DOI: 10.3745/KIPSTD.2009.16D.5.809

# RIA환경에서 SAML을 이용한 SSO에 관한 연구

조 동 일[†] · 류 성 열[††]


## 요 약

도메인 내부 시스템간의 인증 통합에만 초점이 맞추어져 있던 지금 까지의 SSO와는 다르게 SAML을 이용한 SSO는 기존 웹 환경에서 서로 다른 도메인 간의 인증을 지원할 수 있어, RIA에 적용되면 높은 시너지 효과를 기대할 수 있다. 하지만 SAML을 이용한 SSO에 관한 연구는 서버 간 인증 정보 교환에 대한 연구가 주를 이루고 있어, 클라이언트에서 많은 처리를 수행하는 RIA환경에 적용하기 위해서는 RIA와 SP, SP와 IDP 그리고 RIA와 IDP간의 통신 프로토콜의 정의에 관한 연구가 필요하다.

본 연구는 SAML을 RIA에 적용하기 위한 아키텍처를 제안하고 아키텍처 안에서의 통신 프로토콜을 정의하였다. 그리고 제안한 아키텍처를 실제 구현하여 RIA의 데이터 통신 프레임워크로 많이 사용되는 DWR에 적용하고 다른 SSO 솔루션과 비교하여 그 활용성을 확인하였다.


**키워드 : SAML(Security Assertion Markup Language), 단일인증, RIA(Rich Internet application)**

# A SAML based SSO method in RIA environment

Dongil-Il Cho[†] · Sung-Yul Rhew[††]


## ABSTRACT

Current SSO has focused on authenticated integration between inter systems in a domain.On the contrary, becasuse SSO using SAML can support integration between different domains, once it is used in RIA, we can expect highly synergic effect. However, because researches on SSO using SAML are mainly those on authenticated information exchange between servers, a special research is needed in order to be applied in RIA environment, which conducts numerous managements in client.

This study proposes architecture and explain a practice structure in order to apply SAML to RIA. Also, this study has embodied the proposed architecture and applied it on DWR, which is used mostly as Data communication framework of RIA, and verified the useablity of this architecture.


**Keywords : SAML(Security Assertion Markup Language), SSO(Single sign-on), RIA(Rich Internet application)**

## 1. 서 론

SSO(Single sign-on : 단일 인증)는 단 한번의 인증으로 서로 다른 시스템의 서비스들를 동시에 제공할 수 있다는 장점 때문에 EP(Enterprise Portal) 및 EAI(Enterprise Application Integration)등에 필수 요소로 사용 되고 있다. OASIS에서 SAML(Security Assertion Markup Language)을 발표하기 이전까지 SSO는 도메인 내부 시스템의 인증을 통합하는 것에 초점이 맞추어져 있었고, 뚜렷한 표준 없이 구현하는 밴더에 따라 제각각 다른 기술과 표준을 사용하여 상호 호환성을 기대하기 어려웠다. SAML은 환경의 변화 없이, 서로 다른 도메인 간 인증 정보 교환을 가능하게 하는 인증 표준이다. SAML은 웹 환경에서 인증 데이터를 웹 브라우저를 통해 전송하여 도메인간 인증정보를 교환하고, 네트워크 보안을 위해 XML 전자 서명과, PKI(Public key infrastructure)를 이용한 암호화를 지원한다[3]. SAML SSO 시스템을 이용한 서비스 시스템은 크로스 도메인 서비스를 사용자에게 제공할 수 있게 되었다. 이 기술은 최근 각광 받고 있는 RIA(Rich Internet Application) 기술과 접목 할 경우 많은 장점을 도모할 수 있다.

RIA는 브라우저와 서버간 비동기적 데이터 통신을 지원하고 동적으로 UI를 생성하여 높은 사용자 경험을 제공할 수 있다[7]. 이 서비스는 여러 RIA 응용프로그램과 협력할 경우 더 높은 수준의 서비스를 제공할 수 있기 때문에 크로스 도메인 SSO을 적용할 경우 많은 장점을 도모할 수 있다.

※ 본 연구는 숭실대학교 교내 연구비 지원으로 이루어졌음.
† 정 회 원 : 숭실대학교 컴퓨터공학과 박사과정
†† 종신회원 : 숭실대학교 컴퓨터학부 교수
논문접수 : 2009년 3월 26일
수 정 일 : 1차 2009년 5월 14일, 2차 2009년 6월 2일
심사완료 : 2009년 6월 4일

하지만 기존 SSO 솔루션은 서버와 서버간의 사용자 인증 처리에만 중점을 두고 있기 때문에, 사용자 브라우저에서 독립 응용프로그램으로서의 역할을 수행하는 RIA환경에 SSO를 적용하기 위해서는 추가적인 연구가 필요하다. RIA에 SAML을 적용하기 위해서는 클라이언트 환경에서 구동하는 RIA 응용프로그램을 별도의 인증 개체로 정의하고 RIA와 SP, SP와 IDP 그리고 RIA와 IDP간의 통신 프로토콜과 RIA의 비동기적 데이터 통신 매커니즘을 지원하기 위한 인증 아키텍처가 정의 되어야 한다.

