HTML5 Web Security

Thomas Röthlisberger – IT Security Analyst
thomas.roethlisberger@csnc.ch
What is this talk about?
Agenda

What is HTML5?

Vulnerabilities, Threats & Countermeasures

Conclusion

Demo CORS

Demo Web Workers

Quiz and Q&A
The Voting Device
It enables you to participate on votings

The device has no batteries, so it works autarkic

You power it by shaking it until green light flashes
The Voting

Let’s give it a try...
What is HTML5?
History

- HTML 4.01
- XHTML 1.0
- XHTML 1.1

- XHTML 2.0

- WHATWG

- Web Applications 1.0

- HTML5
HTML5 is not finished!

The specification achieved CANDIDATE RECOMMENDATION status on 17 December 2012. Newest version is from 6 August 2013.

However, it is still a draft version and may be updated.

**THE FUTURE**

According to Google search results

- US Debt reaches 97% of GDP
- US unemployment falls to 2.8%
- Restored caliphate unifies Middle East
- Lake Mead evaporates
- Kilimanjaro snow-free

**2022**
- HTML 5 FINISHED
- Newspapers become obsolete and die out

**2023**
- Jesus returns to Earth (again)
- US Debt passes 100% of GDP
- All unprotected ancient forests gone from Pacific Northwest

(http://xkcd.com/887/)
HTML5 TEST - http://html5test.com/

Your browser scores 503 out of 555 points.

You are using Chrome 31 on Windows 8.
Overview

The Web Sockets API

Custom scheme and content handlers

Cross-Origin Resource Sharing

Web Messaging

Location information

Data for offline use

Javascript threads

Data stored on client

Geolocation API

Offline Web Application

Web Storage

Web Worker

<iframe src="anydomainA.csnc.ch [...]

<iframe src="anydomainB.csnc.ch [...]

<iframe src="untrusted.csnc.ch [...] IFrame Sandboxing

Javascript

© Compass Security AG www.csnc.ch
Vulnerabilities, Threats and Countermeasures \textit{(if any)}
Cross-Origin Resource Sharing

- The Web Sockets API
- Custom scheme and content handlers
- Cross-Origin Resource Sharing
- Web Messaging
- Location information
- Data for offline use
- Javascript threads
- Data stored on client

- Geolocation API
- Offline Web Application
- Web Storage
- Web Worker

<iframe src="anydomainA.csnc.ch [...]"
<iframe src="untrusted.csnc.ch [...]"

Iframe Sandboxing

JavaScript
Cross-Origin Resource Sharing I
GET / HTTP/1.1
Host: domainB.csnc.ch
Origin: http://domainA.csnc.ch

HTTP/1.1 200 OK
Content-Type: text/html
Access-Control-Allow-Origin: http://domainA.csnc.ch
CORS – Vulnerabilities & Threats I

Accessing internal websites

Scanning the internal network
Remote attacking a web server

Easier exploiting of Cross-Site Request Forgery (XSRF)

Establishing a remote shell *(DEMO)*
Countermeasures

Use the **Access-Control-Allow-Origin** header to restrict the allowed domains.

Never set the header to `*`.

Do not base access control on the origin header.

To mitigate DDoS attacks the Web Application Firewall (WAF) needs to block CORS requests if they arrive in a high frequency.
Web Storage

- The Web Sockets API
- Custom scheme and content handlers
- Cross-Origin Resource Sharing
- Web Messaging
- Location information
- Data for offline use
- Javascript threads
- Data store
- Geolocation API
- Offline Web Application
- Web Worker
Web Storage

Local Storage Example

Please insert your String to be saved in Local Storage here ...

Click this link to save the String into the Local Storage of your browser (key=TestValue).
Click this link to load the saved value from the Local Storage of your browser (key=TestValue).
Click this link to delete the saved value from the Local Storage of your browser (key=TestValue).

Value of field TestValue:

Please insert your String to be saved in Local Storage here ...

```
window > history

- globalStorage 0 items in Global Storage
- history 1 history entries
- innerHeight 382
- innerWidth 510
- length 0

- localStorage 1 item in Storage TestValue="Please insert your String to be saved in Local Storage here ..."

location

locationbar

menubar

http://internet.csnc.ch/HelloWorld/localStorageExample/list ( constructor=Location, host="internal.csnc.ch", more... )
BarProp ( constructor=BarProp, visible=true )
BarProp ( constructor=BarProp, visible=true )
```
Web Storage – Vuln. & Threats

Session Hijacking
- If session identifier is stored in local storage, it can be stolen with JavaScript.
- No HTTPOnly flag.

