XML 보안기술을 적용한 안전한 프로그램 공유시스템 설계

오인원*, 박현동**, 서상원***, 서인석***, 류재철*

*충남대학교 컴퓨터과학과, **대덕대학, ***국가보안기술연구소

Design of secure program sharing system using XML security

In-won Oh*, Hyun-dong Park**, Sangwon Seo***, Inseog Seo***, Jae-cheol Ryou*

*Department of Computer Sience Chungnam National Univ. **Daeduk college, ***National Security Research Institute

요 약

웹 서비스는 기존 분산 처리기술과 인터넷 기술에 기반한 기술이며, 이기종 시스템간의 데이터를 자연스럽게 공유하고 통신하는 것을 목적으로 등장한 기술이다. 또한 웹 서비스 기술로 구현된 프로그램은 필요한 곳에서 proxy object를 생성해서 사용할 수 있으므로, 다른 개발 환경에서 작성된 프로그램이라도 호출해서 사용할 수 있게 되었다. 본 논문에서는 프로그램을 공유하는 과정에서 발생할 수 있는 보안 요구사항을 도출해 보고 보안을 위해 XML Signature와 XML Encryption을 적용한 안전한 프로그램 공유 시스템을 제시해 본다.

I. 서론

웹 서비스(Web Services) 기술은 그 동안 WWW(World Wide Web)로 대표되던 인터넷 서비스 시장에 새로운 형태의 서비스 모델을 가능하게 하고 있다. 각 하나의 서버와 브라우저로 구성되는 기존의 서버/클라이언트 구조에서 벗어나 서버와 클라이언트의 구별 없이 모든 노드가 서버와 클라이언트가 될 수 있는 구조로 구성되기 때문에 기존 환경에서보다 더 많은 데이터의 공유가 가능하다. 또한, 기존 구조에서는 데이터만을 공유하는 형태의 서비스가 제공되었으나, 웹 서비스 기술을 이용하면 데이터뿐만 아니라 프로그램 자체를 상호간에 공유할 수 있다. 프로그램을 공유하는 기술은 나아가서 소프트웨어의 판매방법도 변화시킬 수 있는 데, 소프트웨어를 패키지 형태로 일괄 구매하는 방식에서 탈피하여 특정 소프트웨어가 필요할 시에는 이를 다운로드하여 사용하고 사용량에 따라 요금을 지불하는 형태도 가능하다.

웹 서비스를 통해 전송되는 정보의 형태는 대부분 XML(eXtensible Markup Language) 형태로 구성되어 있다. HTML(Hyper Text Markup Language)은 단지 정보의 내용을 표현하는 데에 국한되는 기능을 제공하지만, XML은 각 정보들의 관계를 표현할 수 있기 때문에 HTML과 비교할 때에 융통성이 좋은 기능을 제공할 수 있다. 이러한 웹 서비스 기술이 기존 환경의 상당 부분을 개선하고 사용자들에게 더 많은 사용상의 편리함을 제공할 것이라는 예측도 있으나, 더 많은 보안상의 취약성을 가지고 있다는 단점도 가지고 있다. 본 논문에서는 기업 내의 부서들끼리 상호간의 프로그램을 공유할 수 있는 모델을 웹 서비스 기술을 이용하여 제안하고, 이를 안전하게 운영할 수 있는 방법에 대해 설명한다.

II. 웹 서비스 개요

웹 서비스는 기술적인 관점과 비즈니스적 관점으로 나누어 볼 수 있으며, 두 가지 관점을 종합하면, 웹 서비스는 인터넷과 같이 공개적인 네트워크 및 관련 표준을 통해 단일한 기업 내부 또는 다수의 기업간에 기존의 응용프로그램을 운영체제 및 프로그램 언어에 상관없이 상호운영이 가능하도록 해주는 표준화된 소프트웨어 기술로서 거래

업체간의 필요한 서비스를 발견, 제공하여 다양한 비즈니스를 가능케 해주는 것이다[1].

웹 서비스가 가지는 궁극적인 의미는 기존 패키지 상품으로서의 소프트웨어 개념을 향후에는 "서비스로서의 소프트웨어"로 완전히 전환한다는 데에 있다. 즉, 기존의 상품 판매자와 구매자가 서비스의 공급자와 소비자로 바뀌게 되고, 솔루션 업체들도 이와 같은 모델을 가장 잘 수용하는 방향으로 제품을 만들어야 한다는 서비스 지향적 패러다임(Service Oriented Paradigm)을 제공한다는 점이다[1].

웹 서비스의 모든 기술은 XML을 기반으로 하고 있다. XML은 데이터의 공개표준이므로 이기종 시스템간의 정보교환에 널리 사용될 것이므로, 기업간 솔루션 통합의 선두역할을 하며 기업의 비즈니스 수행능력과 활동영역을 확장시켜준다.