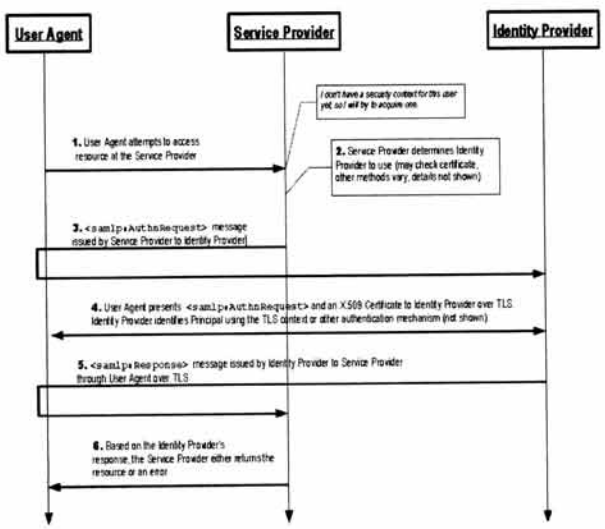
## 2. 관련 연구

### 2.1 SAML을 이용한 웹 응용프로그램 단일인증 기법

SAML은 OASIS에서 발표된 XML기반 인증정보의 교환을 위한 프레임워크이다. SAML은 도메인간의 인증 및 인가 정보를 브라우저를 경유하여 서로 주고 받아 인증 및 인가를 처리한다[2, 3].

사용자가 SP(Serivce Provider)에 서비스를 요청하면 SP에서는 인증된 사용자인지를 검증하고 인증되지 않았을 경우 AuthnRequest 메시지를 생성하여 브라우저에 IDP(Identity Provider)로 전달을 요청한다. 사용자 브라우저는 SP로부터 전달받은 SAML 메시지를 IDP로 전달하고, IDP는 요청 메시지를 받아 사용자의 인증을 처리한다. 인증이 완료되면 Response 메시지를 생성하여 사용자 브라우저에 SP로 전달을 요청하고, 브라우저는 IDP로부터 받은 메시지를 SP로 전달한다. SP는 전달 받은 SAML을 분석하여 사용자의 인증을 처리한다[4].

SAML은 확장성이 매우 뛰어나고, 오픈 표준이기 때문에 보다 먼 개발을 위한 기준으로 활용될 수 있다[13]. 특히 별도 프로그램의 인스톨 절차가 필요없고 현재 웹 환경에서 플랫폼에 상관없이 구동할 수 있기 때문에 RIA에 적용이 용이한 구조를 가지고 있다.




(그림 1) SAML 인증 프로세스[4]

### 2.2 일반 웹 응용프로그램과 RIA

데스트톱 소프트웨어의 많은 특징과 기능을 가지고 온라인에서 동작하는 RIA의 중요성과 대중성은 날로 증가하고 있다[10]. RIA는 새로운 기술이 아닌 이전부터 있어 왔던 AJAX, JavaScript, CSS 등의 기술을 이용해 비동기적 서버 통신, 동적 UI생성 등으로 풍부한 사용자 경험을 제공한다[7].

국내에서 큰 인기를 얻었던 X-Internet의 경우 주로 IE (Internet explorer)에서 구동하며, ActiveX 기술을 이용하여 브라우저에 플러그인 되어 동작하는 구조로, OS 및 브라우저간 호환성이 떨어지고, 보안에 취약하다는 문제가 있다. 반면 RIA에서 사용하는 Ajax 기술은 Web 2.0의 개방성, 확장성, 표준화를 지원하고, ActiveX의 설치를 요구하지 않는다. 이는 검증되지 않은 ActiveX를 설치함으로써 발생할 수 있는 보안문제를 해결할 수 있을 뿐만 아니라 별도의 모듈 없이 OS 및 브라우저간 호환성을 제공한다. 또한 브라우저에서 동작하는 Javascript 코드는 브라우저의 sandbox 안에서 구동하기 때문에 안전하고 보안의 위험이 적다[10].

RIA기술은 기존 웹기술들에 비해 서버측과 클라이언측 모두 동적이고 인터랙티브한 기술들을 말한다. 특히 컴파일된 코드의 실행이 아닌 XML과 해당스크립트를 실시간에 해석하여 수행하기 때문에, 사용자와 상호작용을 더욱 긴밀하게 할 수 있다. 또한 RIA는 Javascript를 이용해 서버의 많은 비즈니스 로직을 클라이언트 브라우저에서 동작 시키므로, 여러 컨텐츠 서버에서 비동기 적으로 정보를 수집해 서비스 하는 것이 가능하며, 지능적인 비동기 요청을 이용하여 네트워크 트래픽을 줄이는 새로운 클라이언트-서버 아키텍처를 제공한다[11].

### 2.3 RIA 기반 단일 인증 기법

대부분의 인터넷 응용프로그램은 보다 많은 기능과 컨텐츠를 제공받기 위해 사용자 인증을 필요로 하는데 RIA는 동적으로 컨텐츠를 생성하기 때문에 사용자의 인증 요구와 권한 체크 요청이 비동기적으로 발생한다[12]. 또한 여러 서버의 컨텐츠를 이용할 경우 사용자와 서버간의 인증이 필요하게 되는데, 이때의 인증 처리는 RIA의 장점을 침해하지 않으면서 처리할 수 있어야 한다.

