情 報 保 護 學 會 誌
第18卷 第3號, 2008. 6

웹 2.0 환경에서의 보안 문제와 효율적 웹 검사 방안

이 창 우*, 김 창 희**, 이 준 호***

요 약

인터넷에 대한 지식 없이도 누구나 직접 참여하고 생성하고 공유할 수 있는 웹 2.0 환경은 새로운 사용자들을 웹으로 끌어 들이게 되었고 이로 인해 웹은 또 다른 보안 위협의 유통 채널로 확고하게 자리 잡게 되었다. 이와 같은 웹의 보안 문제는 기존의 웹 1.0의 보안 솔루션으로는 웹 2.0의 동적인 페이지와 복잡해진 서버 환경에 대응하기에 한계가 존재한다. 하지만 웹 보안에 대한 인식 부족 및 사용자와 웹 서비스 업체의 보안 불감증으로 인하여 웹 보안에 대한 연구와 솔루션이 부족한 실정이다.

본 논문에서는 웹 2.0 환경에서의 보안 문제를 분석하고 보안 문제 해결을 위한 효율적 웹 검사 방안을 제시해본다. 이와 같은 연구로 제시된 내용을 통해 현재 웹 2.0환경에서의 보안 문제에 대한 '백신'이외의 또 하나의 필수 보안 솔루션이 필요하다는 사회적 인지와 인식이 확립되고 웹 브라우저 보안 문제 해결을 위한 서비스가 보다 많이 제공되길 기대해 본다.

Ⅰ. 서 론

텍스트와 이미지로 일방적인 정보 제공형태에 불과했던 인터넷은 이제 커뮤니티와 블로그 등의 활성화에 따라, 사용자들의 참여와 개방으로 발전하고 있으며 국내외에서 Web 2.0의 기술을 적용하는 사례가 늘고 있다. Web 2.0은 일방적으로 정보를 제공받는 웹 1.0과 달리 블로그, 검색 등을 활용해 스스로 정보 및 네트워크를 창조하고 공유하는 플랫폼으로서의 웹으로, 웹 비즈니스의 국내외 주요 개발자들에 의해 실 서비스로 구현되고 있다. 이런 활성화 된 Web 2.0은 기존의 웹이 가지고 있는 취약점 뿐만 아니라, 추후 거론될 Server Side Script, AJAX(Asynchronous Javascript and XML), API 기술 등이 적용됨에 따라 웹에서의 침해 행위는 더욱 증가할 것으로 예상된다.

따라서 본 논문에서는 웹 2.0의 보안 문제점을 분석하여 성능에 영향을 주지 않는 해결 방안을 연구하고자 한다. 본 논문의 2장에서는 웹 2.0의 적용된 기술[1]을 정의한다. 3장과 4장에서는 웹 2.0 환경에서의 보안 문제점과 현재 사용되는 웹 보안 솔루션의 한계를 분석하여 5장에서 효율적인 해결 방안을 제안한다. 마지막으로 6장에서는 결론 및 향후 연구방향을 제시한다.

Ⅱ. 웹 2.0 개념과 적용 기술

2.1 웹 2.0 정의

웹 2.0은 인터넷의 새로운 발전 방향을 함축적으로 표현하는 용어로 기존의 웹과 완전히 다른 2.0 버전이라는 의미이다. 웹 2.0의 개념[2]은 O'Reilly와 MediaLive International의 컨퍼런스 브레인스토밍 세션에서 Dale Dougherty에 의해 시작되었다. 웹 2.0은 처음 그 용어가 사용된 2004년 이후로부터 급속하게 국내외의 주목을 받고 있으며, 닷컴의 버블 붕괴와 함께 새로운 닷컴 기업의 도약에 대하여 설명할 수 있는 용어로 채용되고 있다. 웹 2.0이 단순한 마케팅 용어로 그치게 될지, 새로운 웹 생태계의 변화를 정확하게 짚어낸 용어가 될지는 아직 확실하지 않지만 주요 웹 비즈니스에서 실 서비스로 구현되고 있으며, Tim O'Reilly가 주장하는 웹 2.0의 주요 내용[3]은 다음과 같다.

* 안철수연구소 서비스개발팀 SiteGuard 선임연구원 (comring@ahnlab.com)
** 안철수연구소 서비스플랫폼팀 SiteGuard Project Manager (changhee@ahnlab.com)
*** 안철수연구소 서비스플랫폼팀 SiteGuard 기획파트 (leej@ahnlab.com)

- The Web as Platform
- Harnessing Collective Intelligence
- Data is the Next 'Intel Inside'
- End of the Software Release Cycle
- Lightweight Programming Models
- Software Above the Level of a Single Device
- Rich User Experience

또한 웹 2.0의 주요 내용을 보안 관점으로 해석하면 다음과 같다.

■ 개인화

웹 1.0에서 제공하는 정보는 항상 일방적이었다. 즉 지역과 사용자, 문화 등과 상관없이 Web에서는 같은 정보만을 볼 수 있었다. 하지만 웹 2.0에서는 개인의 기호, 성향, 지역 등에 따라 이를 분석하여 맞춤 정보를 제공할 수 있다. 아마존에서 책을 몇 번 구매하게 되면 같은 분류의 신간 도서를 추천해주는 것을 그 예로 들 수 있다. 결국 웹 2.0으로 인해 개인의 다양한 신상정보와 기호정보, 거래정보 등에 대한 노출과 도용의 위협이 웹 1.0에 비해 상당히 크다고 볼 수 있다.

■ UCC

UCC는 User Created Contents로, 웹 2.0의 의미를 그대로 표현한 개념으로 미국에서는 창작의 개념이 강조된 UGC(User Generated Content)로 표현이 된다. 국내에서는 주로 자체 제작 동영상을 대표하는 용어로 사용되지만, 사용자가 제작한 모든 콘텐츠를 의미한다. 웹 2.0에서는 사용자가 만든 콘텐츠가 서비스 업체의 플랫폼을 통해 다른 사용자에게 쉽게 공유된다. 하지만 콘텐츠의 제작자가 악의적인 목적으로 악성 요소를 쉽게 공유할 수 있다는 점에서 웹을 통한 위험의 노출은 더욱 증가되고 있다.

