Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.26, NO.4, Aug. 2016 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2016.26.4.1025

HTML5 웹 서비스 환경에서의 개인정보 침해 탐지 모듈 구현*

한 미 란,[1†] 곽 병 일,[1] 김 환 국,[2] 김 휘 강[1‡]
[1]고려대학교 정보보호대학원, [2]한국인터넷진흥원

Implementation of the Personal Information Infringement Detection Module in the HTML5 Web Service Environment*

Mee Lan Han,[1†] Byung Il Kwak,[1] Hwan Kuk Kim,[2] Huy Kang Kim[1‡]
[1]Graduate School of Information Security, Korea University,
[2]Korea Internet & Security Agency

요 약

ActiveX와 같은 비표준 기술에 기반을 둔 인터넷 환경을 개선하기 위해 국제 웹 표준 활용 기술인 HTML5의 전환이 진행되고 있다. 웹 페이지를 만들기 위한 기본 프로그래밍 언어인 HTML5 (Hyper Text Markup Language 5)는 HTML4보다 보안을 고려하여 설계되었다. 그러나 새롭게 추가된 신기술들로 인해 웹 기반 취약점에 대한 공격 범위가 넓어졌으며, HTML4 환경에서 발생하던 다양한 보안 위협들이 그대로 상속되고 있다. 본 논문에서는 스크립트 기반 사이버 공격 및 HTML5 기능 악용 스크립트 중 개인정보 침해가 발생할 수 있는 부분에 초점을 두었으며, 어떤 공격에 악용되어 개인정보가 침해되는지 실제 재현하였다. 또한, 개인정보 침해를 유발하는 공격에 대해 클라이언트 기반으로 진단 가능한 플러그인 타입의 탐지 모듈을 구현하였다. 진단 스캐너를 이용한 공격 탐지는 HTML5 기반 웹 어플리케이션이나 웹 페이지에 대한 신뢰도를 자가 진단한 후 웹 서비스를 이용하므로 HTML5의 취약점 공격에 대해 사전대응이 가능하며, 새로운 취약점 발생 시 탐지 기능 추가가 자유로워 확장성이 용이하다.

ABSTRACT

The conversion of the international standard web utilization HTML5 technology is being developed for improvement of the internet environment based on nonstandard technology like ActiveX. Hyper Text Markup Language 5 (HTML5) of basic programming language for creating a web page is designed to consider the security more than HTML4. However, the range of attacks increased and a variety of security threats generated from HTML4 environment inherited by new HTML5 API. In this paper, we focus on the script-based attack such as CSRF (Cross-Site Request Forgery), Cookie Sniffing, and HTML5 API such as CORS (Cross-Origin Resource Sharing), Geolocation API related with the infringement of the personal information. We reproduced the infringement cases actually and embodied a detection module of a Plug-in type diagnosed based on client. The scanner allows it to detect and respond to the vulnerability of HTML5 previously, thereby self-diagnosing the reliability of HTML5-based web applications or web pages. In a case of a new vulnerability, it also easy to enlarge by adding another detection module.

Keywords: HTML5, Personal Information Infringement, Weakness, Vulnerability

Received(05. 02. 2016), Modified(07. 12. 2016), Accepted(08. 04. 2016)
* 본 논문은 미래창조과학부 및 정보통신기술진흥센터의 정보통신·방송 연구개발사업의 일환으로 수행하였음. [14-912-06-002, 스크립트 기반 사이버 공격 사전 예방 및 대응 기술 개발]
† 주저자, blosst@korea.ac.kr
‡ 교신저자, cenda@korea.ac.kr(Corresponding author)

I. 서 론

국내 웹 사이트에서는 동영상 재생과 파일 업로드, 인터넷뱅킹 등의 웹 서비스 사용을 위해 특정 운영체제와 브라우저에 종속적인 ActiveX를 광범위하게 사용하고 있다. 다운로드 서비스, 컴퓨터 내부의 파일 생성 및 삭제, 음악 재생 등 사용자의 인지 없이 웹페이지 접속만으로 실행할 수 있는 ActiveX의 편리한 기술은 그러나 악성코드가 퍼지기 쉬운 환경을 제공하게 되었다. 이에 따라 2012년 방송통신위원회는 보도 자료를 통해 인터넷 환경 개선을 위한 표준화 추진 및 공공과 민간의 웹 사이트에 대한 HTML5 전환 지원을 보도한 바 있다. 웹 기반 기술인 HTML5는 비표준 기술인 ActiveX 사용을 줄일 수 있는 것은 물론, 구글과 애플이 주도하고 있는 운영체제 플랫폼에 대한 의존성도 줄일 수 있는 최적의 대안으로 꼽히고 있다. 온라인 쇼핑 등 전자상거래 분야에 활용하고, 파급효과가 있는 공공 기관 사이트 및 중소기업 사이트 200개에 대해 HTML5로의 전환을 지원하여 국내 웹의 조속한 전환을 유도한다는 것이다[1].

HTML5는 웹 문서를 만들기 위한 기본 프로그래밍 언어인 HTML의 최신 규격으로, 기존의 HTML 기능에 다양한 애플리케이션을 표현하고 제공하도록 진화한 웹 프로그래밍 언어이다. HTML5의 핵심은 브라우저를 통해 접근 가능한 시스템 자원을 API 도구를 통해 대폭 보강하는 것으로써, 웹 브라우저 내부 프로그램에 API를 제공하여, 작성된 코드가 웹 브라우저 상에서 다양한 응용 프로그램으로 동작하도록 하는 것이다[2]. 그러나 canvas/embed를 포함한 각종 태그 및 속성, CORS, Web Storage, Web Socket, Geolocation, File API 등 HTML5 대표적인 신규 기술들을 통해 새로운 보안 위협이 발생할 수 있으며, 컴퓨터와 스마트폰의 브라우저뿐만 아니라, 자동차, 가전제품 등 다양한 영역의 브라우저에도 적용되어 공격자들이 악용할 수 있는 공격 범위를 더 넓게 만들고 있다[3]. 또한, HTML4 환경에서 발생하던 보안 위협들이 HTML5에서 그대로 상속되고 있으며, 표준화된 인증 절차 미비, 웹 개발자, 일반 사용자, 웹 사이트 단말, 웹 어플리케이션 등 웹과 관련된 모든 요소가 신규 보안 위협에 취약할 수 있다. 특히, CSRF (Cross-Site Request Forgery), Cookie Sniffing 취약점과 CORS (Cross-Origin Resource Sharing), Geolocation API 기능을 악용한 개인정보 침해가 우려 된다[4].

