7

웹 2.0과 Ajax 보안 취약점

목포대학교 ▌김미선
동신대학교 ▌김진보
전남대학교 ▌양형초
전남대학교 ▌김용민
목포대학교 ▌서재현

1. 서 론

웹 2.0은 사용자들에 의해 콘텐츠가 제작되고 웹을 통해 참여하고 공유하는 개방성을 특징으로 하고 있다. 초창기 웹은 중앙 집중적인 정보 전달 기능을 가지고 있었으나, 웹 2.0은 사용자들에 의해 발생하는 콘텐츠를 통하여 정보를 획득하고 전달하는 기능으로 변화되었으며, Ajax(Asynchronous JavaScript and XML)와 RSS(Really Simple Syndication) 등과 같은 사용자 중심의 기술이 발전하게 되었다.

이러한 웹 어플리케이션들은 자바스크립트와 XML로 이루어져 있으며 RIA(Rich Internet Application) 환경을 제공하고 있다. 하지만, 웹 2.0 기반의 대화형 웹페이지를 구현하기 위해 보안상 취약점이 많은 자바스크립트의 사용량이 급증한 것이 가장 큰 문제점으로 지적할 수 있다. 자바스크립트가 기반이 되어 사용자의 RIA 환경을 제공함에 있어서 자바스크립트의 인코딩은 상당히 많은 방법들을 가지고 있다. 이러한 이유로 사용자가 직접적으로 웹 사이트에 올린 수많은 코드 중 크로스사이트 스크립팅이나 데이터 조작 같은 악성코드를 발견하기란 쉽지 않다. 이것은 기존 웹 환경에 존재하고 있는 보안 취약점들도 완전히 방어하지 못하고 있는 상태에서 웹 2.0 환경이 도래하고 있는 것이라 할 수 있다. 또한 기존의 웹이 가지고 있는 보안 취약점들을 그대로 포함하면서 웹 2.0 환경에서 발생할 수 있는 새로운 취약점들이 추가되어 보안상 상당한 문제점을 야기 할 수 있다.

본 연구에서는 웹 2.0의 국내·외 동향과 웹 2.0에 따른 해킹 기술의 변화 과정을 살펴보았으며, 웹 2.0의 대표 기술인 Ajax에서 발생할 수 있는 보안 취약점에 대해 연구하였다.

2. 관련연구

2.1 웹 2.0

웹 2.0이라는 용어는 오라일리와 미디어라이브 인터내셔널의 컨퍼런스 브레인스토밍 세션에서 "웹에 일종의 전환점을 찍은 닷컴 붕괴와 새 시대의 등장을 어떻게 표현할 수 없을까?"라는 논의를 시작하였고, 이를 웹 2.0이라 부르기로 하였다. 웹 2.0을 통해 살펴볼 수 있는 큰 변화들은 다음과 같다[1,2].

첫 번째, 콘텐츠 유통과 상거래 방식의 변화를 의미한다. 블로그 등을 통한 사용자 중심의 콘텐츠 생산방식과 RSS를 통한 콘텐츠의 소비 방식이라는 변화가 생겼고, 오픈 API와 웹 서비스, 그리고 매쉬업 등을 통해 새로운 서비스 개발에 소비자가 참여하여 만들 수 있는 환경으로 변화되고 있다. 두 번째, 브라우징 방식의 변화이다. 기존과 같이 HTML에 기반으로 단순하게 브라우징하는 형태에서 탈피하여, 다양한 사람들과 정보들 사이의 관계를 이용하는 소시얼(social) 브라우징이나 태깅/폭소노미(folksonomy)를 이용하는 네비게이션 방식과 같은 변화들이 나타나고 있다. 세 번째, 웹 응용 환경의 변화로 웹 응용이 단순한 HTML 기반의 브라우징이 아니라, 웹 서비스와 개방형 API에 기반한 하나의 복합 응용의 형태로 사용자들에게 다가가고 있으며, 그러한 RIA와 Ajax 등의 클라이언트 확장 기술을 통해 웹 응용의 범위를 넓히고 있다. 네 번째, 서비스 제공방식의 변화이다. SOA(Service-Oriented Architecture) 및 SaaS(Software as a Service)와 같은 소프트웨어 패러다임과도 연관을 맺고 있으며, 웹 서비스와 매쉬업 등을 통해 서로 다른 서비스를 융합하여 새로운 서비스를 손쉽게 만드는 환경과 SOW(Statement of Work)와 같은 서비스 기반의 환경으로 변화하고 있다.

