특 집

웹 서비스 보안

이해규*, 이상수**, 김문규***

● 목 차 ●

1. 서 론

인터넷을 통한 웹의 발전은 클라이언트(웹 브라우저)와 서버(웹 서버)간에 정보전달의 활성화를 가져왔다. 이러한 환경에서 클라이언트와 서버간의 정보전달의 주요 대상은 정보의 내용(Content)이었다. 웹 서비스는 내용 중심의 인터넷 환경을 기능 중심으로 바꾸는 일종의 새로운 진보의 움직임이라고 볼 수 있다. 즉 인터넷상의 모든 Application들이 어떤 표준화된 인터페이스를 통해서 서로 간의 기능을 공유할 수 있게 함으로써 내용 중심의 공유에서 한 단계 진화된 기능 중심의 공유로 인터넷 이용 환경을 한층 발전시킨다는 것이다. 이러한 웹 서비스 개념은 Platform 독립적으로 데이터 형태를 표현할 수 있는 XML(eXtensible Markup Language)의 활성화와 WSDL(Web Service Description Language)과 SOAP(Simple Object Access Protocol), 그리고 UDDI(Universal Description, Discovery, and Integration) 표준의 제정 및 발전으로 인터넷 환경에서 구축되기 시작하면서 좋은 출발을 보이는 듯 했으며 이러한 중심에는 전 세계 인터넷 시장을 움직이는 MS와 IBM이 주도적인 역할을 하였다. 그러나 이러한 확산의 흐름은 웹 서비스가 인터넷상에서 활성화되기 위해서 본질적으로 해결해야할 문제들이 나타나면서 이의 해결에 많은 노력들이 진행되고 있다. 이러한 문제들 중에서 가장 초점이 되고 있는 것 중에 하나가 바로 웹 서비스 Security 문제이다. 이미 W3C(World Wide Web Consortium)에서는 Security를 고려한 XML의 확장된 여러 Specification들을 발표하였고 SOAP의 Security 측면을 강화하기 위한 추가적인 작업을 진행하고 있다. 산업계에서는 현재 MS와 IBM이 공동으로 W3C의 표준 Specification을 기반으로 웹 서비스 Security에 대한 표준 Specification 작업을 진행 중이다. 산업계의 이러한 움직임은 W3C의 여러 XML 표준 Specification으로 웹 서비스 Security의 표준을 위한 기반은 조성되었다는 판단 하에 업계 중심의 웹 서비스 Security 표준을 만들어서 실제 비즈니스에 하루라도 빨리 반영시키자는 의도로 보인다. 본 논문에서는 이러한 산업계의 웹 서비스 표준화 움직임에서 나온 결과를 중심으로 웹 서비스와 웹 서비스 Security Architecture에 대해서 전반적으로 언급하였다.

* KT 서비스개발연구소 선임보연구원
** KT 서비스개발연구소 선임연구원
*** KT 서비스개발연구소 플랫폼연구팀장

2. 웹 서비스 아키텍쳐

웹 서비스의 정의는 여러 가지가 있다. 이는 웹 서비스를 보는 관점에 따라 조금씩 달라지는 것이지만 여러 가지 정의들에서 공통적으로 흐르는 개념을 토대로 종합적으로 정의하자면 웹 서비스는 "여러 가지 다양한 Application들이 각자의 Program-to-Program 인터페이스, 통신 프로토콜, 등록(Registry) 서비스를 통해서 통신하고 서로간의 서비스를 수행하게 하는 진화된 분산 컴퓨팅 Architecture"라고 할 수 있다. 웹 서비스는 Cross-Platform, Program-to-Program 통신을 활성화하기 위해서 W3C에 의해서 만들어진 여러 가지 표준들에 의해서 이루어졌다. W3C는 웹 서비스를 위한 공식적인 표준 Specification으로서 SOAP과 WSDL을 제정하였으며 UDDI는 공식적인 W3C 표준은 아니나 곧 표준으로 제정될 것으로 보인다.

2.1 요소 기술

웹 서비스를 이루는 주요 요소 기술들에 대해서 간략하게 설명하면 다음과 같다.