따라서 본 연구에서는 HTML5를 기반으로 하는 웹 서비스의 안전성 검증을 위해 HTML4에서 발생하는 스크립트 기반 공격과 HTML5 API 중에서 개인정보 침해가 발생할 수 있는 기능을 대상으로 하여 구체적으로 어떤 공격에 악용되어 개인정보가 침해되는지 실제 재현해 보았다. 특정 공격에 대한 취약점이 존재할 경우, 클라이언트 기반의 사전 탐지 및 대처가 가능하도록 하여 피해를 예방하고 경감시킬 수 있는 방법에 대해 서술하고자 한다. 2장에서는 웹에서 발생할 수 있는 프라이버시 위험에 대한 조치 사항과 최근 어떤 연구들이 진행되는지 설명하였으며, 3장에서는 CSRF, Cookie Sniffing 취약점과 CORS, Geolocation API 기능을 악용한 개인정보 침해요소에 대해 상세 기술하였다. 4장에서는 실제 개인정보가 어떻게 침해되는지 재현하였고, 이러한 공격을 사전에 미리 탐지 할 수 있는 모듈에 대해 설명하며, 마지막으로 5장에서는 결론으로 마무리 하였다.

II. 관련 연구

전 세계 오픈소스 기반의 웹, 어플리케이션, 소프트웨어 보안 표준 기술을 선도하고 있는 OWASP (The Open Web Application Security Project)는 웹에 관한 정보노출, 악성 파일 및 스크립트, 보안 취약점 등을 연구하며, 웹 애플리케이션 취약점 중에서 발생 빈도가 높고, 보안상 큰 영향을 줄 수 있는 취약점을 선정하여 문서로 공개하고 있다. 2013년 웹 보안과 관련한 10대 취약점과 2014년 모바일 웹 관련 10대 취약점, 그리고 프라이버시 위험에 대한 보고서를 발표하였다[5]. HTML5에 특정된 취약점이 공개된 것은 아니지만, 기존 HTML4 환경에서 발생하던 공개된 취약점들은 HTML5에서도 그대로 상속될 수 있다. Table 1.은 2016년 OWASP에서 발표된 10대 프라이버시 위험에 대한 조치 사항이다. 그러나 OWASP 취약점 발표에도 불구하고 여전히 국내 사이트 상당수는 해킹에 무방비로 노출되어 있다고 한다. 웹 사이트 개발 시 보안성을 염두에 두지 않고, OWASP나 행정자치부의 시큐어코드 등 가이드라인에 따라 제작하지 않을 경우, 취약점에 그대로 노출될 수 있는 것이

Table 1. OWASP Top 10 Privacy Risks Project

NO	Title	Frequency	Impact
P1	Web Application Vulnerabilities	High	Very high
P2	Operator-sided Data Leakage	High	Very high
P3	Insufficient Data Breach Response	High	Very high
P4	Insufficient Deletion of personal data	Very high	High
P5	Non-transparent Policies, Terms and Conditions	Very high	High
P6	Collection of data not required for the primary purpose	Very high	High
P7	Sharing of data with third party	High	High
P8	Outdated personal data	High	Very high
P9	Missing or Insufficient Session Expiration	Medium	Very high
P10	Insecure Data Transfer	Medium	Very high

다.

우리나라 개인정보보호법 제26조 업무위탁에 따른 개인정보의 처리 제한에 따르면, 개인정보 처리자가 제3자에게 개인정보의 처리 업무를 위탁하는 경우에는 개인정보의 기술적 관리적 보호조치에 관한 사항에 따라야 한다고 기술하고 있으며, OWASP 10대 취약점 방어, 국정원 홈페이지 8대 취약점 방어, 주기적인 개인정보 노출 점검을 통해 대응해야 한다고 말하고 있다. 개인정보보호법 제29조 안전조치의무에 따르면, 개인정보 처리자는 개인정보가 분실·도난·유출·위조·변조 또는 훼손되지 않도록 내부 관리계획 수립, 접속기록 보관 등 대통령령으로 정하는 바에 따라 안전성 확보에 필요한 기술적·관리적 및 물리적 조치를 하여야 한다고 기술하고 있으며, 웹사이트의 개인정보 노출 현황에 대해 자체 감사 및 주기적 보고를 해야 한다고 제시하였다[6]. 이러한 개인정보 처리 및 관리 지침에도 불구하고, 인터넷의 개인정보 침해와 오남용은 항상 논쟁의 대상이 되곤 한다. 최근 연구에 따르면, 기업들은 개개인에 특화된 광고를 보여주기 위해 쿠키를 오남용하는 사례가 많으며, 외부 도메인으로부터 설정된 Third-Party Cookies를 많이 사용한다. 그러나 웹 페이지 방문자의 경우 자신의 개인정보가 의도하지 않게 사용되는 것을 막을 방법이 없다. Cookies 정보는 HTTP Request에 의해 전송되기 때문에, 제3자에 의해 패킷 스니핑 될 수도 있어 데이터 저장에 있어 안전하지 않다[7]. HTML5 위치 정보 API는 웹을 통해 위치 기반 서비스를 가능하게 하지만, 사용자의 위치 정보를 쉽게 노출 시킬 수 있다[8].