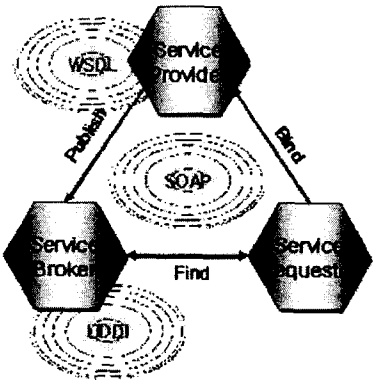
결론적으로 웹 서비스는 기존 분산 처리기술과 인터넷 기술에 기반한 기술이며, 이기종 시스템간의 데이터를 자연스럽게 공유하고 통신하는 것을 목적으로 하고, 표준 인터넷 프로토콜을 사용하여 다양한 프로그램을 액세스할 수 있는 기술이라고 요약할 수 있다.

1. 웹 서비스의 서비스 지향적 구조

웹 서비스는 서비스 지향적 구조(Service Oriented Architecture : SOA)를 구현한 것이다. 웹 서비스 모듈은 웹 서버에서 컴포넌트와 같은 역할을 수행한다. 웹 서비스와 클라이언트는 메시징 방식으로 통신을 하며 클라이언트가 웹 서비스에 요청 메시지를 보내면 웹 서비스는 그 요청에 따른 작업을 하고 필요시에는 결과 값을 클라이언트에 메시지로 보내준다. 서비스 지향 환경을 위해서는 다음과 같은 세 가지 조건을 만족해야 한다.

- 표준 웹 프로토콜을 통해 웹 사용자들에게 유용한 기능을 표시한다.
- 사용자와 대화할 수 있는 클라이언트 응용 프로그램을 만들 수 있는 인터페이스를 세부적으로 설명하는 방법을 제공해야 한다.
- 잠재적인 사용자가 웹 서비스를 쉽게 찾을 수 있도록 등록되어 있어야 한다.

[그림 1]은 웹 서비스의 구성요소들을 표현한 것이다. 이를 두 가지 관점에서 보면, 웹 서비스 객체 관점과 웹 서비스 동작 관점으로 나누어볼 수 있다.



[그림 1] 웹 서비스 구성도

1) 웹 서비스 객체 관점

웹 서비스는 서비스 요청자(Web Service Requestor), 서비스 제공자(Service Provider), 서비스 중개자(Service Broker)로 구성된다. 서비스 요청자는 웹 서비스를 사용하길 원하는 클라이언트이며, 서비스 중개자를 통해 서비스에 대한 검색을 하고, 서비스 제공자에게 필요한 서비스를 바인드 해서 사용한다. 웹 서비스 제공자는 웹 서비스를 만들어 클라이언트에게 제공하며, 서비스 중개자에게 서비스에 대한 인터페이스와 사용 방법 등의 정보를 등록한다. 웹 서비스 중개자는 이들의 중간에서 서비스를 중개해주는 역할을 하며, 일종의 전화번호부와 같은 역할을 한다.

2)웹 서비스 동작 관점

웹 서비스는 등록(Publish), 검색(Find), 바인드(Bind)로 구성되어 있다. 등록은 서비스 제공자가 자신이 만든 서비스를 서비스 중개자에게 등록하는 과정이며, 검색은 서비스 요청자가 자신이 원하는 서비스가 있는지 검색하는 것이다. 바인드는 서비스 요청자가 중개자로부터 서비스 사용방법을 받아와 서비스 제공자에게 연결해서 서비스를 이용하는 과정이다.

2. 웹 서비스의 표준 프로토콜

현재의 웹 서비스를 가능하게 대표적인 기술 요소로는 XML을 기반으로 한 SOAP(Simple Object Access Protocol), WSDL(Web Services Description Language), UDDI(Universal Description Discovery and Integration)가 있으며, 세 가지 핵심 기술에 대해 살펴보도록 하겠다.

1) SOAP

SOAP은 분산 환경에서 구조화된 정보를 교환하기 위한 목적으로 고안된 XML 기반의 경량 프로토콜이다. SOAP의 설계 목표는 단순성(Simple)과 확장성(Extensible)이다. SOAP은 인터넷과 응용 프로그램 간 통신을 위해서 디자인 되었으며, 정보를 데이터 교환을 위한 여러 가지 규칙을 갖고 있다. SOAP은 호출하려는 메소드가 어떻게 구현되어 있는지 모르며, 다만 통신하는데 필요한 규칙을 정의할 뿐이다. 이러한 특징 때문에 SOAP은 어떤 운영체제나 프로그래밍 언어에도 중립적이다. SOAP은 1999년 초에 0.9 버전이 나왔으며, 현재 1.2 버전까지 발표된 상태이며, W3C에서 SOAP의 웹 서비스 관련 표준화 작업을 진행하고 있다[2][3][4].