■ SNS

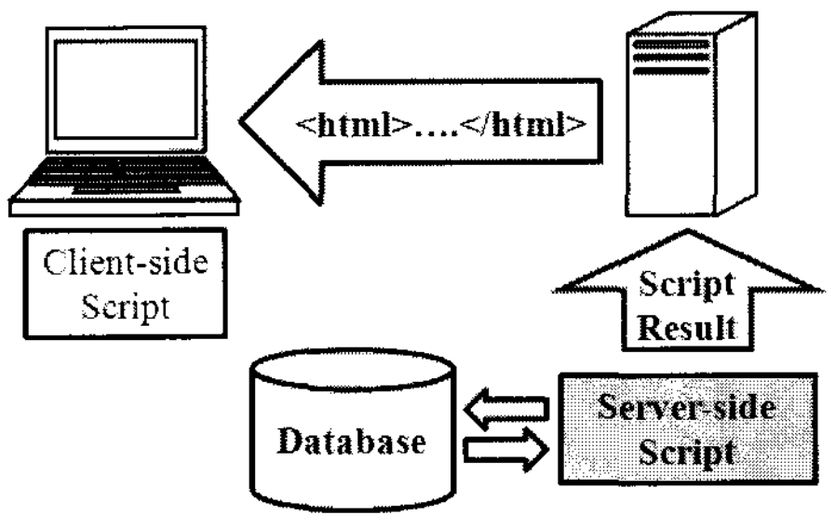
소셜 네트워크 서비스(SNS;Social Network Service) 역시 사용자 참여와 공유, 개방 등의 웹 2.0의 핵심 가치가 서비스에 반영되어 있는 웹 2.0 시대의 Killer Application이다. 소셜 네트워크 서비스란[4] 사회적 관계 개념을 인터넷 공간으로 가져 온 것으로, 사람 간의 인맥구축과 네트워크 형성을 지원하는 서비스로 이와 관련된 다양한 서비스들이 등장하며 확산이 가속화되고 새로운 사회 트렌드로 자리 잡고 있으나 이용자들의 상호신뢰 관계는 과다한 개인정보 노출이나 연쇄적인 해킹을 용이하게 할 수 있다.

2.2 웹 2.0 적용 기술

웹 2.0의 적용 기술(인프라 기술)은 복잡하고 진화하고 있으며, 대표적 기술은 AJAX, Open API, LAMP, XML, RSS, REST 등이 있다. 이 중에서 보안적 위험을 가지는 기술만을 정리해 보면 아래와 같다.

■ Server-Side Script와 데이터베이스 연동

Server-Side 스크립트는 웹 서버가 사용자(웹 브라우저)의 요청에 의해 웹 서버에서 스크립트를 실행하여 동적인 HTML 페이지를 제공하도록 구현한 기술이다. 이는 주로 데이터베이스나 기타 저장된 데이터를 통하여 대화형 웹 사이트(Interactive Web Site)를 제공한다.[5] PHP나 ASP 혹은 JSP가 이에 해당되며, 웹 2.0에서는 양방향 소통을 위한 웹을 구현하기 위해서 필수적인 요소다. 사용자 PC에서 신원을 증명하는 정보를 서버에 보내고, 서버에서는 그 사용자에게 맞는 데이터를 DB에서 찾아 동적인 웹을 구성하고 이를 결과로 전송하거나, 사용자가 만든 데이터를 가공하여 DB에 저장하고 이를 다시 가공하여 다른 사용자에게 제공하게 된다.



〔그림 1〕 Server-side Script

■ AJAX

AJAX(Asynchronous Javascript and XML)는 XHTML, CSS(Cascading Style Sheets), 자바스크립트 등의 기술이 고루 섞여 대화형 웹 애플리케이션을 만들 수 있게 하는 웹 프로그래밍 기술의 복합체로 비동기식 자바스크립트와 XML의 줄임말이다. AJAX의 특징으로는 XML 기반으로 브라우저와 웹 서버간의 데이터량이 줄어들어 응답성이 향상되고 웹 서버의 부담이 줄어들게 된다. 또한 웹에서 서비스를 이용할 때 ActiveX와 같은 별도의 프로그램을 설치하거나 해당 기능을 갖

춘 새 창을 띄울 필요가 없으며, 사용자가 웹 상의 자료의 위치를 직접 편집하는 등의 커스터마이징을 가능하게 해준다. 비동기 통신이 가능하므로 '새로고침'과 같은 사용자의 액션이 없어도 스스로 동적인 웹 페이지의 생성과 갱신이 가능하게 되었다. AJAX로 인해 개인화된 웹을 넘어서 웹 애플리케이션이 만들어지고 있으며 Cloud Computing이 실현되어 가고 있는 것이다.

■ Open API

API(Application Program Interface)의 사전적 의미는 애플리케이션 개발을 위한 함수의 집합으로 특정 소프트웨어나 프로그램의 기능을 다른 프로그램에서도 활용할 수 있도록 표준화된 인터페이스를 공개(Open)하는 것을 의미한다. 웹 서비스들은 자사의 서비스 구성요소를 모듈화 시킨 API를 공개해 이용자가 이를 활용해 다양한 서비스를 스스로 제작할 수 있도록 지원하고 있다. 이러한 Open API가 SNS와 포털을 중심으로 가속화됨에 따라 가까운 미래에는 누구나 자신의 입맛에 맞는 인터넷 서비스를 스스로 만들고 공유할 수 있게 될 것이다. 이와 같은 변화는 웹 2.0과 맞닿아 있으며 Open API는 새로운 서비스를 만들어 낼 수 있는 도구로서 다른 웹 2.0 서비스와 마찬가지로 이용자의 자발적인 참여와 공유가 없으면 정보가치를 창출할 수 없다.