이처럼 웹 2.0은 "참여", "공유", "개방"이라는 3개의 키워드로 대표될 수 있으며, 웹 1.0 서비스와 비교하여 보다 많은 사용자들의 참여를 적극적으로 유도

표 1 웹 1.0과 웹 2.0의 콘텐츠 환경 비교

구 분	웹 1.0	웹 2.0
제작 및 유통 방식	소수의 대형 제작사에 의해 콘텐츠가 제작되고 포털 사이트 등을 통해 다수에게 유통	다수의 사용자가 손쉽게 콘텐츠를 생산(User Created Content)하고 다수에게 유통 및 공유
기반 지식	콘텐츠 공유를 위해 전문적인 지식이 필요함	블로그, 미니홈피 등을 통해 손쉽게 콘텐츠를 공유하며, RSS 등으로 쉽게 유통됨
제공 환경	HTML의 한계로 웹 브라우저에서 제공되는 범위에서 표현됨	Ajax, Atlas 등등의 기술을 통해 웹 자체가 하나의 플랫폼화 되어감

하는 서비스 모델로 확장성과 개방성으로 나누어 생각할 수 있다. 표 1은 웹 2.0이 사용자 중심의 환경으로 발전함에 따라 웹 1.0과 웹 2.0에서의 콘텐츠 환경의 변화에 대해 비교한 것이다[3].

2.2 Ajax

Ajax는 'Asynchronous JavaScript + XML'의 줄임말로 '비동기 자바스크립트 XML'이다. Ajax는 자바스크립트 렌더링 엔진을 이용한 기술로, Ajax를 이용할 경우 플래시나 액티브엑스(ActiveX) 의존도를 많이 벗어날 수 있다. 대표적으로 구글과 야후, 아마존 등의 여러 서비스에서 Ajax 기술을 활용하고 있다. Ajax라는 낱말은 Garrett이 2005년 'A New Approach to Web Applications'이라는 에세이에서 'Ajax'라는 낱말로 이 기술을 소개한 이후 널리 알려진 것으로 알려졌다[4].

Ajax는 웹 프로그래밍의 한 종류로 하나의 기술이 아니라 여러 가지 기술이 복합된 방법론 또는 기술 덩어리를 뜻한다. Ajax에 사용된 기술을 보면 XHTML과 CSS를 사용한 표준 설계에 동적 표시, DOM을 사용한 상호작용, XML과 XSLT를 이용한 자료 교환과 제어, XmlHttpRequest를 이용한 비동기 자료 검색과 모든 것을 결합하여 단계별 수행하도록 하는 자바스크립트 등이 고루 섞여 있다. 따라서, '브라우저와 서버 사이의 전송에는 XML을 사용하고, 사용자의 브라우저 화면의 인터페이스로는 자바스크립트를 이용하는 기술'로 개념을 설명할 수 있다. 기술적으로 보자면 '웹서버-브라우저'의 구조 사이에 Ajax가 중간에 위치한 '웹서버-Ajax엔진-브라우저'의 구조로 변경되었음을 알 수 있다. 그림 1은 웹 어플리케이션의 전통적인 모델과 Ajax의 어플리케이션의 모델의 차이점을 표현 한 것이다[5,6].

