SOiVA 메타데이터 보안 시스템 설계 및 구현*

김영만[1], 김형환[2], 정기용[1], 정태성[1]
[1]국민대학교 컴퓨터공학부
[2]한국전자통신연구원
e-mail: [1]{ymkim, admin, motoko}@kookmin.ac.kr, [2]hhkim@etri.re.kr

Design and Implementation of SOiVA Metadata Security System

Youngman Kim[1], Hyung Hwan Kim[2], Kiyong Jung[1], Taesung Jung[1]
[1]School of Computer Science, Kookmin University
[2]Electronics and Telecommunications Research Institute

요 약

SOiVA는 양방향 동영상 서비스를 위한 TTA 정보통신 표준이다. SOiVA 서비스는 다양한 정보를 표현하기 위해 메타데이터를 이용하는데, 이 메타데이터에는 그 중요성 때문에 반드시 제 3자로부터 보호되어야 하는 내용이 포함되어 있다. 이 중 일부가 외부에 노출되거나 유실·조작될 경우 큰 문제를 야기할 수 있으므로 반드시 적합한 보안 시스템이 필요하다. 본 논문에서는 전자서명과 암호화 등의 기술을 이용해 SOiVA 메타데이터 보안 시스템을 설계하고, 타당성을 검증하기 위해 시스템을 구현하고 예제 메타데이터에 대하여 시험하였다.

1. 서론

IPTV는 통신과 방송의 융합서비스로 최근 전 세계 통신 사업자는 물론 기존 방송사업자들에게 큰 관심을 받고 있는 분야이다. 하지만 IPTV는 방송국에서 Video를 제작하여 일방적으로 서비스하는 형태로 양방향성이 충분히 지원되지 못하며 사용자의 요구에 따라 개인화 되어 있지 못해 iVideo의 특성을 만족하지는 못한다.[1]

이런 단점을 보완하고 진정한 의미로서의 iVideo의 특성을 만족시킬 수 있는 기술로 TTA의 정보통신 표준인 SOiVA가 존재한다. SOiVA를 이용하면 상호간의 iVideo 서비스를 동적으로 발견하고 이를 P2P로 연결하여 그 콘텐츠를 볼 수 있으며, 내재된 하이퍼링크를 이용해 관련된 정보나 서비스를 열람 또는 사용할 수 있다.[2]

한편 SOiVA 서비스는 iVideo 및 서비스에 대한 정보를 표현하기 위해 메타데이터를 이용한다. 그런데 이 메타데이터에는 개인정보나 금전거래 정보 등 그 중요성 때문에 반드시 제 3자로부터 보호되어야 하는 내용이 포함되어 있다. 이 중 일부가 외부에 노출되거나 유실·조작될 경우 큰 문제를 야기할 수 있으므로, 메타데이터를 보호할 수 있는 보안 시스템이 필요하다.

본 논문에서는 전자서명과 암호화 등의 기술을 이용해 SOiVA 메타데이터 보안 시스템을 설계한다. 2장에서는 SOiVA 및 널리 쓰이는 XML 기반 보안 기술인 XML Signature와 XML Encryption을 소개하고, 3장에서는 본 논문에서 주제로 삼고 있는 SOiVA 메타데이터 보안 시스템을 설계한다. 이어서 4장에서는 보안 시스템을 구현한 프로그램을 작성하여 예제 SOiVA 메타데이터를 이용해 시험하고 마지막으로 5장에서 결론을 맺는다.

* 본 연구는 한국전자통신연구원이 주관하는 지식경제부 "SOA기반 양방향 동영상 응용서비스 기술표준개발" 과제로부터 지원받아 수행하였음.