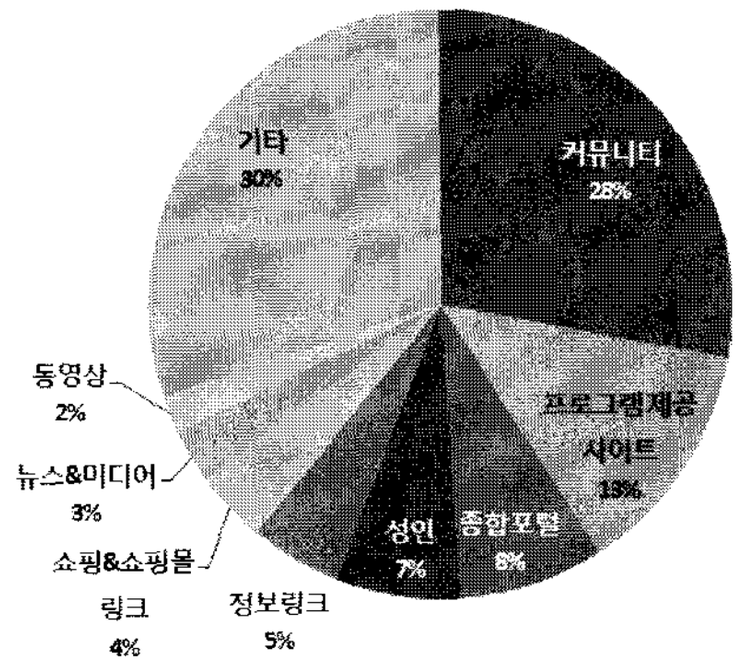
Ⅲ. 웹 2.0의 보안 문제점

웹 2.0은 위에서 기술한 바와 같이 많은 장점을 가져다 주지만 그 확산의 범위와 속도만큼 역기능 또한 증가하고 있다. 이 장에서는 이러한 웹 2.0의 보안적인 문제점들을 나열하고 현재의 웹 보안 솔루션의 한계에 대해 기술 한다.

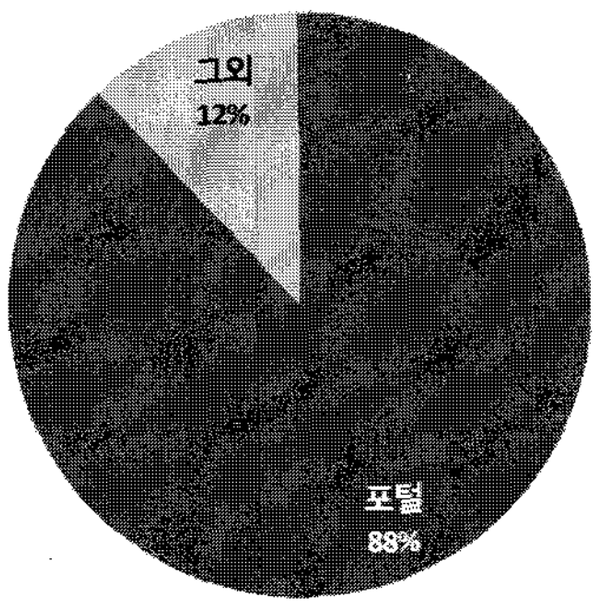
한국정보보호진흥원의 2007 정보보호 실태조사[표 1]에 따르면, 바이러스 감염 경로별 피해 경험에서 프로그램 다운로드가 88.6%, 그 뒤로 특정 웹사이트, 동영상, 메신저를 통한 감염 피해가 70.2%로 집계되었다. 특히 '특정 웹사이트 방문, 동영상, 메신저 이용 등에 의한 감염' 항목은 2007년에 신규로 추가된 항목이다.

〔표 1〕 바이러스 감염 경로별 피해 경험

감염 경로	2005년	2006년	2007년
프로그램 다운로드	87.1%	88.7%	88.6%
특정 웹사이트, 동영상, 메신저	-	-	70.2%
전자우편	45.2%	42.3%	42.4%
내부 네트워크	37.0%	36.0%	37.3%
외부 강제침투	26.1%	24.8%	26.6%
저장매체	21.0%	19.4%	19.8%



〔그림 2〕 악성코드 유포지의 서비스별 분포



〔그림 3〕 서비스 형태별 위험 사이트 경유지 분포

[그림2]는 안철수연구소에서 자체 집계한 자료로 현재 국내 사이트 중 6만 여개의 악성 코드를 유포하는 사이트를 수집하여 서비스별로 분류한 자료이다.그림을 보면, 커뮤니티와 프로그램 제공 사이트, 그리고 포털 사이트를 중심으로 웹이 악성코드의 유포지로 악용되고 있는 것을 알 수 있다.

또한 [그림3]과 같이 위험 사이트의 경유지는 인터넷 사용자의 거의 대부분이 방문하는 포털 사이트가 대부분(약 88%)을 차지하고 있으며 그 중에서도 블로그와 카페와 같은 UCC가 상당 부분을 차지하고 있어, 인터넷 사용자 누구에게나 쉽게 위험 사이트에 노출될 수 있다.

3.1 UCC의 보안 문제

웹 2.0의 대표적 서비스 특징인 UCC는 사용자가 콘텐츠를 생성하여 다른 사용자에게 공유할 수 있는 서비

스 사이트에 공개(업로드)하게 된다. 이렇게 공개된 콘텐츠는 해당 서비스 사이트 뿐만 아니라 타 검색 서비스를 통해서 불특정 다수의 사용자들에게 쉽게 노출된다. 구글과 같은 검색 서비스가 유투브의 동영상이나 블로그의 글, 카페의 게시글 등을 검색해주는 것이 그 예로 볼 수 있다.

여기서 만약 콘텐츠의 제작자가 악의적인 의도를 가지고 있다면 이러한 공유와 검색이 충분히 악용될 수 있는 조건을 갖추고 있다. 한 사례를 보면, 금전적 목적과 타인의 개인정보 수집을 목적으로 자극적이거나 관심을 가질만한 주제의 동영상과 동영상 플레이어를 만들어 공개하는 경우가 있다. 동영상 플레이어의 내부에는 키보드의 입력을 가로채는 키로깅 기능이 탑재되어 있다. HTML로 콘텐츠를 작성하면서 '동영상을 보려면 플레이어를 설치해야 한다'는 내용을 같이 작성한다. 이 콘텐츠를 블로그나 카페 등에 업로드하게 되면 검색 서비스나 포털사이트의 "검색 엔진"을 통하여 이를 쉽게 다른 사용자에게 공유할 수 있다. 이 콘텐츠를 접한 다른 사용자는 동영상을 보려고 플레이어를 설치하지만 키로거와 같은 해킹툴에 의해 자신의 개인정보가 유출된다.

3.2 IFRAME Injection

최근 인터넷 침해 사고 중에 게임 아이템을 탈취하거나, 사용자 계정 정보를 가로채는 사건들이 발생하고 있으며, 실례로 국내 한 연구기관이 IFRAME Injection을 통해 악성코드 유포 및 해킹 경유지로 악용된 사고가 있었다[6].