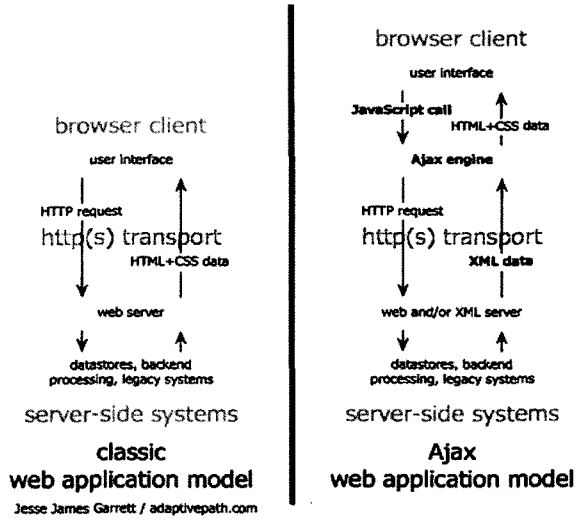


그림 1 웹 어플리케이션 전통 모델과 Ajax 모델 비교

Ajax가 현재 웹 서비스 기술로 각광받는 이유로는 ActiveX 기술의 대안으로써 대부분의 웹 브라우저에서 기본적으로 지원하며, 특정 모듈의 설치 과정이 필요 없다. 또한 웹상의 RIA 인터페이스 구현으로 웹 페이지의 어플리케이션화, 데스크톱 어플리케이션 정도의 인터페이스를 구현할 수가 있다. Ajax는 비동기식 통신을 사용하여 웹 사용자로 하여금 요청 서비스에 대한 응답 속도 향상, 대기시간 불필요, 트래픽 감소, 비용절감의 이점을 가지고 있으며, 표 2는 Ajax의 장점과 단점을 보여준다[7,8].

2.3 OWASP Top 10 2007

OWASP(Open Web Application Security Project)는 웹 애플리케이션에서 가장 심각하다고 생각되는 10가지 보안 취약점에 대하여 명세하였다. 2004년도에 최초 버전이 발표되었고, 3년 만에 새로운 버전인 'OWASP

표 2 Ajax의 장점과 단점

장 점	단 점
· 페이지 이동 없이 고속으로 화면 전환 가능 · 서버 처리를 기다리지 않고, 비동기 요청이 가능 · 수신하는 데이터량을 줄일 수 있고, 클라이언트에게 처리를 위임할 수 있음	· Ajax를 쓸 수 없는 브라우저 · Http 클라이언트의 기능이 한정 · 페이지 이동 없는 통신으로 인한 보안상의 문제 · 지원하는 Charset이 한정 · 디버깅이 용이하지 않음 · 요청을 남발하면 역으로 서버 부하가 늘 수 있음

Top 10 2007'을 발표하였다. OWASP 10대 보안 취약점 2007년 개정판은 2006년 MITRE 취약점[9]을 기준으로 작성되었으며, OWASP는 가장 공통적인 웹 어플리케이션 보안 취약점에 대해 개발자, 설계자, 아키텍트 등 관련된 모든 사용자에 대하여 보안 교육을 하고자 하는 목적과 보안상 안전한 웹 프로그래밍을 하도록 안내하기 위함이라 할 수 있다. 이것은 최근 웹 환경의 변화와 함께 웹 2.0 시대에 맞추어 웹 보안에 대해 가이드라인을 제시한 것이라 할 수 있다[10].

OWASP 10대 웹 취약점은 2004년 발표 이래 웹 보안 제품이나 웹 애플리케이션의 기능 요구사항 표준으로써 중요한 지표로 활용되고 있다. 실제로 국내·외 주요 보안 기업들이나 포털 사이트들에서 OWASP 10대 취약점을 활용하여 대응방안을 개발하고 있으며, 웹의 보안을 점검하는 주요 척도로 이용되고 있다.

표 3은 OWASP Top 10 2007의 취약점 항목과 그 내용을 개략적으로 보인 것이다[10].

