>를 추가하여 데이타를 삽입하고, XML 문서의 헤더를 파싱하여 네임스페이스를 추가하기만 하면 된다. 그림 5는 유효성 유지를 위해 갱신된 XML 문서의 예이다. 그림에서 표시된 부분이 변환된 부분이며, 이 부분은 복호화 과정을 거쳐 원본 문서의 형태로 복원된다.

4. 설계 구현 및 적용사례

4.1 시스템 설계

보안 XML 문서의 유효성을 유지하기 위해서는 암호화된 데이타를 갖는 XML 문서이외에 이 XML 문서에 필요한 XML 스키마, 그리고 데이타가 복호화된 후 XML 스키마의 무결성을 검증하는데 필요한 XML 스키마 전자 서명이 요구된다. 본 논문에서 제안하는 유효성을 지원하는 XML 암호화 시스템의 전체적인 흐름도는 그림 6과 같다. 그림에서 실선은 데이타의 암호화 작업의 흐름을, 점선은 데이타의 복호화 작업의 흐름을 나타낸다. 주요 구성 요소는 XML 문서 데이타 보안 처리 부분과 XML 스키마 데이타 처리 부분으로 이루어진다. XML 문서 데이타 보안 처리 부분에서는 기존의 XML 데이타 암호화 기능뿐만 아니라 XML 문서의 암호화에 따른 유효성 유지를 위한 XML 스키마 변환 사항을 고

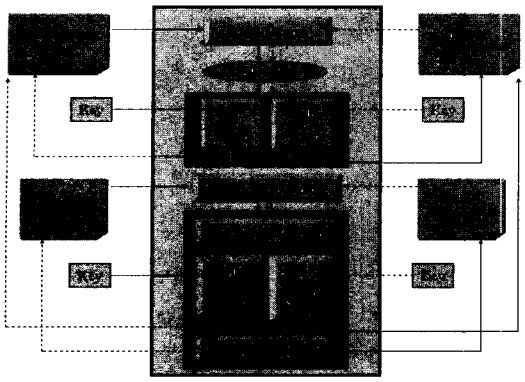


그림 6 시스템 구성도

려하여 적용하게 된다. 그리고 XML 스키마 데이타 처리 부분은 XML 스키마에 대한 무결성 및 부인 봉쇄 기능을 지원하는 기능을 갖는다. 따라서 최종적으로 얻어지는 산출물은 암호화된 데이타와 XML 스키마 전자 서명을 포함한 XML 문서, 그리고 이 문서의 유효성을 유지하도록 지원하는 변환된 XML 스키마이다. 각 구성 요소에 대한 세부 설계 사항은 다음과 같다.

4.1.1 XML 파서

XML 파서는 제안된 시스템뿐만 아니라 일반적인 XML 관련 애플리케이션에 있어서 핵심적인 요소이다. XML 파서는 XML 문서뿐만 아니라 XML의 메타 데이타인 DTD 또는 XML 스키마를 해석하여 필요한 데이타를 추출하며 문서의 유효성 검증과 같은 오류 추적에도 이용된다. 제안된 시스템에서는 DOM 구조를 이용한 파서를 사용한다. XML 문서의 파싱을 통해 얻어진 DOM 트리는 문서의 구조적 정보 및 각 엘리먼트간의 상호 관계정보의 정의가 가능하다. DOM을 기반으로 하여 암호화 및 복호화를 위한 엘리먼트 이름과 데이타의 검색 및 추출을 통해 효과적인 데이타 처리를 수행한다.