2. 관련 연구

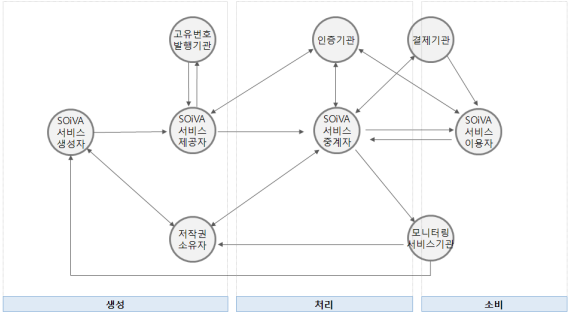
2.1. SOiVA

2.1.1. 용어정의

SOiVA(Service Oriented interactive Video Application)는 SOA를 기반으로 iVideo 응용 서비스를 안정되게 주고받을 수 있도록 동영상 기반의 웹을 실현하는 기술이다.

iVideo(Interactive Video; 양방향 동영상)는 선형적인 비디오 또는 영상 정보와 사용자와의 상호작용을 혼합하는 기술이다. 동영상 자체에 각종 정보를 포함하여 동영상을 보는 도중에 사용자가 쉽게 동영상에서 정보를 보고 타 서비스를 요청할 수 있다.

2.1.2. SOiVA 서비스 운영 모델



(그림 1) SOiVA 서비스 운영 모델

SOiVA 서비스는 다양한 개인, 기업, 기관 등이 서비스에 관여하는 액터가 되어 운영된다. (그림 1)은 SOiVA 서비스 운영 모델을 도식화하여 나타낸 것이다. 동그라미는 액터를, 화살표는 서비스 트랜잭션을 의미한다. 다음은 본 논문과 관계된 주요 액터에 대한 설명이다.[3]

1) SOiVA 서비스 제공자: 판매하고자 하는 서비스에 대한 iVideo를 생성하여 서비스를 제공한다.
2) SOiVA 서비스 중개자: SOiVA 서비스 제공자와 이용자를 1:1로 중개하는 역할을 한다.
3) SOiVA 서비스 이용자: 서비스에 대한 정보를 검색하고 양방향 동영상을 제공받아 상품을 구매한다.
4) 인증기관: SOiVA 서비스 제공자와 중개자, 이용자를 인증하고 상호간에 전송되는 정보를 관리한다.

2.1.3. SOiVA 메타데이터

SOiVA 메타데이터는 iVideo에 대한 정보를 비롯 iVideo가 제공하는 각종 서비스에 대한 정보를 통합하여 XML 포맷으로 표현한다. 서비스에 대한 정보는 개별 서비스의 이름, 카테고리, 키워드, 가격 등의 정보와 서비스 제공자, 서비스 중개자에 대한 정보를 포함한다.

또한 SOiVA 메타데이터에는 결제 정보를 비롯해 높은 보안성을 요구하는 메타데이터가 존재하는데, 이러한 데이터들은 보안이 필요치 않은 일반적인 메타데이터와 혼재되어 있다.

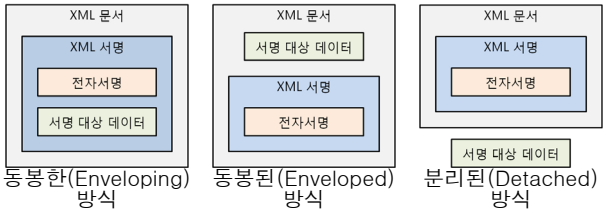
2.2. XML Signature

XML Signature는 디지털 아이템에 전자 서명을 생성하여 XML 포맷으로 나타내는 W3C 표준으로, XML 문법, 처리 규칙, 권장 알고리즘 등을 정의하고 있다. 특히 XML 문서에 전자서명을 첨부하고자 할 경우에는 문서의 일부, 즉 특정 엘리먼트만을 대상으로 서명을 생성하는 등의 다양한 옵션을 제공한다.[4][5][6]

한편 전자서명은 디지털 아이템의 해시 값으로부터 생성되므로, 원본 디지털 아이템이 XML 문서일 경우 같은 의미의 문서라 할지라도 공백 등으로 인해 해시 값이 달라질 수 있다. 이 문제를 방지하기 위해 XML 문서에 서명을 첨부할 때에는 해시 생성 이전에 반드시 문법적 차이를 없애는 정규화 과정이 필요하다.

