정 보 보 호 학 회 지
제25권 제5호, 2015. 10

HTML5를 이용한 자바스크립트 기반 공격 기법 연구

윤 수 진*, 정 종 훈**, 김 환 국***

요 약

HTML5가 발표되고, 웹상에서 HTML5 신요소와 자바스크립트를 이용한 다양한 기능 구현이 가능하게 되었다. 기존 웹에서는 외부 플러그인을 이용하여야 가능했던 기능들이 웹의 자체 기능으로 구현이 가능해졌다. 하지만 이로 인해 기존에 플러그인인 ActiveX, Flash를 이용했던 공격 대신 HTML5와 자바스크립트를 이용한 공격이 주목 받고 있다. 본 논문에서는 HTML5와 자바스크립트를 이용한 공격들을 살피고, 공격의 파급력과 대책의 중요성을 확인한다.

Ⅰ. 서 론

HTML은 웹 페이지를 작성하기 위한 언어로, 대부분의 웹 페이지가 HTML을 이용하여 작성된다. 웹 사용은 점점 증가하였고, HTML의 사용도 증가했다. 그러나 1997년 HTML4 이후 xHTML 등이 등장하였으나, 기능 확장은 매우 더디었다. 오랜 시간 HTML은 발전이 없었으나, 웹의 사용량 증가에 따라 사람들은 다양한 기능을 웹에 구현할 필요성이 생겼다. 이를 위해 ActiveX, Flash, SilverLight와 같은 플러그인으로 동적인 기능을 구현하였다. 그리고 2009년, 새로운 표준인 HTML5으로 HTML은 크게 기능이 향상된다. 기존 플러그인으로 구현되었던 기능을 HTML5 자체 API와 속성을 이용하여 구현할 수 되었다.

다양한 기능을 추가 플러그인 없이 구현할 수 있다는 장점으로 HTML5는 기존 HTML4 문서를 대체하고 있다. 하지만 이런 다양한 기능을 악용한 공격과 공격 시도들이 증가하고 있다. 아직 국내에서는 2013년 6월 25일에 정부기관을 대상으로 발생한 DDoS 공격 외에는 관련 공격 사례가 없어서 주목받지 못하고 있다.

본 논문에서는 기존에 발표된 HTML5와 자바스크립트 공격을 이용하여, 위험성을 지적한다.

Ⅱ. HTML5

HTML은 웹 문서를 만들 때 쓰이는 프로그래밍 언어이다. HTML5는 HTML의 최신 버전으로 2009년부터 진행되어 2014년 10월 28일에 차세대 웹 표준으로 확정되었다.

HTML5의 주요 특징은 Semantic Tag과 CSS3와 신규 API이다.

Semantic Tag는 기존에 웹 문서 내에 콘텐츠에 대한 정보를 Tag에 명시할 수 있도록 제공되는 Tag이다. 기존에는 식으로 웹 프로그래머가 해당 콘텐츠가 그림임을 표현했다면, HTML5에서는 <figure>라는 태그를 써서 해당 콘텐츠가 그림임을 문서에 표시할 수 있다. <figure>외에 <menu>, <nav> 등과 같은 Semantic Tag가 추가되었다.

CSS3(Cascading Style Sheet 3)는 주로 웹 페이지의 시각적 구성을 제공하는 언어이다. CSS2보다 미디어 매체에 대한 기능 지원이나 Animation같은 동적인 구성을 지원한다.