최근의 사이트는 대부분 사용자가 업로드 하는 콘텐츠에 HTML을 허용하고 있다. HTML을 허용함으로써 사용자가 좀더 다양한 콘텐츠의 제작이 가능하게 되었다. 하지만 이런 허용이 악의적 사용자에게도 그 악용의 기회를 제공하고 있다.

요즘 인기 있는 카페나 블로그를 예로 들면, 카페나 블로그는 대형 사이트를 중심으로 서비스되고 있다. 악의적인 사용자가 카페에 업로드 할 글(text)을 작성하면서 HTML에 IFRAME을 삽입하고 IFRAME의 source는 다른 서버로 연결할 수 있다. IFRAME의 내용을 서비스하는 사이트의 한 페이지처럼 위장하고 그럴듯한 이유를 제시하며 사용자에게 계정 정보의 입력을 유도하게 된다. 이를 접한 사용자는 이러한 위장 사실을 모른 채 자신의 개인 정보를 입력하게 되고, 이 정보는 고스란히 악의적인 의도를 가지고 있는 작성자의 서버에 전송된다.

또한 IFRAME injection의 다른 위험은 웹 브라우저의 취약점을 이용하는 것이다.

MS04-013 MHTML URL 처리 취약점

```
ms-_its:_mhtml:_file://C:\nosuchfile.mht!_
http://www.example.com//exploit._chm::exploit.html
```

악성코드 실제 이용 사례

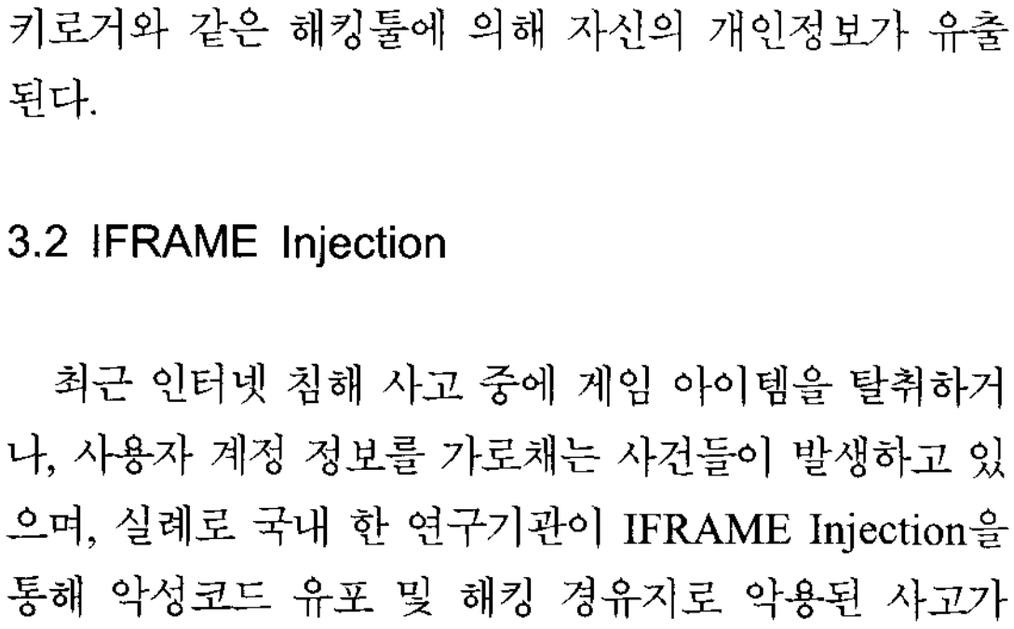
```
document.write('<object
  date= &#109&#115&#45&#105&#116&#115&#58&#109&#1
  04&#116&#109&#108&#58&#102&#105&#108&#101&#58&
  #47&#47&#67&#58&#92&#102&#111;o.mht!'+'http://game
  xxxxx.biz//adv//070//targ.ch'+'m::/targ'+'et.htm
  type= text/x-scriptlet></ob'+'ject>');
```

〔그림 5〕 Internet Explorer 취약점(MS04-013) 사례

웹 브라우저는 사용자가 요청한 URL을 파싱하고 렌더링(rendering)하여 그에 상응하는 콘텐츠를 사용자에게 보여준다. 하지만 이런 과정에는 항상 취약점이 존재한다. 웹 브라우저의 취약점을 이용하는 exploit code와 exploit후에 실행될 코드를 포함한 콘텐츠를 웹 서비스 사이트에 업로드 하게 되며, 이런 사실은 웹 서비스 관리자가 바로 알기 어렵다. 다른 사용자가 이 콘텐츠의 URL로 접속할 때 만약 이런 웹 브라우저의 취약점이 패치되어 있지 않다면 악성코드가 실행된다.

3.3 Cross-Site Script

Server-side Script와는 달리 Client-side Script(Javascript, VBScript 등)는 Client에서 웹 브라우저에 의해 해석되고 실행되어 웹 페이지의 동적인 표현을 가능하게 한다. 또한 DOM(Docu- ment Object Model)을 조작하거나 쿠키(cookie)를 제어할 수 있다. 따라서 이런



〔그림 4〕 OO연구기관 악성코드 유포 및 해킹 경유지 사고

스크립트는 제한적으로 해당 도메인에서만 가능하다면 문제가 없겠지만, 만약 다른 도메인에서 실행된다면 보안 상 위험 요소가 될 수 있다.

예를 들어, A라는 웹 서버로 서비스가 운영되고 있다고 가정하자. 해킹 혹은 위에서 거론한 사용자의 의도에 의해서 HTML에 스크립트가 삽입되었다. 삽입된 스크립트는 실제로 B서버에 존재하는 스크립트 파일이다. 이 B서버의 스크립트에서 쿠키 정보를 읽어서 B서버로 전송하는 코드가 있다면, 이런 사실을 모르는 서비스 이용자는 A서버의 웹페이지에 접근하게 되고, 자신도 모르는 사이에 B서버의 스크립트가 실행되어 쿠키 정보가 B서버에 전송되게 되는 것이다.

많은 서버와 브라우저가 패치되었음에도 불구하고 아직도 cross-site script가 허용되는 곳이 많이 존재하며, CGI를 이용하여 cross-site script의 실행을 차단하는 브라우저를 우회하기도 한다.