Table 2. The pros and cons of the method of vulnerability detection in HTML5

Rule-based Detection [9][10]	
Pros	• high-accuracy detection of the attack pattern corresponding to the defined rules • providing precise evidence for detection result
Cons	• Difficulty of detection for the attack based on the undefined rules • Requiring the large data set for efficient detection

Sequence-based Detection [11][12]	
Pros	• Detailed analysis is possible
Cons	• Requiring a lot of time for analysis • Difficulty of detection for the complex and sophisticated attack

HTML 웹 페이지에서 발생할 수 있는 공격 벡터 탐지에 대한 연구는 오용탐지(misuses detection) 기법의 한 종류인 규칙 기반과 이상탐지(anomaly detection) 기법의 한 종류인 행위 기반으로 분류하여 정리하였다. 규칙 기반이란 알려진 공격 패턴과 유사한 형태의 행위가 발견되면 침입으로 판정하는 탐지 방법이며, 행위 기반은 일반적인 행위 패턴으로부터 벗어난 비정상적인 행위나 사용에 근거한 이상 침입을 탐지하는 방법이다. 우선 규칙 기반의 연구 중에서 Dong et al.의 논문은 HTML4 및 HTML5에 대한 XSS 공격 벡터들을 조사하여 3가지 transform 기법을 사용하였다. XSS 스크립트를 생성하고 E-mail 프로토콜 규격의 구성요소 중 5가지 항목에 선택적으로 삽입 후, 실제 Web-mail system에 전송하여 테스트하였다[9]. Bates et al.의 논문에서는 인젝션을 차단하기 위한 Client-side XSS 필터를 HTTP Request에서

Table 3. CAPEC attack vector, CWE and CVE for XSS and CSRF based on script

	CAPEC Attack Vector [14]		CWE		CVE	
	ID	Description	ID	Description	ID	Description
XSS	18	Embedding Scripts in Non-Script Elements	79	• Trustless data is entered in the web application via the web request. • The web application dynamically creates a web page containing trustless data. • When the web pages are generated, web application doses not block the content executed by the web browser such like Javascript, HTML tag, Mouse event, Flash, ActiveX. • Victim visits the web page created by the web browser with a malicious script. • Because the script result from the web page transmitted at the web server , victim's web browser execute the malicious script in the contents of domain of the web server.	2008-5080	Chain: protection mechanism failure allows XSS
	19	Embedding Scripts within Scripts			2006-4308	Chain: only checks "javascript:" tag
	32	Embedding Scripts in HTTP Query Strings				
	63	Simple Script Injection			2007-5727	Chain: only removes SCRIPT tag, enabling XSS
	85	AJAX Fingerprinting			2008-5770	Reflected XSS using the PATH_INFO in a URL
	86	Embedding Script (XSS) in HTTP Headers			2008-4730	Reflected XSS not properly handled when generating an error message
	91	XSS in IMG tag			2008-5734	Reflected XSS sent through email message
	106	Cross Site Scripting through Log Files			2008-0971	Stored XSS in a security product.
	198	Cross-Site Scripting in Error Pages			2008-5249	Stored XSS using a wiki page
	199	Cross-Site Scripting Using Alternate Syntax			2006-3568	Stored XSS in a guestbook application
	209	Cross-Site Scripting Using MIME Type			2006-3211	Stored XSS in a guestbook application using a javascript: URI in a bbcode img tag
	243	Cross-Site Scripting in Attributes				
	244	Cross-Site Scripting via Encoded URI Schemes			2006-3295	Chain: library file is not protected against a direct request, leading to reflected XSS
	245	Cross-Site Scripting Using Doubled Characters, e.g. %3C%3Cscript				
	246	Cross-Site Scripting Using Flash				
	247	Cross-Site Scripting with Masking through Invalid Characters in Identifiers				

Table 3. continued

	CAPEC Attack Vector [14]		CWE		CVE	
	ID	Description	ID	Description	ID	Description
CSRF	62	Cross-Site Request Forgery	352	• When the web server is designed to receive a request without any mechanism ensured whether the request is intentionally transmitted from the client, an attacker can make deceive client request to the web server for response. • Cause of the unintended code execution or data exposure via URL, IMG loading and XMLHttpRequest.	2004-1703	Add user accounts via a URL in an IMG tag
					2004-1995	Add user accounts via a URL in an img tag
					2004-1967	Arbitrary code execution by specifying the code in a crafted IMG tag or URL
					2004-1842	Gain administrative privileges via a URL in an IMG tag
					2005-1947	Delete a victim's information via a URL or an IMG tag
	111	JSON Hijacking			2005-2059	Change another user's settings via a URL or an IMG tag
					2005-1674	Perform actions as administrator via a URL or an IMG tag
					2009-3520	Modify password for the administrator
					2009-3022	CMS allows modification of configuration via CSRF attack against the administrator
					2009-3759	Web interface allows password changes or stopping a virtual machine via CSRF

콘텐츠를 비교하는 방법을 사용하기도 하였다. Client-side XSS 필터는 브라우저 HTML 파서와 자바스크립트 엔진 사이에 위치시키고, 구현한 필터는 기본 구글 크롬 환경에서 사용 가능하다[10].

행위 기반의 연구 중에서 Gol et al.의 논문은 White-box 테스트를 이용하여 취약점 및 버그를 테스트하고 기존의 자동화된 웹 취약점 스캐너의 성능을 개선할 수 있는 방법을 제안하였다. 이 방법은 어플리케이션의 내부 구조와 동작을 검사하는 소프트웨어 테스트 방식으로 소스 코드에서 코드의 결함이나 취약한 라인을 추적할 수 있다. 내부 소스코드의 동작을 개발자가 추적할 수 있기 때문에, 동작의 유효성뿐만 아니라 실행되는 과정을 살펴봄으로써, 코드가 어떤 경로로 실행되며, 불필요한 코드와 테스트되지 못한 부분을 확인할 수 있다[11]. Choo et al.의 논문에서는 브라우저에서 일어나는 연쇄 행위를 알아내기 위하여 API call과 브라우저 내 일어나는 행위와 이벤트를 분석하는 방법을 제안하기도

하였다. 스크립트 언어는 각 명령마다 순차적으로 일어나기 때문에 연쇄적으로 나타나는 행위의 순서를 파악하는 것이 가능하다. 운영체제와 독립된 가상의 브라우저를 사용하여 HTML5의 잠재적인 공격을 발견하였고, 공격자의 연쇄 행위 패턴을 정의하였다[12]. Table 2.는 웹 취약점 탐지 방법을 규칙 기반과 행위 기반 연구로 분류하여 장단점을 기술한 내용이다.