신규 API는 주로 자바스크립트(Javascript)로 제공된다. 자바스크립트(JavaScript)는 객체 기반 스크립트 프로그래밍 언어로, 웹 사이트에 동적인 기능을 수행하는데 쓰인다. 현재 다양한 API가 권고안으로 확정되거나

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 정보통신·방송 연구개발사업의 일환으로 수행하였음 [B0101-15-0230, 스크립트 기반 사이버 공격 사전 예방 및 대응 기술 개발]
* 한국인터넷진흥원 모바일보안기술개발팀 (sjyoon@kisa.or.kr)
** 한국인터넷진흥원 모바일보안기술개발팀 (jjh2640@kisa.or.kr)
*** 한국인터넷진흥원 모바일보안기술개발팀 (rinyfeel@kisa.or.kr)

〔표 1〕 HTML5를 이용한 공격

유형	공격
악성 스크립트 삽입	HTML5 신규 태그를 이용한 악성 스크립트 삽입
	CSS를 이용한 악성 스크립트 삽입
	SVG를 이용한 악성 자바스크립트 삽입
정보 유출	ClickJacking
	WebStorage 유출
	WebSocket Hijacking
	WebSocket 데이터 탈취
	SVG를 이용한 Keylogger
	네트워크 정보 유출
	위치 정보 유출
정보 조작	WebStorage 조작
	WebSQL 삽입
	WebCache 조작
	History 조작
요청 위조	CORS를 이용한 CSRF
	WebSocket Hijacking
DoS 공격	WebStorage Fill up
	WebWorker Botnet
	Client DoS

진행 중이다. 대표적인 신규 API는 WebWorker, WebSocket, IndexedDB 등이 있다. 각 기능은 공격에서 언급될 때 살펴보도록 한다.

Ⅲ. HTML5를 이용한 공격

HTML5를 이용한 공격은 HTML5의 신규 태그 뿐 아니라 자바스크립트를 이용한 API를 통해 이루어진다.

3.1. 공격 개요

HTML5를 이용한 공격은 주로 정상 웹 사이트에 악성 자바스크립트를 삽입하거나, 아니면 아예 악의적인 사이트를 구축한다. 그 뒤 사용자가 웹 사이트에 접근하면 악의적인 기능이 웹 페이지에서 실행되면서 공격이 시작된다.

3.2. 악성 자바스크립트 삽입

악성 자바스크립트 삽입은 XSS(Cross Site Scripting)이라고 불린다. 정상 사이트에 취약점을 이용하여 웹 페이지에 공격자가 악성 자바스크립트를 삽입할 수 있다. 여기서는 모든 악성 스크립트 삽입을 다루지 않고, HTML5의 요소를 악용한 악성 스크립트 삽입만을 다룬다.

3.2.1. HTML5 신규 태그를 이용한 악성 자바스크립트 삽입[1]

HTML5 신규 태그인 <video>, <audio> 등을 이용하여 태그의 속성에 자바스크립트를 삽입한다.

```
〈video poster=javascript:alert(1)//〉〈/video〉
```
〔1〕

위의 태그는 태그 <video>의 poster 속성에 자바스크립트를 삽입하여 실행한 예다. poster 속성은 비디오 재생 전에 플레이어 화면에 미리 보여줄 이미지를 가리킨다. 위와 같은 악성 스크립트 삽입 방법은 웹 브라우저별로 작동하는 것과 작동하지 않는 것이 있다. 더 많은 삽입 방법은 지면상 다루지 않는다.

3.2.2. CSS를 이용한 악성 자바스크립트 삽입[1]

CSS에서는 웹 페이지의 시각적 표현을 위한 많은 요소들이 있는데, 그 중 링크를 가리키는 -o-link, -o-link-source 등의 요소가 악성 자바스크립트 삽입에 쓰인다.

〈a style="-o-link:'javascript:alert(1)';-o-link-source:current"〉X〈/a〉

위와 같이 -o-link에 자바스크립트를 삽입하면 웹 페이지 로딩과 함께 실행된다. 이 역시 신규 태그를 이용한 것과 마찬가지로 웹 브라우저에 따라 실행되지 않기도 한다.

3.2.3. SVG를 이용한 악성 자바스크립트 삽입[1]

SVG는 이미지를 벡터로 표현한 이미지 포맷으로, HTML5에서는 <svg>라는 태그로 SVG 이미지를 지원한다. <svg>내에 이미지를 표현할 수 있도록 태그들이 정의되어 있다.