SAML의 주요 기능은 사용자가 한 도메인에서 인증을 받고 다른 도메인에서 재인증 없이 자료를 사용할 수 있게 하는 SSO 이다[5]. SAML은 기존에 지원하는 기술을 이용해 도메인간의 SSO를 지원하기 때문에 RIA와 연동할 경우 높은 시너지 효과를 발휘할 수 있다. 하지만 현재까지의 SAML 관련 연구는 서버차원의 인증을 주로 다루고 있어서, RIA에 적용하기 위해서 Ajax 어플리케이션의 장기 지속성, 구속성, 비동기성과 같은 재인증을 촉발하는 특징들을 지원하기 위해 기존과 다른 인증 메커니즘을 필요로 한다[1].

### 2.4 FLEX의 LiveCycle ES

어도비사의 Flex는 대표적인 RIA 응용프로그램 개발툴로서 독보적인 미려한 UI를 제공하여 널리 쓰여지고 있다. 어

도비에서는 상대적으로 미약한 Flex의 서버 모듈을 보완하기 위해 엔터프라이즈 서비스 기반의 LiveCycle ES라는 서버 프레임워크를 제공한다.

LiveCycle ES는 Java 기반의 서버 프레임워크로 Flex로 개발된 RIA 응용프로그램과 상호 작용한다. LiveCycle ES의 SSO는 중앙 제어 방식으로 LiveCycle ES로 개발된 여러 웹 응용프로그램의 인증을 단일화 한다[15]. LiveCycle ES는 HTTP Cookie를 이용하여 사용자 인증 정보를 저장한다. 각각의 LiveCycle ES 서버 응용프로그램은 브라우저에서 전달 받은 인증 Cookie를 이용하여 단일 인증을 실현한다.

LiveCycle ES의 SSO은 Flex 응용프로그램에 쉽고, 서비스 중단 없는 인증 서비스를 제공할 수 있지만 HTTP Cookie를 사용하기 때문에 서로 다른 도메인간의 서비스를 제공할 수 없고, Cookie 자체가 가지는 휘발성 및 보안성의 문제를 그대로 가진다는 단점이 있다.

## 3. SAML 기반 RIA SSO

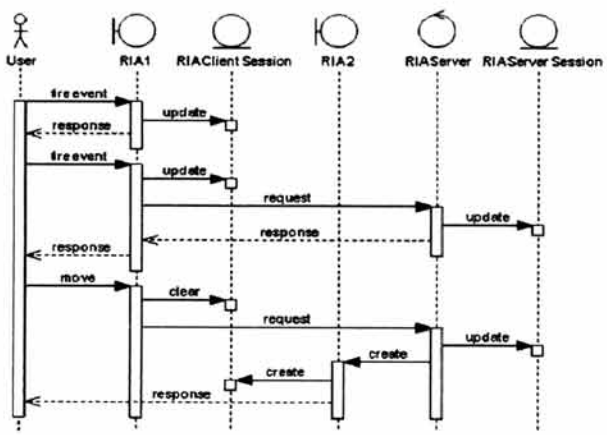
### 3.1 RIA의 실행 구조

#### 3.1.1 RIA의 실행 구조

RIA는 브라우저에서 서버와 통신하여 서버의 특정 컨텐츠를 제공받아 처리하는 부분과, 브라우저 내부의 자원만으로 처리하는 프로세스로 나눌수 있다[10]. 각 프로세스를 처리하기 위해 사용자 세션은 서버와 브라우저 각각에서 유지된다.

RIA는 브라우저 내부의 RIA세션을 유지하면서 화면 단위의 사용자 요청을 처리하고, 필요에 따라 서버와 통신하여 정보를 처리한다.

다른 프로그램으로의 전환과 같은 사용자의 요청이 발생하면 RIA는 현재의 브라우저의 자원을 해제하고 이동할 화면의 정보를 서버에 요청한다. 서버는 사용자가 요청한 RIA UI를 브라우저에 전달하고 전달된 RIA UI는 새로운 브라우저 세션을 생성하고 사용자의 요청을 처리한다.




(그림 2) RIA의 데이터 처리 시퀀스 다이어그램

서버는 사용자 정보를 세션에 저장하고, 브라우저의 요청을 세션을 참조하여 처리한다. 브라우저 세션은 RIA 응용프로그램 단위로 생성되고 유지된다. 브라우저의 세션은 서버의 세션정보를 참조하여 생성되고, 브라우저 내부 프로세싱에 의해 갱신 되며, 생성 이후 프로그램 종료까지 무결성은 RIA 내부에서 보장 한다.

