XML Transform 알고리즘의 구현

박종현[0] 김병규 강지훈 한우용
충남대학교 컴퓨터과학과
{ jhpark, yourovin, jhkang }@cs.cnu.ac.kr, wyhan@etri.re.kr

An Implementation of The XML Transform Algorithms

Jong-Hyun Park[0], Byung-Kyu Kim, Ji-Hoon Kang, & Woo-Yong Han
Dept. of Computer Science, Chungnam National University

요 약

인터넷상에서 메시지 교환을 위하여 XML의 사용이 급증함에 따라 XML문서의 보안이 필요하게 되었고, 이에 W3C는 XML-Signature 표준안을 제안 하고 있다. XML-Signature 표준 스펙에서는 서명할 문서의 내용을 선택 하는 방법으로 Transform 알고리즘들을 제안하고 있고, 그 알고리즘들은 서명자가 원하는 문서의 일부분만을 선택하거나, 변형하는 방법들을 기술하고 있다. 서명 시스템은 그런 Transform 알고리즘을 사용하여 문서의 전체 혹은 원하는 부분만을 선택하여 서명 함으로써 서명의 생성 및 검증의 처리속도를 높일 수 있고, 송·수신 시 효율을 높일 수 있고, 기존의 문서를 재사용 할 수 있는 등의 장점을 제공 하고 있다.

본 논문에서는 위와 같은 처리를 할 수 있는 4가지 Transform 알고리즘(XPath, XSLT, Enveloped, Base64 Transform)과 XML문서들의 무결성을 유지하기 위해 W3C의 Canonical XML 스펙을 기반으로 하는 Canonicalization Transform 알고리즘을 설계, 구현하였다. 이 Transform 알고리즘들은 XML 디지털 서명 뿐만 아니라 문서를 선택적으로 변환하는 응용등에서 사용할 수 있다.

1. 서론

점점 거대화 되고 있는 인터넷상의 데이터 교환을 위하여 XML의 사용은 급증하고 있다. 이에 따라 사용자 인증, 메시지 무결성, 메시지 송.수신, 부인 봉쇄 서비스를 제공하는 XML 디지털 서명이 필요하게 되었고, 이와 관련하여 W3C에서는 XML Signature 표준 스펙을 제안하고 있다. 그 스펙에서는 XML 디지털 서명을 작성하는데 필요한 XML 구문과 처리 규칙들을 기술하고 있고, 서명하고자 하는 문서 중 일부 만을 선택하여 처리하거나, 다른 형태의 문서로 변환하는 등 다양한 처리 방법을 제공하는 5가지 Transformation 알고리즘을 기술 하고 있다.[XML 1998, Dsig 2001, MK2000]

XML문서의 전자 서명 시 Transform 알고리즘은 XML이외의 문서를 서명할 수 있도록 하거나, 기존문서의 일부분만을 선택 또는 변형하여 서명하려는 문서를 생성한다. 또한, 문서의 무결성을 위하여 Canonicalization 알고리즘을 이용하여 문서를 변환한다.

본 논문에서는 XML문서의 전자 서명을 생성 및 검증하는데 반드시 필요로 하는 XML 문서의 무결성 및 선택 기능을 제공하는 Transform 알고리즘 S/W를 설계하고 구현한 내용을 기술하고 있다. 각각의 Transform 알고리즘들은 독립적으로 동작할 수 있도록 하고, XML문서를 교환하는 시스템에서 문서의 표현과 변형 및 무결성을 보장하기 위해서 필요에 따라 개별적으로 모듈을 사용 가능하도록 설계하였다. 본 논문에서 구현한 Transformer의 모듈로는 XSLT, XPath Filtering, Enveloped, Base64, Canonicalization 알고리즘이 있으며, 각각의 알고리즘들은 독립적으로 작업을 수행하므로 응용의 요구에 따라 적절하게 포함시켜 사용할 수 있다.

* 이 연구는 BK21 대전.충남 정보통신 인력 양성 사업단의 지원을 받았음.

2. Transform 알고리즘의 구성요소

2.1 XPath Filtering Transform

XPath Filtering Transform은 원본 XML문서 중 서명할 특정 부분을 XPath로 표현한 것을 XPath Syntax와 Semantics를 이용하여 필터링 한다. [Dsig 2001, XPath 1999]

XPath Filtering Transform을 사용하여 처리 할 경우 기존에 존재하는 문서의 일부분을 XPath 표현만으로 재사용 할 수 있고, XML-Signature에서 활용할 경우 반드시 필요한 부분에만 서명할 수 있어 효율적으로 사용할 수 있다. Xpath 표현이 들어간 문서는 [그림 1]과 같다.

```
<?xml version="1.0"?>
<PurchaseOrder>
  <Item> . . . </Item>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"> . . .
   <Reference URI = "http://www.카드회사정보.xml">
     <Transform Algorithm = "http://www.w3.org/TR/1999/REC-xpath-19991116">
       <XPath xmlns:Credit="http://cs.cnu.ac.kr/~wins/credit#">
       ancestor-or-self::credit:InfoCard
       </XPath>
     </Transform>
   </Reference> . . .
  </Signature>
</PurchaseOrder>
```