4.1.2 XML 암복호화 모듈

XML 데이타 암호화 및 복호화 모듈은 XML 보안에 있어 핵심적인 모듈이다. 서론에서 언급한 바와 같이 XML 데이타 암호화는 크게 두 가지로 분류되는데 문서 전체를 암호화하는 방법과 엘리먼트를 기반으로 암호화하는 방법이 있다. 제안된 시스템에서는 기본적인 구조로 [4]를 따른다. 부가적으로 웹 페이지 상에서 암호화 알고리즘을 선택하여 암호화가 가능하도록 지원한다. 이를 위해 웹 페이지 생성을 자바 서블릿(servlet)으로 지원한다. 선택이 가능한 암호화 알고리즘에는 DES, Triple DES, SEED, IDEA 등의 블록 대칭키 암호기가 있다.

4.1.3 XML 문서 변환기

암호화 및 복호화 작업을 통해 얻어진 데이타로 변환 후 XML 스키마에 대한 무결성을 지원하기 위하여 XML 스키마의 전자 서명값을 XML 문서 내에 추가한다. 이 모듈의 세부적인 작동 과정은 그림 7과 같다. 즉, 3.4 절에서 다룬 예와 같이 XML 문서의 특정 부분만을 파싱 과정을 통해서 데이타를 삽입하도록 한다. 원본 문서로의 복구과정은 파싱 후 삽입된 정보를 추출하여 삭제하는 과정을 거치도록 한다.

4.1.4 XML 스키마 파서

XML 스키마 파서는 XML 스키마의 구조를 파악하는 부분으로서 XML 문서의 기반이 되는 메타 컨텐츠가 보관된 XML 스키마로부터 필요한 정보를 추출하기 위한 것이다. 현재 개발되어 있는 다양한 XML 스키마 파서를 이용할 수 있다.

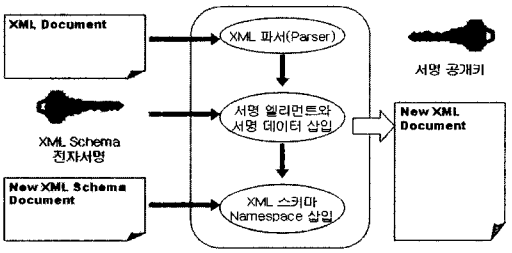


그림 7 XML 문서의 유효성 유지를 위한 XML 문서 변환 모듈

4.1.5 XML 스키마 DOMHash

XML 스키마의 표준형태를 생성하는 부분으로 해시테이블에 XML 스키마에 대한 엘리먼트,속성, 엔티티 등의 선언 데이타들을 저장한다.

4.1.6 XML 스키마 전자 서명 생성 모듈

XML 스키마 DOMHash로 생성된 해시테이블과 사용자의 서명값을 기반으로 XML 스키마에 대한 전자 서명을 생성한다. 전자 서명 생성 및 검증과 관련된 부분을 XML 암호화 라이브러리의 클래스 연동을 통해 해결한다.

4.1.7 XML 스키마 변환기

XML 데이타의 암호화 및 복호화 작업 후 변환된 데이타 타입을 지원하기 위하여 변환된 형태의 XML 스키마를 생성한다. 이 모듈은 4.1.3의 모듈과 유사한 형태를 갖는다.

4.2 구현 및 적용 사례

4.2.1 시스템 개발 환경

시스템 개발 환경은 다음과 같다. 제안된 시스템은 내

부적으로 데이타를 처리하며 이러한 처리된 데이타의 적절한 사용환경이 구축되기 위해 부가적인 적용 환경이 요구되었다. 이를 위한 시스템 구축 환경도 포함시켰다.

- 운영체제 : Sun Sparc5.8(Server), Windows2000 Professional(Client)
- 웹 서버 : Apache Web Server1.3.22, Jakarta Tomcat 3.2.4
- 데이타베이스 : mySQL3.22.32(Sun-Solaris2.7-Sparc)
- 사용언어 : JDK1.3, Java Servlet(JSDK2.1)
- 라이브러리 : Xerces3.1, Xalan2.0.0, JAXP1.1, Oracle-XMLSchema1.0.1, JCE1.2

4.2.2 적용사례

제안된 시스템은 임의로 구축된 모의 전자상거래 사이트에 적용하여 테스트하였다. 적용될 전자 상거래 사이트는 XML로 구축되었으며 이러한 환경 하에서 XML 문서는 정보 교환을 위해 XML 문서의 유효성을 필수적으로 유지해야 한다. 제안한 시스템을 이용하여 이러한 사이트에 적용하여 기존의 XML 보안 기능 이외에 XML 문서의 유효성을 유지하는 사례를 보고자 한다.