3. 웹 공격 기법의 변화

1990년대 후반부터 2000년 초반까지 웹 서비스의 폭발적인 증가로 인해 대부분의 기업들이 웹을 이용하여 기업을 홍보하고 고객에게 정보를 제공하기 시작했다. 당시만 해도 보안에 대한 일반인들의 관심이 없었기 때문에 웹서비스에 대한 보안 요소를 전혀 고려하지 않았고 아주 간단한 공격만으로도 손쉽게 시스템을 공격할 수 있었다. 그러나 최근에 대기업부터 중소기업까지 방화벽과 침입탐지시스템(IDS)을 도입하면서 실제로 외부에서 원격으로 시스템의 취약한 데몬을 이용한 공격이 상당히 어려워졌으며 시스템 자체에 대한 공격도 점차 줄어들었다. 방화벽으로 인해 웹서비스가 이루어지는 80번 포트를 제외하고 다른 포트들은 서비스를 제한하는 방법 등이다. 이렇게 웹 서비스 이용의 증가와 시스템에 대한 보안 솔루션의 도입으로 인해 해커들은 대상 시스템을 침투하기 위해 80번 웹서비스 포트를 집중적으로 공략하게 되었다.

표 4는 세대별 공격 기법의 변화를 나타내고 있다. 1세대의 공격 기법은 일반적으로 OS 자체에 대한 취약점의 공격으로 내부공격(Local Attack)이 주로 이루어졌으며, 이후 2세대에는 취약한 데몬(daemon)을 이용한 공격인 외부공격(Remote Attack)이 주로 수행되었다. 이와 같은 시기에 3세대 해킹 기법으로는 TCP/IP 프로토콜의 취약점을 이용한 다양한 공격이 이루어졌고, 4세대 해킹이라고 볼 수 있는 80번 포트를 이용한 웹 애플리케이션, 웹서비스에 대한 공격이 새롭게 나타났다.

표 3 OWASP Top 10 2007

취약점	요 약
크로스 사이트 스크립팅(XSS)	· 콘텐츠를 암호화나 검증하는 절차 없이 사용자가 제공하는 데이터를 애플리케이션에서 받아들이거나, 웹브라우저로 보낼 때마다 발생 · 공격자가 희생자의 브라우저 내에서 스크립트를 실행하게 허용함으로써 사용자 세션을 가로채거나, 웹사이트를 손상하거나 웜을 심는 것 등 가능
인젝션 취약점	· 인젝션은 사용자가 입력한 데이터가 명령어나 질의문의 일부분으로 인터프리터에 보내질 때 발생 · 공격자가 삽입한 데이터에 대해 인터프리터는 의도하지 않은 명령어 실행 및 데이터 변경 가능
악성 파일 실행	· 공격자가 악의적인 코드와 데이터의 첨부를 허용함으로써 전체 서버 훼손과 같은 공격 가능 · 악성 파일 실행 공격은 PHP, XML, 그리고 사용자로부터 파일명이나 파일을 받아들이는 프레임워크에 영향을 줌
불안전한 직접 객체 참조	· 직접 객체 참조는 개발자가 파일, 디렉토리, 데이터베이스기록 혹은 키 같은 내부구현객체에 대한 참조를 URL 혹은 폼 매개변수로 노출시킬 때 발생 · 공격자는 이러한 참조를 조작해서 승인없이 다른 객체에 접속 가능
크로스 사이트 요청 변조(CSRF)	· 로그온 한 희생자의 브라우저가 사전 승인된 요청을 취약한 웹 애플리케이션에 보내도록 함으로써 희생자의 브라우저가 공격자에게 이득이 되는 악의적인 행동을 수행하도록 함
정보 유출 및 부적절한 오류 처리	· 애플리케이션은 다양한 애플리케이션 문제점을 통해 의도하지 않게 자신의 구성 정보, 내부 작업에 대한 정보를 누출하거나 또는 개인정보 보호 위반 가능
취약한 인증 및 세션 관리	· 자격증명과 세션토큰은 종종 적절히 보호되지 않는 경우, 공격자는 다른 사용자인 것처럼 보이게 하기 위하여 비밀번호, 키, 혹은 인증 토큰을 손상시킴
불안전한 암호화 저장	· 웹 애플리케이션은 데이터와 자격증명을 적절히 보호하기 위한 암호화기능을 사용 않음
불안전한 통신	· 애플리케이션은 민감한 통신을 보호할 필요가 있을 때 네트워크 트래픽 암호화 시 종종 실패함
URL 접속 제한 실패	· 애플리케이션이 권한 없는 사용자에게 연결주소나 URL이 표시되지 않도록 함으로써, 민감한 기능들을 보호함 · 공격자는 이러한 약점을 이용하여 이 URL에 직접 접속함으로써 승인되지 않은 동작 수행가능