Ⅳ. 현재 웹 보안 솔루션의 한계

사실 지금까지 언급한 웹의 보안 문제를 위한 솔루션으로는 웹 방화벽과 웹 스캐너, 웹 서버용 보안소프트웨어를 대표로 들 수 있다. 물론 일부 기능이 추가되어 업그레이드된 솔루션이 있을 수 있으나, 기본적인 방식은 대동소이하므로 동일한 문제점을 가지고 있으며, 이 장에서는 3가지 솔루션에 대한 한계점을 분석한다.

결론부터 말하자면 웹은 여러 서버에 존재하는 리소스로 구성된다는 점과 웹 2.0으로 동적인 웹 페이지가 구성되기 때문에 현재 솔루션의 한계가 있다.

4.1 웹 방화벽

기존의 방화벽이나 침입방지시스템(IPS)는 OSI 7 Layer에서 전송계층 이하(포함)를 대상으로 하고 있다. Deep Packet Inspection과 같이 응용 계층까지 검사하는 기능이 포함되고 있지만, 비정상 프로토콜을 탐지하기 위한 수준에만 그치고 있다. 웹 방화벽은 HTTP에 특화된 방화벽으로 2005년 중국발 SQL injection과 같은 웹해킹이 유행하면서 널리 알려지게 되었다.

웹 방화벽의 역할은 HTTP 트래픽을 좀더 심도 있게 검사하는 것이다. 보통 방화벽과 IPS 뒤에 위치하며, 방화벽과 IPS를 통해서 통과된 세션 중 HTTP 세션을 검사하게 된다. GET이나 POST 혹은 그 외의 요청을 제어할 수 있으며, HTTP 세션 관리가 가능하다. 세션을 관리할 수 있기 때문에 로그 관리가 용이하며 웹 애플리케이션의 취약점도 막을 수 있다. 결과적으로 불필요한 트래픽을 감소시켜 웹 서버의 부하를 줄일 수도 있다. 하지만 위에서 설명한 웹 2.0의 보안 위험까지 해소하기에는 다음과 같은 한계가 있다.

■ 웹 콘텐츠를 검사할 수 없다.

3.1절에서 설명한 UCC의 위험을 예로 들어보자. 웹 방화벽은 기본적으로 HTTP 상의 문제점을 검사하게 된다. 물론 최근 웹 방화벽에는 업로드 되는 콘텐츠에서 위험 요소가 있는지 검사하는 기능이 들어가 있기도 하다. 하지만 웹이라는 것은 여러 서버의 리소스가 하나의 페이지를 구성하게 된다. 한 예로, 노출되는 것은 하나의 웹 페이지이지만, 이를 구성하는 이미지는 다른 서버에 존재하는 경우는 비일비재하다. 콘텐츠를 구성하는 요소의 실제 경로를 분석하여 각 요소를 검사할 수도 있겠지만, 그 모든 요소를 검사하기에는 많은 시간이 소요되는 문제점과 redirect되는 경우 등으로 인하여 현실적으로 불가능함이 있다.

■ 동적인 웹 페이지를 검사 할 수 없다.

개인별 맞춤 페이지나 UCC 등 동적인 구성은 웹 2.0의 특징이다. 웹 방화벽은 inbound traffic을 검사하는 것이 주된 기능인데, 업로드 되는 콘텐츠에서 동적으로 페이지를 변경하도록 하는 자바스크립트가 삽입되어 있는 경우, 웹 방화벽에서는 이 스크립트가 결과적으로 어떤 페이지를 만들어 낼지 모를 수밖에 없다.

스크립트뿐만 아니라 쿠키도 마찬가지이다. HTTP 세션의 브라우저와 서버 간의 세션을 쿠키로 세션 id를 부여하는 경우가 많다. 사용자가 웹 서버에 로그인하면 사용자의 id를 쿠키로 남겨 두고 이후 브라우징에서 이 쿠키의 유무로 사용자를 확인하고 사용자 맞춤 페이지를 제공하는 경우가 있다. 이는 웹 방화벽이 쿠키가 어떻게 생성될지 모르기 때문에 모든 웹 콘텐츠의 위험 요소를 검사할 수가 없다.

4.2 웹 스캐너

웹 스캐너는 웹 서버의 취약점 발견에 주력하는 제품이다. 구축 완료된 웹 애플리케이션이나 웹 페이지상의 취약점을 찾아내기 위해서 비정상적인 입력값을 전송하는 등 시뮬레이션을 통하여 그 취약점을 찾는 스캐너이

다. 하지만 웹 2.0의 문제점은 서버상의 문제점이 아닌 콘텐츠가 보안 문제의 핵심이기 때문에 웹 스캐너로는 이러한 콘텐츠의 보안상 위험요소를 찾아낼 수 없다. 물론 최근 솔루션에는 안티바이러스 기능이 추가되기도 하지만 다음에 기술할 서버용 보안 솔루션과 마찬가지의 문제점을 가진다.

4.3 서버 보안 솔루션

클라이언트 PC의 바이러스와 스파이웨어 감염 검사를 위해 보안 솔루션(주로 백신)을 설치하는 것과 마찬가지로 서버를 위한 보안 솔루션도 존재한다. 이들의 역할은 서버에 업로드되는 binary를 검사하여 악성코드 감염 여부를 검사하는 것이다. 또한 사용자가 웹을 통해서 binary를 다운로드하려고 할 때 검사해주기도 한다.

이 솔루션에도 마찬가지의 문제점이 있다. 솔루션이 적용된 서버의 바이너리만을 검사한다는 점이 그것이다. 위에서 기술한 것과 같이, 대부분의 웹 서비스는 단일 웹 서버로만 구성되지 않는다. 링크를 통해 다른 서버에 존재하는 바이너리를 다운로드 경로로 제공하기도 하고, 다른 서버의 ActiveX가 설치되도록 구성하기도 한다. 따라서 적용된 서버만을 검사하는 것은 여전히 서비스 이용자에게 악성 코드가 실행될 여지를 남기게 된다. 또한 웹 스캐너의 한계점과 마찬가지로 시간과 리소스의 한계가 있기 때문에 모든 서버의 모든 트랜잭션에 대한 검사는 현실적으로 불가능하다.