서버는 최초 인증 이후 고수준의 서비스에 대한 요청 또는 서버 인증이 만료되거나 다른 액티비티에 의해 인증이 종료되는 현상이 발생되어, 재인증을 필요로 할 수 있다. 그리고 브라우저의 서버 요청이 각각 다른 서버에서 구동할 경우 최초 인증에 대한 문제가 발생할 수 있다. RIA는 프로세스 도중 사용자의 인증문제가 발생하더라도 브라우저의 사용자의 상태가 보존되어 인증 이후 재작업 없이 지속적인 프로세스 진행이 가능하여야 한다.

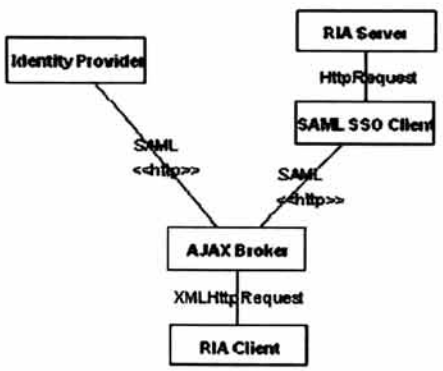
본 연구에서는 서버의 요청이 처리 될 때 초기 SSO인증을 처리하고, 프로세스 도중 인증 세션이 만료 되더라도 브라우저의 세션을 보존한 상태에서 SSO를 처리하여 브라우저에서는 인증 이후 다음 프로세스를 연속적으로 처리할 수 있도록 하는 SAML기반 SSO 아키텍처를 제시한다.

### 3.2 SAML 기반 RIA SSO의 아키텍처

일반적인 SAML SSO 솔루션은 SP의 서비스 처리전(웹서버 플러그인 등)에 사용자 인증에 관한 체크를 수행한다. 인증되지 않았거나 인증이 만료 되었다면 SAML SSO 클라이언트는 IDP로 인증을 요청하고, IDP는 인증 프로세스를 수행하게 된다[4].

RIA에서는 Client에서 발생하는 서버 요청이 비동기적으로 발생하기 때문에 서버에 요청하는 객체인 XMLHttpRequest를 관리하는 모듈이 필요하다. 이 모듈은 RIA와 SSO 서버의 통신을 관리하며, RIA에 적합한 SSO를 수행한다.

SAML은 서버간 인증 정보를 주고 받을 때 브라우저를 경유하게 된다[3]. RIA환경에서는 서버와 브라우저간의 통신은 HTML기반의 통신과 AJAX를 이용한 XML기반의 데이터 통신 두가지가 모두 가능하다. HTML기반 통신은 기존 SAML SSO 솔루션에서 처리가 가능하지만, XML기반의 데이터 통신은 별도의 처리가 필요하다. 또한 AJAX통신은 비동기적으로 발생할 수 있기 때문에, 인증을 처리하는 도




(그림 3) RIA에서 SAML SSO 커뮤니케이션

중에도 요청이 발생할 수 있다. 이때 시스템은 발생하는 요청을 저장해 두었다가 인증 완료후 요청을 처리하는 기능이 필요하다. RIA SAML SSO에서는 이 문제를 모듈간 프로토콜과 AJAX Broker를 통해 해결한다.

### 3.2.1 AJAX Broker의 구조 및 동작 방법

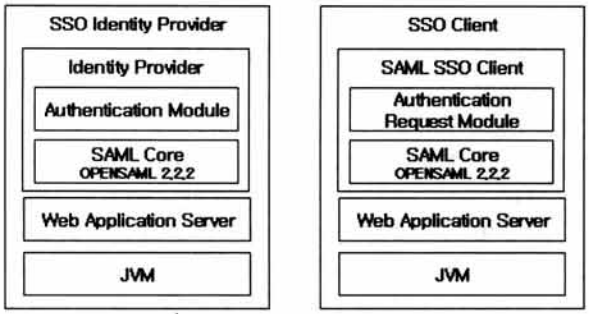
AJAX Broker는 SAML SSO 시스템과 RIA를 인터페이스하는 모듈로 RIA에 플러그인 되어 동작한다.

RIA의 비동기적 서비스 요청은 AJAX Broker에서 Queue를 통해 관리한다. AJAX Broker는 인증에 실패해 서버로부터 서비스의 결과를 받지 못한 request와 인증중 발생한 request를 Queue에 저장하고, 인증이 완료되면 자동으로 Queue에 담긴 request를 재실행 하여 서비스를 재개한다.

XMLHttpRequestPool은 XMLHttpRequest를 생성하는 클래스로 XMLHttpRequestFactory를 이용해 XMLHttpRequest를 생성하고 XMLHttpRequestWrapper로 래핑하여 리턴한다. RIA에서는 AJAX Broker를 plugin하기 위해 XMLHttpRequestPool을 이용하여 XMLHttpRequest를 생성해야 한다. XMLHttpRequestFactory는 브라우저 마다 다른 XMLHttpRequest의 생성 패턴을 추상화한다. XMLHttpRequestWrapper는 SAML SSO를 처리하기 위해, 서비스 요청시점의 request 정보를 보존하고 서비스 인증 이후 재요청을 처리할 수 있는 기능을 제공한다. Auth 클래스는 SP의 응답이 인증 요청인지를 식별하고 인증을 처리하는 기능을 수행하고, AJAX 요청을 Queue를 통해 관리하며 인증 완료 후 Queue의 요청을 재전송 하는 작업을 수행한다.