III. 보안약점과 취약점

스크립트 기반의 사이버 공격은 XSS 공격, SQL 인젝션 공격, 웹기반 DDoS공격, Web Shell 공격, ActiveX 취약점 공격, PHP 사이트스크립트 공격으로 정리할 수 있으며, HTML5 API 기능으로 인한 신규 취약점은 신규 태그/속성을 이용한 Tabnabbing 공격, CORS를 이용한 CSRF 공격, Web Storage 정보탈취, Web Socket을 이용한 사설 네트워크 정보 수집, Geolocation을 이용한 사용자 위치 추적, File API를 이용한 사용자 시스템 파일 탈취, Web Workers를 이용한 DDoS, Cross-Document Messaging 공격으로 정리해 볼 수 있다. 본 논문에서는 기존 HTML4에서 발생하는 스크립트 기반 공격인 CSRF, Cookie Sniffing 취약점과 HTML5의 주요 보안 취약점 중 개인정보 침해요소가 있는 CORS, Geolocation API 기능을 악용한 공격에 대해 상세 기술하였다. 이 중 이미 공개된 XSS와 CSRF에 대한 공격 벡터와 보안약점 그리고 취약점을 매칭하여 Table 3. 과 같이 정리하였다[13,14]. 보안약점은 보안취약점이 될 수 있는 소프트웨어의 결함, 실수, 버그와 같은 것을 말하고, 취약점은 공격자가 이용하였을 경우 시스템의 보안정책을 침해하게 되는 시스템의 허점을 말한다. 소프트웨어 보안약점의 주요 유형에서도 입력 데이터 검증 및 표현, API 오용의 경우 웹 개발에서 자주 발생할 수 있는 보안약점이다[15].

3.1 Cross-Site Request Forgery (CSRF)

CSRF 공격이란 사용자가 자신의 의지와는 무관하게 공격자가 의도한 행위의 HTTP Request를 특정 웹 서버에 요청하게 만드는 공격으로 공격자는 이미지 태그, XSS 등 다양한 공격 벡터를 이용하여 피해자가 변조된 HTTP Request를 발생시키도록 만든다. HTTP Request를 받은 타겟 서버는 현재 요청이 올바른 경로를 통한 정상적인 요청인지 검증 절차를 마련하여 방지해야 한다. CSRF 공격과 XSS 공격은 공격 대상과 방식에서 차이를 보인다. XSS의 경우 공격자가 세션 하이재킹, 웹 사이트 변조, 악성 스크립트 삽입 등을 통하여 사용자의 브라우저에서 악성 스크립트를 실행하는 것으로 사용자 정보나 세션 탈취가 목적인 반면에 CSRF의 경우에는 사용자의 권한으로 데이터 삭제, 변조 등의 HTTP Request를 공격 대상 서버에 전송하여 서버로 하여금 공격자가 원하는 작업을 수행하도록 만드는 것이 목적이다.

3.2 Cookie Sniffing

XSS를 이용한 Cookie Sniffing은 공격자가 삽입한 악성 스크립트에 의해 사용자의 Cookie 정보가 공격자에게 전송되어 특정 사용자의 웹 브라우저에 있는 Cookie를 훔치거나 엿보는 기술이다. XSS 공격은 공격자가 의도적으로 브라우저에서 실행될 수 있는 악성 스크립트를 웹 서버에 입력 또는 출력 시 위험한 문자를 필터링 하지 않고 처리하는 애플리케이션의 개발 과정에서 발생한다. E-mail, 게시글 등에 삽입된 악성 스크립트를 이용해 개인정보 유출, 바이러스 배포 및 세션 재사용 공격, CSRF 공격, 피싱 공격 등의 공격 수행이 가능하다. Sniffing에 사용되는 Cookie 파일은 웹 사이트에 접속할 때 자동적으로 만들어지는 임시 파일로 이용자가 본 내용, 상품 구매 내역, 신용카드 번호, 계정, 비밀번호, IP 주소 등의 정보를 담고 있는 일종의 정보파일이다. 주로 웹 기반의 환경에서 사용되며, 이용자가 웹 사이트에서 로그인한 이후 동일 웹 사이트에서 다른 서비스 페이지를 방문할 때 회원정보를 계속해서 입력하는 불편함을 없애는 기능을 하고 있다. Cookie 파일은 계정과 비밀번호, 개개인의 행위 패턴을 엿볼 수 있는 중요한 정보를 가지고 있기 때문에 유출 시 사생활 침해나 기타 개인정보 유출 등의 심각한 피해를 입을 수 있다. XSS 취약점은 비교적 쉽게 공격할 수 있으며, 웹 애플리케이션 개발 시 제거되지 않아 매우 광범위하게 분포되기도 한다.