표 4 세대별 공격 기법

1세대 공격 기법	Operation System Hacking : Local Attack OS 자체의 취약점 공격
2세대 공격 기법	Operation System Hacking : Remote Attack OS에서 실행되는 취약한 Daemon 프로그램 공격
3세대 공격 기법	NetWork System Hacking : Network Attack TCP/IP 모델의 특성을 이용한 공격
4세대 공격 기법	Application Hacking : Web Application Attack Firewall 우회 가능한 HTTP Port를 이용한 공격

그림 2는 해킹 형태의 변화로 기존의 공격대상이 웹 환경으로 변화하는 것을 나타내고 있으며 전문지식이 없어도 공격 툴에 의존하여 웹을 공격하고 있다는 것을 반영하고 있다. 웹 2.0을 표방하는 대부분의 사이트가 보안 취약점에 대하여 허술하게 대응하고 있어 고객정보가 쉽게 누출되는 등 치명적인 보안상 결함을 가지고 있다.

4. Ajax의 보안 취약점과 대응 방안

4.1 Ajax의 보안 취약점

Ajax에서 사용되는 XMLHttpRequest는 HTTP 방식과 동일하게 GET, POST 방식으로 요청을 보내고 문자열이나 XML 형태로 응답을 받기 때문에 HTTP 트래픽을 훔쳐봄으로써 정보가 노출될 수 있고, 이런 문제로 인하여 Ajax가 보안에 취약하다고 할 수 있다. 다음은 Ajax의 주요 취약점들에 대한 것이다.

4.1.1 크로스 사이트 스크립팅(XSS)

크로스 사이트 스크립팅은 해커들이 웹 애플리케이션에 몰래 침투하기 위해 이용하는 가장 일반적인 애플리케이션 계층 공격 중에 하나이다. 크로스 사이트 스크립팅은 자바스크립트를 통해 가능한 공격이므로, 자바스크립트로 구성된 Ajax에게 치명적인 공격이 될 수 있다. Ajax가 비동기 방식으로 웹 페이지를 불러오는 과정에서 악의적인 스크립트가 삽입된다면, Ajax의 특성상 사용자는 공격당하고 있다는 사실을 모른 채 공격 코드가 사용자의 시스템에서 백그라운드로 실행된다. 또한 Ajax는 한번 신뢰된 사이트는 지속적으로 신뢰를 함으로 방화벽 등을 우회할 수가 있다는 것이 가장 큰 문제점이라 할 수 있다.

만약 악의적인 사용자가 악성 스크립트를 삽입할 수 있다면, 클라이언트의 쿠키 및 패스워드, 개인정보 등을 훔칠 수 있을 뿐 아니라, 크로스 사이트 스크립팅을 이용해 취할 수 있는 수많은 공격들을 행할 수 있다.

4.1.2 자바스크립트 소스코드 누출

Ajax는 많은 부분이 자바스크립트로 구성되어 있다. 하지만, 자바스크립트 코드는 HTML 안에 포함되어 소스보기를 통해서 누구나 볼 수 있으며, 이는 클라이언트의 동작방식이나 서버와 클라이언트 사이의 동작방식에 대해서 누구나 알 수 있다는 의미를 가진다. 특히, 클라이언트 측에서 코드를 숨길 수 없다는 것은 공격자가 클라이언트의 코드를 수정하거나 유사한 동작방식으로 Ajax 애플리케이션을 만들어 악용할 수 있는 등의 위협 상황을 포함하고 있다. 물론 소스 코드를 알아보기 어렵게 해주는 엔코더(encoder) 등을 이용할 수도 있지만, 키 값에 대한 관리 문제가 있으며, 소스 코드를 완전히 숨기는 작업은 어렵다고