2) WSDL

WSDL은 웹 서비스와 그에 따른 모든 메소드를 표현하기 위해 설계된 언어이며, XML 형식을 사용해서 웹 서비스에서 지원하는 메소드 호출이나 인터페이스에 대한 설명을 기술한 문서이다. WSDL의 목표는 '웹 서비스를 기술하는 것'이며, SOAP 메시지 집합 및 해당 메시지가 교환되는 방법을 설명하고 있다. 장점은 WSDL 또한 SOAP과 같은 프로그래밍 언어 중립적이며, 표준 기반이므로 다양한 플랫폼과 프로그래밍 언어에서 액세스할 수 있는 웹 서비스 인터페이스를 설명하기에 적합하고, 어떠한 시스템이라도 웹 서비스의 메소드를 해석하고 무슨 SOAP 메시지가 만들어지고 전송되었는지를 이해할 수 있다는 점이다[5][6].

3) UDDI

UDDI는 비즈니스 기업과 그들이 제공하는 서비스들에 대한 정보를 구조화된 방법으로 저장하는 개방형 레지스트리이다. 클라이언트는 UDDI를 통해서 필요한 웹 서비스를 검색 할 수 있다. UDDI는 크게 서비스에 대한 기술(description) 부분, 탐색을 위한 표준 기반의 규격(specifications) 부분, 웹 상에서의 비즈니스 레지스트리의 공동 운영(operation) 부분으로 이루어지며 XML 파일 형태로 되어있다. UDDI는 탐색 및 등록을 위해 SOAP API를 사용하며 이를 위해 XML, HTTP, SOAP 등과 같은 인터넷 표준을 기반으로 하고 플랫폼과 업체에 중립적이며 산업체 전반으로부터 지원을 받고 있다. 간략히 말해 UDDI 레지스트리는 새로운 인터넷에 대한 전화번호부라 할 수 있다. 이것은 비즈니스가 어디서 서비스되고 있고, 개발자들이 어디에 위치를 지정했으며 가능한 서비스를 어디서 사용할 수 있는지를 나타낸다[7]. UDDI 정보등록의 세 가지 유형은 다음과 같다.

- 화이트 페이지(White pages) : 기업에 대한 기본 접근 정보와 회사명, 주소, 연락처 설명 등을 포함한다.
- 엘로우 페이지(Yellow pages) : 다양한 분류법으로 웹 서비스를 기술한 정보이다.
- 그린 페이지(Green pages) : 여러분이 회사에서 호스트하는 웹 서비스의 행위와 지원 기능을 기술한 정보이며, 웹 서비스의 집합적 정보와 웹 서비스의 위치 정보이다.

Ⅲ. 웹 서비스 보안 기술

효과적인 보안기술을 적용하지 않은 상태에서의 웹 서비스는 그 활용범위가 넓지 못 하다. 아직까지 SOAP과 같은 요소기술을 개발한 주체에서는 보안 문제를 향후에 해결해야 할 과제로 남겨 놓고 있다. 웹 서비스에 높은 수준의 보안기능을 제공하는 것은 쉬운 일이 아니다. 이는 웹 서비스가 제공해 줄 수 있는 응용 서비스의 구조 자체가 기존의 WWW보다 복잡한 구조를 가지고 있기 때문이다. 그러나, MicroSoft나 IBM과 같이 웹 서비스의 확산을 주도하고 있는 주체에서는 웹 서비스의 보안을 위한 요소기술을 선보이고 있으며, 이러한 기술들이 대부분 사실상의 표준으로 여겨지고 있는 추세이다. 이들 중에서 가장 중요한 기술로서는 WS-Security를 들 수 있으며, WS-Security는 다른 기술들의 근간을 이루고 있다.

1. 보안요구사항

웹 서비스에서 요구하는 보안기능은 일반적인 인터넷 통신의 것과 유사하다. 특히, WWW의 보안요구사항과 비교하면 동일한 보안기능을 요구하면서 그 이외에 몇 가지를 추가적으로 요구하는데, 이는 기존의 WWW에서 제공하지 않는 기능을 제공하는 데에 있어서 발생하는 것이다. 웹 서비스에서 요구하는 보안 요구사항을 정리하면 다음과 같다.

1) 메시지 기밀성

모든 인터넷 통신에서 기본적으로 요구하는 보안기능으로써 특정 메시지의 내용을 정당한 사용자만이 읽을 수 있게 함으로써 내용의 불법적인 노출을 방지하고자 하는 것이다. 대칭키 암호방식을 이용하여 제공하는 것이 일반적인 방법인데, 기존에는 SSL(Secure Socket Layer)이나 IPSec처럼 통신 채널을 모두 암호화하는 터널링 방법을

많이 사용하고 있다. 그러나, 웹 서비스에서는 다양한 형태의 통신 모델을 제공하기 때문에 터널링 기법의 암호화 방법은 융통성이 떨어지므로, 하나의 메시지 내부에서도 특정 부분만을 암호화하는 방법을 사용해야 한다.