3.3 Cross-Origin Resource Sharing (CORS)

CORS는 HTML5 XML Level2가 지원하는 기

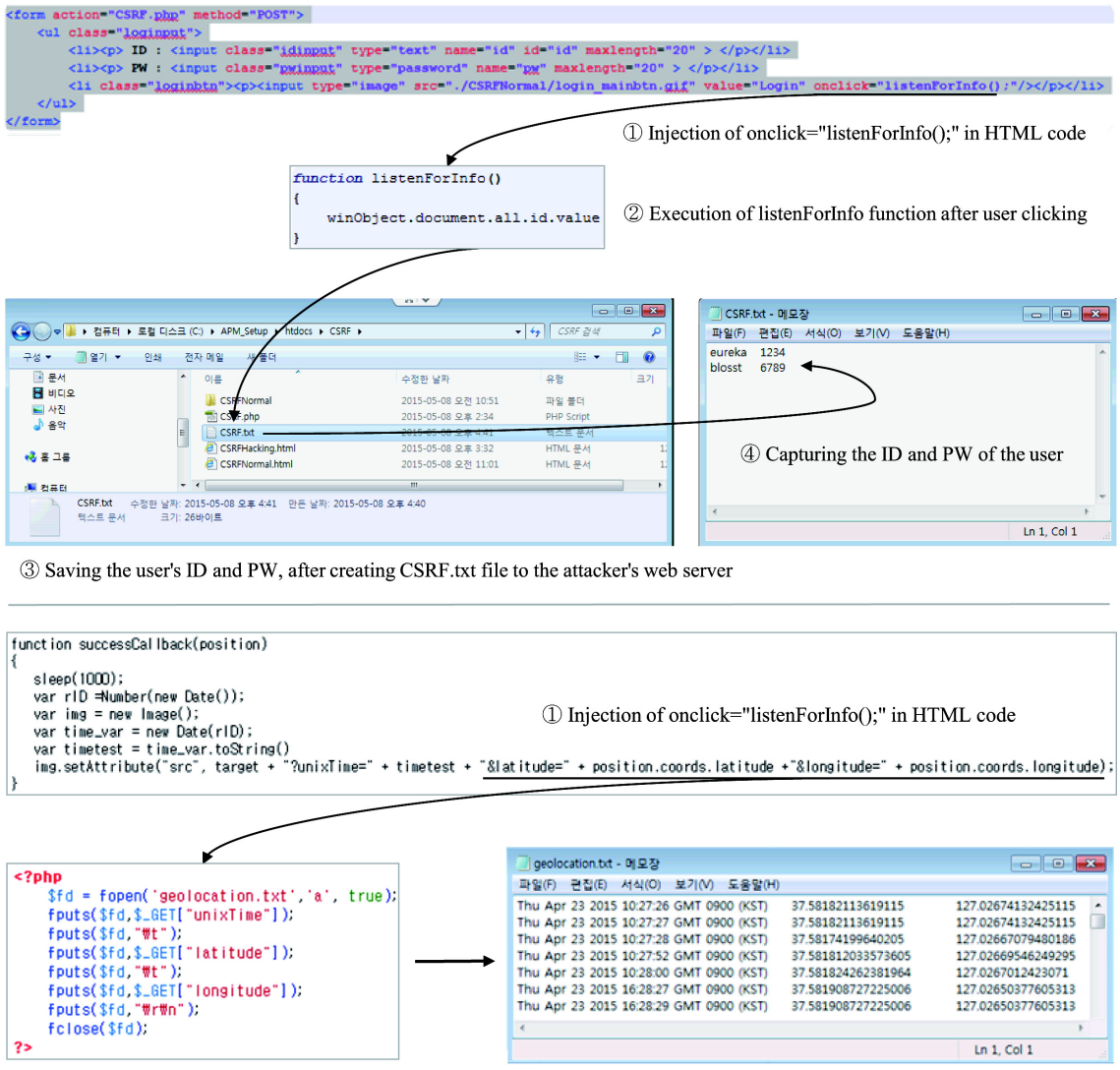

Fig.1. Infringement reappearance of ID and Password by CSRF attack (Top) and location-tracking by Geolocation API (Bottom)

술로 브라우저가 XHR (XMLHttpRequest)을 이용하여 Cross-Origin 요청을 가능하게 한다. 이러한 CORS가 등장하게 된 것은 최근 정보를 제공하는 사이트에서 각종 컨텐츠와 서비스를 가져와 새로운 서비스를 재창출하는 매쉬업 형태의 사이트가 증가함에 따라 XHR을 이용하여 다른 도메인에 리소스를 요청하지만 SOP (Same Origin Policy)에 위배되어 사이트 개발이 어렵다는 문제점 때문이다. HTML5 XML Level2가 지원되기 이전의 XHR Level1에서는 SOP에 제한을 받아 COR을 발생시킬 수 없었고, HTML5에서 지원하는 XHR Level2는 COR을 지원해 CORS가 가능하게 되었다. 즉, CORS를 통해 서로 다른 도메인 간 자원을 공유하고 SOP우회가 가능해진 것이다. CORS 방식 중 특정 사이트에 대한 접근을 허용하는 방식은 가장 간단하고 쉬운 방식으로 API 서버의 설정에서 모든 소스로부터 들어오는 API 호출을 허용하게 한다. 이는 HTTP로 API를 호출하였을 경우 HTTP Response에 응답을 주면서 HTTP Header에 Request Origin(요청을 처리해줄 수 있는 출처)를

명시 하는 방식이다.

3.4 Geolocation API

HTML5에서는 브라우저에 내장된 Geolocation API를 통해 사용자 디바이스의 위치정보를 획득할 수 있다. 브라우저의 기본 기능으로 디바이스의 종류와는 무관하게 실행되며 경도, 위도, 이동 방향, 속도, 현재 시간 등의 다양한 정보를 제공한다. 기존의 IP 기반 위치정보는 City level이라 거리 오차가 컸으나 HTML5 Geolocation API의 경우 접속기기의 물리적 위치가 Street level로 노출된다. IP 주소뿐만 아니라 GPS, RFID, WiFi와 Bluetooth MAC address, GSM/CDMA cell ID 등을 통해 위치 정보를 산출한다. 브라우저는 위치 계산에 필요한 정보를 Location Service에 전송하여 사용자의 위치정보(경도, 위도)를 제공받는데, Location Service에 따라 위치를 계산하는 알고리즘과 사용하는 데이터베이스(IP, Cell tower ID에 맵핑되는 물리적 주소)가 다르므로 사용하는 Location Service에 따라 위치정보가 다르게 계산될 수 있다. GPS 모듈이 없고, 인터넷에 연결되어 있지 않을 경우 Geolocation API를 이용해 위치를 계산할 수 없지만, 이미 캐시된 위치정보를 사용할 수 있다. 공격자는 사용자의 지리적 위치 정보를 자신에게 전송하는 위장 웹 사이트를 개설하거나 해킹을 통해 기존 서비스 중인 웹 페이지를 변조할 수 있다. 웹 브라우저를 통해 사용자가 해당 웹 페이지에 방문하면 악성 스크립트가 실행되어 Geolocation API를 통해 획득한 사용자의 위치 정보가 공격자에게 전송될 수 있다.