### 3.2.2 IDP 및 SAML SSO Client의 구조

SAML Client와 IDP의 구조는 기존 SAML기반 SSO 솔루션과 다르지 않다. (그림 5)는 본 연구에서 구현된 SAML SSO Client와 IDP의 구조이다.



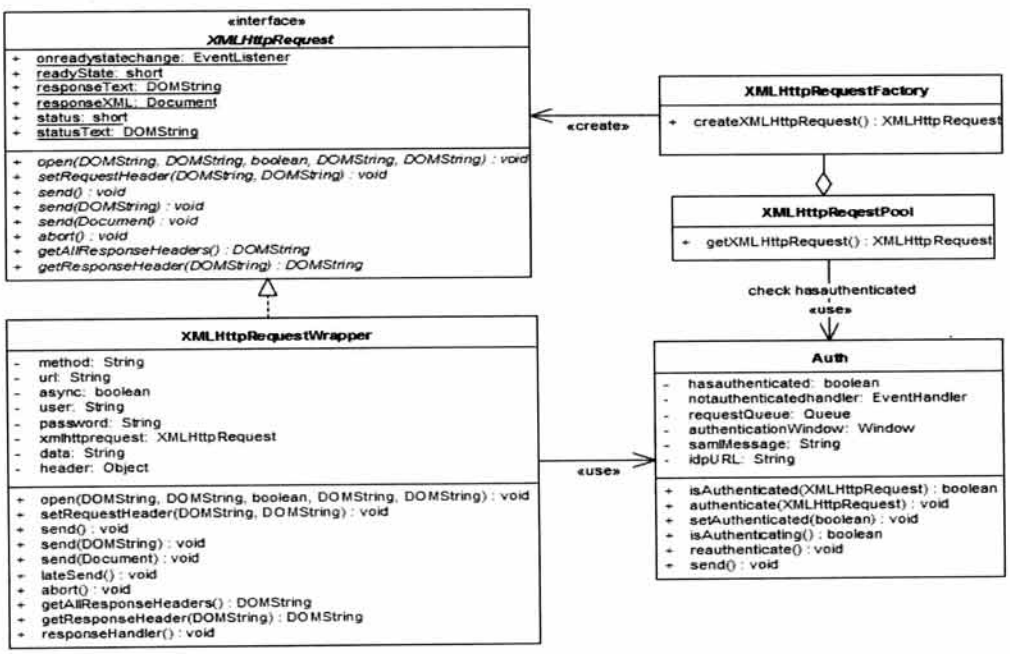

(그림 5) IDP 및 SAML SSO Client 시스템 구조

본 연구에서 개발된 시스템은 www.opensaml.org에서 제공되는 opensaml 2.2.2를 이용하였고 Java언어를 기반으로 개발하였다[9]. 개발된 시스템은 시스템의 범위를 줄이기 위해 인증만을 처리하며, 보안성을 제공하기 위해 XML전자서명과 PKI를 이용한 암호화를 지원한다.

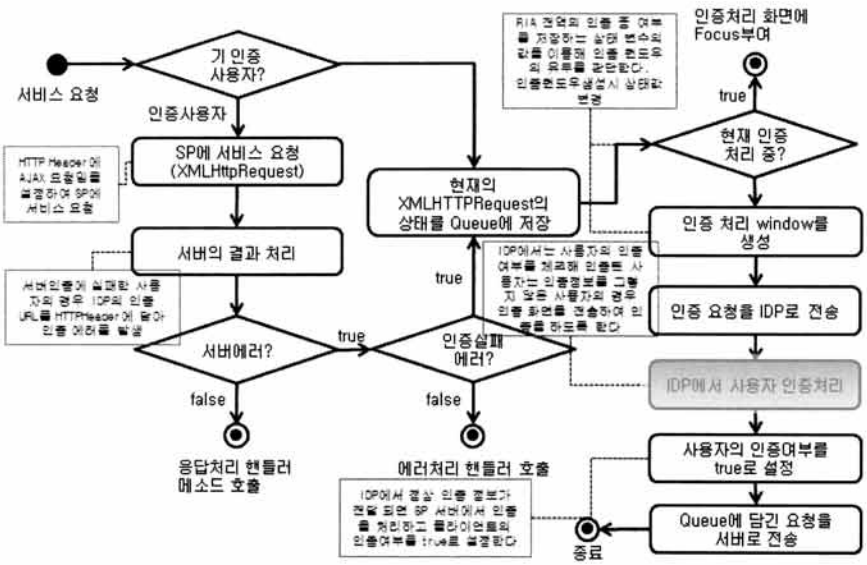
### 3.3 모듈간통신 프로토콜 및 인증 절차

### 3.3.1 모듈간 SAML 통신 프로토콜

본 연구에서는 RIA의 두가지 통신 모드를 해결하기위해 HTTP 상태 코드와 HTTP Header를 이용한다. AJAX Broker는 서버에 서비스를 요청할 때 예약된 HTTP Header로 Ajax 요청임을 명시하여 SP로 전송한다. SP의 SSO 클라이언트는 사용자 인증여부를 체크해 인증이 필요한 경우 전달 받은 HTTP Header 값을 이용하여 인증 이후 SP에서 인증을 처리할 URL을 선택하고 IDP에 전송할 SAML을 생성한다.