V. 웹 2.0의 특징에 따른 보안 문제 해결 방안

웹 2.0의 특징과 그에 따른 보안 위험 요소가 존재하며, 그 웹 서비스는 각 요소가 다수 서버에 의해 구성되고, 동적으로 페이지와 정보가 생성되기 때문에 현재의 웹 보안 솔루션만으로는 분명 한계가 있다. 그 한계의 원인은 웹 방화벽 같은 웹 보안 솔루션들이 현실적으로 인터넷 상의 모든 서버에 적용될 수 없으며, 또한 현재의 보안 솔루션은 중간 단계에서 검사하기 때문에 인터넷 이용자는 항상 위험한 사이트에 노출될 수밖에 없다.

결국 동적이든 정적이든 모든 웹 상의 페이지를 보고, 개인 정보가 저장된 쿠키가 있는 단말기 즉 클라이언트 PC의 웹 브라우저를 통하여 웹의 위협을 검사하는 방안으로 지금까지 언급한 문제들과 웹 보안 솔루션의 한계를 극복할 수 있다. 이 마지막 장에서 클라이언트 PC에서 웹을 검사하는 기존의 방법들과 그 문제점을 분석하여 효과적으로 검사할 수 있는 방안을 제시한다.

5.1 기존 클라이언트에서의 웹 검사 방법과 문제점

■ 로컬 웹 프록시

로컬 웹 프록시는 클라이언트 보안 솔루션에서 주로 사용하는 방법이다. 클라이언트에 프록시를 설치하여 80번 포트에서 아웃바운드 연결이 발생할 때 이를 프록시가 받는다. 프록시는 웹 서버와 웹 클라이언트(웹 브라우저)의 중계역할을 하므로 서버에서 전송된 데이터에 유해한 요소가 있는지 검사하게 된다.

이 방법의 장점은 브라우저가 쿠키를 저장하고 저장된 쿠키에 의해서 요청이 만들어지기 때문에 개인화된 페이지까지 검사할 수 있다는 점이다. 한 예로 사용자가 서버에 로그인한 후 보게 되는 웹 메일 페이지를 들 수 있다.

하지만 두 가지 단점이 있는데 스크립트에 의해서 동적으로 구성되는 웹 페이지를 미리 예측하기가 힘들다는 점과 속도가 느리다는 점이다. 프록시에서 스크립트를 파싱하더라도 사용자 입력값에 의해서 스크립트는 다른 결과를 만들기 때문에 스크립트가 어떤 결과를 만들어 낼지는 예측하기 힘들다. 또한 스크립트를 사전에 분석하려면 시간이 걸린다. 그리고 프록시이기 때문에 중간에 네트워크의 연결이 한 번은 더 필요하므로, 웹 브라우징의 속도를 저하시키며 로컬 시스템의 자원을 많이 사용해야만 한다.

■ 네트워크 필터링

네트워크를 필터링 하기 위한 방법은 여러 가지가 있으며 대표적인 방법이 SOCKET Hooking과 Network Filtering이다. 어떠한 형태라도 결국 필터링이라는 점에서 동일하다.

필터링하여 웹을 검사하고자 하면 다음과 같은 절차가 필수적이다.

a. 검사할 네트워크 트래픽 선택
b. HTTP State Inspection
c. HTML 추출
d. HTML 파싱
e. 각 요소 검사

결국 필터링하여 웹을 검사하는 것은 웹 브라우저의 역할과 동일하여, 하나의 웹 페이지를 브라우징하기 위

해 같은 절차를 두 번 수행하여 속도 저하를 가져올 수 밖에 없다. 더욱이 위 모든 과정은 사용자 레벨이나 커널 레벨 모두 동일하게 네트워크의 timeout 시간 이내에 동작해야 한다는 점은 검사의 제약사항이기도 하다.

5.2 성능 저하 없는 웹 검사 방안

위와 같이 웹 2.0의 특징에 따른 보안 문제는 현재의 웹 보안 솔루션으로는 한계가 있으므로, 클라이언트 PC에서 접근하는 방식을 취해야 하나 그 클라이언트 PC의 성능 저하 없이 빠르게 검사하는 것이 필요하다. 이는 웹 브라우저 extension과 적절한 검사 시점으로 그 해답을 찾을 수 있다.

5.2.1 웹 브라우저에서 바이너리의 실행 시점부터 검사

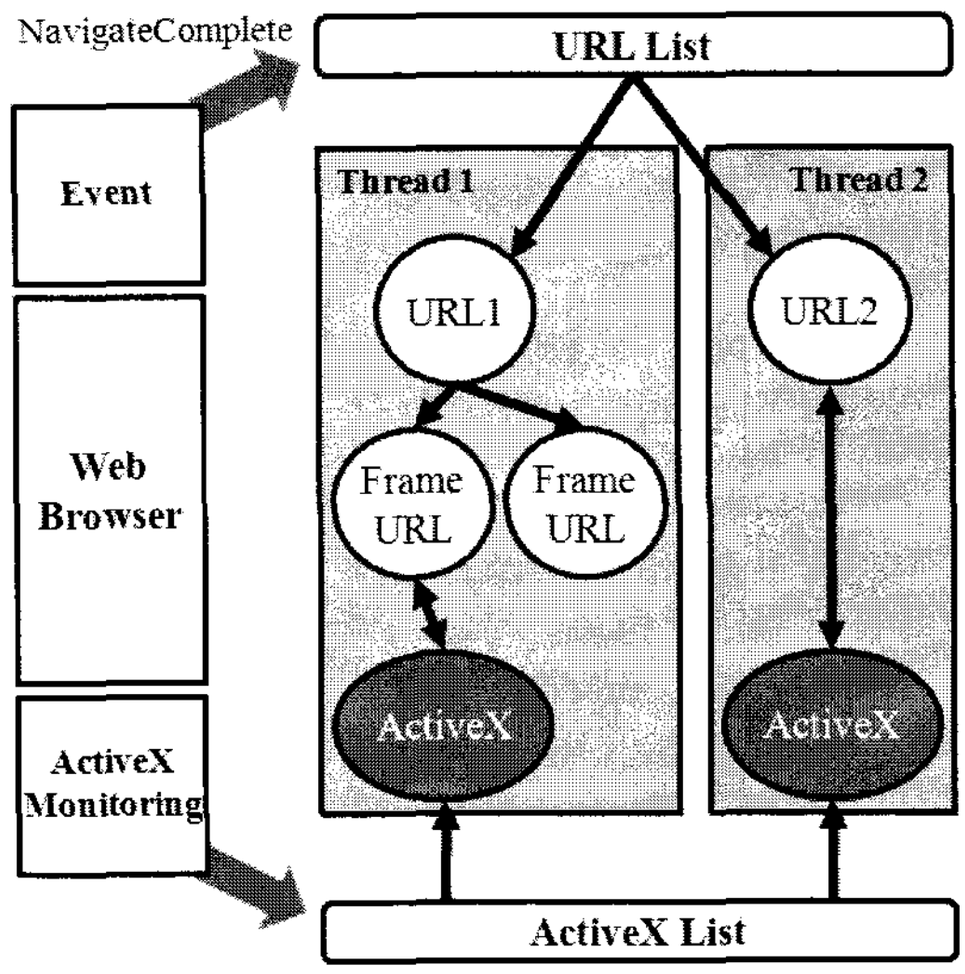
IFRAME injection이나 해커에 의해서 변조된 웹 페이지에서 악성코드가 실행되는 것은 결국 웹 브라우저 프로세스에 의해서 바이너리가 실행되는 것이다. 이러한 점을 착안하여 웹 브라우저에서 새로운 프로세스가 실행되는 시점에서 검사를 시작한다. 이때 실행되려는 프로세스 혹은 관련 바이너리가 악성코드인지를 검사하고 실행되는 과정에 문제가 없는지, 즉 정상적인 HTML(script)에 의해서 실행되는 것인지를 검사하는 것이다. 이렇게 하면 웹을 필터링하거나 파싱하는 일은 필요 없게 된다. 따라서 로컬 시스템의 속도 저하를 최소화 할 수 있다.