SVG는 이미지 파일로도 저장이 가능하다. SVG 이미지와 <svg> 태그 내에는 자바 스크립트를 쓸 수 있게 <script> 태그를 안에 넣으면 웹 페이지에서 SVG 이미지 구현과 함께 <script> 태그 내의 자바스크립트도 같이 실행된다.[1,2]

3.3. 피해 유형에 따른 HTML5를 이용한 공격

다음은 기존에 발표된 HTML5를 이용한 공격을 피해 유형에 따라 나눈 것이다.

3.3.1. 정보 유출

HTML5에서 다양한 기능을 제공하면서, 이에 따라 웹 페이지에서 처리하는 정보가 많아졌다. 또한 WebSocket, XMLHttpRequest 등을 이용한 외부와의 통신 기능이 제공되면서, 정보 전송도 쉬워졌다. 정보 유출은 사용자의 정보를 수집하여 외부로 보내는 공격을 가리킨다.

3.3.1.1. ClickJacking

ClickJacking은 웹 페이지내에 사용자가 혼동할 수 있는 요소를 삽입하여, 공격자가 의도한 행동을 유도하는 공격이다. CSS 3에서부터 개체의 opacity(불투명도)를 이용할 수 있어, 투명한 프레임을 덧씌워서 사용자를 혼란시키는 식으로 공격한다.[3,4] 주로 투명한 프레임 위에 다른 링크를 추가하여 사용자를 악의적인 사이트로 유인하거나, Drag-Drop 속성과 File API를 이용하여 사용자의 파일을 가로챈다.

Drag-Drop 속성과 File API를 이용하는 경우, 사용자는 자신의 파일을 Drag하여 특정 장소에 Drop한다. 이렇게 하면 특정 장소에 정의된 Drop 옵션은 파일을 인식하고 외부로 보낼 수 있다. 이를 이용한 공격은 사용자가 Drag-Drop을 통해서 자신의 파일을 올리고자 했던 위치에 투명한 프레임을 올려두어 사용자의 파일을 가로채는 방법이다.[2,5] 그림 1과 같이 원래 업로드하는 위치에 공격자의 투명한 프레임을 겹쳐두었다. 만약 사용자가 투명한 프레임에 파일을 올리게 되면 파일은 정상 사이트가 아닌 공격자에게 전송된다.

〔그림 1〕 Clickjacking 예시 화면. 투명 처리된 iframe에 파일을 올리면 공격자에게 파일이 전송됨

3.3.1.2. WebStorage 유출

WebStorage 유출은 HTML5 신규 기능인 WebStorage내 정보를 가져가는 공격이다.[3,4] WebStorage API는 각 도메인별로 Key-Value 쌍 형태의 정보를 브라우저에 저장/사용하는 기능이다. WebStorage는 저장/사용되는 공간을 가리킨다. WebStorage는 지속되어 저장되는 LocalStorage와 당시 세션에만 유효한 SessionStorage로 이루어진다. 여기서는 LocalStorage의 정보를 유출하는 것을 가리킨다. LocalStorage의 정보는 한 번 저장되면 해당 사용자의 브라우저에 계속 남아있다. 만약 이 LocalStorage에 중요한 정보를 저장한 상태에서, 악성 스크립트가 삽입

되면 해당 정보를 빼내서 다른 곳에 보낼 수 있다. 그 탓에 LocalStorage에 계정 정보와 같은 예민한 정보는 저장하지 않도록 권하고, 저장하게 될 경우 암호화를 하도록 권유한다.