2) 메시지 무결성

메시지가 전송 도중에 불법적으로 변경되었는지의 여부를 확인할 수 있게 해 주는 기능으로 주로 해쉬함수와 공개키 암호방식이 사용된다. 기밀성에서와 마찬가지로 터널링 방식 즉, 전송되는 메시지 전체를 하나의 단위로 하여 무결성 코드를 제공하는 방법은 웹 서비스에 적합하지 않다. 무결성이 요구되는 일부분에 대해서만 무결성으로 제공하는 방법이 필요하다.

3) 사용자 인증

통신 당사자 즉, 송신자와 수신자가 상호간에 상대방의 신원을 확인할 수 있는 방법이 필요하다. 특히, 웹 서비스는 하나의 트랜잭션이 완료되기까지 하나 이상의 중간노드를 거쳐야 하는 상황이 발생할 수있기 때문에 중간노드들과 관련된 사용자 인증 기술이 필요하다.

2. 보안요소기술

현재까지 제안된 웹 서비스의 보안을 위한 기술들 중에서 대표적인 기술을 간단하게 소개한다.

1) WS-Security

WS-Security는 2002년 4월에 Version 1.0이 발표되었고, 같은 해 8월에 Version 1.0의 addendum이 발표되었다. MS, IBM과 Verisign이 협력하여 작성한 후에 이를 OASIS에 제출하였고, Sun은 OASIS를 통해 규격의 개선 작업에 참여하고 있다. 향후에 Global XML Architecture의 한 부분으로 WS-Security를 이용할 계획을 가지고 있다. WS-Security의 목적은 전송되는 SOAP 메시지에 기밀성과 무결성을 제공하여 보호하고, 사용되는 공개키 등의 키 정보 즉, 보안토큰(Security Token)을 안전하게 전파하는 데에 그 목적으로 두고 있다. XML-Signature, XML-Encryption 등 기존의 관련 표준을 모두 수용함으로써 기존 환경에 변화를 주지 않는다는 것이 특징이다.

가) 보안토큰 전파

규격의 주요 목적 중의 하나가 보안토큰을 SOAP 메시지를 이용하여 다른 노드에 전달하는 것이다. 보안토큰이란 사용자의 신원을 증명할 수 있는 Claims의 집합을 의미한다. 보안토큰을 다루는 방법으로는 3 개의 element를 사용하는 방법이 있다.

- Username Token - 사용자의 ID와 패스워드를 전송하는 방식이다. 가장 기초적인 사용자 인증 방법으로 패스워드가 평문으로 전송되어 취약성을 내포하고 있지만, SSL 등과 같은 전송계층에서의 보안기술이 사용되는 환경에서는 가장 간단하게 사용할 수 있는 방법이다.
- BinarySecurityToken - 이 방법을 사용하면 사용자가 자신의 X.509 인증서나 Kerberos에서 사용하는 티켓을 포함하여 전송할 수 있다.
- SecurityTokenReference - 이 헤더에는 실제로 SecurityToken을 포함시키지 않고, 다른 헤더에 있는 인증정보를 참조하거나 또는 특정한 사이트에 존재하는 정보를 참조하도록 한다. [그림 2]를 보면, www.XYZShoes.com 사이트의 특정 위치에 있는 X.509 형식의 인증서를 참조하도록 하고 있다.

```
<wsse:SecurityTokenReference
  xmlns:wsse="http://schemas.xmlsoap.org/ws/2-02/04/secext">
    <wsse:Reference
      URI="http://www.XYZShoes.com/tokens/XYZ#X509token">
</wsse:SecurityTokenReference>
```

[그림 2] SecurityTokenReference의 예제

나) 메시지 무결성

SOAP 메시지 내부의 특정 부분을 지정하여 이에 송신자의 전자서명을 계산하여 전송하는 방법으로 메시지의 무결성을 보장한다. 대상이 되는 부분의 해쉬함수 계산결과와 이를 송신자의 비밀키를 이용하여 계산한 전자서명 결과를 함께 포함한다. 이를 검증하기 위해 필요한 공개키 등의 정보는 앞에서 설명한 보안토큰을 이용하여 전송한다.

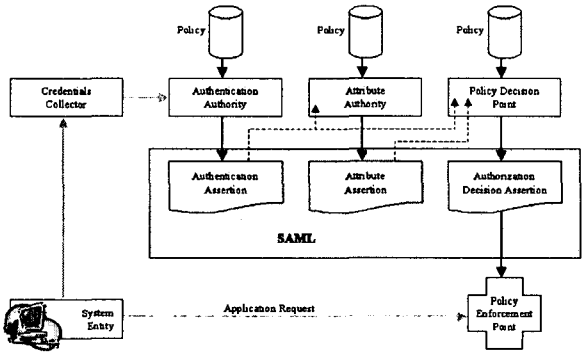
다) 메시지 기밀성

SOAP 메시지의 <Header> 부분에 포함되는 내용의 전체 또는 일부를 특정 수신자만이 복호화할 수 있도록 암호화한다. 먼저, 대상이 되는 부분을 대칭키 암호방식을 이용하여 암호화하고 사용된 대칭키를 수신자의 공개키로 암호화하여 기밀성을 보장한다. 만약, 하나의 메시지에 각 수신자별로 열람할 수 있는 부분이 따로 존재한다면 송신자는 수신자별로 데이터를 구분하고 이를 별도의 대칭키를 이용하여 암호화함으로써 수신자별

기밀성을 제공할 수 있다.