또한 대상 데이터를 문서 내에 포함시키거나 URI를 링크하는 등의 방법으로 전자서명을 대상 데이터와 연관 지을 수 있다. 이는 전자서명과 원본 데이터의 위치 관계에 따라 다음과 같이 세 가지 방식으로 나뉘며, (그림 2)에서 이를 도식화하여 나타내고 있다.

- 동봉한(Enveloping) 방식: 서명 대상 데이터가 XML Signature 엘리먼트 내부에 존재한다.
- 동봉된(Enveloped) 방식: XML Signature 엘리먼트가 서명 대상 XML 문서 내에 존재한다.
- 분리된(Detached) 방식: 서명 대상 데이터가 XML Signature 문서 외부에 URI 형태로 존재한다.



(그림 2) XML Signature의 방식

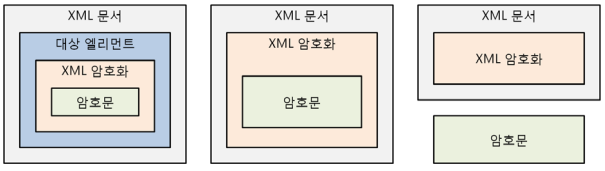
2.3. XML Encryption

XML Encryption은 디지털 아이템을 암호화하여 XML 포맷으로 나타내는 W3C 표준으로, XML Signature와 유사하게 XML 문법, 처리 규칙, 권장 알고리즘 등을 정의하고 있다. 특히 XML 문서를 암호화하고자 할 경우에는 문서의 일부, 즉 특정 엘리먼트만을 암호화하는 등의 다양한 옵션을 제공한다.[7]

한편 XML Encryption의 엘리먼트는 XML Signature와 대조적으로 대부분이 선택적이다. 이는 전자서명과 암호화의 성격 차이에서 기인한 것으로, 전자서명은 누구나 검증할 수 있어야 하지만 암호화는 오로지 수신자만이 해독할 수 있으면 되기 때문이다. 즉, 암호화 알고리즘이나 키 등 송신자와 수신자가 사전에 합의할 수 있는 사항은 모두 생략이 가능하다.

또한 XML Signature와 유사하게 XML 문서를 대상 데이터와 연관 지을 수 있다. 그러나 XML Signature와 달리 대상 데이터가 아닌 암호화된 데이터를 연관 지으며, 동봉된(Enveloped) 방식은 존재하지 않고 동봉한(Enveloping) 방식은 다시 두 가지로 나눌 수 있다. 암호화된 데이터를 문서 내에 포함시키거나 URI를 링크하는 등의 방법은 동일하다. 구체적인 방식은 다음과 같으며, (그림 3)에서 이를 도식화하여 나타내고 있다.

- 동봉한(Enveloping) 방식: 암호화된 데이터가 XML Encryption 엘리먼트 내부에 존재한다. 즉, 문서의 일부분을 암호화할 경우 XML Encryption 엘리먼트가 암호화 대상 엘리먼트 또는 그 내용을 대체하여 위치하게 된다. 이 때 대상 엘리먼트의 내용만을 암호화할 경우 콘텐츠 암호화 방식, 엘리먼트까지 통째로 암호화한 경우 엘리먼트 암호화 방식으로 구분한다.
- 분리된(Detached) 방식: 암호화된 데이터가 XML Encryption 엘리먼트 외부에 URI 형태로 존재한다.



(그림 3) XML Encryption의 방식

3. SOiVA 메타데이터 보안 시스템

3.1 개요

서론에서 기술한 것처럼 SOiVA 메타데이터에는 전자서명과 암호화 등 보안 시스템이 반드시 필요하다. 이 때 W3C의 표준을 그대로 적용하기에는 불필요한 요소가 많이 존재하고 있으며 또한 일부는 SOiVA 서비스 운영 모델에 적합하지 않다. 따라서 SOiVA 메타데이터 보안 시스템은 XML Signature/Encryption을 기반으로 하여 그 중 일부를 수용하고 추가적으로 필요한 부분은 SOiVA 메타데이터를 확장하여 정의할 것이다.