Disclosure of Confidential Data
- If sensitive data is stored in the local storage, it can be stolen with JavaScript.

User Tracking
- Additional possibility to identify a user.

Persistent attack vectors
- Attack vectors can be stored persistently in the victim’s browser.
Countermeasures

Use cookies instead of Local Storage for session handling.

Do not store sensitive data in Local Storage.
Offline Web Application

The Web Sockets API

Custom scheme and content handlers

Cross-Origin Resource Sharing

Web Messaging

Iframe Sandboxing

Location information

Data for offline use

Javascript threads

Geolocation API

Web Storage

Web Worker

Stored on client
<!DOCTYPE HTML>
<html manifest="/cache.manifest">
<body>
...

Example cache.manifest
CACHE MANIFEST
/style.css
/helper.js
/csnc-logo.jpg
NETWORK:
/visitor_counter.jsp
FALLBACK:
/ /offline_Error_Message.html
Cache Poisoning
- Caching of the root directory possible.
- HTTP and HTTPs caching possible.

Persistent attack vectors
- Attack vectors can be stored persistently in the victim’s browser.

User Tracking
- Additional possibility to identify a user.
- Unique identifiers could be stored along with the cached files.
Offline Web Application – Attack 1/2

Unsecured Network

1: Request to any.domain.com
2: GET / HTTP/1.1
3: HTTP/1.1 200 OK
4: Content of any.domain.com with additional hidden
   <iframe src="http://www.filebox [...]"
5: Request to www.filebox-solution.com
6: Faked content with CACHE MANIFEST /
Offline Web Application – Attack 2/2

7: Request to www.filebox-solution.com
   Poisoned content is loaded from browser cache

8: POST /listener.do HTTP/1.1
   username=admin&password=compass.123

9: JavaScript Execution

10: Login request

11: HTTP/1.1 200 OK
   Login successful
User Training

=> Do not accept caching of web applications!

=> Clear the cache including Local Storage and Offline Web Applications!
Web Messaging

- The Web Sockets API
- Custom scheme and content handlers
- Cross-Origin Resource Sharing

- Location information
- Data for offline use
- Javascript threads
- Data stored on client

- Geolocation API
- Offline Web Application
- Web Storage
- Web Worker

- domainA.csnc.ch
- domainB.csnc.ch
- domainC.csnc.ch
Web Messaging

Embedding HTML Page
html5demos.com

postMessage()

<iframe src="jsbin.com" [...]>

Send Message:

```javascript
var win =
document.getElementById("iframe").contentWindow;
win.postMessage(
    document.getElementById("message").value,"http://jsbin.com"
);
```
Web Messaging

Embedding HTML Page
html5demos.com

postMessage()

<IFrame src="jsbin.com" [...]>

Receive Message:

window.onmessage = function(e){
    if ( e.origin !== "http://html5demos.com" ) {
        return;
    }

    document.getElementById("test").innerHTML = e.origin + " said: " + e.data;
};
Web Messaging

Embedding HTML Page
html5demos.com

postMessage()

&lt;IFrame src="jsbin.com" [...]\

Stealing confidential data

 раств. Sensitive data may be sent accidently to a malicious IFrame.

Expands attack surface to the client

 раств. IFrames can send malicious content to other IFrames.
 раств. Input validation on the server is not longer sufficient.
Web Messaging - DEMO

Embedding HTML Page
html5demos.com

postMessage()

<iframe src="jsbin.com" [...]>

postMessage (cross domain)

Message: "(document.cookie);" />

Daten absenden

Die Seite mit der Adresse http://jsbin.com meldet:

_Utma=77969039.1540083565.1385043415.1385043415.1385043415.1;
_Utmb=77969039.3.10.1385043415; __utmcc=77969039;
_Utmz=77969039.1385043415.1.1.utmcsr=html5demos.com|utmccn=(referral)utmcmd=referral|utmctr=0
Countermeasures

The target in postMessage() should be defined explicitly and not set to *.