- SOAP - Integration : 가볍고 확장 가능하게 만들어진 XML 기반의 프로토콜이며 Loosely Coupled 환경에서 정보 교환을 위해 사용된다. 기존의 통신 API(Application Program Interface)나 RPC(Remote Procedure Call)과 같은 역할을 한다. 다시 말해서 각각의 Application들이 서로 연결하고 통신하며 서비스를 호출하는데 사용하는 Action Words의 집합이다.
- WSDL - Behavior : XML로 만들어지며 Application들이 통신하는 방법을 규정한다. 즉 상호간에 정보가 공유될 수 있도록 서로 공통적인 방식으로 데이터 형식, Message, 동작, 통신포트 형식, 통신 프로토콜, 서비스 등을 표현할 수 있게 일종의 Template 역할을 해준다.
- UDDI - Location : Registry/Directory의 Evolving 표준으로서 UDDI.org에 의해서만들어졌으며 W3C에서는 공식적인 표준으로 채택을 검토중이다. 웹 서비스 Application들에 대한 정보들을 담고 있는 Directory-Like Repository이다.
- XML : 각기 다른 Application간에 Content나 데이타를 공유할 수 있도록 만들어진Human and Machine Readable한 Meta-Language이다. 종래의 HTML(Hyper Text Markup Language)가 주로 content와 graphic의 표현에 국한한 것에 비해 데이터, Syntax, Schema와 Semantics를 표현하는 방법을 제공한다.
- HTTP : 인터넷상에서 데이터를 전송하는 Application Layer의 통신프로토콜이다. 현재 대부분의 웹 서비스에서의 SOAP 메시지는 HTTP상에서 전송되고 있으나 웹 서비스가 꼭 HTTP상에서만 이루어지는 것은 아니다.

2.2 특징

웹 서비스의 특징을 몇 가지 측면에서 정리하면 다음과 같다.

2.2.1 분산 컴퓨팅 기술 측면

가. Loosely-Coupled Application

웹 서비스의 특징 중에서 가장 중요한 것 중의 하나가 바로 Loosely-Coupled Application 측면이다. 종래의 Tightly-Coupled Application에서는 Application 간의 통신에 관련된 모든 것이 미리 개발 프로그래머에 의해서 정해졌다. 즉 개발 프로그래머가 정한 방식대로 Application들은 정해진 Application과 정해진 방식대로 통신을 하였으며 Connection 중에는 계속적인 관리가 요구되었다. 이러한 두 Application간의 Hardwiring은 Quality-of- Service, Security, Privacy, Data Integrity, Complex Transaction Processing 관점에서 장점을 지닌다. 이러한 장점으로 인해서 기존의 Enterprise 컴퓨팅의 대부분이 이러한 구조이다. 반면에 Loosely-Coupled Application에

서는 Application간의 통신이 표준화된 인터페이스를 통해서 이루어지기 때문에 개발 프로그래머가 미리 정의할 필요가 없으며 Registry 서비스에 의해서 원하는 Application이 자동적으로 검색된다. 따라서 개발자들의 부담이 줄어들며 관리가 쉬워지며 유연성과 Interoperability가 제공된다. 이 두 구조의 장단점은 서로 Trade-off 관계이나 향후 분산 컴퓨팅 환경이 점차적으로 Cross-platform간의 Interoperability가 강조되어 가는 추세이기 때문에 Loosely-Coupled 구조의 단점을 보완해가면서 이 구조를 활성화시키려는 움직임으로 가고 있다.

2.2.2 Dynamic Look-up

Application이 자동적, 동적으로 자신이 필요한 Application을 찾아서 원하는 기능을 수행하는 것을 의미한다. 즉 Application은 UDDI 서비스를 이용해서 자동적으로 원하는 상대 Application을 찾고 자동적으로 서로 통신하는 방법을 맞추고, 기존의 상대 Application을 찾지 못할 때는 동적으로 다른 Application을 찾아서 원하는 기능을 수행한다. 이 모든 것이 인간의 개입 없이 이루어지게 된다[3].

2.2.3 Cross-Platform, Program-to-Program 통신

표준화된 SOAP, WSDL을 사용해서 인터페이스를 하기 때문에 웹 서비스는 Heterogeneous 환경, 즉 Cross-Platform에서의 Interoperability를 제공한다. 종래의 다른 분산 컴퓨팅 구조들도 이러한 점을 지향하려고 했지만 결국 표준화되지 않는 인터페이스로 비용과 관리, 확장성 문제로 인해서 모두 실패하였다.