작동 방식은 제안된 시스템의 적용을 위하여 다음과 같이 요구사항을 최소화하였다.

- 오디오 CD를 온라인으로 판매한다.
- 제품에 대한 CD 타이틀과 카탈로그 번호를 기반으로 검색이 가능하다.
- 재고 정보를 알 수 있으며, 간단한 쇼핑 카트의 기능을 가지고 있다.
- 결재시 구매자의 개인 정보에 대하여 XML 문서를 기반으로 데이타를 처리한다.

전체적인 적용 사례 시스템의 구조는 그림 8과 같다. 사용자 인증 모듈 및 데이타 검색, 데이타 저장, 데이타베이스 관리, 결재시의 보안 시스템 등으로 구성된다.

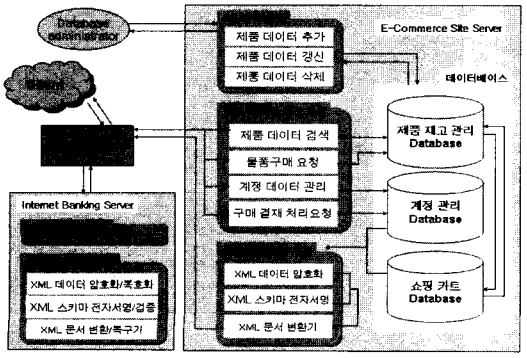


그림 8 적용 사례 시스템의 전체 구조

그림 9는 적용사례인 모의 전자상거래 사이트에 대한 간단한 실행 예이다. 그림과 같은 실행 결과들을 바탕으로 최종 단계인 구매 결재 과정에서 제안된 시스템이 적용되었다.

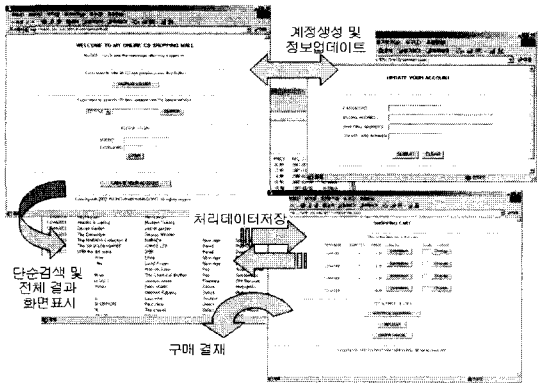


그림 9 제안된 시스템이 적용된 사이트의 작동 예

4.2.2 XML 스키마 전자 서명 생성결과

이 절에서는 제안된 시스템이 적용된 사이트에서 동작하는 과정 및 결과를 보여준다. 제안된 시스템은 클라이언트의 결재 과정에서 필요한 카드 정보에 대한 데이타 암호화와 화면에 표시된 XML 문서의 메타 데이타인 XML 스키마에 대한 전자 서명값을 추가하여 이를 표시하고 있다. 그리고 이 문서의 정보를 취한 사이트에서는 서명에 대한 검증 작업을 거쳐 신뢰성 있는 메타 정보를 기반으로 만들어진 XML 데이타에 대하여 안전하게 전송되었음을 확인할 수 있다.