3.2. SOiVA 메타데이터 보안 모델

3.2.1. SOiVA 메타데이터 전자서명 모델

SOiVA 메타데이터는 이미 원본 데이터가 XML 포맷으로 존재하며 일부 엘리먼트만 선택적으로 서명할 수 있어야 한다. 따라서 동봉된(Enveloped) 방식을 사용해 원본 메타데이터에 전자서명을 첨부하는 것이 적합하다.

정규화 알고리즘으로는 유일한 W3C 표준인 c14n을, 해시 알고리즘으로는 XML Signature 표준에서 추천하는 유일한 알고리즘인 SHA-1을 사용한다. 전자서명 알고리즘은 RSA-SHA1 알고리즘을 사용하는데, 오랜 시간동안 널리 사용되어 대부분의 환경에 컴포넌트가 존재하는 등 호환성이 높기 때문이다.[8]

한편 SOiVA 서비스 운영 모델 내에는 인증기관이 존재하며 전자서명과 관련된 모든 액터는 인증기관으로부터 사전에 키를 발급받게 된다. 따라서 서명 검증을 위한 키는 문서에 포함시킬 필요가 없다. 대신 서명자는 항상 인증기관으로부터 발급받은 자신의 개인 키를 이용해 전자서명을 첨부하고, 수신자는 서명자의 ID를 통해 인증기관으로부터 서명자의 공개 키를 얻어 이를 이용해 서명을 검증한다. 이렇게 하면 문서 내에 서명 키를 첨부하는 것보다 신뢰성을 높일 수 있으며 메타데이터가 경량화된다. 따라서 SOiVA 메타데이터 전자서명에는 서명 키에 관한 엘리먼트가 제거된 XML Signature 문법을 사용한다.

또한 전자서명 자체에 관한 메타데이터를 별도의 <SignatureInformation>엘리먼트로 구성한다. 그 구조를 간략히 나타내면 다음과 같다.

<Identifier>: 전자서명의 ID이다.

<SignerID>: 서명자의 ID이다.

<SignedDate>: 서명 시각이다.

<SignedItem>: 해당 메타데이터가 나타내는 <Signature> 엘리먼트의 Id 속성이다.

<SignatureInformation> 엘리먼트는 서명이 첨부될 때마다 SOiVA 메타데이터의 루트 아래에 위치하는 <SignatureMetadata> 엘리먼트의 자식 엘리먼트로 추가된다. 이렇게 하면 한 문서에 여러 액터의 서명이 첨부되어도 관리하기 수월하며, 서명과 관련된 추가적인 메타데이터의 소요가 발생할 경우에도 추가가 용이하다.

3.2.2. SOiVA 메타데이터 암호화 모델

전자서명과 마찬가지로, SOiVA 메타데이터는 원본 데이터가 XML 포맷으로 존재하며 일부 엘리먼트만 선택적으로 암호화할 수 있어야 한다. 또한 암호화로 인해 이 양식이 해쳐져서는 안 되며 엘리먼트 이름 자체를 보호할 필요는 없으므로 동봉한(Enveloping) 콘텐츠 암호화 방식을 사용한다.

문서의 암호화에는 본문은 대칭 키 알고리즘을 적용하여 암호화하고 암호화에 쓰인 대칭키를 비대칭 키 알고리즘을 이용해 암호화하는 방법을 사용한다. 이를 통해 비대칭 키 알고리즘의 느린 수행속도를 보완할 수 있다. 대칭키 알고리즘으로는 XML Encryption 표준에서 필수로 요구하는 알고리즘 중 보안성이 가장 우수한 AES-256을 사용한다. 비대칭 키 알고리즘으로는 전자서명과 같은 이유로 RSA를 사용한다.

한편 역시 전자서명과 마찬가지 이유로 대칭 키를 암호화하는데 쓰인 비대칭 키에 대한 정보는 문서에 포함시키지 않는다. 대신 발송자는 대칭 키를 암호화할 때 항상 인증기관으로부터 얻은 수신자의 공개 키를 이용한다. 수신자는 자신의 개인 키로 대칭 키를 복호화할 수 있다. 따라서 SOiVA 메타데이터 암호화에는 비대칭 키에 관한 엘리먼트가 제거된 XML Encryption 문법을 사용한다.