[그림 1] XPath Filtering Transform선언을 포함한 문서

[그림1]에서 볼 수 있듯이 PurchaseOrder 엘리먼트의 하위 엘리먼트에 서명하고자 하는 문서를 Reference 엘리먼트의 URI로 지정하고 있고, Reference 엘리먼트의 하위 엘리먼트로 해당 URI에 적용하고자 하는 Transform 알고리즘을 선언하여준다. [그림 1]의 경우 와 같이 Transform 알고리즘이 XPath Filtering Transform을 선언 했다면, 반드시 Transform 엘리먼트의 하위 엘리먼트에 XPath 엘리먼트를 가져야 하고, XPath 엘리먼트의 내용은 서명 하려는 문서에 적용시켜야 할 XPath 표현을 기술한다. XPath Transformer는 카드회사.xml 문서에서 Namespace [XML-ns 1999]의 prefix가 credit인 곳에서 이름이 InfoCard인 엘리먼트와 모든 자식 노드를 서명하기 위한 부분으로 선택 한다.

2.2 XSLT Transform

XSLT Transform은 아래 문서에서 Reference Element의 URI 속성이 가리키는 문서를 XSLT의 Syntax와 Semantics를 따라서 다른 문서로 변형한다.[Dsig 2001, XSLT 1999]

```
<?xml version="1.0"?>
<PurchaseOrder>
  <Item> . . . </Item>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"> . . .
    <Reference URI = "http://www.제조정보.xml">
      <Transform Algorithm = "http://www.w3.org/TR/1999/REC-xslt-19991116">
        <xsl:stylesheet xmlns:xsl="http:// www.w3.org/1999/XSL/
          Transform"  xmlns="" version="1.0">
          <xsl:output method="xml" indent="yes"/>
          <xsl:template match="/">
            <date>
              <xsl:value-ofselect="/order/date/year"/><xsl:value-of
              select="/order/date/month"/><xsl:value-of
              select="/order/date/day"/>
            </date>
          </xsl:template>
        </xsl:stylesheet>
      </Transform>
    </Reference> . . .
  </Signature>
</PurchaseOrder>
```

[그림 2] XSLT Transform선언을 포함한 문서

XSLT Transform의 표현은 위의 [그림 2]에서 볼 수 있듯이 XPath Filterring Transform과 유사한 구조를 갖지만 Transform 엘리먼트의 속성으로 XSLT 알고리즘 URI를 선언해야 하고, Transform 엘리먼트의 하위 엘리먼트 부분에 반드시 stylesheet 엘리먼트를 생성하여 reference하는 문서에 적용시켜야 할 XSLT 표현을 기술 한다. 위 예제에서 서명할 문서를 얻기 위해 XSLT Transform수행을 마친 결과는 년/월/일의 내용을 가지는 data엘리먼트를 얻는다.

2.3 Enveloped Transform

Enveloped Transform은 서명하려는 대상 문서내에 또 다른 서명을 포함하고 있는 경우 서명부분을 제외한 원본 문서만을 추출해 낸다.[Dsig 2001]

아래 [그림 3]의 경우 수신자 측에서는 문서의 유효성 검사를 하기 위해 Reference엘리먼트의 URI를 읽는다. 이때, URI가 공백인 경우 서명 하고자 하는 문서가 자신이 되는 것을 나타낸다. 하지만 원본문서는 서명문서를 내포하고 있으므로 순수한 원본문서만을 추출하기 위해서는 Transform 엘리먼트를 가지는 내포된 문서(서명문서)를 제거하여야 한다. 이 경우 Enveloped Transform을 사용하여 내포된 서명문서를 제외한 원본문서를 얻어낸다.

```
<?xml version="1.0"?>
<PurchaseOrder>
  <Item> . . . </Item>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#" > . . .
    <Reference URI = " ">
      <Transform Algorithm = "http://www.w3.org/TR/2000/CR-xml-c14n-20001026"/>
      <Transform Algorithm = "http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
    Other Transform Algorithm
    . . .
    </Reference>
    Other Reference
    . . .
  </Signature>
</PurchaseOrder>
```

[그림 3] Enveloped Transform선언을 포함한 문서

2.4 Base64 Transform

Base64 Transform의 기본적인 스펙은 MIME를 따르며[MIME 1996], Enveloped Transform과 마찬가지로 엘리먼트의 내용을 가지지 않는 Empty 엘리먼트 이다. Base64 Transformer는 Reference하는 문서가 Base64로 Encoding되어 있는 문서일 경우 Base64로 Decoding하는 Transform 알고리즘으로 reference된 문서가 Raw Data일 경우 유용하게 사용된다.