The receiving IFrame should not accept messages from any domain. E.g. `e.origin == "http://internal.csnc.ch"`

The received message needs to be validated on the client to avoid malicious content being executed.
Custom scheme and content handlers

- The Web Sockets API
- Custom scheme and content handlers
- Cross-Origin Resource Sharing

HTML Overview

- iframe src="anydomainA.csnc.ch" ...
- iframe src="untrusted.csnc.ch" ...

- Web Messaging
- Web Worker
- Geolocation API
- Offline Web Application
- Web Storage

- Location information
- Data for offline use
- Javascript threads

Data stored on client
Custom scheme and content handlers

Stealing confidential data
- An attacker tricks the user to register a malicious website as the e-mail protocol handler.
- Sending e-mails through this web application gives the attacker access to the content of the e-mail.

User Tracking
- Additional possibility to identify a user.
- Unique identifiers could be stored along with the protocol handler.
Countermeasures

User Training

=> Do not accept registration of protocol handlers!
Web Sockets API
Web Sockets API

User Agent

1: GET / HTTP/1.1

1.1: HTTP/1.1 200 OK

2: Upgrade: WebSocket

2.1: HTTP/1.1 101 Web Socket Protocol Handshake

Full-Duplex TCP-Channel established
Web Sockets API – Vuln. & Threats

Cache Poisoning
A misunderstanding proxy could lead to a cache poisoning vulnerability.
\[\text{Fixed} \] by introducing masking of the web socket data frames.

Scanning the internal network
\[\text{The browser of a victim can be used for port scanning of internal networks.}\]

Establishing a remote shell
\[\text{Web Sockets can be used to establish a remote shell to a victim’s browser.}\]
Countermeasures

The risks of the Web Sockets API needs to be accepted.

The user could disable it in the browser.
Geolocation API
Geolocation API

Finding your location: found you!

User Tracking

- User tracking based on the location of a user.
- If users are registered, their physical movement profile could be tracked.
- The anonymity of users could be broken.
User Training

=> Do not accept to share location information!
Web Workers

- The Web Sockets API
- Custom scheme and content handlers
- Cross-Origin Resource Sharing

- Web Messaging
- IFrame Sandboxing

- Location information
- Data for offline use
- Javascript threads

- Data stored on client

- Geolocation API
- Offline Web Application
- Web Storage
Web Workers

Web Workers provide the possibility for JavaScript to run in the background

Prior to Web Workers using JavaScript for long processing jobs was not feasible because
+ it is slower than native code and
+ the browsers freezes till the processing is completed

Web Workers alone are not a security issue.

But they can be used indirectly for launching work intensive attacks without the user noticing it.
Worst Case Scenarios

Web Workers $\p Leftrightarrow$ Feature!

Cracking Hashes in JS Cloud (DEMO).


Conclusion
Some HTML5 features are the vulnerabilities themselves.
Not all issues can be mitigated through secure server-side implementation.
Cross-Site Scripting (XSS) becomes even worse.
USE IE 6

;)
DEMO – Exploiting Cross-Origin Resource Sharing

Shell of the Future
Simplified:

Victim’s hijacked browser

XSS

shell of the future console
DEMO – Exploiting Web Workers

Ravan
DEMO – Web Workers – Ravan

http://www.andlabs.org/tools/ravan.html
DEMO – Web Workers – Ravan

http://www.andlabs.org/tools/ravan.html

14d6a3e0201f58bfe7c01e775973e80e
Quiz and Q&A
DEMO – Web Workers – Ravan

http://www.andlabs.org/tools/ravan.html

HTML5 Web Security Video (May 2011):
http://www.youtube.com/watch?v=Eju4e5mhEN0

Test HTML5 security yourself:
https://www.hacking-lab.com/sh/Gb5VF4q
References

- Master Thesis „HTML 5 web security“
  Michael Schmidt
  31 March 2011

- Article „HTML5 web security“ (extract of master thesis)
  Michael Schmidt, Thomas Röthlisberger
  6 December 2011

- Attack and Defense Labs
  Lavakumar Kuppan
  http://www.andlabs.org

- HTML 5 Demos and Examples
  Remy Sharp (https://twitter.com/rem)
  http://html5demos.com/

- A vocabulary and associated APIs for HTML and XHTML
  W3C, HTML5 specification
  6 August 2013
  http://www.w3.org/TR/html5/