4.2.4 XML 스키마 전자 서명 검증

그림 10과 그림 11은 구매 결재를 위한 사이트에서 이 문서의 신뢰성을 검증하는 단계를 보여준다. 내부적

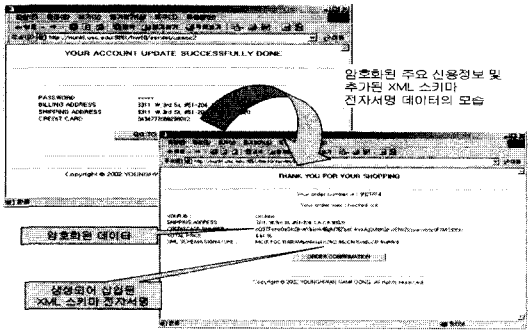


그림 10 암호화된 데이타 및 XML 스키마 전자 서명이 포함된 XML 문서

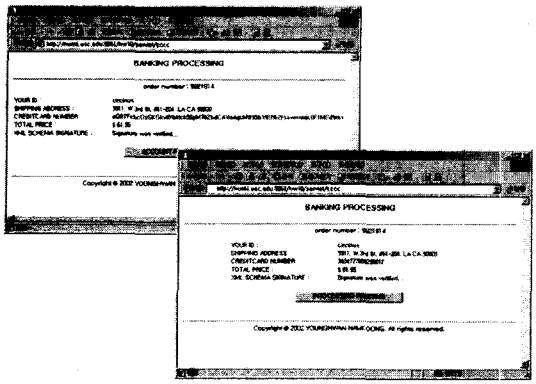


그림 11 XML 스키마 전자 서명의 검증 결과 및 복호화된 XML 문서

으로 서명자의 공개키를 입력받은 후 3.3 절의 알고리즘을 기반으로 하여 서명 검증 작업을 거쳐 XML 문서의 메타 데이타인 XML 스키마의 무결성이 유지되었음을 확인할 수 있다.

5. 분석 및 평가

본 논문에서 제안한 시스템은 XML의 메타 데이타인 XML 스키마에 초점을 두고 구현된 것으로 XML 데이타에 관점을 두고 개발된 기존 시스템과는 평가대상이 다르기 때문에 절대적인 평가는 어렵다. 뿐만 아니라 S2ML 및 SAML과 같은 마크업 언어는 암호화된 데이타, 인증 데이타, 전자 서명 데이타 등의 효율적인 교환과 공유를 목적으로 제작된 것으로 실제 시스템의 동작과는 관련이 없다는 문제점이 있다. 그러나, 데이타 처리 과정에서 적용된 알고리즘에 따른 속도와 시스템 확장성 및 신뢰도 등을 통해서 제안된 방법을 기존의 연구들과 평가해 보았다.

5.1 XML 스키마 전자 서명 속도 측정

5.1.1 실험 환경 설정 및 계산 복잡도 비교

전자 서명의 처리속도는 서명 알고리즘에 따라 많은 차이를 보이는 것이 일반적이다. 또한 서명키 값의 강도(strength)에 따라서도 약 3~4 배의 차이를 보인다. 본 논문에서는 실험을 위한 서명 알고리즘으로 MD5/RSA (RSA 키의 강도는 1024 bit)을 사용하였으며, 하드웨어 환경은 Pentium II 350 Mhz, RAM 128 MB, 운영체제는 Windows 2000이었다. 실험 데이타는 임의의 XML 스키마 문서를 대상으로 해시 연산 및 전자 서명 생성과 검증 작업에 걸린 시간을 측정하여 얻어진 데이타를 바탕으로 계산되었다. 가장 큰 차이는 XML 스키마의 파싱과 정규화된 형태로 변환하는 부분으로 [14]에서는 정규화 XML을 사용한 반면, 본 논문에서는 [11]에서 제시한 DOMHash를 사용하였다. 정규화 XML의 경우 각 엘리먼트를 트리의 노드로 보았을 때, 각 노드를 순회하여 정규화된 형태를 생성하게 되므로 노드를 방문하는 시간을 $O(1)$이라고 가정할 경우 n 개의 노드에 대하여 계산 복잡도는 일반적으로 $O(n)$임을 알 수 있다. 한편 DOMHash의 경우도 해시 테이블을 작성하는데 걸리는 시간도 원본 XML 문서 또는 XML 스키마 문서를 읽어들이는데 걸리는 시간과 동일하므로 n 개의 엘리먼트를 갖는 경우 계산 복잡도는 $O(n)$이다. 그러나 정규화 XML의 경우 작업의 결과로 정규화된 XML 형태의 문서를 생성하며 전자 서명을 위한 디스크 I/O 과정이 요구된다. 따라서 해시 테이블을 저장한 상태에서 전자 서명을 생성하는 DOMHash에 비해 불필요한 반복 작업으로 인한 속도 저하가 나타난다.