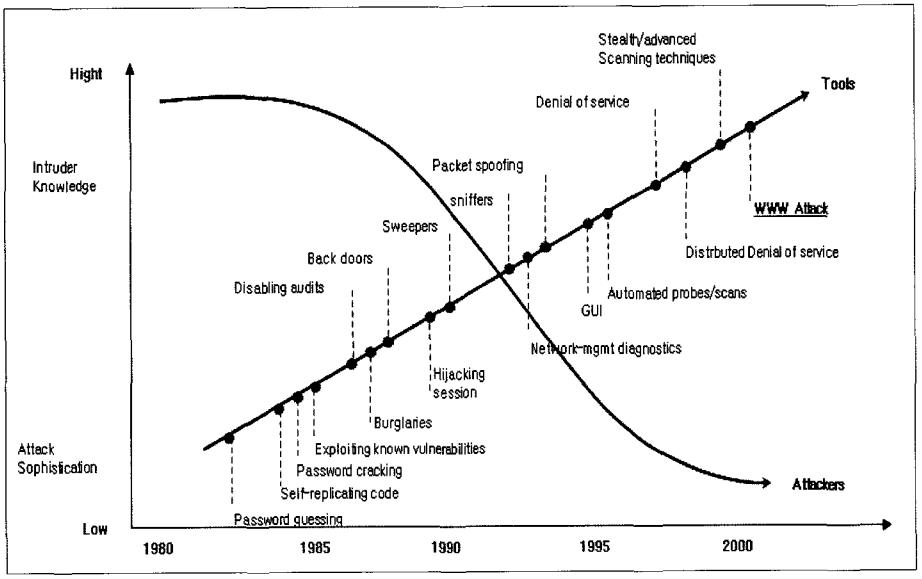


그림 2 해킹 형태의 변화

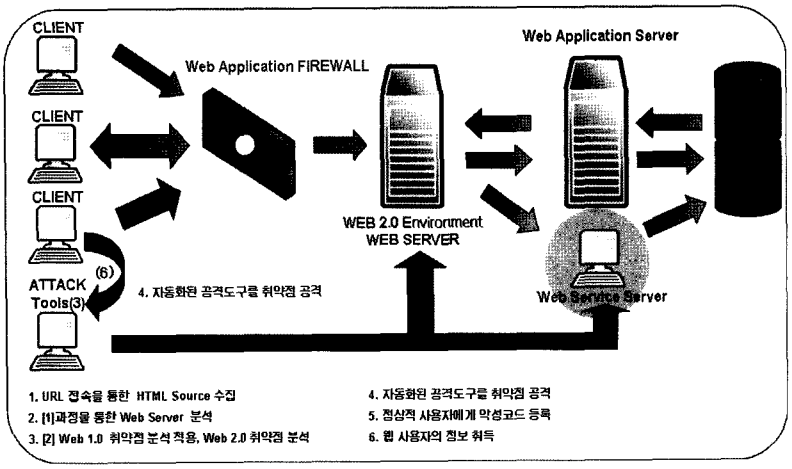


그림 3 Ajax의 취약점 공격 시나리오

할 수 있다.

이와 같이, Ajax와 같이 자바스크립트로 구성된 클라이언트 애플리케이션에 포함되어 있는 민감한 정보나 동작 방식이 포함되어 있을 경우 그 정보들은 위험에 노출되어 있다고 볼 수 있다.

그림 3은 Ajax의 자바스크립트의 소스코드 누출에 대한 취약점을 이용하여 이루어지는 형태이다. Ajax를 통한 공격의 순서는 URL 접속을 통한 HTML, JavaScript 소스 코드 수집 및 분석, 자동화된 공격도구를 이용한 취약점 공격, 정상적인 사용자에게 악성코드 등록, 웹 사용자의 정보 및 웹서비스 제공 콘텐츠의 불법적 획득이라는 과정에 의해서 이루어진다.