2.2.4 해결해야 할 문제들

Bloor Research NA(North America)라는 연구 기관에서 최근에 IBM에 제출한 웹 서비스에 관한 보고서에서는 웹 서비스가 활성화되기 위해서는 해결해야 할 문제로 다음의 항목들을 지적하였다[3].

- ○ Security/Privacy
- ○ Messaging/Routing
- ○ Quality-of-Services/Reliability
- ○ Transaction-Handling
- ○ Management
- ○ Performance
- ○ Interoperability

이 항목들에 대한 구체적인 설명은 본 논문의 범위 밖이라서 생략하였고 문제로 지적된 항목 중의 하나인 Security에 관한 기술 동향에 대해서 설명하겠다.

3. 웹 서비스 보안 아키텍쳐

웹 서비스가 지향하는 Cross-Platform 환경에서의 기업 내, 기업 간의 Application들이 원하는 방식으로 서로간의 기능을 공유하기 위해서는 필수적으로 대두되는 문제가 Security이다. 웹 서비스 Security는 기존의 Tightly-Coupled 분산 컴퓨팅이나 인터넷 웹 사이트 Security 기술의 단순한 적용만으로는 충족될 수 없다.

3.1 배 경

웹 서비스 Security를 고려할 때 흔히 쉽게 생각할 수 있는 것이 기존의 웹 사이트에서 사용하고 있는 Security 기술이다. 웹의 통신 프로토콜이 HTTP이기 때문에 웹 사이트의 Security는 HTTP상에서 이용 가능한 Authentication 위주의 Security 기술을 주로 적용해 왔다. HTTP의 Basic Authentication은 ID/ Password만으로 Authentication을 하는 가장 간단한 웹 사이트 Security이다. 이 방식은 사용자의 Password가 Plain Text 형태 그대로 인터넷상을 떠다니기 때문에 공격자에 의해서 노출될 위험성이 높

다. 이를 완화시키기 위해서 Password에 대한 Digest를 생성하여 이를 전송하는 방식 역시 Digest로 Password를 유추하기 힘들다 하더라도 Digest가 Plain Text로 날아가는 것은 마찬가지다. 이를 해결하기 위해서 검증된 가장 널리 쓰이는 방식이 SSL(Secure Socket Layer)를 사용하여 Line Encryption을 하는 방식이다. Line Encryption은 Point-to-Point Security 방식으로서 전송되는 모든 데이터가 전송 노드 사이에서 Encryption된다. 각 노드 단위로 Encryption/Decryption 되기 때문에 Multi-Hop Topology의 Security로서는 부적합하다. 사용자 쪽의 부담 때문에 웹 사이트의 Authentication 방식으로 널리 쓰이지는 않지만 PKI(Public Key Infrastructure)나 Kerberos 등의 암호학적인 방식도 있지만 상호 호환성이 없기 때문에 Cross-Platform 환경에 적용하기는 힘들다. 또한 Firewall 외부에 주로 서버를 운용하는 웹 사이트와는 달리 웹 서비스는 기업 내의 Application간의 통신이 필요하기 때문에 Firewall을 통과하면서 Security를 지원하는 방식이 요구된다. 이러한 상황에서 웹 서비스에 적합한 새로운 Security Architecture의 필요성이 대두되었다.

3.2 요구 사항

웹 서비스가 이루어지기 전에 Requester와 웹 서비스는 상호간에 서로를 확인해야 하며, Integrity, Confidentiality, Non-Repudiation등의 필수적인 Security 기능이 만족되어야 한다. 웹 서비스는 각각의 Security Policy를 갖고 있는 서비스 주체(Subject)간의 Trust, Federation 설정을 통한 상호 협력적인 방식으로 서비스가 이루어지기 때문에 웹 서비스 Security Architecture는 Security 기술 측면뿐만 아니라 비즈니스 프로세스 측면에서도 고려해야 한다. 따라서 웹 서비스 Security Architecture는 Heterogeneous 환경에서 다양한 Security 기술들의 Interoperability와 비즈니스 프로세스 측면의 Security를 효율적이고 안전하게 지원하기 위해 유연성 및 확장성이 있는 구조이어야 한다. 또한 Requester와 웹 서비스 사이에 여러 Intermediary가 존재하는 Multi-Hop Topology이기 때문에 양단 간의 End-to-End Security가 지원되어야 한다. 웹 서비스 Security Architecture의 요구 사항을 정리하면 다음과 같다.