5.1.2 XML 스키마 전자 서명 속도 측정 결과

실험 결과, XLM 스키마 전자 서명의 생성속도는 평균 300.57 bytes/sec 정도이다. 이는 DOMHash를 기반으로 XML 스키마의 구조를 파싱 및 정규화 XML 스키마로 변환하기 때문이다. 반면 정규화 XML을 적용하였을 경우 230.88 bytes/sec로 나타났다. 또한 검증 과정에서의 속도는 DOMHash를 적용한 경우 669.84 bytes/sec였으며 정규화 XML의 경우 400.47 bytes/sec로 나타났다. 결과적으로 DOMHash를 적용한 본 시스템의 서명 처리속도가 약 1.3~1.7 배정도 빠른 속도를 보였다. 그러나 DOMHash와 정규화 XML의 처리속도만 고려한다면 DOMHash의 경우 1468.15 bytes/sec이고, 정규화 XML의 경우는 593.38 bytes/sec였다. 결국 DOMHash를 적용한 경우 약 2.47 배의 속도향상을 기대할 수 있음을 알 수 있다.

5.2 XML 스키마 관점에서의 비교

본 논문에서 제안한 기법과 [5]의 S2ML 및 상용 시스템인 [14]의 XSS(XML Security Suite)와의 기능을 비교한 결과는 표 2와 같다. 주요 비교 대상은 유효성 및 보안성을 고려하여 평가하였다. 표 1에서 제시된 바와 같이 본 논문에서 제안한 방법은 XML 문서의 유효

표 1 XML 스키마 전자 서명의 처리속도

(단위: bytes/sec)

	DOMHash 적용	정규화 XML 적용
메시지 다이제스트	1468.15	593.38
전자 서명 생성	300.57	230.88
전자 서명 검증	669.84	400.47

성을 고려한다는 측면에서 타 시스템과의 큰 차이를 보이고 있다. 유효성이 고려되지 않는 경우 애플리케이션 특히 XML 문서의 파싱 과정에서 오류가 발생할 가능성이 있다. 또한, XML 스키마를 대상으로 구현되었기 때문에 XML 스키마의 지원에 있어 정규 XML 스키마의 지원 및 XML 스키마 전자 서명 시스템 등 XML 스키마를 고려하지 않았던 기존 시스템과 비교하여 XML 네임스페이스를 기반으로 작동하는 XML 문서 처리에 효과적인 장점이 있다. 또한 기존 연구에서는 암호화되지 않은 XML 문서내의 데이타에 대한 처리가 요구될 경우에도 필수적으로 복호화 작업이 요구되었으나, 이러한 XML 문서의 유효성 유지는 복호화 작업 없이도 다양한 데이타 처리 작업이 가능하도록 한다. 따라서 적용 가능한 작업에 대하여 확장성이 높아지며 특히 불필요한 작업을 최소화하여 작업 속도를 향상시킬 수 있는 장점이 있다.

제안된 기법 및 XSS와 같은 시스템이 S2ML과 비교하여 가장 큰 장점은 S2ML 등의 마크업 언어는 그 자체로 암호화 작업을 수행할 수 없는 한계를 가지고 있다. 즉, S2ML 또는 SAML은 암호화 및 서명된 데이타의 교환을 목적으로 제작된 마크업 언어이며, 이는 집합의 개념으로 볼 때 XML의 부분집합에 속하기 때문이다.