3.3.1.3. WebSocket Hijacking

WebSocket Hijacking은 사용자 모르게 WebSocket 연결을 맺는 것을 말한다.[4,6] WebSocket API은 HTML5에서 추가된 API로 웹 브라우저에서 WebSocket 서버와 연결을 맺는 기능이다. WebSocket은 웹 페이지에서 Socket 통신과 같이 정보를 주고 받을 수 있다. WebSocket Hijacking은 WebSocket 서버에서 적절한 인증 절차를 거치지 않을 경우, 임의로 연결을 맺을 수 있는 문제점을 악용한 공격이다.

3.3.1.4. WebSocket 데이터 탈취

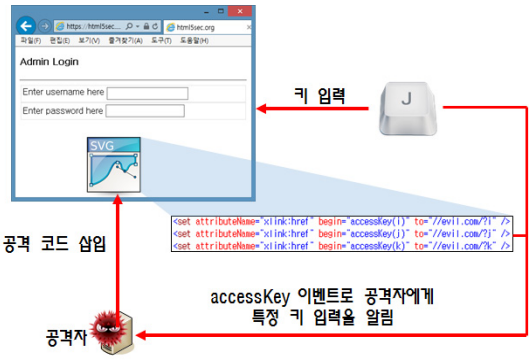
WebSocket 데이터 탈취는 기존에 맺어진 WebSocket 통신 내용을 탈취하기 위하여 WebSocket의 onmessage 핸들러를 이중으로 정의하는 공격이다.[4] onmessage 핸들러는 WebSocket으로부터 데이터를 받는 이벤트를 관리하는 핸들러이다. 공격자는 onmessage 핸들러를 이중으로 정의하여 정상 사이트의 핸들러 기능을 무력화하고, 공격자가 정의한 핸들러를 통해 정보를 빼낼 수 있다.

3.3.1.5. SVG를 이용한 Keylogger 공격

SVG를 이용한 Keylogger 공격은 SVG의 accessKey 이벤트를 이용하여 사용자의 키입력을 감지하는 공격이다.[7-9]

〈set attributeName="xlink:href" begin="accessKey(a)" to="//evil.com/?a" /〉

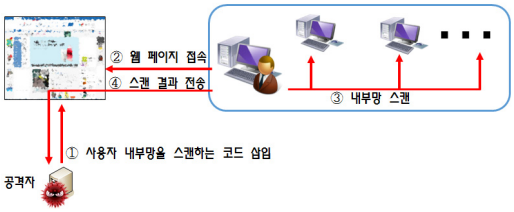
앞의 태그는 SVG 이미지 내에서 accessKey 이벤트가 추가된 태그이다. accessKey(a)는 a키가 눌렸을 때, xlink:href 행위가 일어나는 걸 가리킨다. xlink:href는 화면 전환 없이 링크에 접속하는 것으로 to에 있는 URL에 접속한다. a키가 눌리면 accessKey 이벤트가 작동하여, evil.com/?a에 접속 정보를 남기게 되고, 공격자는 자신의 사이트의 접속 정보에 따라 사용자가 입력한 키를 추측할 수 있다. 위의 태그를 각각 키에 대응하게 만들어주면 대부분의 키입력을 확인할 수 있다.



〔그림 2〕 SVG를 이용한 Keylogger 공격

3.3.1.6. 네트워크 정보 유출

네트워크 정보 유출은 WebSocket, XMLHttpRequest를 이용하여 사용자의 내부 네트워크의 IP, Port가 사용 가능한지 확인하는 공격이다.[2,10] 사용자가 웹 페이지에 접근하면, 웹 페이지는 내부 네트워크에 WebSocket, XMLHttpRequest를 이용하여 통신을 시도한다. 이 때 응답이 있으면 해당 포트와 IP가 사용 가능하다고 판단한다. 네트워크 정보 유출은 사용자의 웹 브라우저에서 일어나기 때문에 외부에서 내부 네트워크를 탐색하는 것과 달리 내부에서 내부 네트워크를 탐색하는 것과 마찬가지이다. 내부에서 일어나는 접속 시도이기 때문에 이를 악의적으로 판단할 가능성이 낮다. 이렇게 수집된 네트워크 정보를 외부에 보내면, 향후 다른 공격에 악용될 수 있다.