- SOAP Message Confidentiality/Integrity/Authentication
- Requester와 웹 서비스의 상호간 Authentication
- Non-Repudiation
- Authorization에 따른 서비스의 Access Control
- End-to-End Security
- Challenge/Response 형태의 Security Context 설정
- 키 교환 및 Derived key
- Multiple Trust Domains 환경에서의 Trust/Federation의 설정 및 관리

3.3 접근 방법

웹 서비스 Security의 접근 방법은 다음과 같이 크게 두 가지로 나눌 수 있다.

3.3.1 Line/Network Level Security

Transport나 Network Layer차원의 통신프로토콜 지원을 받아서 Security를 지원하거나 IP Blocking을 통한 패킷 필터링으로 접근 제어를 하는 방식이 이에 속한다.

가. SSL

Application Layer와 Transport Layer 중간에서 서버와 클라이언트 양단 간의 Handshaking을 통한 Line Encryption으로 Security를 지원하는 방식으로 현재 많이 쓰이고 있다. SSL은 검증된 프로토콜이지만 End-to-End Security가 지원되지 않고 Line Encryption으로 인해 성능 측면에서 큰 영향을 받기 때문에 Credential과 같은 주요 정보를 전송할 때만 사용해야 한다[4].

나. VPN (Virtual Private Network)

VPN은 인터넷과 같은 공중망에서 구성된 Virtual Network를 의미하며 IPSec을 이용해 Network-Level Authentication, 데이터 Integrity/Encryption을 제공한다. 송신자와 수신자 사이에서 형성된 임시 Connection상에서의 Packet Tunneling으로 Long-Term Point-to-Point Security를 지원한다. Requester의 IP가 미리 고정적으로 알려지는 웹 서비스에 적용 가능하며 Connection이 Long-Term인 관계로 성능 면에서 문제가 있을 수 있다.

다. Firewall

Firewall은 외부 네트웍과 내부 네트웍 사이에서 Source IP 주소 혹은 포트 번호를 통한 Packet Filtering 기능으로 내부 네트웍을 보호하는 역할을 한다. 웹 서비스에서 적용될 때는 IP Blocking으로 허용되지 않은 접근을 차단한다. 따라서 Requester가 불특정한 웹 서비스 경우에는 적용이 곤란하다.

3.3.2 Message/Content Level Security

Message/Content Level에서 Security의 기본 기능을 제공하는 것을 의미한다. 즉 Line/Network Level Security의 지원 없이 Application의 Message 자체에서 내부적으로 Security 메카니즘을 가지고 있다. 메시지에 단순히 Credential 넣어서 전송하는 것으로는 Security가 지원되지 않으며 메시지에 암호학적인 여러 기법을 사용하여 Authentication, Integrity, Confidentiality, Non-Repudiation을 제공한다. Message/Content Level Security에서 중요한 점은 End-to-End Security를 제공한다는 것이다. 이와 같은 장점 때문에 최근의 웹 서비스 Security의 흐름은 Message/Content Level Security로 가고 있다. 이에 크게 기여한 것이 W3C의 Security관련 여러 XML Specification들이다. 웹 서비스는 근본적으로 Requester와 웹 서비스 상호간의 SOAP Message 교환이라고 볼 수 있기 때문에 가장 기본적인 웹 서비스 Security는 XML, 즉 SOAP Message Security부터 시작해야 한다. 이런 관점에서 산업계에서 웹 서비스 Security Specification으로 가장 먼저 만든 것이 W3C의 Security 관련 XML Specification들을 기반으로 하여 SOAP Message Security를 기술한 WS-Security이다. WS-Security에 대해서는 뒤에서 보다 자세히 살펴보겠다.