(그림 4) Ajax Response Broker 클래스다이어그램




(그림 6) RIA 기반 SAML 인증 절차 액티비티

SAML의 생성이 완료 되면 SP는 HTTP 상태 코드를 401(Unauthorized)로 설정하고, Ajax 요청인 경우에는 HTTP Header로 IDP의 URL을 HTTP Body에는 SAML을 요청을 담아 전송하고, 일반 HTTP 요청인 경우에는 SAML을 IDP로 전송하는 코드가 포함된 리다이렉션 HTML을 브라우저로 전송한다.

SP로부터 응답을 받은 AJAX Broker는 HTTP 상태 코드로 인증의 필요 여부를 판단하고 인증이 필요한 경우 응답 정보에서 SAML을 추출하여 IDP로 전달한다. 전달은 팝업 윈도우를 통해 전달하며 이때 사용된 팝업 윈도우는 AJAX Broker에 저장해 현재 인증 프로세스가 진행중인지를 판단하는데 사용한다.

3.3.2 SAML 기반 RIA SSO의 사용자 인증 절차

RIA기반의 SAML SSO 인증절차는 다음과 같다.

AJAX Response Broker는 다음과 같은 절차로 RIA와 SP 그리고 IDP와 연동한다.

1. RIA는 XMLHttpRequestPool을 이용해 획득한 XMLHttpRequest의 send() 메소드를 호출한다. XMLHttpRequestWrapper는 현재 세션이 인증처리 되었는지를 확인한다

1.1 인증처리가 되었을 경우

1.1.1 SP로 서비스를 요청한다

1.1.2 SP로부터 응답을 받으면 IDP로 인증을 요청하는 응답인지를 체크한다

1.1.2.1 IDP로 인증을 요청하는 응답인 경우

1.1.2.1.1 (1.2.1)를 수행한다.

1.1.2.2 정상적인 서비스 결과데이터인 경우

1.1.2.2.1 등록된 응답 핸들러를 실행한다.

1.2 인증처리가 되지 않았을 경우

1.2.1 AJAX Response Broker는 현재 요청정보를 Request Queue에 저장하고 현재 인증중인지 검사한다

1.2.1.1 인증중이 아닐 경우

1.2.1.1.1 인증 윈도우를 생성하고 IDP로 인증을 요청한다. 인증 윈도우는 내부에 저장한다.

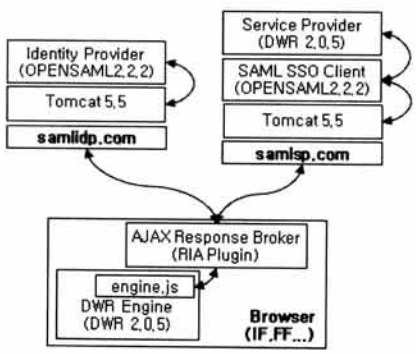
1.2.1.1.2 IDP의 사용자 인증이 완료되면 AJAX Response Broker는 SP에 사용자 세션을 생성을 요청하고 생성이 완료되면 Auth 클래스의 인증여부를 true로 설정한다

1.2.1.1.3 AJAX Response Broker는 인증여부가 true로 설정될 때 Request Queue에 있는 요청을 SP로 재전송 하여 인증전과 인증중 발생한 요청을 처리한다.

## 4. DWR을 이용한 검증 및 비교

### 4.1 DWR 적용 사례

본 연구에서는 제안한 SAML 기반 RIA SSO 솔루션의 활용을 검증하기 위해 RIA 기반의 데이터 통신 솔루션으로 많이 활용되고 있는 DWR에 제안 솔루션을 다음과 같이 적




(그림 7) DWR에 적용

용시켰다[8].

적용 환경은 다음과 같다.

- JDK 1.5.0/tomcat 5.5.26/dwr 2.0.5/opensaml 2.2.2/saml2.0

별도의 Tomcat 서버에 IDP와 SP를 구축하고 각각 samlidp.com과 samlsp.com 이라는 별도 도메인을 구성하여 테스트 하였다.

IDP와 SAML SSO Client는 Opensaml 2.2.2기반으로 구현 하였다. SP에서는 *.dwr 요청에 대해 인증을 수행하도록, SP의 web.xml에 SAML SSO Client를 필터로 등록하여, DWR 요청에 전처리를 수행할 수 있도록 설정하였다.

브라우저에서 동작하는 부분은 DWR에서 런타임 엔진에 해당하는 engine.js에 XMLHttpRequest를 생성하는 부분을 AJAX Response Broker를 이용하도록 소스를 수정하여 적용 시켰다.

마지막으로 IDP와 SP에 각각 public key와 private key를 생성하고 각 서버의 public key를 교환하여 상호 교환하는 SAML 문서를 전자서명과 검증을 수행할 수 있도록 하였다.