2) SAML(Security Assertions Markup Language)

XML 기반 프레임워크로 하여금 인증, 권한부여와 관련된 정보를 상호 교환할 수 있도록 한다. 이 정보들은 Security assertions라고 불리며 사용자, 웹 서비스를 포함하여 디지털 ID를 소유한 모든 객체와 관련된 정보를 포함한다. [그림 3]은 SAML 도메인 모델을 설명하고 있다. 사용자나 웹 서비스의 신원에 대한 정보가 수집되면 먼저 Authentication authority가 인증작업을 수행한 후에 Authentication assertion을 발행한다. 이 Authentication assertion은 Attribute authority와 Policy decision point로 전달되어 Attribute assertion과 Authorization assertion을 발행하는 데에 이용된다.



[그림 4] SAML 도메인 모델

이와 같이 SAML은 사용자와 웹 서비스의 신원을 확인하고 각각에 대한 접근권한을 부여하는 데에 있어 앞서 발생한 이벤트의 결과를 이용하게 되므로 응용 시스템의 동작을 단순화시킬 수 있다.

3) XACML(XML Access Control Markup Language)

XACML은 접근제어와 관련한 정책정보가 표현되고 전달되는 방법에 대해 정의하고 있다. 접근의 주체와 객체간의 규칙을 정의함으로써 정책 작성자는 어떤 웹 서비스가 특정 XML 문서에 어느 정도의 권한을 갖게 되는 지를 융통성 있게 정의할 수 있다. 예를 들어, XACML은 간호사가 환자의 의료기록 중에서 자신의 업무에 필요한 부분만큼의 자료 일부를 열람할 수 있도록 해 준다.

4) XML Signature

XML Signature란 임의의 디지털 컨텐츠에 대한 디지털 서명을 표시하기 위해 쓰이는 XML 문법이다. XML Signature 문법은 고수준의 확장성과 유연성을 갖도록 설계되어 있으며, 이런 개념들은 문법 안에 추상적으로 들어가 있기 때문에, 어느 서명에나 사용할 수 있다. XML Signature는 PKI기반의 인증서를 사용하여 XML 문서에 서명을 한다[8].

5) XML Encryption

XML Encryption은 메시지의 기밀성을 보장하기 위해 사용한다. XML Encryption은 다른 데이터를 암호화하는 과정과 같아 보이나, 두 가지 점에서 차이가 있다. 하나는 입력 데이터를 옥텟 대신 웹 자원으로 제공했다는 것이고, 다른 하나는 최종 암호화 데이터의 형식이 이진 데이터가 아니라 XML이라는 점이다 특히, 입력 데이터를 URI 방식으로 제공한다는 특징이 있다[8].

Ⅳ. 프로그램 공유를 위한 보안설계

1. 기존 시스템의 문제

기존의 WWW기반 프로그램과 분산 환경에서의 프로그램들은 통신을 하는 양단간에 시스템을 구성하고 있는 운영체제와 프로그램을 작성한 언어에 상당한 영향을 받았다. 그래서 서로 통신하고자 하는 경우 먼저 데이터 통신을 위한 협약이 이루어지고, 그것을 기반으로 통신하고자 하는 양단의 시스템 환경을 일치시켜야 하는 문제가 있었다. 또한 각 시스템에서 구현한 프로그램을 재사용하기 힘들었다. 프로그램의 재사용성을 높이기 위하여 프로그램을 컴포넌트화하여 필요한 곳에서 호출해서 사용하도록 시도하였다. 그러나 기존의 RPC 방식에서도 이기종 간에 프로그램을 호출해서 사용하는 일은 결코 쉬운 일이 아니었다. 특정 포트를 통해 통신이 이루어지는데, 대부분의 기업은 방화벽을 사용하고 있고, 이 방화벽에 막혀서 원활한 통신이 이루어지지 않는 경우도 많았으며, 원활한 통신을 위해 포트를 많이 열다보면 보안을 걱정하지 않고 사용하기란 힘들었다. 기존 시스템에서는 이러한 문제점들을 안고 있었기에 상호운영성이 좋지 않았다. 그러나 웹 서비스는 데이터 표준으로 자리 잡은 XML을 기반으로 한 표준 메시징 방식을 사용하므로 구현된 언어나 운영체제, 시스템 환경에 관계없이 서로 메시지를 교환할 수 있게 되었다. 또한 SOAP은 기존 통신 프로토콜과 바인딩하여 메시지를 전송하므로 보안을 위해 설치된 방화벽이 있을 때에도 만약 HTTP 프로토콜과 바인딩하여 메시지를 전송할 경우 방화벽에 막

히지 않고 통신할 수 있는 장점이 부각되고 있다. 또한 서비스 제공자가 자신이 만든 응용프로그램을 개인의 PC에 설치하는 방식이 아니고, 웹을 통해 필요한 서비스 사용자에게 제공하고, 프로그램의 관리는 서비스 제공자가 맡아서 처리하므로 사용자는 내부적으로 버전이 변경되거나 프로그램에 수정이 이루어져도 다시 프로그램을 설치하거나 자신의 시스템을 변경하는 번거로움 없이 서비스를 사용할 수 있다.