3.4 제안된 웹 서비스 Security Architecture

웹 서비스 Security의 방향이 Message/Content Level Security로 잡혀가는 흐름 속에서 웹 서비스 Security의 요구 사항을 만족시켜 줄 수 있는 웹 서비스 Security Architecture의 모델을 기반으로 Security 요구 사항을 충족시켜 주는 표준 Specification의 필요성이 확산되었다. 표준 Specification 없이 기업들이 나름대로의 웹 서비스 Security 솔루션을 적용한다면 Interoperability가 떨어지게 되고 이를 맞추기 위해서 또 추가적인 작업이 소요되는 경우가 발생하게 된다. 이러한 상황을 인식하고 MS와 IBM은 위에서 언급한 웹 서비스 Security 요구 사항을 반영한 웹 서비스 Security Architecture를 제안하였다.

3.4.1 사상

실제 비즈니스 환경에서 웹 서비스 Security의 요구 사항을 만족하는, 포괄적이고 유연한, 표준에 기반 한 웹 서비스 Security Architecture를 구축할 수 있는 기술적인 전략을 제시한다. 포괄적이란 의미는 Security 기술적 측면(Line/Network Level Security, Message/Content Level Security)과 비즈니스 프로세스 측면(Policy, Trust, Federation)을 모두 포괄한다는 의미이다. 유연하다는 것은 특정 Security 기술에 의존적이지 않은 추상화된 기능적인 요구 사항으로 구성되며 향후의 진화된 기술 역시 쉽게 반영할 수 있는 구조라는 의미이다. 웹 서비스 Security의 요구 사항들은 모듈화/계층화된 여러 Specification들로 나뉘어서 기술되었다. 제안된 웹 서비스

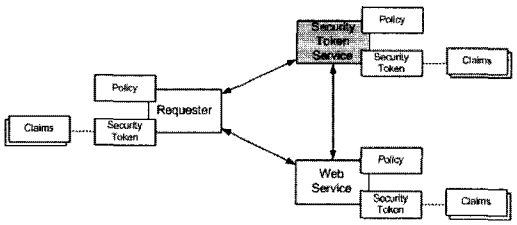
Security Architecture는 위의 사상을 반영할 수 있는 Security 모델을 토대로 요구 사항을 만족하는 모듈화/계층화된 Specification들로 구성된다.

3.4.2 Security 모델

(그림 1)에 웹 서비스 Security 모델을 나타내었다. 그림의 이해를 돕기 위해서 기본 적인 용어들에 대해서 간단하게 설명하면 다음과 같다.

가. 기본 용어

1) Claims : 웹 서비스의 Requester에 대한 표현들이다. 즉 이름, Identity, Key, Privilege 등이다.
2) Security Token : Security와 관련된 정보를 말한다. 즉 X.509 인증서, Kerberos 티켓과 Authenticator, Username 등이다.
3) Policy : 웹 서비스가 Requester의 Authentication이나 Authorization을 위해서 요구하는 Claims와 관련된 정보들이다.



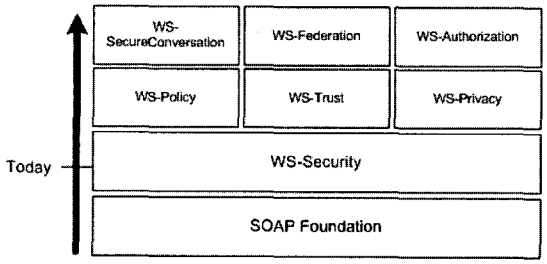
(그림 1) 웹 서비스 Security 모델

나. 개념

(그림 1)에서 알 수 있듯이 세 개의 주체들은 모두 같은 요소(Claims, Security Token, Policy)들을 포함하고 있다. 이것이 의미하는 것은 각각의 주체는 모두 서로의 역할을 할 수도 있다는 것이다. 웹 서비스는 자신의 Policy에 따라 서비스 이용을 위한 Claims를 요구하며 Requester는 웹 서비스가 요구하는 Claims의 Proof-of-Possession으로서의 역할을 하는 Security Token을 SOAP Message에 포함시켜 전송한다. Requester가 요구되는 Claims를 갖고 있지 않으면 제3의 Security Token 서비스에 이를 요청할 수 있다.