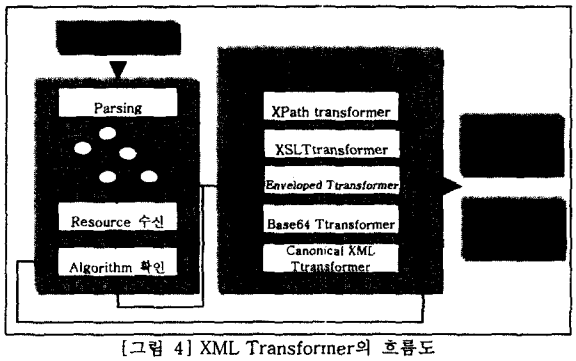
2.5 Canonical XML Transform

Canonical XML Transform은 Canonical XML Version 1.0을 따른다.[XML-C14N 2001] 이 Transform은 논리적으로는 동일하나 물리적으로는 여러 가지 표현이 존재할 수 있는 XML 문서를 일관된 형태로 변형하는 Canonicalization을 수행한다. 즉, 문서의 처리 과정에서 파서 등에 의해서 나타나는 문서의 물리적인 변형을 처리하여 주므로 상호 교환되는 XML문서의 무결성을 보장하여 준다. 때문에 XML-Signature와 같이 Digest 값을 구해야 하는 시스템에서는 문서의 물리적 구조 또한 일관된 형태를 유지하여야 하므로 반드시 필요한 모듈이다. Canonical XML Transform을 선언은 [그림 3]과 같다.

3. Transform 알고리즘의 설계 및 구현

3.1 XML Transformer의 흐름도

XML Transformer의 처리흐름은 [그림 4]와 같다. 원본 XML 문서를 구문분석 한 후 DOM Tree를 생성하여 문서의 정보를 얻고[DOM 1998], Reference된 Resource를 수신한다. 수신한 Resource는 원본문서에서 선언한 알고리즘을 확인하여 적절한 Transform을 수행하고, 다음 알고리즘의 선언들이 존재하면 모든 Transform을 처리 할 때 까지 반복하여 수행하게 된다.

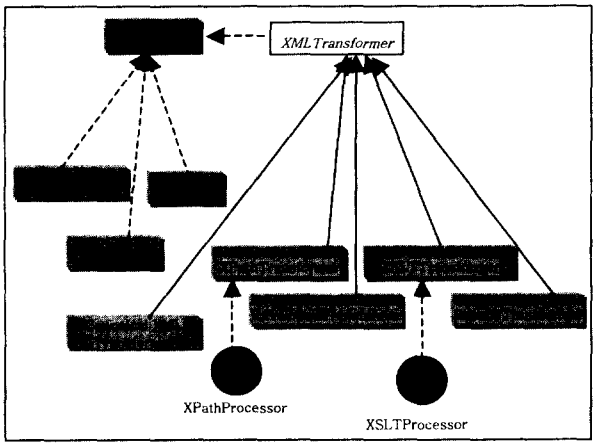


[그림 4] XML Transformer의 흐름도

XML 전자 서명과 같은 응용인 경우, XML Transformer의 출력 값은 Digest 값을 얻기 위한 입력으로 제공되며, 응용의 요구에 따라서 적절하게 사용가능 하다.

3.2 XML Transformer의 구조

본 논문에서 구현한 XML Transformer의 구조는 아래 [그림 5]와 같다. XML Transformer는 XML 디지털 서명과 같은 응용(예: 그림5의 TransformProcessor)이 사용할 수 있는 인터페이스를 제공하고있다. TransformProcessor는 Reference 엘리먼트에 선언되어 있는 문서를 가져오고(GetResource) 관리 하며, 선언된 알고리즘을 비교하여 적절한 Transform을 적용한다. 이때에 각 Transformer마다 입력 값으로 Resource를 넣어주고, Transform된 결과는 다시 Resource를 update시켜서 다른 Transformer의 입력 값으로 제공된다.



[그림 5] XML Transformer의 구조

각각의 Transformer는 적절한 Processor를 이용하는데 XPath Transform은 XPath Processor를 이용하고, XSLT Transform은 XSLT Processor를 이용하여 Transform한다.