또한 암호화 사유, 암호화 시각 등의 암호화 관련 메타데이터를 <EncryptionInformation>엘리먼트로 구성한다. 그 구조를 간략히 나타내면 다음과 같다.

<Identifier>: 암호화의 ID이다.

<EncryptionNote>: 암호화 사유 등 간단한 설명이다.

<EncryptedDate>: 암호화 시각이다.

<EncryptedItem>: 해당 메타데이터가 나타내는 <EncryptedData> 엘리먼트의 Id 속성이다.

<EncryptionInformation> 엘리먼트는 문서의 일부가 암호화될 때마다 SOiVA 메타데이터의 루트 아래에 위치하는 <EncryptionMetadata> 엘리먼트의 자식 엘리먼트로 추가된다.

3.3. SOiVA 메타데이터 보안 절차

3.3.1. 암호화 및 전자서명 절차

송신자는 먼저 메타데이터에 정규화 알고리즘을 적용하고 전자서명이 필요한 엘리먼트에 자신의 비밀 키로 RSA-SHA1 전자서명을 생성한다. 이후 전자서명에 대한 메타데이터인 <SignatureInformation> 엘리먼트를 생성해 SOiVA 메타데이터에 첨부한다.

암호화가 필요한 엘리먼트가 존재할 경우에는 대상 엘리먼트를 임의의 대칭 키로 암호화한다. 이후 인증기관으로부터 수신자의 공개 키를 구해 대칭 키를 수신자의 공개 키로 암호화한다. 마지막으로 암호화에 대한 메타데이터인 <EncryptionInformation> 엘리먼트를 생성해 SOiVA 메타데이터에 첨부한다.

이 때 기존에 전자서명이 첨부되어 있거나 암호화된 엘리먼트가 존재하더라도 별도의 처리를 해 줄 필요가 없으며, 특정 엘리먼트에 전자서명과 암호화가 모두 필요할 경우에는 우선 암호화를 진행한 후 전자서명을 생성하는 순으로 처리한다.

3.3.2. 복호화 및 서명 검증 절차

수신자는 우선 <SignatureMetadata> 엘리먼트와 <EncryptionMetadata> 엘리먼트를 탐색해 SOiVA 메타데이터에 첨부된 모든 전자서명과 암호화에 관련된 메타데이터를 확인한다. 이후 가장 최근에 수행한 전자서명 또는 암호화부터 차례대로 검증 또는 복호화한다.

전자서명의 검증을 위해서는 인증기관으로부터 서명자의 공개 키를 구해 전자서명 값을 해석하여 해시 값을 얻는다. 이 값과 서명된 엘리먼트의 해시 값, <DigestValu

e>의 해시 값이 모두 일치하는지 여부를 확인한다.

암호화 된 엘리먼트가 존재할 경우에는 우선 수신자의 개인 키로 대칭 키를 복호화한 후 복호화한 대칭 키로 암호화 된 엘리먼트를 복호화한다.

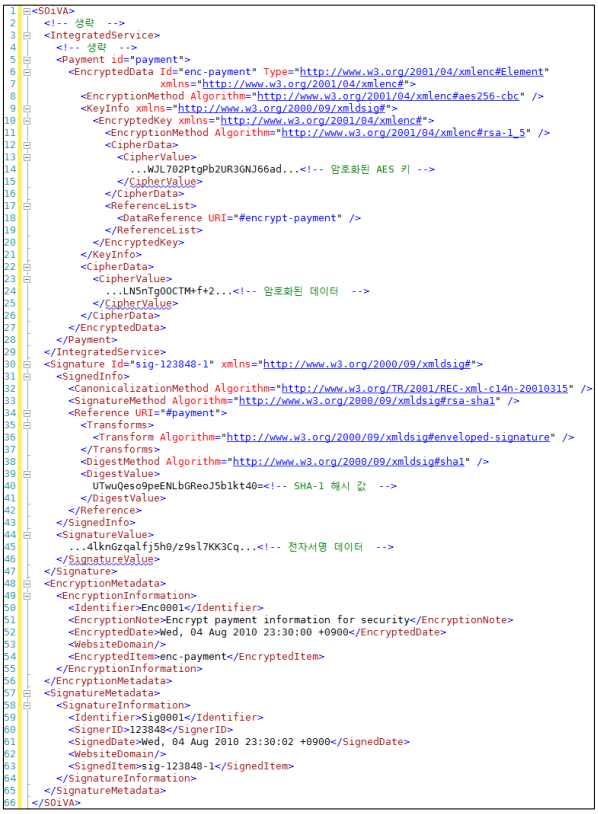
모든 절차가 끝나면 전자서명 및 암호화에 관련된 엘리먼트를 모두 제거해 원본 메타데이터를 얻을 수 있다.