2. 프로그램 공유 시 보안고려사항

1) 보안 요구사항 도출

웹 서비스가 다양한 형태의 데이터와 프로그램을 쉽게 공유할 수 있다는 장점에 대해서 언급했지만, 기존의 기술보다 훨씬 유연하고, 확장성이 있는 만큼 보안에 대해서도 각별히 주의해야 한다. 앞 장에서 보았듯이 기존의 WWW기반 환경에서 제공되던 서비스가 그대로 이전되어 오는 형태이므로 WWW기반 환경에서 필요로 했던 메시지 무결성, 메시지 기밀성, 사용자 인증, 부인봉쇄 등의 보안기능이 웹 서비스에서도 필요하다. WWW보안을 위해 사용되었던 기술을 웹 서비스에 그대로 접목시켜 활용할 수도 있다. 그러나 웹 서비스만의 확장성을 감안한다면 웹 서비스에 맞는 보안 기술을 더 발전시키고, 이를 적용하여 서비스를 만들어야한다. 본 논문에서는 웹 서비스를 기반으로 한 프로그램을 공유하는 과정에서 발생할 수 있는 보안 요구사항 중 메시지 기밀성, 무결성, 사용자 인증, 부인봉쇄를 제공하는 시스템을 설계한다.

2) XML 보안기술 적용

설계하고자 하는 시스템에서 기존에 제시되고 있는 전송계층에서의 암호화방식인 SSL과 IPSec을 사용할 수도 있다. 그러나, WWW에 맞게 설계된 SSL과 IPSec은 WWW보다 확장된 기능을 가질 수 있는 웹 서비스에서 필요한 보안 기능을 모두 만족시키기 힘들다. SSL과 IPSec은 양단간 통신 채널을 형성하여 안전하지만, 패킷 전체를 암호화하여 보내므로 전송 속도에 영향을 줄 수 있으며, 암호화가 필요하지 않은 부분까지 모두 암호화가 되므로 오버헤드가 크다.

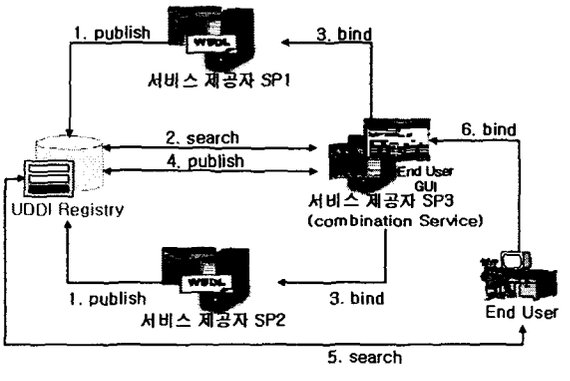
또한 웹 서비스에서 송신자와 수신자 사이를 이동하는 XML 메시지는 SOAP 프로토콜을 이용하는 경우가 많은데, SOAP 메시지는 다양한 통신 프로토콜과 바인딩이 가능하며, SOAP 메시지는 바인딩 되는 프로토콜의 보안특징을 따른다. 따라서 SOAP 메시지가 송신자에서 수신자까지 가는 중간에 여러 중개자를 거칠 때마다 만약 바인딩되는 프로토콜이 다르다면 각 프로토콜마다 지원되는 보안기능이 다르기 때문에 문제가 발생할 수 있으며, SOAP 메시지가 각 노드에서 암호화/복호화가 수행될 수도 있으므로 그 과정에서 데이터가 공격자에게 누출될 수도 있다.

기존 보안기술만으로 웹 서비스에 기반한 시스템에 적용하기에는 이러한 문제점들이 지적되기에 웹 서비스에서는 XML Signature와 XML Encryption 기술을 사용한다.

3. 프로그램 공유시스템 설계

1) 프로그램 공유 시스템 구성도

[그림 4]는 본 논문에서 제시하려는 프로그램 공유시스템을 웹 서비스의 일반적인 관점에서 봤을 때의 모습이다.



[그림 4] 프로그램 공유시스템

[그림 4]에서 서비스 제공자 SP1, SP2가 제공하는 웹 서비스를 SP3가 서비스 요청자가 되어 서비스 바인딩을 해서 사용하며, SP3가 다시 웹 서비스 제공자로서의 역할을 하는 모습이다. [그림 4]에서 SP1과 SP2는 웹 서비스를 만들고 서비스하는 주체가 된다. 사용자가 SP1과 SP2로부터 받을 수 있는 서비스는 각각에서 제공하는 프로그램 모듈이 될 수도 있고, 각각의 서버에서 제공하는 데이터가 될 수도 있다. 다음은 [그림 4]를 프로세스가 진행되는 순서대로 설명한 것이다.