4.1.3 Client-Side 보안

일반적인 웹 환경에서는 Sever-side에서 여러 작업을 진행하고 처리하지만, Ajax 기반의 서버에서는 필요한 데이터만을 받아오고, 데이터 처리는 클라이언트의 Ajax 엔진이 담당하면서 서버 보다는 클라이언트의 비중이 더 증가되었다. 전달해야 하는 데이터의 양이 줄어들어 서버의 처리 부담이 감소한다는 것은 분명히 장점이지만, 클라이언트의 비중이 증가한다는 것은 서버에서의 보안 뿐 만 아니라 클라이언트에 대한 보안에도 많은 노력을 기울여야 한다는 것을 의미한다[11].

실제로 서버의 보안이 철저하다고 하여도 클라이언트의 취약한 부분을 통해서 서버의 보안 정책을 우회하여 공격을 행할 수 있으며, 이는 곧 서버와 통신하는 다른 클라이언트에게도 영향을 미칠 수 있다.

4.1.4 웹 플랫폼에서의 취약점

Ajax는 일반 웹 애플리케이션이 가지고 있는 보안 취약점들을 가질 수 있다. Ajax 역시 웹 애플리케이션이고 웹 플랫폼을 기반으로 하기 때문에 이러한 취약점들에 대해 자유로울 수가 없다. 또한, 추가적으로 Ajax의 특징인 XML을 이용한 데이터 교환, XMLHttpRequest를 이용한 비동기 통신, 자바스크립트를 이용한 클라이언트 애플리케이션들과 합쳐져서 새로운 보안 문제들이 나타날 수 있다.

4.2 Ajax 취약점 대응 방안

Ajax의 보안은 기존 웹과 유사하다고 할 수 있다. 기본적으로 자바스크립트에서와 같이 동일 도메인으로만 요청이 가능하도록 제한이 되어 있고, 쿠키를 사용한 인증도 가능하다. 하지만 이런 방식이 데이터 자체의 안정성을 보장하지는 못하기 때문에 다음과 같은 기법이 사용될 수 있다.

첫 번째, 자바스크립트의 특수문자를 처리하는 것이다. 크로스 사이트 스크립팅은 자바스크립트에 악의적인 코드를 삽입하는 것으로써, 악성 스크립트 실행 시 필요한 '<', '>' 등과 같은 문자들을 다른 문자로 교체한다던지 원천 봉쇄하는 방법을 통해 악의적인 자바스크립트로 인한 공격들에 대해 방어를 할 수 있다.

두 번째, 토큰(token)의 사용이다. UUID(Universally Unique IDentifier)와 같은 토큰을 사용함으로써 유효한 사용자의 요청에만 응답할 수 있다. 세션에 키를 저장한 후 동일서버로 일정한 시간 안에 XMLHttpRequest 요청이 들어오는 경우만 응답을 하는 방식으로 유효성을 체크할 수 있다.

세 번째, 문자열의 암호화이다. XMLHttpReqeust는 문자열로 응답을 주기 때문에 문자열 자체를 암호화해서 데이터를 전송할 수 있다. 하지만 암호화된 내용을 자바스크립트로 기술하는 과정에서 키나 암호화 알고리즘이 노출될 수 있는 위험이 있다. 이 문제를 해결하기 위해서는 Ajax 관련 js 파일을 매번 서버에서 새로 생성하여 줌으로써 해킹의 위험성을 줄이는

방법이 있다.

위의 방법을 통해 어느 정도의 데이터를 보호할 수는 있지만 완벽한 보안이란 존재하지 않는다. 특히 Ajax로 데이터를 주고받는 경우에 사용자 인증 과정을 꼭 거쳐야 하고, 민감한 내용은 가급적 보안이 되는 통신을 사용하는 것이 좋은 방법으로 고려된다.