IV. 방법론

4.1 개인정보 침해 재현

본 장에서는 스크립트 기반의 공격 CSRF와 HTML5 신규태그 취약점 Geolocation API 공격에 대해 개인정보 침해가 어떻게 발생할 수 있는지 테스트 용 악성 웹 페이지를 제작하여 진행해 보았다. 특정 웹사이트가 사용자의 웹 브라우저를 신뢰하는 상태를 이용하여, 로그인한 상태에서 사이트 간 요청 위조 공격 코드가 삽입된 페이지를 열면, 공격 대상이 되는 웹사이트는 위조된 공격 명령이 신뢰할 수 있는 사용자로부터 발송된 것으로 판단하게 되어 공격에 노출될 수 있다. 앞서 설명 드렸던 html 소스 내 CSRF 공격에 취약한 태그 중 Form 태그를 이용한 공격을 시도해 보았다. onclick 속성을 통해 함수를 호출하고, 함수 호출 시 계정과 비밀번호를 공격자의 웹서버로 전송하는 것을 확인할 수 있었다.

특정 목적을 위해 나의 위치를 공개하지 않은 이상 사용자 동의 없이 웹 브라우저 사용 시 GPS 정보가 쉽게 노출되지는 않는다. 그러나 팝업 메시지 혹은 경고 메시지라 할지라도 무의식중에 클릭하거나 방심한 상태로 개인정보 사용자 동의를 하는 경우가 있다. 또한, GPS 디바이스와 인터넷이 연결되어 있지 않더라도 캐시된 과거 위치정보가 사용되어 웹 브라우저 사용자의 정보가 노출될 가능성도 존재한다. GPS 추적 코드를 테스트용 웹사이트에 삽입한 후 이 웹사이트에 접속했을 경우 웹서버에 Geolocation 정보가 저장되도록 하였다. 스마트폰을 통해 취약한 웹사이트 접속 시 초단위로 변동되는 GPS 정보가 저장될 수 있음을 확인하였다. Fig.1.은 CSRF 공격을 이용한 계정, 비밀번호 정보 침해 과정과 Geolocation API를 이용한 위치 정보 침해 과정을 재현해 본 것이다.

4.2 탐지 모듈 구현

본 논문에서 제안하는 개인정보 침해 탐지 모듈은 Fig. 2.와 같이 구성되어 있다. 구현된 탐지 모델은 크롬 확장 프로그램으로 설정하여 설치 및 관리가 가능하고, 크롬 브라우저 오픈 시 동시에 연동된다. 이렇게 연동된 탐지 모듈은 사용자가 확장 프로그램 아이콘을 클릭하여 실행되며, 아이콘 클릭 시 파싱된 웹 페이지를 검색한다. Fig.2. 탐지 모듈 에서는 스크립트 기반 공격과 HTML5 API 기능취약점을 이용한 공격으로 나뉘어 탐지할 수 있도록 하였고, 대응 모듈에서는 취약점 유무를 확인하여 경고 혹은 안전 메시지를 출력하도록 하였다.

사용자가 웹 브라우저를 신뢰한다는 점을 악용하여, 사이트 간 요청 위조 공격 코드가 삽입된 페이지를 클릭할 경우, 공격 대상이 되는 웹 사이트는 위조된 공격 명령이 신뢰할 수 있는 사용자로부터 발송된 것으로 판단할 수 있다. 이러한 공격에 노출된 게시판 등을 스캔하여 경고 메시지 출력한다. Iframe hidden 속성 확인, Image 링크, From 태그

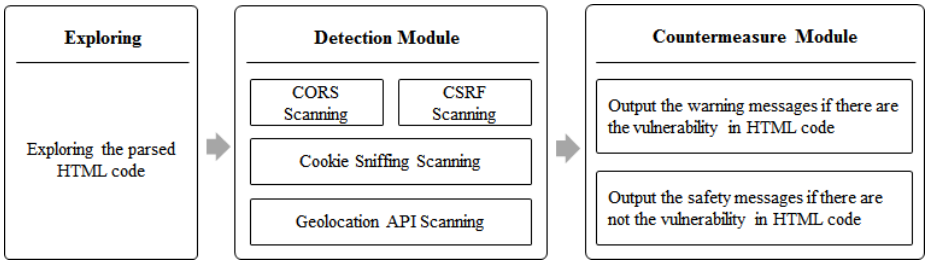

Fig. 2. Module of detection and countermeasures against the attack of HTML5 API and script vulnerability

csrf_token 속성 확인을 통해 CSRF 관련 악성 스크립트가 삽입되어 있는지 확인한다.

Cookie Sniffing 공격의 경우 현재 방문 중인 웹 페이지의 HTML 소스코드에서 Cookie 값에 접

Pseudo code for detection algorithm

```
Input  : HTML source of Web page
Output : result ∈ {abnormal, normal}

// CORS Attack
if withCredentials() = true then
     result = abnormal
     print "Not safe: this browser is CORS use"
else if
     result = normal
     print "Safe: this browser is CORS not use"
end if

// CSRF Attack
if from iframe-tag to /iframe-tag command ∃ .php || .asp || .html || visibility:hidden = true then
     result = abnormal
     print "iframe tag : not safe"

     if from img src-tag to /img src-tag command ∃ .gif || .jpg || .bmp then
          result = abnormal
          print "img tag : not safe"

          if from form-tag to /form-tag command ∃ csrf.token || token then
               result = abnormal
               print "form tag : not safe"
          else if
               result = normal
               print "form tag : safe"
     else if
          result = normal
          print "img tag : safe"
else if
     result = normal
     print "iframe tag : safe"
end if

// Cookie Sniffing
if document.cookie() = true and  XMLHttpRequest() = true and
  post() = true || get() = true then
     result = abnormal
     print "Cookie Sniffing : not safe"
else if
     result = normal
     print "Cookie Sniffing : safe"
end if

// Geolocation API Attack
if getCurrentPosition() = true || watchPosition() = true || cords = true || maximumAge = true then
     result = abnormal
     print "exposure-of-Real-time-GPS-information : not safe"
else if
     result = normal
     print "exposure-of-Real-time-GPS-information : safe"
end if
```