5.2.2 웹 브라우저 Extension 이용

현재 많이 사용되는 웹 브라우저들은 모두 extension (또는 plug-in)을 허용하고 있다. 웹 브라우저 별로 제공하는 형태가 차이는 있으나 브라우징 관련 이벤트를 전달해 준다는 점과 DOM에 접근할 수 있다는 점에서 공통점을 갖는다. 또한 Event handler를 등록하여 사용자가 접근하는 웹 페이지에 대한 정보를 얻게 되면 세션을 별도로 관리해야 하는 부담을 덜 수가 있다. 적절한 이벤트가 발생한 시점에서 DOM을 검사하게 되면 동적으로 생성되는 페이지까지 검사할 수 있다.

아래는 위의 두 가지 방안으로 악성코드를 배포하는 사이트를 추출하는 과정을 마이크로소프트의 인터넷 익스폴로어를 기준으로 기술한 것이다.

■ 악성 ActiveX의 배포 사이트 URL 확인 절차

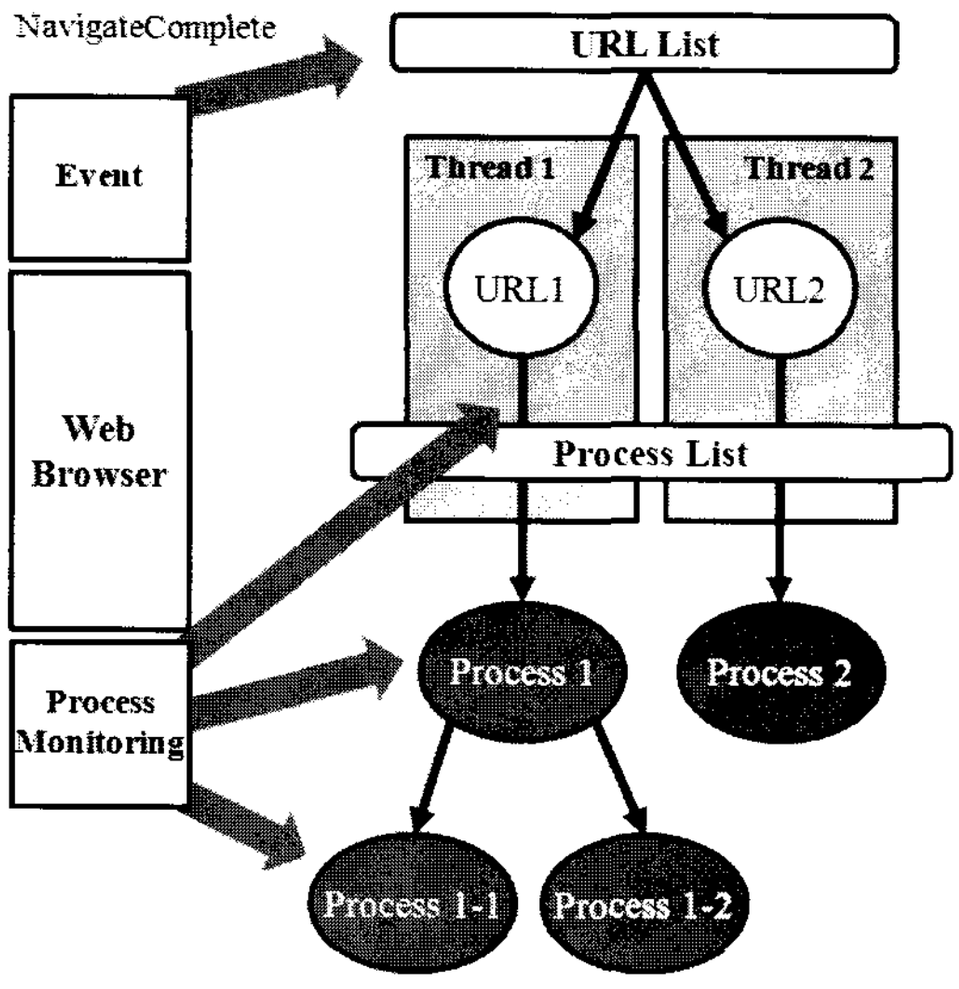


〔그림 6〕 악성 ActiveX 배포 URL 확인 절차

1. 브라우저에 ActiveX가 loading 되는 것을 모니터링한다.
2. NavigateComplete Event가 호출될 때 URL 목록에서 해당 Thread에 URL 노드가 있는지 확인한다.
3. URL 노드가 없다면 Main URL로써 최상위 노드를 등록한다.
4. NavigateComplete 이벤트가 호출되었는데도 이미 Thread에 노드가 있다면 이것은 Frame URL로 등록한다.
5. Browser가 ActiveX를 Loading 하려고 할 때 해당 ActiveX를 검사하여 악성 여부를 확인한다.
6. ActiveX를 Load하는 Thread ID를 참고로 URL 목록에서 해당 ActiveX가 포함된 URL 노드를 찾는다.
7. 서비스로 해당 파일 악성코드 검사를 요청.
8. 악성코드 유무 확인 후 삭제.
9. 사용자에게 위험 정보를 알림.