4. SOiVA 메타데이터 보안 시스템 구현 및 시험

3장에서 설계한 보안 시스템을 구현한 프로그램을 작성하여 시스템 타당성을 검증하였다. 이 프로그램을 이용해 다양한 예제 SOiVA 메타데이터에 전자서명 첨부 및 검증, 암호화 및 복호화 등의 테스트를 수행한 결과 이상 없이 구동함을 확인할 수 있었다.

```
<SOiVA>
  <!-- 생략 -->
  <IntegratedService>
    <!-- 생략 -->
    <Payment id="payment">
      <Identifier>P332325</Identifier>
      <ChosenService>
        <ID>SID0001</ID>
        <Amount>1</Amount>
      </ChosenService>
      <PaymentMethod>Credit Card</PaymentMethod>
      <Payer>
        <Identifier>123848</Identifier>
        <UserName>hongan</UserName>
        <LastName>Nguyen</LastName>
        <FirstName>Hong An</FirstName>
        <Email>hongan@abc.com</Email>
        <Address>Jeongneung-gil 77, Seongbuk-gu, Seoul (136-702)</Address>
        <Phone>010-1234-5678</Phone>
      </Payer>
      <Purchaser>
        <Identifier>123848</Identifier>
        <UserName>hongan</UserName>
        <LastName>Nguyen</LastName>
        <FirstName>Hong An</FirstName>
        <Email>hongan@abc.com</Email>
        <Address>Jeongneung-gil 77, Seongbuk-gu, Seoul (136-702)</Address>
        <Phone>010-1234-5678</Phone>
      </Purchaser>
      <Receiver>
        <Identifier>4684357</Identifier>
        <UserName>motoko</UserName>
        <LastName>Jung</LastName>
        <FirstName>Tae Sung</FirstName>
        <Email>tsjung@abc.com</Email>
        <Address>Gamasan-gil 340, Guro-gu, Seoul (152-701)</Address>
        <Phone>010-8852-6472</Phone>
      </Receiver>
    </Payment>
  </IntegratedService>
</SOiVA>
```

(그림 4) 원본 SOiVA 메타데이터



```
<SOiVA>
  <!-- 생략 -->
  <IntegratedService>
    <!-- 생략 -->
    <Payment id="payment">
      <EncryptedData Id="enc-payment" Type="http://www.w3.org/2001/04/xmlenc#Element"
                     xmlns="http://www.w3.org/2001/04/xmlenc#">
        <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes256-cbc" />
        <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
          <EncryptedKey xmlns="http://www.w3.org/2001/04/xmlenc#">
            <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5" />
            <CipherData>
              <CipherValue>
                ...WJL702PtgPb2UR3GNJ66ad...<!-- 암호화된 AES 키 -->
              </CipherValue>
            </CipherData>
            <ReferenceList>
              <DataReference URI="#encrypt-payment" />
            </ReferenceList>
          </EncryptedKey>
        </KeyInfo>
        <CipherData>
          <CipherValue>
            ...LN5nTgOOCTM+f+2...<!-- 암호화된 데이터 -->
          </CipherValue>
        </CipherData>
      </EncryptedData>
    </Payment>
  </IntegratedService>
  <Signature Id="sig-123848-1" xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
      <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
      <Reference URI="#payment">
        <Transforms>
          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
        <DigestValue>
          UTwuQeso9peENLbGReoJ5b1kt40=<!-- SHA-1 해시 값 -->
        </DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>
      ...4lknGzqalfj5h0/z9sl7KK3Cq...<!-- 전자서명 데이터 -->
    </SignatureValue>
  </Signature>
  <EncryptionMetadata>
    <EncryptionInformation>
      <Identifier>Enc0001</Identifier>
      <EncryptionNote>Encrypt payment information for security</EncryptionNote>
      <EncryptedDate>Wed, 04 Aug 2010 23:30:00 +0900</EncryptedDate>
      <WebsiteDomain/>
      <EncryptedItem>enc-payment</EncryptedItem>
    </EncryptionInformation>
  </EncryptionMetadata>
  <SignatureMetadata>
    <SignatureInformation>
      <Identifier>Sig0001</Identifier>
      <SignerID>123848</SignerID>
      <SignedDate>Wed, 04 Aug 2010 23:30:02 +0900</SignedDate>
      <WebsiteDomain/>
      <SignedItem>sig-123848-1</SignedItem>
    </SignatureInformation>
  </SignatureMetadata>
</SOiVA>
```