3.3 각 Transform간의 문서 처리

원본 XML문서는 여러 개의 Reference를 가질 수 있고, Reference엘리먼트의 속성으로 가지는 URI를 표현하는 방식은 RFC2396을 따른다.[URI 1998] 또한 Reference엘리먼트 안에는 여러 개의Transform 알고리즘을 선언할 수 있지만 각 Transform 알고리즘의 입력 데이터의 형식과 출력 데이터의 형식은 미리 정의되어 있다. XPath와 Enveloped Transform의 경우에는 입력형식과 출력형식 모두 Node-Set으로 정의 되어 있고, XSLT와 Base64 Transform은 입력과 출력형식 모두 OctetStream으로 정의되어 있다. 만약 XSLT Transform을 수행한 결과가 XPath Transform을 수행해야 하는 경우 XPath Transform이 요구하는 입력형식을 따르기 위하여 XSLT Transform의 출력 값인 OctetStream을 Parsing하여 Node-Set으로 변환하여야 하고, 반대의 경우에도 마찬가지로 요구된 입력의 형식을 따르기 위하여 W3C에서 제안한 Canonical XML에 따라 Canonicalization한 후 Transform을 수행하여야 한다.[XML-C14N 2001] 때문에 각 Transformer마다 요구되는 입력의 형식에 맞도록 변환하는 기능을 구현하여 여러 개의 Transform이 유기적으로 수행될 수 있도록 하였다.

4. 관련 연구

XML Transformer는 현재 W3C에서 제안한 XML-Signature Syntax and Processing 스펙의 일부분으로 XML 전자 서명을 구현하기 위해서는 반드시 필요한 모듈이다. 하지만 아직까지 XML-Signature 스펙이 표준화 되지 않은 상태이기 때문에 완벽한 구현을 한 곳은 없다고 볼 수 있다. 현재 국외에서는 표준화 작업에 참여하고 있는 IBM사가 개발 중에 있고[IBM 2001], WebSig Consortium에서는 실제 여러 여행사들 사이의 e-Commerce 응용을 목적으로 개발 중에 있지만[GPSM2000], IBM사를 제외한 기관에서는 5가지 Transform을 모두 완벽하게 지원하는 곳은 아직까지 없다.

5. 결론

XML 기술이 인터넷 e-비즈니스 시스템등에서 메시지 교환 형식으로 이용되면서 이들 XML 문서의 보안은 필수적 요구조건이 되고, 안전한 비즈니스를 수행을 위해서 XML 디지털 서명을 반드시 지원하여야 한다. XML 디지털 서명에서는 선택적으로 문서를 얻어내고 변형하여 서명하기 위한 Transform 알고리즘(XPath, XSLT, Enveloped, Base64)과 XML 문서의 무결성을 보장하기 위한 Canonicalization 알고리즘을 포함하는 XML Transformer를 구현 하였다. 구현된 Transform 알고리즘들은 독립적으로 수행 되므로 응용에 따라 특성에 맞게 선택적으로 사용될 수 있다.

향후 Canonical XML과 유사한 기능을 하는 DOMHash를 구현하여 비교 분석하고, 실제 전자서명 시스템에 연계하여 통합하는 테스트가 필요하다.

6. 참고 문헌

[DOM 1998] W3C, Document Object Model (DOM) Level 1, Recommendation, Oct. 1998. (http://www.w3.org/TR/REC-DOM-Level-1).

[Dsig 2001] W3C, XML-Signature Syntax and Processing, Candidate Recommendation, 19-April-2001. (http://www.w3.org/TR/xmldsig-core/).

[GPSM2000] C. Geuer-Pollmann, C. Puland, P. Sklavos, & M. Moula "Digital Signatures for Web Content", e-2000 eBusiness and eWork Conference, 2000.

[IBM 2001] IBM, The XML Security Suite, 2001. (http://www.trl.ibm.com/projects/xml/xss4j/docs/dsig.html)

[MIME 1996] N. Freed & N. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part One, Format of Internet Message Bodies. November 1996. (http://www.ietf.org/rfc/rfc2045.txt).

[MK2000] T. Miyazawa & T. Kushida "An Advanced Internet XML/EDI Model Based on Secure XML Documents", ICPADS'00 Workshops, 2000

[URI 1998] (URI)Uniform Resource Identifiers: Generic Syntax. T. Berners-Lee, R. Fielding, L. Masinter. August 1998. (http://www.ietf.org/rfc/rfc2396.txt).

[XML 1998] W3C, Extensible Markup Language (XML) Version 1.0, Recommendation 10- February-1998. (http://www.w3.org/TR/1998/REC-xml-19980210).

[XML-C14N 2001] W3C, Canonical XML Version 1.0, Recommendation 15 March 2001. (http://www.w3.org/TR/xml-c14n)

[XML-ns 1999] W3C, Namespace in XML, Recommendation 14 January 1999. (http://www.w3.org/TR/REC-xml-names/).

[XPath 1999] W3C, XML Path Language (XPath) Version 1.0, Recommendation 16 November 1999. (http://www.w3.org/TR/xpath).

[XSLT 1999] W3C, XSL Transformations(XSLT) Version 1.0, Recommendation 16 November 1999. (http://www.w3.org/TR/xslt)