3.4.3 Specifications 계층 구조

MS와 IBM에서 제안하고 표준화를 위해서 작업중인 웹 서비스 Security Specification의 Roadmap은 (그림 2)와 같다. 현재 WS-Security가 발표되어 OASIS(Organization for the Advancement of Structured Information Standards)에 제출되어 검토되고 있는 상황이며 나머지 Specification에 대한 작업은 진행 중이다. (그림 2)에서 알 수 있듯이 Security에 관련된 Specification은 요구 사항별로 모듈화 되었으며 완성 단계별로 계층화되어서, 지금 현재 작업중인 Intial Specification과 이후에 완성될 예정인 Follow-On-Specification 의 두 그룹으로 계층화되었다.



(그림 2) 웹 서비스 Security Specifications

Roadmap에 있는 Specification들을 간략하게 소개하면 다음과 같다[1].

가. Initial Specification

- WS-Security : SOAP 메시지에 Signature와 Encryption을 하는 방법
- WS-Policy : SOAP 메시지를 처리하는 각각의 Actor들이 각각의 Security 정책을 나타내는 방법
- WS-Trust : Actor들간에 Direct나 Brokered Trust를 설정하는 모델과 방법
- WS-Privacy : Requester와 웹 서비스간에 Privacy 정보들에 대한 정책이 어떻게 기술되고 반영되는지에 대한 모델

나. Follow-On-Specification

- WS-SecureConversation : Requester와 웹 서비스간에 Security Context를 교환하는 방법, 상호간에 키 설정하는 방법
- WS-Federation : actor들간에 Federation을 어떻게 설정하고 관리하는지에 대한 방법
- WS-Authorization : 웹 서비스에 대한 접근제어 정책이 어떻게 표현되고 관리되는지에 대한 방법

3.4.4 WS-Security

MS와 IBM이 VeriSign과 함께 만든, 웹 서비스 Security의 기반이 되는 Specification으로서 SOAP Message에 대한 Message/Content Level Security에 대한 내용을 기술하고 있다. SOAP Message가 XML을 사용해서 구성되기 때문에 WS-Security Encryption/Signature 등의 Security를 위한 XML의 Specification들을 기반으로 이를 SOAP Message에 적용하는 방식을 기술하고 있다[2].

가. 목 적

SOAP Message에 대한 Integrity, Confidentiality, Authentication 제공을 통해서 웹 서비스 Application이 안전하게 SOAP Message 교환을 할 수 있게 한다. WS-Security에서는 크게 3가지 주요 메카니즘을 지원한다.

- Security Token Propagation
- Message Integrity
- Message Confidentiality

WS-Security는 특정한 Security 기술이나 Security 프로토콜에 의존적이지 않은 추상적인 모델의 유연한 방식을 제안하고 있다. 즉 다시 말해서 WS-Security의 초점은 설정된 Session을 통해서 Security Context와 Policy Agreement에 대한 Message Security를 제공하는 Single-Message Security Language를 표현하는 것이다.

나. 요구사항

WS-Security에서 요구사항으로 취급한 항목들은 다음과 같다.

- Multiple Security Token for Authentication or Authorization
- Multiple Trust Domains
- Multiple Encryption Technologies
- End-to-End Message-Level Security

다. Message Security 모델

앞에서 언급한 웹 서비스 Security Architecture의 모델을 따르고 있다.

라. 관련된 XML 표준들

WS-Security의 암호학적 메카니즘은 W3C가 제정한 다음의 두 표준에 기반을 두고 있다.

1) XML Encryption Syntax and Processing
- XML형태의 문서를 Encryption하는 Syntax와 Processing에 대한 내용으로 현재 Candidate Recommendation 상태이다.

2) XML Signature Syntax and Processing
- XML digital Signature의 Syntax와 Processing에 관한 내용으로서 Integrity, Message Authentication, Signer Authentication을 제공하는 방식을 기술하고 있으며 현재 Proposed Recommendation 상태이다.

WS-Security와 직접적인 관련은 없으나 키와 관련된 표준이 있다.

3) XML Key Management
- XML Encryption과 Signature를 위해 키를 분배하고 등록하는데 관련된 프로토콜을 정의한다.