(그림 5) 서명 첨부 및 암호화된 SOiVA 메타데이터

테스트 프로그램은 자바 언어로 작성하였으며, JDK/JRE 1.6을 이용해 컴파일 및 실행하였다. 자바는 JCA(Java Cryptography Architecture)와 JCE(Java Cryptography Extension)를 통해 암호화 기술 및 전자 서명 기술을 제공하므로 이를 사용해 보안 기술을 적용하였다. 한편 XML 파싱을 위해 파서인 Apache Crimson 라이브러리를, base 64 인코딩을 Apache Common Codec 라이브러리를 이용했다.[9][10][11][12]

(그림 4)는 실제 테스트에 사용된 SOiVA 메타데이터 중 일부를 나타낸 것이다. (그림 5)는 테스트 프로그램을 이용해 /SOiVA/IntegratedService/Payment 엘리먼트를 암호화하고 서명을 첨부한 후의 모습이다.

5. 결론

SOiVA 서비스가 성공적으로 안착하기 위해서는 몇 가지 걸림돌을 극복해야만 한다. 그 중 한 가지가 iVideo 및 SOiVA 메타데이터에 대한 보안이다. 본 논문에서는 SOiVA 메타데이터의 보안을 위해 XML Signature와 XML Encryption에 기반을 둔 보안 시스템을 설계하였고 타당성을 입증하기 위해 시스템을 구현하여 다양한 SOiVA 메타데이터에 대해 성공적인 테스트결과를 도출하였다.

참고문헌

[1] 권영환, 최준균, “IPTV 기술 및 표준화 동향”, 텔레콤 22권 1호, p.21, 2006.

[2] TTA, “ICT Standardization Roadmap 2010”, p.185, 2010.

[3] TTAK.KO-09.0044, “SOiVA 서비스 시나리오”, p.9, 2008.

[4] W3C, “XML Signature Syntax and Processing (Second Edition)”, http://www.w3.org/TR/xmldsig-core, 2008.

[5] 김주한 외 7명, 웹서비스 보안 기술의 표준화 및 시장 동향, 전자통신동향분석 20권 1호, pp.44-45, 2005.

[6] 이준동 외 4명, “유비쿼터스 환경에서 프라이버시 보호를 위한 XML 전자서명 인증 시스템”, 한국정보기술학회논문지 6권 2호, pp.78-79, 2008.

[7] W3C, “XML Encryption Syntax and Processing”, http://www.w3.org/TR/xmlenc-core, 2002.

[8] W3C, “Canonical XML”, http://www.w3.org/TR/2001/REC-xml-c14n-20010315, 2001

[9] “Java Cryptography Architecture(JCA) Reference Guide”, http://download.oracle.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html

[10] “JavaTM Cryptography Extension (JCE) Reference Guide”, http://download.oracle.com/javase/1.4.2/docs/guide/security/jce/JCERefGuide.html

[11] “Crimson”, http://xml.apache.org/crimson

[12] “Commons Codec”, http://commons.apache.org/codec