〔그림 3〕 네트워크 정보 유출

3.3.1.7. 위치 정보 유출

위치 정보 유출은 HTML5에서 추가된 Geolocation API를 이용하여, 사용자 단말의 위치 정보를 유출하는 공격이다.[2] 사용자 단말이 이동 단말일 경우, 위치 정보는 더 민감한 정보가 된다. 웹 브라우저에서 사용자에게 Geolocation API 사용 허가를 요청하나, 한 번 허가가 된 이후로는 사용 허가를 묻지 않으므로, 위치 정보

를 사용하는 정상 사이트에 위치 정보 유출하는 악성 스크립트가 삽입되면 지속적인 유출로 이어질 수 있다.

3.3.2. 정보 조작

정보 조작은 사용자의 웹 브라우저 내 정보를 조작하여 공격자가 의도한 행동을 하게 만드는 공격이다.

3.3.2.1. WebStorage 조작

WebStorage 조작은 WebStorage에 저장된 정보를 조작하는 공격이다.[3] WebStorage 내 정보가 웹 페이지에서 쓰일 때, 해당 정보를 스크립트 삽입 등을 이용하여 변경하여 웹 페이지에서 악의적인 행동을 하도록 한다.

3.3.2.2. WebSQL 삽입

WebSQL 삽입은 기존의 SQL 삽입 공격을 WebSQL에 적용한 것이다.[3] WebSQL은 HTML5에서 웹 페이지에서 DB 형태로 정보를 저장하기 위한 안으로 제안되었는데, 현재는 표준 등록이 거절된 상태이다. 몇몇 브라우저에서 제안서의 내용을 실제로 구현하여 제공하고 있으나, WebSQL 대신 IndexedDB가 표준으로 채택됨으로써 실제 공격에 쓰일 가능성은 낮아졌다.[11] 대신 IndexedDB를 조작하는 공격이 출현할 가능성이 높다.

3.3.2.3. WebCache 조작

WebCache 조작은 WebCache 내용을 공격자가 유도한 행위를 하도록 조작하는 공격이다.[2,4,12] WebCache는 오프라인일 때를 대비하여 미리 저장하는 리소스를 가리키는데, 만약 이 WebCache가 조작된 상태에서 오프라인이 되면 해당 웹 페이지는 조작된 WebCache 리소스를 사용하게 되고, 악의적인 행위를 할 수 있다.

3.3.2.4. History 조작

History 조작은 웹 브라우저의 웹 페이지 탐색 정보를 제어하는 History API를 악용한 공격이다.[2] History API는 웹 브라우저에서 웹 페이지 탐색을 한 기록을 제어할 수 있다. 웹 브라우저에서 제공하는 이전 페이지나 다음 페이지의 기록을 삭제하거나, 추가할 수 있다. 이를 이용하여 이전 페이지에 굉장히 많은 수를 추가하여 이전 페이지로 가기를 이용해서는 현재 페이지나 특정 페이지를 못 벗어나도록 할 수 있다. 단, 이는 현재 접속 중인 웹 페이지와 같은 도메인 내에서만 가능하기 때문에 실제 공격의 파급력은 낮다.

3.3.3. 요청 위조

요청 위조(Request Forgery)는 사용자가 의도하지 않은 요청을 보내게 함으로써 악의적인 행동을 하게 만드는 공격이다.

3.3.3.1. CORS를 이용한 CSRF

CORS(Cross Origin Resource Sharing)를 이용한 CSRF(Cross Site Request Forgery)는 서로 다른 출처로 리소스를 교환하는 CORS 상태를 이용하여 외부로 거짓된 메시지를 보내는 공격이다.[2,3] HTML5 이전에는 CORS가 불가능하였으나, HTML5 도입과 함께 CORS가 가능해졌다.

http://www.example.com/index.php?page=http://www.attacker.com/exploit.php

http://www.example.com/#http://www.attacker.com/stealDetails.php

위의 예시와 같이 example.com이 아닌 attacker.com으로 요청을 보내게 할 수 있다.[12] 만약 CORS를 막고 싶으면 웹 사이트에서 외부 Origin으로부터의 호출을 막아야하나, 그렇지 않은 웹 사이트는 CORS를 이용한 CSRF 공격에 취약하다.

