HTML5

Eoin Keary
CTO BCC Risk Advisory

www.bccriskadvisory.com
www.edgescan.com

Magic Quadrant for Global MSSPs

Hype Cycle for Application security 2015

Where are we going?

HTML5

- WebSockets
- **AngularJS**
- **HTML5 Sinks**

WebSockets:

Full duplex communications between client or server to a resource.

A secure version of the WebSocket protocol is implemented in Firefox 6,

Safari 6,

Google Chrome 14,

Opera 12.10

Internet Explorer 10.

```
Request:
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: x3JJHMbDL1EzLkh9GBhXDw==
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13
Origin: http://example.com

Response:
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept: HSmrc0sMlYUkAGmm5OPpG2HaGWk=
Sec-WebSocket-Protocol: chat
```

This helps ensure that the server does not accept connections from non-WebSocket clients – Abuse?

It is the web server's responsibility to verify the *Origin* header in the initial HTTP WebSocket handshake. If the Origin header is not properly checked, the application may be vulnerable to CSRF

Sec-WebSocket-Key header field in the client's handshake were " x3JJHMbDL1EzLkh9GBhXDw== ", the server would append the string "258EAFA5-E914-47DA-95CA-C5AB0DC85B11" to form the string " x3JJHMbDL1EzLkh9GBhXDw== 258EAFA5-E914-47DA-95CA- C5AB0DC85B11".
The server would then take the SHA-1 hash of this and return the value.

RISK ADVISORY

WebSockets

```
if(window.WebSocket) {

        /*browser has websocket support
        var sock = new WebSocket('ws://server:8181');
        …..
        sock.onopen = function(event) {
                /*Open... sock.send() */
        }

        sock.onmessage = function(e) {
                /*Received message */
                e.data();…

        }

        sock.onclose = function(event) { /* Connection closed
        }
```

RISK ADVISORY

ProTip

- You should use the secure wss:// protocol over the insecure ws:// transport.

- Never Tunnel via websockets from the browser to say a database! XSS attacks can attack such connections.

- CSRF and WebSockets

- Process the messages received by the websocket as data.
  - Never try to assign it directly to the DOM nor evaluate as code.
  - If the response is JSON, **never** use the insecure eval() function; use the safe option JSON.parse() instead.

Websockets Authentication

Authentication
WebSockets do not handle authentication, instead normal application authentication mechanisms apply, such as cookies, HTTP Authentication or TLS authentication.

Input Validation
As with any data originating from untrusted sources the data should not be trusted.

Websockets Authentication

- Check the Origin: header in the Websockets handshake. Though it might be spoofed outside a browser, browsers always add the Origin of the page that initiated the Websockets connection.

- Always validate data coming through a WebSockets connection.

RISK ADVISORY

AngularJS Pitfalls

AngularJS uses stuff like:

{{}} are expressions in AngularJS. They are parsed and executed

{{5+5}} <- output is 10

<div ng-init="some.js.code.here">

or

{{constructor.constructor('alert(1)')()}}

AngularJS uses it's own parser; If there's an injection source, no actual antiXSS filter will be able to stop these attacks.

HTML5 Sinks

Formaction:

```
<form id="test"></form><button form="test" formaction="javascript:alert(1)">X</button>
```

```
<form><button formaction="javascript:alert(1)">clickme</button>
```

- Don't allow users to submit markup containing "form" and "formaction" attributes or transform them to bogus attributes. HTML Attribute encoding of user data should prevent injection of formaction

More HTML 5 Elements

<video>:

```
<video vid1=javascript:alert(1)//></video>
<video><src onerror="alert(1)">
```

Oninput:

```
<body oninput=alert(1)><input autofocus>
```

Srcdoc:

```
<html><body><iframe src=“”
srcdoc=<script>alert(123)</script></body>
```

More HTML 5 elements

HTML5 offers the <picture> element.

"srcset" attribute allows to trigger load events:

```
<picture><img srcset="x" onerror="alert(1)"></picture>
```

Other elements which need to be considered

<TABLE>	<FRAMESET>	<BASE>	<OBJECT>	<EMBED>
<TITLE>	[if] (conditional comments)	<LINK>	<STYLE> @import data	

Local Storage

- Also known as “WebStorage” , “DOM Storage”
- Supported by:

HTML5 Storage support

IE	Firefox	Safari	Chrome	Opera	iPhone	Android
8.0+	3.5+	4.0+	4.0+	10.5+	2.0+	2.0+

- Accessed via “LocalStorage” object:

```
var foo = localStorage.getItem("bar"); localStorage.setItem("bar",
foo);
```

Local Storage

- it's recommended not to store any sensitive information in local storage.
  - Malware;
  - Unencrypted storage on local machine
  - Shared computer environment etc etc
- *sessionStorage* instead of localStorage should be considered if persistent storage is not needed.
  - sessionStorage object is available only to that window/tab until the window is closed.

Local Storage

- XSS can steal data similar to session cookie attacks.
- XSS can also load malicious data into local storage.
- getItem() and setItem() calls in Javascript are sources and sinks for localstorage attacks.
- Multiple applications on the **Same Origin** would share the same localstorage…beware. One vulnerability could result in many applications being attacked!!!

iFrame Sandboxing

Sandboxed frames

- Use the sandbox attribute of an iframe for untrusted content. Rendering content based in input from an untrusted source.

- The sandbox attribute of an iframe enables restrictions on content within a iframe.
- The following restrictions are active when the sandbox attribute is set:
  - All markup is treated as being from a unique origin.
  - All forms and scripts are disabled.
  - All links are prevented from targeting other browsing contexts.
  - All features that triggers automatically are blocked.
  - All plugins are disabled.

iFrame Sandboxing

```
<iframe src="demo_iframe_sandbox.htm"
sandbox="<VALUE>"></iframe>
```

- If <VALUE> is specified as an empty string (sandbox=""), the sandbox attribute enables a set of extra restrictions for the content in the inline frame.
- The value of the sandbox attribute can either be an empty string (all the restrictions is applied), or a space-separated list of pre-defined values that will REMOVE particular restrictions.

RISK ADVISORY

iFrame Sandboxing

```
<iframe src="demo_iframe_sandbox.htm"
sandbox=“<VALUE>"></iframe>
```

Value	Description
"“ (empty)	Applies all restrictions below
allow-same-origin	Allows the iframe content to be treated as being from the same origin as the containing document
allow-top-navigation	Allows the iframe content to navigate (load) content from the containing document
allow-forms	Allows form submission
allow-scripts	Allows script execution

RISK ADVISORY

iFrame Sandboxing

- In old versions of user agents where this feature is not supported, this attribute will be ignored. - backward compatable
- iFrame Sanboxing can be used as an additional layer of protection.
- It may be an option to check if the browser supports sandboxed frames and only show the untrusted content if supported.
- Apart from this attribute, to prevent Clickjacking attacks and unsolicited framing it is encouraged to use the header X-Frame-Options which supports the deny and same-origin values.