마. 메카니즘

SOAP Message는 헤더와 바디로 구성된다. 헤더에는 Message와 관련된 메타 데이터를 표현하며 실질적인 데이터 내용은 바디에 나타난다. 따라서 Security와 관련된 정보들은 헤더에 한 Element로 나타나게 된다. WS-Security에서는 SOAP Message와 관련하여 Security의 기본 기능인 Confidentiality,

Integrity, Authentication 및 SOAP Message 송신자에 대한 Authentication을 지원한다. 여기에 사용되는 암호학적인 방법은 대칭키/공개키 Encryption/ Decryption, Digital Signature이다.

1) Security Token Propagation

Requester 혹은 웹 서비스의 Name, Password, X.509 인증서, Kerberos 티켓, 세션키 등 Authentication에 관련된 정보를 나타내는 Security Token을 SOAP 메시지 헤더에 나타낸다. WS-Security는 어떠한 형태의 Authentication 정보도 포함시킬 수 있는 유연하고 확장된 형태의 구조를 지원한다. Authentication 정보는 세션 키로 Encryption 되거나 송신자의 개인키로 Digital Signature 값과 함께 전송된다.

2) Message Integrity

XML Digital Signature Specification에 따르는 Digital Signature 방식을 지원한다. Digital Signature는 헤더에 위치하며 헤더와 바디에 있는 데이터의 전부 혹은 일부에 대해서 Digital Signature를 생성할 수 있다. 생성하는 과정을 단계별로 나누면 다음과 같다.

- ㅇ Reference Element에서 지정하는 데이터를 추출
- ㅇ 데이터 정규화
- ㅇ Digital Signature가 정확히 적용되는 데이터로 변형
- ㅇ Digest값 생성
- ㅇ Digest값을 개인 키로 Encryption하여 Digital Signature 생성

SOAP 헤더에는 Digtal Signature의 검증에 필요한 송신자의 X.509 인증서와 함께 각각의 단계에서 쓰인 알고리즘이 어떤 것인지에 대한 정보도 포함된다.

3) Message Confidentiality

XML Encryption Specification의 Encryption 방식을 기반으로 한다. SOAP 헤더와 바디의 일부 혹은 전부나 Attachment를 Encryption 할 수 있다. Encryption에 관련된 정보는 헤더에 위치하며 Encryption되기 전의 SOAP Message 부분이나 Attachment는 Encryption된 Cipher Text로 대치된다. Encryption 과정을 단계별로 나누면 다음과 같다.

- ㅇ Encryption하려는 부분이나 Attachment를 지정
- ㅇ Encryption 한 후 원문을 대치
- ㅇ 필요한 경우 Encryption에 사용된 키를 수신자 공개키로 Encryption

4) 추가적인 Security 고려사항

사실 Digital Signature만으로는 완전한 Authentication이 지원되지 않는다. 이미 사용된 Signature 값을 재 사용하는 Replay 공격에 대한 취약점이 있기 때문이다. 이는 돈 거래가 수반되는 상황에서는 매우 심각한 문제를 야기할 수 있다. 또한 서명자에 의한 서명 부인 문제도 발생한다. 일반적인 Digital Signature에서 발생하는 문제기 이를 SOAP Message에 적용할 때도 역시 발생하기 때문에 이에 대한 해결책이 고려되어야 한다. Signature와 함께 난수 값, Timestamp, Sequence Number, Expirations, Message Correlation 등의 정보를 같이 전송하는 방안을 고려해야 한다.

바. 에러 처리

WS-Security에서는 Security 정보를 처리하면서 발생한 에러의 종류를 "unsupported"와 "failure"의 두 가지로 나눴다. "unsupported"의 경우는 수신자는 송신자에게 "supported" 형식에 대한 정보를 주어야 한다. "failure"의 경우는 DOS (Denial Of Service) 공격의 가능성 때문에 응답하지 않는다.