1. Publish - 서비스 제공자 SP1과 SP2에서 서비스를 만든다. 서비스에 대한 WSDL 파일도 생성 후 UDDI Registry에 서비스 정보를 등록한다.

2. Search - SP3는 UDDI에 접속해서 필요한 서비스를 검색하고, SP1과 SP2에서 자신이 원하는 서비스를 하고 있음을 확인한다.

3. Bind - SP3는 SP1과 SP2에서 제공하는 서비스를 바인드 해서 또 다른 새로운 서비스를 생성한다. 이 시점에서 SP3는 서비스 요청자이다.

4. Publish - 새로 생성한 웹 서비스를 UDDI Registry에 등록한다.

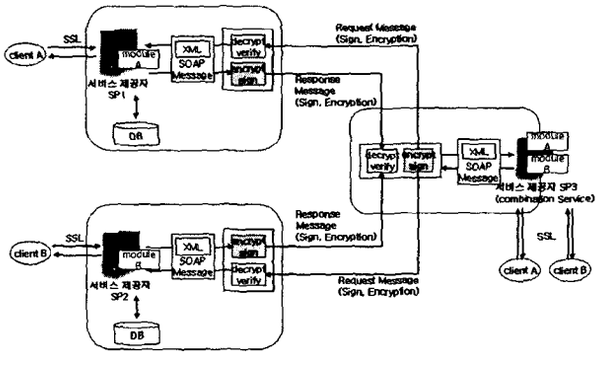
5. Search - End User는 UDDI에서 서비스를 검색하고, 자신이 원하는 서비스를 찾는다.

6. Bind - End User는 자신이 사용하려는 서비스 제공자에게 서비스를 요청한다. 이 시점에서 SP3는 서비스 제공자이다.

서비스 제공자 SP1, SP2는 B2B환경에서는 각기 다른 기업이 될 수 있고, 기업 내의 환경이라면 각기 다른 부서 또는 각기 다른 응용프로그램이라고 생각할 수 있다. 서비스 제공자 SP3는 처음엔 서비스 요청자로서 SP1과 SP2의 서비스를 요청하고 사용하지만, combination service를 생성 후 또 다른 서비스 제공자로서의 역할을 할 수 있다.

2) 프로그램 공유 보안시스템

[그림 4]는 기업 내 부서 간 프로그램을 공유하기 위한 시스템 구성도이다. [그림 4]에 보안기술을 적용시킨 프로그램 공유를 위한 보안시스템은 [그림 5]와 같다.



[그림 5] 프로그램 공유를 위한 보안시스템

[그림 5]에서는 UDDI에 서비스를 등록하고, 검색하는 부분은 생략하였고, 서비스 제공자와 서비스 요청자의 상호 작용에 중점을 두었다. 웹 서비스 제공자 SP1과 SP2가 속한 영역은 물리적으로 분리되어 있고, SP1과 SP2는 자신이 속한 부서에서 사용할 프로그램을 제공해준다. 클라이언트는 자신이 속한 부서에서 제공하는 프로그램만 접속해서 사용할 수 있다. 클라이언트 A는 서비스 제공자 SP2의 프로그램을 사용할 수 없으며, 클라이언트 B 역시 서비스 제공자 SP1의 프로그램을 사용할 수 없다.

SP3는 웹 서비스 기술을 이용해 구현된 SP1의 프로그램과 SP2의 프로그램을 호출해서 사용한다. 내부적으로는 SP1과 SP2에서 제공하는 서비스를 사용하기 위한 Proxy Object를 생성하고, Proxy Object의 메소드를 호출함으로써 SP1, SP2의 프로그램을 사용한다. 클라이언트 A와 클라이언트 B는 이제 SP3에서도 SP1이나 SP2에서와 같은 서비스를 제공받을 수 있다. 여기까지 보면 프로그램을 공유하는 데에 보안기술이 필요하지 않은 것처럼 보인다. 그러나 SP1, SP2에서 저장하고 있는 데이터가 SP3로 전송될 시에 데이터가 노출될 위험이 있다. 또한 앞에서 언급한 것처럼 서비스 요청자가 서비스를 이용한 시간이나 기능에 따라 요금부과를 하려 한다거나, 유효한 사용자만 프로그램을 사용할 수 있도록 제한하기 위해서는 보안기술이 필요하다. 앞으로의 웹 서비스의 발전 방향이 이러하므로 보안기술의 접목 또한 반드시 필요하다.