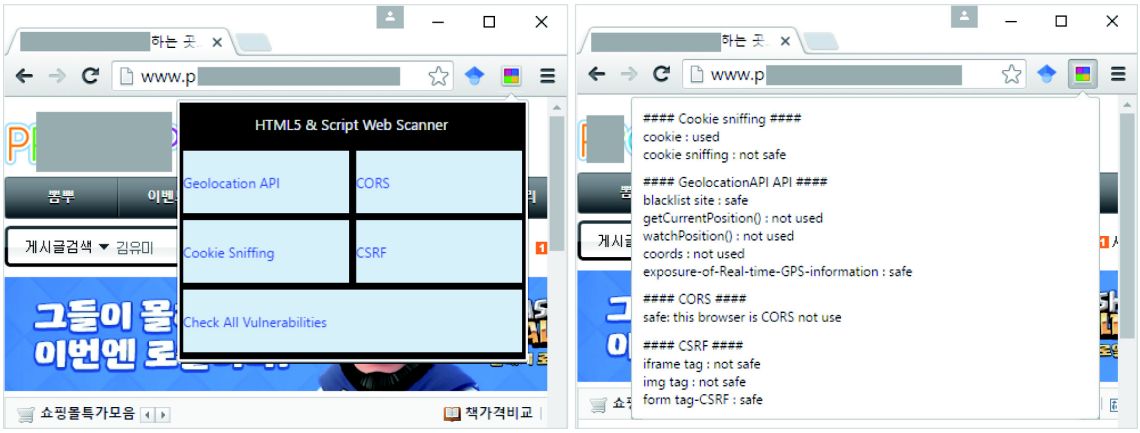

Fig. 3. Web scanner testing for HTML5 and script attack

근 여부를 검색한다. Cookie에 접근하면서 Get, Post 등을 통해서 보내는 소스가 존재하는지 확인하고, 존재할 경우 경고 메시지를 출력한다.

CORS를 사용하기 위해서는 도메인간의 신뢰관계가 형성이 되어야하는데 XHR Level2를 사용한 Request에는 COR을 발생시킨 도메인의 정보를 포함하는 Origin 헤더가 포함되어 있으며, 해당 Request의 Response에는 Access-Control-Allow-Origin 헤더를 포함하여 허용 여부를 확인하게 한다. 쿠키정보가 포함된 COR에 대한 응답은 Access-Control-Allow-Credentials가 true인 정보가 포함되어야 한다. 공격자는 CORS를 사용하기 위해 맺은 신뢰관계를 이용하여 타겟 서버에 공격을 수행한다. Access-Control-Allow-Origin의 값을 all 로 허용할 경우 모든 도메인의 자원 공유 요청을 허용하는 것임으로 개인정보 탈취가 가능하다. 현재 방문 중인 웹 페이지의 HTML 소스코드에서 CORS 기능을 사용하는지에 대해 검사하고, CORS에 대해 태그 범위 내 withCredentials 속성(옵션, 쿠키 값을 남기기 위해 사용하는 기능)을 이용하여 웹 페이지 탐색 결과를 사용자에게 알려준다. withCredentials가 탐색 되었을 경우 확실히 CORS가 사용되었음을 알 수 있다.

특정 목적을 위하여 공개가 허용되지 않은 상태일 때 웹 브라우저 오픈 시 사용자 동의 없이 GPS 정보가 노출되는지 확인하여 경고 메시지를 출력한다. GPS Device와 인터넷 연결이 되지 않을 경우 캐시된 과거 위치정보가 사용되어 웹 브라우저 사용자의 정보가 노출될 수 있다. 현재 방문 중인 웹페이지의 HTML 소스코드를 검색하여 위치 정보를 취득할 수 있는 함수 getCurrentPosition(), watchPosition(), cords, maximumAge가 존재하는지 검색하고, 피싱, 도박 사이트, 적대국 관련 사이트 등에 Access가 되었을 경우 경고 메시지를 출력한다.

기존 일부 상용화 제품의 경우 관리자 측면에서 웹 브라우저의 취약점을 탐지 하다 보니, 많은 탐지 기능을 포함하게 되어 탐지 시간이 길어지는 경우가 있다. 탐지를 수행하기 전 각 웹 사이트 개별로 정보를 기입한 후 탐지를 시작하므로 수행하는데 있어 번거로움이 존재한다. 또한, 웹 사이트 운영자가 악의를 가지고 웹 페이지를 만들거나, 의도하지 않게 웹 페이지가 공격에 노출된 경우라면 웹 사이트의 신뢰성이 훼손된 상태가 되고, 이런 경우 사용자 측면에서의 탐지가 필시 필요하다. 본 논문에서 제안한 탐지 모듈은 운영자 측면에서 관리가 어려운 웹 사이트의 취약점을 사용자 측면에서 수시로 수행 가능하도록 한 것이며, 규칙 기반의 알고리즘이기 때문에 새로운 취약점이 발견되었을 경우 규칙 추가가 용이하다. Fig.3.은 실제 서비스 중인 웹 사이트를 대상으로 본 논문에서 제안한 탐지 모듈을 수행한 결과이다.