표 2 XML 스키마 관점에서의 시스템 비교

	XML 문서의 유효성 유지	XML 문서의 원형 보존	XML 문서에 대한 확장성	정규화 XML 스키마 지원	XML 스키마 무결성 지원	XML 자바 라이브러리 호환성
XSS	-	-	O	-	-	O
S2ML	-	O	O	-	-	-
제안된 방법	O	O	O	O	O	O

6. 결 론

XML은 문서의 내용과 표현을 적절하게 분리하고 트리 구조 형태의 데이타 저장방식을 지원함으로써 정보의 구조적 표현을 가능하게 하였다. 이를 바탕으로 효율적인 정보 교환 및 공유 검색을 용이하게 하는 장점을 부각시켰다 그러나, 보안에 대한 취약성으로 인해 많은 정보가 노출됨으로써 전자 비즈니스 및 전자 상거래와 같은 안전한 정보 교환이 요구되는 환경 하에서 많은 정보 범죄를 야기할 수 있는 문제점을 드러내었다. 이러한 문제점에 대하여 XML 전자 서명, XML 암호화 기법, XML 접근 제어와 같은 다양한 측면의 연구 결과가 제시되었다. 각각의 방법은 보안 분야에서 세분화된 방법을 XML에 적용시킨 것으로 상호 보완적인 측면이 있다.

그러나, XML 데이타 암호화로 일어날 수 있는 구조적인 XML 유효성 위반 문제 및 XML 문서의 기반이 되는 메타 데이타를 보유한 XML 스키마에 대한 보안 문제는 간과되어 왔다. 본 논문에서는 이러한 XML 보안의 취약점을 파악하고, 전자 비즈니스 환경에서 필수적으로 요구되는 XML 문서의 유효성을 유지하는 XML 보안 기법을 제안하고 이를 기반으로 한 시스템을 설계 및 구현하였다. 제안한 방법은 기존의 엘리먼트 기반 XML 데이타 암호화 기법과 XML 전자 서명 연구를 기반으로 한다. 기존의 XML 문서의 암호화는 네트워크 상의 불안전한 데이타 전송을 보완하는 장점이 있으나 XML 문서의 구조보다는 XML 구문법에 초점을 두어 구현되었기 때문에 XML 문서의 유효성을 위반하는 문제점이 있었다. 이를 위해 본 논문에서는 XML 스키마의 갱신된 버전을 생성하며 이를 다중 XML 스키마 참조를 이용하여 XML 문서에서 유효성 유지가 가능하도록 하는 방법을 사용하였다. 또한 XML 문서의 메타 정보인 XML 스키마에 대한 무결성을 지원하기 위하여 XML 스키마의 구조를 DOMHash를 이용하여 정규화된 형태로 변환하고 이를 기반으로 전자 서명을 생성하여 XML 문서에 첨부시키는 방법을 이용하였다. 이는 SAML과 같은 보안만을 위한 마크업 언어의 사용을 필요로 하지 않는다. 본 시스템은 유효성이 요구되며 SOAP과 같은 웹 서비스를 기반으로 하는 다양한 전자 비즈니스 및 전자 상거래 환경에서 쉽게 이식되어 사용될 수 있는 장점이 있다. 추후 연구과제로는 XML 데이타 보안과 XML 전자 서명에 한정된 연구에서 확장하여 XML 접근 제어 연구와의 호환성 문제, 자바 언어로 구현되어 야기되는 속도 문제, 현재 새롭게 제안된 SAML과의 호환성 문제, 개선된 정규화 XML 스키마 기법 등을 생각해 볼 수 있다.