테스트는 Internet Explorer v7, Firefox v3, Crome v1에서 진행 하였으며 결과 화면은 다음과 같다.


(그림 8) DWR 적용 테스트 결과

① DWR 데모 페이지에서 SP에 서비스를 요청하는 정보를 입력
② IDP의 로그인 화면 출력
③ 로그인 이후 DWR 데모 페이지에서 서비스 결과 화면 출력

추가로 다음과 같은 DWR 데모에 적용하였을때에도 이상없이 구동하였다.

프로토콜 유효성은 테스트케이스에 적용했을 때 시나리오 대로 정상동작하는지 여부를 검증한 것이고, 서비스 지속성은 제안한 아키텍처 프로토콜의 강점인 RIA의 서비스 중단 없이 재인증 및 인증을 처리할 수 있는 특징을 검증한 것이다.

〈표 1〉 테스트 케이스별 적용 결과

| 테스트케이스 | 프로토콜 유효성 | 서비스 지속성 |
|---|---|---|
| Dynamic Text | ○ | ○ |
| Resource Forwarding | ○ | ○ |
| Editable Table | ○ | ○ |
| Fast Address Entry | ○ | ○ |
| General Interface | ○ | ○ |
| Anti-Spam mailto | ○ | ○ |
| Server-Side Clock | ○ | ○ |
| Javascript Chat | ○ | ○ |
| Java Chat | ○ | ○ |

### 4.2 다른 SSO 시스템과 비교

RIA는 다음과 같은 특징을 가진다[14].

ⓐ 별도의 설치 작업 없이 사용자의 접근이 가능
ⓑ 클라이언트에서 강력한 응용프로그램 프로세스를 지원
ⓒ 네이티브 데스크탑 GUI와 밀접하게 매칭되는 풍부하고 미려한 사용자 인터페이스를 제공
ⓓ 보다 효과적인 정보 전달을 가능
ⓔ 매우 높은 사용의 용이성을 제공
ⓕ 유지보수를 단순화

다음은 서버기반 인증 솔루션인 JetSpeed2와 어도비의 LiveCycle ES의 SSO과 본 연구에서 제안한 아키텍처를 RIA에 적용한 결과 RIA의 특징의 보존 여부에 대한 비교표이다.

〈표 2〉 RIA에 SSO 적용성 비교

| 특징 | JetSpeed2 | LiveCycle ES | RIA기반 SAML SSO |
|---|---|---|---|
| ⓐ | Y | N | Y |
| ⓑ | N | Y | Y |
| ⓒ | Y | Y | Y |
| ⓓ | N | Y | Y |
| ⓔ | Y | Y | Y |
| ⓕ | N | Y | Y |

비교 결과 RIA기반 SAML SSO는 RIA의 특징을 유지한 상태로 SSO을 지원할 수 있지만, 서버인증 방식의 SSO은 프로세스 관련된 RIA의 기능에 제한을 준다.

<표 3>은 SAML이 다른 SSO에 차별성을 가지는 특징

이고 이를 통해 상기 세 SSO 솔루션을 비교한 표이다[17].

JetSpeed2와 LiveCycle ES의 경우 도메인 제한적인 HTTP Cookie를 사용하고, 도메인에 제한적인 서비스 제공으로 다음과 같은 비교 결과가 도출 되었다.

〈표 3〉 SSO솔루션 비교

| 특징 | JetSpeed2 | LiveCycle ES | RIA기반 SAML SSO |
|---|---|---|---|
| 플랫폼독립적 | N (Java) | N (Java) | Y (Any) |
| 디렉토리와 약결합 | N (Domain) | N (Domain) | Y (Cross Domain) |
| 최종사용자를 위한 온라인 경험의 개선 | N (Domain) | N (Domain) | Y (Cross Domain) |
| 서비스 제공자의 관리 비용 감소 | N (Domain) | N (Domain) | Y (Domain) |
| 전송리스크 감소 | Y (HTTP Cookie) | Y (HTTP Cookie) | Y (HTTP XML) |

## 5. 결 론

RIA는 AJAX 및 동적 UI 생성 기술을 이용하여 기존웹 환경에서 제공하지 못했던 경험을 사용자에게 제공한다. 브라우저에서 많은 기능을 처리하는 RIA는 여러 도메인에 걸친 고수준의 서비스 제공이 가능하고 이때 SSO은 필수적이다.

환경의 변화없이 도메인간의 SSO을 지원하는 SAML은 RIA와 결합하였을 때 높은 시너지 효과를 발휘할 수 있지만, 지금까지의 연구는 서버의 사용자 인증정보 교환에만 초점이 맞추어져 있어, 기존 웹 환경과 다른 서버 통신 메커니즘을 이용하는 RIA환경에 적용하기에는 별도의 연구가 필요하다.

본 연구는 SAML SSO를 RIA환경에 적용하기 위하여 아키텍처를 제안하고, 그 실행구조를 설명하였다. 그리고 제안한 아키텍처를 실제 구현하여 DWR에 적용시켜 그 활용성을 확인하였다.