■ 악성 프로세스를 실행하는 사이트 확인 및 제어

1. ActiveX에 의해서 실행되는 프로세스를 모니터링한다.
2. NavigateComplete 이벤트가 호출될 때 URL 목록에서 해당 Thread에 URL 노드가 있는지 확인한다.
3. URL 노드가 없다면 Main URL로서 최상위 노드



〔그림 7〕 악성프로세스를 실행하는 사이트 확인 및 제어

를 등록한다.

4. ActiveX에 의해서 새로운 프로세스 생성을 시도하면 호출자 Thread ID를 확인하고 URL 목록에서 동일한 Thread ID를 찾고 최상위 URL 노드를 얻는다.
5. 프로세스 목록에 프로세스 노드를 추가하고 관련 URL 노드로 위에서 찾은 노드를 연결한다.
6. 실행되려고 하는 프로세스가 악성코드인지 검사한다.
7. Browser에 의해서 생성된 프로세스가 다시 새로운 프로세스 생성을 시도하면 호출자에 대한 프로세스를 프로세스 목록에서 찾은 후, 새로 생성될 프로세스 노드를 자식 노드로 추가한다.
8. 위의 9와 10번의 과정을 반복한다.
9. 실행되는 프로세스가 악성코드이면 연결되어 있는 URL 노드를 참조하여 악성코드를 실행하려는 사이트의 URL을 알 수 있다.

Ⅵ. 결 론

인터넷은 이제 하나의 Media로서 모든 정보의 흐름이 인터넷을 통해서 이뤄지고 있고, 웹 2.0으로 누구나 참여하고 생성하고 공유할 수 있는 환경이다. UCC, 블로그, 소셜 네트워크 등은 이제 보편화된 서비스로 존재하고 있으며 이를 통해 참여하고 생성한 콘텐츠는 '검색'과 '포털'을 통하여 쉽고 빠르게 공유된다. 또한 동적인 웹 페이지의 구성으로 대화형 인터페이스가 가능하고 개인별 맞춤 웹 환경 구성이 가능해졌다.

하지만 이러한 콘텐츠 제작과 공유는 그 역기능으로 악용되고 있으며 금전적 피해와 개인정보의 유출 피해는 계속 증가할 것이며 그 악용 기법은 더욱 다양해 질 것이다. 이러한 웹의 보안 문제는 기존의 웹 1.0의 보안 솔루션인 웹 방화벽, 웹 스캐너, 웹 서버 보안 솔루션으로는 웹 2.0의 동적인 페이지와 복잡해진 서버 환경에 분명 한계가 존재한다.

그 한계는 클라이언트의 웹 브라우저 보안으로 해결할 수 있으며, 이 방식이 효율적이면서 로컬 시스템에 성능 저하를 최소화할 수 있다. 이를 위해 웹 브라우저의 extension을 이용한 이벤트와 DOM 검사, 실행 프로세스 검사 등 그 방안을 모색했다.

그러나 웹 서비스 업체의 보안 불감증, 웹 브라우저 보안이라는 생소한 보안 영역은 여전히 과제로 남아 있으며 이는 PC에 '백신'이 정착화 된 것과 같이 꾸준한 연구, 다양한 검사 기술과 범위의 확장 그리고 웹 브라우저 보안이라는 '백신'이외의 또 하나의 필수 보안 솔루션이 필요하다는 인지와 인식이 필요하다.

참고문헌

[1] 박정환, 방지호, "Web 2.0 Technical Trend and Web Security Treat Analisys", 한국정보보호진흥원 연구자료, pp.4-6, 2006.

[2] 송복섭, 권수갑, "Web 2.0 개념 및 서비스 동향", 정보통신연구진흥원 주간기술동향, 1296호, pp.14-19, 2007.

[3] 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.

[4] 김성훈, 김인호, 이응용, 백기언, "소셜 네트워크 서비스와 정보보호", 한국정보보호진흥원 CSO Briefing, 기술정책 07-10, pp.2-8, 2007.

[5] Server-Side Scripting, http://en.wikipedia.org/wiki/Server-side_scripting

[6] <사고사례> OO연구기관, 악성코드 유포 및 해킹 경유지 악용 사고, 안철수연구소 보안 정보.

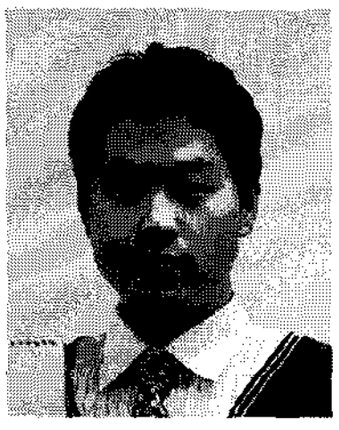
〈著者紹介〉

이 창 우 (ChangWoo Lee)

2002년 2월 : 강원대학교 컴퓨터공학과 졸업
2002년 1월~현재 : 안철수연구소 서비스개발팀 SiteGuard 선임연구원
<관심분야> 네크워크, 컴퓨터 보안, 컴퓨터 아키텍쳐

김 창 희 (Daniel KIM)

2003년 2월 : 대전대학교 정보통신공학과 졸업
1999년~2003년 : (주)인츠 VoIP 팀장
2003년~2005년 : 株式會社 インターネットテレホン VoIP개발팀장
2005년 9월~현재 : 안철수연구소 서비스플랫폼팀 SiteGuard Project Manager
<관심분야> VoIP, IPTV, 이동통신, 정보보호



이 준 호 (Junho Lee)

2004년 2월 : 한국외국어대학교 산업공학과 졸업
2005년~2007년 4월 : 모바일리더 기획마케팅팀 UX파트
2007년 5월~현재 : 안철수연구소 서비스플랫폼팀 SiteGuard 기획파트
<관심분야> 웹 2.0, 보안, 정보보호