4. 결 론

웹 2.0의 도입에 따른 웹 콘텐츠 활용의 특징과 보안 제품이나 웹 애플리케이션 개발 시 표준 지표로써 활용도가 매우 높은 OWASP Top 10 2007 웹 보안 취약점과 웹 2.0 콘텐츠 응용 처리기술인 Ajax에 대해 소개하였다. 그리고 기존의 웹에 대한 보안 방어 대책도 충분히 확보하지 못하고 있는 상황에서 웹 2.0 도입은 보안상 상당한 문제점을 야기 할 수 있으며, 실제로 최근의 웹 공격 기술들이 웹 2.0을 이용하는 특성을 나타내고 있음을 보이며, 웹 2.0의 대표 기술인 Ajax에서 발생할 수 있는 보안 취약점과 대응방안을 기술하였다. 향후에는 Ajax 뿐 만 아니라 광범위한 웹 2.0의 주요 기술들에 대해 보안 취약점을 파악한 후 이를 해결하기 위한 보안 기법에 대한 연구가 필요하다.

참고문헌

[1] Tim O'Reilly, "What is Web 2.0", http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html/, 2005.

[2] 전종홍, 이승윤, "웹 2.0 기술 동향 및 전망", 전자통신동향분석, 2006.

[3] 강은성, "2007년 주요 보안 트렌트", 안철수연구소, 2007.

[4] Dave Crane, Eric Pascarello and Darren James, "Ajax 인 액션", 에이콘, 2006.

[5] 김중태 문화원, "웹사이트 접근의 새로운 혁명 Ajax", http://www.dal.co.kr/blog/2005/11/20051109_ajax_1.html, 2005.

[6] Jesse James Garrett, "Ajax : A New Approach to Web Applications", 2005.

[7] 최범균, "Ajax 프로그래밍," 가메출판사, May 2006.

[8] 위키백과, "Ajax," http://ko.wikipedia.org/wiki/Ajax, Jul. 2007.

[9] MITRE, "MITRE Vulnerability Trends for 2006", 2006.

[10] OWASP, "OWASP Top 10 2007", http://www.owasp.org/index.php/Top_10_2007, 2007.

[11] 유성수, 노봉남, "Ajax 기술과 보안", 한국정보과학회2006년 추계학술대회, 2006.

김 미 선

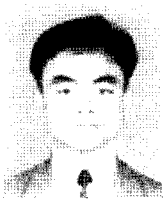
1996 목포대학교 컴퓨터공학과 졸업
2000 목포대학교 컴퓨터공학과 석사
2007 목포대학교 컴퓨터공학과 박사
2007~현재 목포대학교 정보공학부 정보보호전공 초빙교수
관심분야 : 컴퓨터공학, 네트워크보안, 정보보호
E-mail : misun@mokpo.ac.kr

김 진 보

2003 목포대학교 멀티미디어공학전공 졸업
2007 목포대학교 정보보호협동과정 석사
현재 동신대학교 디지털콘텐츠 협동 연구센터 콘텐츠 개발팀
관심분야 : 정보보호, 네트워크보안, 웹서비스 보안, 디지털 콘텐츠 보안
E-mail : progress97@nate.com

양 형 초

2006 목포대학교 이학사
2006~현재 전남대학교 정보보호 협동과정
관심분야 : 웹보안, IPv4/IPv6 보안
E-mail : yang2oon@lsrc.jnu.ac.kr



김 용 민

2002 전남대학교 전산통계학과 이학박사
2004~현재 전남대학교 문화콘텐츠학부
관심분야 : 시스템 및 네트워크보안, 전자상거래보안, Data Assimilation
E-mail : ymkim@chonnam.ac.kr

서 재 현

1985 전남대학교 계산통계학과 졸업
1988 중앙대학교 전자계산학과 석사
1996 전남대학교 전산통계학과 박사
1996~현재 목포대학교 정보공학부 정보보호전공 부교수
관심분야 : 정보보호, 시스템 및 네트워크보안, 컴퓨터 네트워크
E-mail : jhseo@mokpo.ac.kr