클라이언트 A, B가 SP3로 접속해서 서비스를 요청했을 경우 SP1, SP2 그리고, SP3사이에 안전하게 데이터를 교환하기 위해 XML Signature와 XML Encryption을 적용한다. SP3에서 XML 기반의 SOAP Request 메시지가 작성된다. 이 메시지는 SP1 또는 SP2로 전송되기 전에 encrypt/sign 모듈을 거쳐 서명 되고, 암호화 된다. 이 메시지는 목적지에 도착할 때 까지 데이터의 기밀성이 이루어진다. 메시지가 SP1 또는 SP2에 도착하였을 경우 메시지에 대해 복호화가 이루어지고, 전자서명을 확인해서 유효한 사용자인지 판단하고, 메시지의 무결성을 체크한다. 또한 전자서명 된 값을 저장해둠으로써 나중에 생길 수 있는 부인문제에 대한 봉쇄기능을 제공한다. SP1, SP2에서는 요청에 대한 응답 메시지를 생성하고, SP3로 보내기 전에 또 다시 서명과 암호화를 수행한다. SP3가 이 메시지를 받았을 때 메시지를 복호화 하고, 전자서명을 확인한다. 메시지에 이상이 없을 경우 클라이언트 A, B에게 결과를 보여준다.

본 설계에서는 클라이언트가 SP1, SP2, SP3에 접속할 때 아이디/패스워드 기반의 인증방식을 사용하며, 이 과정에서 해킹방지를 위해 SSL을 적용한다. 본 논문에서는 프로그램 공유에 목표를 두었기에 클라이언트의 인증부분을 기존의 Basic Authentication 방식으로 구성한다.

3) 프로그램 공유 보안시스템의 장점과

활용

웹 서비스가 도입되기 전이라면 클라이언트 A가 SP2에서 제공되는 서비스와 클라이언트 B가 SP1에서 제공되는 서비스를 사용하고자 했다면, 양자간 데이터 공유를 위한 합의 과정을 거친 후 SP1의 개발자와 SP2의 개발자는 자신의 시스템에 맞는 프로그램 언어로 새로운 프로그램을 작성했을 것이다. 그러나 웹 서비스 기술을 도입해서 프로그램을 구현한다면 프로그램을 새로 작성하지 않고, 필요한 프로그램을 사용할 수 있게 된다. 개발자의 입장에서는 개발 속도를 향상시키고, 보다 비즈니스 로직에 집중할 수 있게 해준다. 여기에 프로그램 공유 시에 발생할 수 있는 보안 취약점에 대한 부분을 보안 기술을 접목시켜 [그림 5]와 같은 시스템을 구축한다면 교환되는 데이터와 사용하는 프로그램 모두에 안정성을 제공하므로 사용자는 신뢰를 가지고, 필요한 서비스를 사용할 수 있게 된다. 본 논문에서는 기업 내의 프로그램 공유를 목표로 하였지만, 범위를 넓혀 기업 간 시스템 통합에도 이용할 수 있으며, 사용자에게 더 편리한 인터페이스와 고급의 서비스를 제공할 수 있다.

V. 결 론

본 논문에서 살펴 본 바와 같이, 웹 서비스는 기존의 IT 환경에 큰 폭의 변경을 가져올 수 있는 기술이다. 특히, 다른 서버에서 작동하고 있는 프로그램들을 통합하여 사용자가 하나의 서버에 접속하여 원하는 모든 프로그램의 서비스를 받을 수 있다는 점은 수많은 프로그램을 운영하고 있는 기업의 입장에서 볼 때에 업무의 효율성 제고와 하드웨어 및 소프트웨어 유지비용의 절감이라는 이득을 얻을 수 있게 된다. 따라서, 본 논문에서는 웹 서비스를 소개하고 보안기능을 제공하기 위해 발표된 기술에 대해 알아보았다. 그리고, 동일 기업의 다른 부서에서 사용하고 있는 프로그램들을 통합할 수 있는 방법을 설계하였다. 프로그램의 통합은 단순한 기능 통합뿐만 아니라 통합의 과정에서 발생할 수 있는 보안상의 취약성을 보완하기 위하여 XML-Signature, XML-Encryption 기술을 적용하였다. 향후에는 더욱 큰 규모의 프로그램 통합과 여러 기업들 간의 프로그램 통합을 위한 연구가 진행되어야 할 것이다.

참고문헌

[1] 정부연, "웹 서비스의 현황 및 비즈니스 모델의 변화", 정보통신정책 제 14권 15호, 2002년 8월.

[2] Simple Object Access Protocol (SOAP) Version 1.2 : Primer, http://www.w3.org/TR/2002/CR-soap12-part0-20021219/

[3] Simple Object Access Protocol (SOAP) Version 1.2 : Message Framework, http://www.w3.org/TR/2003/REC-soap12-part1-20030624/

[4] Simple Object Access Protocol (SOAP) Version 1.2 : Adjuncts, http://www.w3.org/TR/soap12-part2/

[5] Web Services Description Language (WSDL) Version 1.1, http://www.w3.org/TR/wsdl

[6] Web Services Description Language (WSDL) Version 1.2: Core Language, http://www.w3.org/TR/wsdl12/

[7]http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uddi-spec

[8] Blake Dournaee, XML Security, 2002.