V. 결 론

HTML5는 모바일 장치, 자동차 IVI (in-vehicle infotainment) 플랫폼, 가전제품 등 브라우저가 존재하는 모든 장치에서 사용이 가능하며, 이로 인한 공격 범위 증가는 공격자의 개인정보 침해 공격을 더 용이하게 만들 것이다. 본 연구는 스크립트 기반 사이버 공격 및 HTML5 기능 악용 스

크립트 중 개인정보 침해가 발생할 수 있는 공격에 한정하여 클라이언트 기반으로 진단 가능한 플러그인 타입의 탐지 모듈 구현에 관한 것이다. 이렇게 구현된 진단 스캐너를 이용하여 HTML5 기반 웹 어플리케이션이나 웹 페이지의 개인정보 취약성에 대한 신뢰도를 자가 진단할 수 있으며, HTML 소스 확인 후에 웹 서비스를 이용하므로 HTML5의 취약점 공격에 대해 사전대응이 가능하다. 또한, 추가적인 취약점 발생 시 제한 없이 탐지 기능을 추가할 수 있어 확장성이 용이하다. 다만, 규칙 기반의 탐지 방법을 적용했기 때문에, 공개되지 않은 취약점에 의한 공격일 경우, 탐지가 어려울 수 있다는 제약 사항이 존재한다.

References

[1] "Plans for implementing the dissemination of HTML5," Korea Communications Commission, July. 2012, http://www.kcc.go.kr/user.do?mode=view&page=A05030000&dc=K00000001&boardId=1113&boardSeq=34309

[2] "A vocabulary and associated APIs for HTML and XHTML, W3C," W3C Recommendation, October. 2014, http://www.w3.org/TR/html5

[3] "Security issues in environment of the New HTML5 Web services," Korea Internet & Security Agency, Internet & Security Focus KISA Report, December. 2013, http://www.kisa.or.kr/public/library/IS_View.jsp?mode=view&p_No=158&b_No=158&d_No=117&cPage=19&ST=T&SV=

[4] Ministry of government administration and home affairs, "Exposure Guideline for Web pages personal information," October. 2014, http://www.privacy.go.kr/nns/ntc/selectBoardArticle.do?nttId=5952

[5] OWASP Top 10 Privacy Risks Project, "https://www.owasp.org/images/0/0a/OWASP_Top_10_Privacy_Countermeasures_v1.0.pdf ," April. 2016.

[6] "PERSONAL INFORMATION PROTECTION ACT," Korea Ministry of Government Legislation, http://www.law.go.kr/lsInfoP.do?lsiSeq=142563&chrClsCd=010203&urlMode=engLsInfoR&viewCls=engLsInfoR#0000

[7] West, William, and S.M. Pulimood, "Analysis of privacy and security in HTML5 web storage," Journal of Computing Sciences in Colleges vol.27, no.3, pp.80 - 87, 2012.

[8] H. Kim, S. Lee and J. Kim, "Exploring and mitigating privacy threats of HTML5 geolocation API," Proceedings of the 30th Annual Computer Security Applications Conference. ACM, pp. 306 - 315, December. 2014.

[9] G. Dong, Y. Zhang, X. Wang, P. Wang and L. Liu, "Detecting cross site scripting vulnerabilities introduced by HTML5," In Computer Science and Software Engineering (JCSSE), 11th International Joint Conference on IEEE, pp. 319 - 323, May. 2014.

[10] D. Bates, A. Barth, and C. Jackson, "Regular expressions considered harmful in client-side XSS filters," In Proceedings of the 19th international conference on World wide web. ACM, pp. 91 - 100, April. 2010.

[11] D. Gol and N. Shah, "Web Application security tool to identify the different Vulnerabilities using RUP model," International Journal of Emerging Trends in Electrical and Electronics (IJETEE), vol.11, no.2, June. 2015.

[12] H.L. Choo, S. Oh, J. Jung and H. Kim, "The Behavior-Based Analysis Techniques for HTML5 Malicious features," In Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2015 9th International Conference on. IEEE, pp. 436 - 440, July. 2015.

[13] CAPEC, Common Attack Pattern

Enumeration and Classification, http://capec.mitre.org/index.html
[14] CWE, Common Weakness Enumeration, https://cwe.mitre.org
[15] K. Tsipenyuk, B. Chess and G. McGraw, "Seven pernicious kingdoms: A taxonomy of software security errors," IEEE Security & Privacy, vol.3, no.6, pp.81-84, 2005.

〈저자소개〉

한 미 란 (Mee Lan Han) 학생회원
2002년 2월: 동덕여자대학교 컴퓨터공학 학사 졸업
2004년 5월~2012년 3월: NEXON 해외사업 개발본부
2014년 8월: 고려대학교 정보보호대학원 석사 졸업
2014년 9월~현재: 고려대학교 정보보호대학원 박사과정
〈관심분야〉 사이버 범죄 프로파일링, 데이터마이닝, 온라인게임 보안, 자동차 보안

곽 병 일 (Byung Il Kwak) 학생회원
2013년 2월: 세종대학교 컴퓨터공학과 졸업
2013년 9월~현재: 고려대학교 정보보호학과 석·박사통합과정
〈관심분야〉 온라인게임 보안, 데이터 마이닝, 네트워크 보안, IoT 보안

김 환 국 (Hwan Kuk Kim) 일반회원
2000년 8월: 한국항공대학교 컴퓨터공학과 석사 졸업
2001년~2006년: 한국전자통신연구원 정보보호연구단 연구원
2009년 3월~2011년 2월: 고려대학교 정보보호대학원 박사과정 수료
2007년~현재: 한국인터넷진흥원 보안기술R&D2팀 책임연구원
〈관심분야〉 정보보호, 클라우드 보안, 정보보호 관리체계 인증체계

김 휘 강 (Huy Kang Kim), 종신회원
1998년 2월: KAIST 산업경영학과 학사 졸업
2000년 2월: KAIST 산업공학과 석사 졸업
2009년 2월: KAIST 산업및시스템공학과 박사 졸업
2004년 5월~2010년 2월: 엔씨소프트 정보보안실장, Technical Director
2010년 3월~2014년 12월: 고려대학교 정보보호대학원 조교수
2015년 1월~현재: 고려대학교 정보보호대학원 부교수
〈관심분야〉 온라인게임 보안, 네트워크 보안, 네트워크 포렌식