또한 iframe을 이용하여 공격자가 자신의 웹 페이지를 삽입할 경우, contentWindow.postMessage를 이용하여 iframe 내부와 외부가 정보 교환이 가능하다. Origin 확인을 하지 않을 경우엔 이런 웹 메시징을 이용하여 요청 위조가 발생할 수 있다.

3.3.3.2 WebSocket Hijacking

WebSocket Hijacking은 요청 위조에도 해당하나 정보 유출에서 이미 다루었기 때문에 설명을 생략한다.

3.3.4. DoS 공격

DoS(Denial Of Service)는 웹 페이지의 서비스를 사

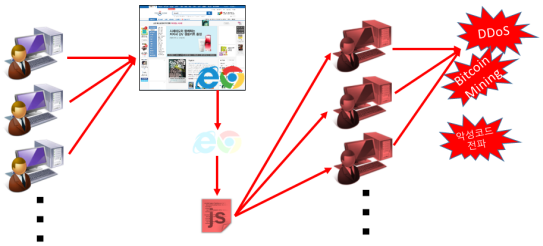
용할 수 없는 상태를 가리킨다. 이러한 상태를 유도하는 경우를 DoS 공격이라 한다.

3.3.4.1. WebStorage Fill up

WebStorage Fill up 공격은 WebStorage에 반복적으로 리소스를 넣어서 웹 브라우저에서 할당한 WebStorage를 가득 채워서 해당 웹 브라우저를 사용하지 못 하도록 만드는 공격이다.[14] 현재에는 각 브라우저 별로 패치가 되어서 불가능한 공격이다.

3.3.4.2. WebWorker Botnet

WebWorker Botnet은 다중 쓰레드를 지원하기 위한 WebWorker를 이용하여 웹 사이트에 접속한 사용자들을 마치 Bot처럼 사용하는 공격이다.[2,13] WebWorker는 기존에 웹 사이트에서 자바 스크립트를 한 쓰레드로 밖에 사용하지 못 하던 문제점을 해결하기 위하여, WebWorker를 이용하여 자바 스크립트를 여러 쓰레드로 만들어서 돌릴 수 있게 하였다. 부담이 큰 연산을 처리하기 위한 기능으로, WebWorker의 자바 스크립트는 웹 페이지 내의 리소스에 접근하지 못 하도록 하는 보안 조치를 취하였다. 그러나 WebWorker 내에서 외부로 요청을 보내는 것에 대해서는 제약이 없기 때문에, WebWorker를 이용하여 외부에 DDoS 공격, 스패밍, 비트코인 채굴 등에 악용될 수 있다.



〔그림 4〕 WebWorker Botnet

3.3.4.3. Client DoS

Client DoS는 사용자 입장에서 웹 페이지의 사용을 저해하는 공격이다.[1] repeat, pattern 등의 요소를 이용하여 웹 브라우저에 부하를 준다. 그러나 이 공격은 웹 브라우저에 따라 불가능하기도 하다.

<input onblur=focus() autofocus>

위의 태그는 입력 값을 받기 위한 태그인데, 속성인 onblur는 해당 태그가 선택되지 않았을 때 취하는 행동을 가리키는 값이다. 이 때, focus()는 해당 태그가 선택을 받도록 하는 API이다. autofocus는 처음에 input 태그가 선택되어 있도록 하는 속성이다. 위의 input 태그가 웹 페이지에 있으면 사용자는 위의 input 태그 외에는 어떠한 것도 선택할 수 없게 된다. 위의 input 태그 외에 다른 값을 선택하면 다시 위의 input 태그로 focus가 이동하기 때문이다. 위의 공격은 client DoS의 예로 웹 사이트 자체에는 부하를 주지 않으나, 실제 사용자 입장에서 웹 사이트의 사용을 불가능하게 만든다.