5. 결 론

인터넷 및 분산 컴퓨팅 환경의 혁신으로 나온 웹 서비스가 활성화되기 위해서 해결해야 할 주요 난제로 지목되고 있는 Security는 기존의 Line/Network Level Security가 End-to-End Security 등 근본적인 웹 서비스 Security를 지원할 수 없음에 따라 중심이 Message/Content Level Security로 옮겨가면서 웹

서비스 Security Architecture와 이를 뒷받침하는 Specification의 제정 움직임이 활발하게 일어나고 있다. 이러한 흐름이 가능했던 것은 표준화 쪽에서 W3C가 XML 및 SOAP의 Security와 관련된 표준 Specification 작업을 활발하게 해왔으며 이를 기반으로 산업계에서 MS와 IBM이 중심이 되어 웹 서비스 Security를 위한 표준 Specification 작업에 적극적인 자세를 보였기 때문이다. 최근에는 이러한 표준화 움직임에 미온적이었던 SUN 역시 동참 의사를 밝히고 있다. 이런 흐름 속에서 MS와 IBM은 웹 서비스를 위한 Security Architecture와 Specification Roadmap을 제안하였다. 현재 Specification Roadmap의 기반이라고 할 수 있는 SOAP Message Security에 관한 내용인 WS-Security가 공개되었으며 OASIS는 이를 긍정적으로 검토하고 있다. OASIS에 참여하고 있는 산업계의 다른 여러 기업들은 이미 WS-Security에 대한 적극적인 지지를 보이고 있다. 제안된 웹 서비스 Security Architecture는 안전한 웹 서비스를 구현하려는 기업들이 기존에 사용하고 있는 Security 기술을 활용할 수 있는 포괄적이고 유연한 구조이다. 이를 토대로 하는 일련의 Specification 작업은 이제 시작 단계이기 때문에, 비록 MS와 IBM등 대기업이 주도하고는 있으나 IT산업계에서 어느 정도의 호응을 얻을지는 사실 미지수다. 하지만 웹 서비스가 표준을 토대로 나왔고 제안된 웹 서비스 Security Architecture 역시 기본 사상은 이를 벗어나 있지 않으며 웹 서비스의 Security는 시급히 해결되어야 할 문제라는 공감대가 많기 때문에 그 전망은 밝다고 하겠다.

참고문헌

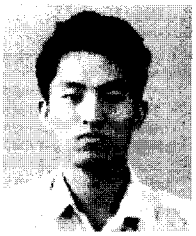
[1] MS, IBM, "Security in a Web Services World: A Propsosed Architecure and Roadmap", April, 2002.

[2] MS, IBM, VeriSign, "Web Services Security (WS-Security)", April, 2002.

[3] Bloor Research, "Web Services Gotchas", July, 2002

[4] Todd Sunsted, "Building Security into Web Services", August, 2001

저자약력

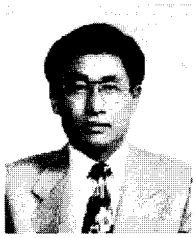


이 해 규

1989년 서울대학교 컴퓨터공학과 (학사)
1991년 서울대학교 컴퓨터공학과 (석사)
1992년 KT S/W연구소 전임연구원
2001년 서울대학교 컴퓨터공학과 (박사과정수료)
현재 KT 서비스개발연구소 선임보연구원
관심분야 : 인증, 접근제어
e-mail : hkrhee@kt.co.kr

이 상 수

1984년 한국항공대학교 전자공학과
1986년 New Jersey Institute of Technology 전자공학 전공
1987년 KT 사업지원단 전임연구원
2000년 IMT-2000기반 멀티미디어서비스 개발
2001년 유무선통합서비스 개발
현재 KT 서비스개발연구소 공통플랫폼연구실장, 선임연구원
관심분야 : Pervasive Computing, CBD, 차세대 플랫폼
e-mail : ssllee@kt.co.kr



김 문 규

1974년 숭실대학교 전산학과
1974년 한국과학기술연구소(KIST) 연구원
1979년 KIST 부산사무소 소장
1987년 한국과학기술원(KAIST) 책임연구원
1988년 KAIST 강남분소 소장
1991년 숭실대학교 정보과학 대학원
1994년 KT S/W연구소 교육훈련팀장
1995년 KT 정책개발총괄팀장
1996년 KT 전산기획국장
2001년 KT 정보보호연구팀장
2002년-현재 KT 서비스개발연구소 플랫폼연구팀장
(경영직)
관심분야 : e-Business, e-Security, 차세대플랫폼
e-mail : mkkim@kt.co.kr