참 고 문 헌

[1] W3C, *"Extensible Markup Language(XML) 1.0,"* 1998, http://www.w3c.org/TR/REC-xml

[2] William J.Pardi, *"XML in Action, Web Technology,"* Microsoft Press, 1999.

[3] W3C, *"XML Signature Syntax and Processing,"* 2000, http://www.w3c.org/TR/xmldisig-core

[4] T. Imamura, H. Maruyama, *"Specification of Element-wise XML Encryption,"* W3C XML-Encryption Workshop, 2000,.

[5] P. Brandt, F. Bonte, *"Towards Secure XML."*

xml-encryption@w3.org Mail Archives, 2000. http://lists.w3.org/Archives/Public/xml-encryption/2000Oct/

[6] M. Kudo, S. Hada, *"XML Document Security based on Provisional Authorization,"* Conference on Computer and Communication Society, Athens. Greece, 2000.

[7] E. Damiani, S Vimercati, S. Paraboschi, P. Samarati, *"Design and Implementation of an Access Control Processor for XML Documents,"* Proceedings of 9th International World Wide Web Conference, Amsterdam, 2000.

[8] E. Bertino, M, Braun, S. Castano, E. Ferrari, M. Mesiti, *"Aurhor-x: a Java-Based System for XML Data Protection,"* Proceeding of the 14th IFIP WG 11.3 Working Conference on Database Security, Schoorl. Netherlands, 2000.

[9] W3C, *"XML Schema Part 1: Structures,"* 2001, http://www.w3.org/TR/xmlschema-1/

[10] W3C, *"Canonical XML Version 1.0,"* 2000, http://www.w3.org/TR/2000/CR-xml-c14n-20001026

[11] H. Maruyama, K. Tamura, N. Uramoto, *"XML and Java, Developing Web Applications,"* Addison Wesley, 1999

[12] W3C, *"Document Object Model(DOM),"* 2001, http://www.w3c.org/DOM

[13] B. McLaughlin, *"JAVA and XML,"* O'REILLY, June, 2000

[14] alphaWorks, *"XSS:XML Security Suite,"* 2002. http://www.alphaworks.ibm.com/tech/xmlsecuritysuite

[15] J. Knudsen, *"Java Cryptography,"* O'REILLY, 1998.

[16] L. D. Stein, *"Web Security, A step-by-step Reference Guide,"* Addison-Wesley, 1997.

남궁 영 환

1999년 고려대학교 컴퓨터학과 학사. 2001년 (주)시큐리티테크놀로지스 연구원. 2001년 고려대학교 컴퓨터학과 이학석사. 2001년 ~ 현재 University of Southern California. 관심분야는 XML보안, XML, 정보통합, 데이타베이스, 에이전트

박 대 하

1992년 고려대학교 컴퓨터학과 학사. 1994년 고려대학교 컴퓨터학과 이학석사. 1996년 고려대학교 컴퓨터학과 박사과정 수료. 1999년 ~ 현재 (주)시큐리티테크놀로지스 책임연구원. 관심분야는 XML 보안, 보안 프로토콜, 이동코드 보안, 임베디드 시스템 보안

허 승 호

1997년 연세대학교 금속공학과 학사. 2000년 연세대학교 금속공학과 공학석사. 2001년 ~ 현재 (주)시큐리티테크놀로지스 선임연구원. 관심분야는 XML 보안, 암호 프로토콜, 시스템 및 네트워 보안, 암호 응용

백 두 권

1973년 고려대학교 수학과 학사. 1976년 고려대학교 산업공학과 공학석사. 1983년 Wayne State University 전산학 석사. 1985년 Wayne State University 전산학 박사. 1986년 ~ 현재 고려대학교 컴퓨터학과 교수. 1991년 ~ 현재 ISO/IEC JTC1/SC32 국내위원회 위원장. 2002년 ~ 현재 고려대학교 정보통신대학장. 관심분야는 보안공학, 소프트웨어 공학, 데이타 공학, 데이타베이스