Ⅳ. 결 론

본 논문에서는 HTML5를 이용한 공격들에 대해 살펴보았다. 대부분의 공격은 웹 사이트 사용을 저해하거나, 사용자의 정보를 유출하는 공격이다. 최근 웹 앱이나 많은 서비스가 사이트로 구현되고 있다. 또한 HTML5를 이용한 모바일 앱 개발도 증가하고 있다. 이런 상황에서 HTML5를 이용한 공격은 다시 한 번 살펴볼 필요가 있다.

아직까지는 웹 서버에 대한 공격에 대한 대비가 대부분이었으나, 이제는 웹 브라우저 상에서 일어나는 공격에 대한 대비가 필요하다. 동적 기능이 제공되는 HTML5를 이용한 공격은 더 증가하고 더 치명적으로 변할 것이다. 향후 현재까지 발표된 공격들을 대상으로 탐지/차단할 방안을 연구할 계획이다.

참 고 문 헌

[1] https://html5sec.org/
[2] TrandLabs, HTML5 OVERVIEW, TrendMICRO, 2011
[3] Shreeraj Shah, HTML5 Top 10 Threats Stealth Attacks and Silent Exploits, Blackhat EU, 2012
[4] KISA, HTML5 개발자를 위한 정보보호 안내서, Dec 2014.
[5] Paul Stone, Next Generation Clickjacking, Blackhat EU, 2010
[6] http://www.christian-schneider.net/CrossSiteWebSocketHijacking.html
[7] https://html5sec.org/keylogger/
[8] Vulnerability Summary for CVE-2011-3663, NVD ,

https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-3663

[9] M. Heiderich, Scriptless attacks Stealing more pie without touching the sill, Journal of Computer Security, p.567-599, July 2014.

[10] andlabs, JS-RECONHTML5 based JavaScript Network Reconnaissance Tool, http://www.andlabs.org/tools/jsrecon.html

[11] http://www.w3.org/TR/IndexedDB/

[12] andlabs, Chrome and Safari users open to stealth HTML5 AppCache attack, http://blog.andlabs.org/2010/06/chrome-and-safari-users-open-to-stealth.html, June 2010.

[13] Kuppan, Attacking with HTML5, Blackhat, 2010

[14] Feross Aboukhadijeh, Introducing the HTML5 Hard Disk FillerTM API,http://feross.org/fill-disk/, Feb 2013.

〈저자소개〉

윤 수 진 (Soojin Yoon)
정회원

2012년 2월 : 고려대학교 정보통신컴퓨터공학부 졸업
2014년 2월 : 고려대학교 정보호호대학원 정보보호학과 석사
2014년 5월~현재 : 한국인터넷진흥원 주임 연구원
관심분야 : 정보보호, 웹 보안, 모바일 보안

정 종 훈 (Jong-Hun Jung)

2003년 2월 : 한국해양대학교 제어컴퓨터공학과 졸업
2010년 2월 : 성균관대학교 정보통신대학원 정보보호학과 석사
2013년~현재 : 한국인터넷진흥원 책임 연구원
관심분야 : 암호학, 정보보호, 개인정보보
호, 모바일 보안, 웹 보안, 클라우드 보안, IoT 보안

김 환 국 (HwanKuk Kim)
정회원

2000년 2월 : 한국항공대학교 컴퓨터공학과 석사
2011년 2월 : 고려대학교 대학원 경정보공학과 박사 수료
2007년~현재 : 한국인터넷진흥원 팀장
관심분야 : 정보보안관리, VoIP 보안, 4G 보안, 네트워크보안