제안한 아키텍처 내부의 모듈은 IDP와 SP 그리고 RIA에서 SAML SSO 인증을 처리하기 위한 RIA 플러그인으로 구성되고, 이 세 모듈은 SAML 프로토콜을 확장한 통신 메커니즘을 통해 상호 연동한다. 제안한 아키텍처는 기존 HTTP에서 제공하는 기능을 이용하여 시스템의 변화 없이 적용 가능하다. 또한 RIA의 특성을 고려하여 화면의 전환 없이 진행중이던 사용자의 상태를 보존하여 인증 이후 지속적인 서비스를 제공할 수 있다.

현재 RIA에서 서버 통신에 많이 사용되는 DWR에 제안한 SAML SSO 구현물과 서버 기반의 SSO 솔루션을 적용하여 비교한 결과, 제안한 아키텍처는 RIA의 장점을 유지할 수 있는 반면 서버 기반 SSO 솔루션은 서버와 통신 부분에서 RIA의 특징을 저해하는 요소를 발생시켰다. 또한 SAML을 적용함으로써 SAML이 가지는 다른 SSO 기술과의 차별성을 도모할 수 있었다.

SAML은 6가지의 통신 프로토콜을 가진다[2]. 본 연구는 인증처리와 해제에 대한 프로토콜만 중점이 맞추어져 있어 향후 나머지 프로토콜의 RIA환경에 적용에 대한 연구가 보완되어야 한다.

## 참 고 문 헌


[1] 남상온,Rolyn C Daguil,'Ajax를 기반으로 한 인증 및 세션 관리', 인터넷정보학회논문지 제7권 제6호, pp.157-174, 2006. 12.

[2] Scott Cantor, "Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS, 2005.

[3] Rod Widdowson, "Identity Provider Discovery Service Protocol and Profile", OASIS, 2008.

[4] Nate Klingenstein, "SAML V2.0 Holder-of-Key Web Browser SSO Profile", OASIS, 2008.

[5] Kyu Il Kim; Hae Kyung Lee; Ung Mo Kim, "Secure Delegation Model based on SAML in Ubiquitous Environments", Information Security and Assurance, 2008. ISA 2008. International Conference on, Vol.24126, pp.117-121, 2008.

[6] 박차라, 임태수, "RIA 기반 개인화 검색을 위한 Widget 응용의 구현", 한국정보과학회 컴퓨팅의 실제 및 레터 제13권 제6호, 2007.

[7] J. Musser, T, O'Relly, O'Relly Radar Team, Web 2.0 Report, O'Relly Media, 2006.

[8] Direct Web Remoting, http://directwebremoting.org

[9] Opensaml, http://www.opensaml.org

[10] Lawton, G., "New Ways to Build Rich Internet Applications", Computer, Vol.41, Issue 8, pp10-12, 2008.

[11] Perez, S.; Diaz, O.; Melia, S.; Gomez, J., "Facing Interaction-Rich RIAs: The Orchestration Model", Web Engineering, 2008. ICWE "08. Eighth, Vol.14-18, pp.24-37, 2008.

[12] Heidenbluth, Norbert; Schweiggert, Franz, "Status Sensitive Components: Adapting Rich Internet Applications to Their Runtime Context", Digital Society, Vol.1-7, pp133-138, 2009.

[13] Gross, T., "Security analysis of the SAML single sign-on browser/artifact profile", Computer Security Applications Conference, Vol.2003, pp.298-307, 2003.

[14] IBM, "How Rich Internet Applications (RIAs) can help business", http://www-01.ibm.com/software/info/web20/mashups-rias/ria.html

[15] adobe, "Programming with LiveCycle ES", http://livedocs.adobe.com/livecycle/8.2/programLC/programmer/help/wwhelp/wwhimpl/common/html/wwhelp.htm

[16] Adobe® LiveCycle® ES, "LiveCycle® ES Services", 2008.

[17] Paul Madsen, NTT, SAML V2.0 Executive Overview Committee Draft 01,OASIS, 2005.


## 조 동 일

e-mail : chodongil@yahoo.co.kr

2003년 수원대학교 기계공학과(학사)

2008년 숭실대학교 정보과학대학원 (공학석사)

2008년~현 재 숭실대학교 컴퓨터공학과 박사과정

2003년~현 재 (주)토마토시스템 기술연구소 선임연구원

관심분야 : BPM, EA, 시멘틱웹



## 류 성 열

e-mail : syrhew@ssu.ac.kr

1976년 숭실대학교 전자계산학과(학사)

1980년 연세대학교 산업대학원 전자계산학과(공학석사)

1997년 아주대학교 컴퓨터공학과(공학박사)

1981년~현 재 숭실대학교 컴퓨터학부 교수

1982년~1995년 숭실대학교 전자계산연구소 및 중앙전자계산소 소장

1997년~1998년 George Mason University 객원 교수

1998년~2001년 숭실대학교 정보과학대학원 원장

2004년~현 재 한국품질재단 운영위원회 위원장

2006년~현 재 공정거래위원회 성과관리위원회위원

2008년~현 재 정보통신연구진흥원 비상임 이사

관심분야 : 소프트웨어 유지보수, 소프트웨어공학 프로세스, 오픈 소스 소프트웨어