Browser Security Track @ OWASP Summit 2011 (14:00 - 15:20 Wednesday Feb 9)

Site Security Policy

D Ross == David Ross, Microsoft
B Sterne == Brandon Sterne, Mozilla
G Heyes == Gareth Heyes
I Fette == Ian Fette, Google
P Uhley == Peleus Uhley, Adobe
M Heiderich == Mario Heiderich, Uni Bochum
J Hodges == Jeff Hodges, PayPal
T Gondrom == Tobias Gondrom, IETF
L Adamski == Lucas Adamski, Mozilla
J Schuh == Justin Schuh, Google
R Hansen == Robert Hansen, SecTheory

[About this document] These are the raw notes from the Summit discussion on Site Security Policy. All the cited people have had the chance to edit these notes but there may still be errors and misunderstandings in here. Along with the notes from the other browser security sessions – DOM sandboxing, HTML5 Security, and EcmaScript 5 Security – these notes will be the foundation of the forthcoming Browser Security Report. If you have any questions regarding this document, please email john.wilander@owasp.org.

[The session kicks off on the topic of X-Frame-Options, XFO]

D Ross: We're considering a third XFO option specifying which site can frame and host you. The caveat is that some sites want to frame content but not be *detected* doing so.

B Sterne: Currently JavaScript frame busting to disallow framing. It doesn't prevent clickjacking. Framing can be very good but is not the same as clickjacking. Many sites want to be framed but not clickjacked.

D Ross: That's why we want a new option.

G Heyes: Spec HTML element ID in options that is allowed to frame

B Sterne: How does that prevent a site from obfuscating?

G Heyes: Even height and width could be specified in x-frame-options

B Sterne: That's going to be hard to specify for the developer. NoScript's ClearClick is doing best currently. The user experience could be enhanced though.

G Heyes: Tooltip can be used to show where the click goes.

D Ross: The more places the user gets to know what they're doing, the better.

I Fette: Playing the Devil's advocate – the more places to look at the *less* you know where to

look.

D Ross: The URL bar is the trust bar and where security relevant info should be. I don't like trust indicators all over.

J Hodges: User studies say users don't look at anything at all. They have no context to understand this. The UI is orthogonal to policy questions.

T Gondrom: I like XFO except for the 'X'. The third option is good too.

[The session then moved on to discuss Content Security Policy]

L Adamski: Something on CSP. Primary purpose is to mitigate code injection attacks. Second is content injection attacks. It has to be a simple policy language. White list instead of black list. Current CSP doesn't require hooks into your content so it most often works with existing sites. Base XSS vectors where data (strings) become code are ruled out by default. Active content is only allowed in files.

I Fette: How do you mix and match?

B Sterne: You add e.g. Google Analytics domain to your white list.

L Adamski: You can set a policy to only allow scripts to load over TLS. Might look as overlap to HSTS but is really complimentary. Disabling stuff should be explicit, such as "Remove XSS protection" instead of "Allow inline scripts". Tries to regulate framing too but still hard to address clickjacking while still allowing framing.

J Schuh: Outer page with CSP incl directives on what can framed in it. Will a dynamically generated page/iframe inherit its policy?

B Sterne: The frame policy of the outer page is applied to loading that frame but is not inherited.

J Schuh: Having to inject policies into all generated frames gets complicated. And how would you do that? Injected meta tags?

L Adamski: Yes, that might be a good use case for meta tag policies.

L Adamski: You can have the CSP in a file but not both. There's also a reporting mechanism that allows reports being sent back to your domain. www.foo.co.uk can send back to report.foo.co.uk but not report.bar.co.uk.

G Heyes: What if we have a malicious CSP server? Will it be able to get reports sent elsewhere?

B Sterne: No.

I Fette: Why are we enforcing same origin for reports? If we trust the CSP served?

L Adamski: There are concerns on leaking information to places outside your domain.

B Sterne: Adam Barth has had suggestions on reporting mechanism and an alternative DOM event reporting will probably be added.

P Uhley: Should CSP apply to plugins or is it HTML-only?

L Adamski: Maybe looking at "code source", "media source" makes more sense to plugins.

P Uhley: There's risk we apply rules they were not intended for. And should we query the policy URI ourselves or to we ask the page?

T Gondrom: If you can combine image and media that's good. At least standards-wise.

L Adamski: If nobody cares we can combine them. But it might be you want to rule out terrifying codecs but not images.

T Gondrom: Better to start with few. More granularity can always be added but never taken away.

I Fette: Should swfs have their own policies? The plugin doesn't know what additional content will be pulled with the swf.

P Uhley: Plugins are treated as iframes basically.

L Adamski: Script source and object source need to be separate? Maybe overriding the scripts source with an object source and thus go three layers.

J Schuh: CSP and cross-origin rules stack. If either is violated the resource request is denied.

B Sterne: We've gone back and forth on the "more buckets" and "less buckets". I tend to like "more buckets". Sites don't have to use any of them. A very simple policy of allow self will suffice for many.

I Fette: You might not realize you have to set object source after setting script source.

L Adamski: The flip side of having few buckets is that when you need to add a little authority you have to open a large part of your policy.

J Schuh: If you allow a plugin to load it will effectively have all access after that.

T Gondrom: I now see that the white listing motivates more buckets.

R Hansen: Will you for instance address CSS overlays (third party apps should not be able to style the page) or redirects with CSP?

L Adamski: Requests are the finest granularity right now. No policies for individual elements.

M Heiderich: What about attacking the reports? Can we XSS it or attack the databases behind a reporting tool? Will there be OSS libs for providing good reporting software providing all necessary security precautions? What about flooding and hiding actual payload in noisy requests?

L Adamski: Right now we do not enforce any integrity or such on reports.

L Adamski: Should we have one policy mechanism to rule them all or a header per problem? Right now the number of headers is growing. CSP right now doesn't solve clickjacking at all.

[The session ended with a short not on HTTP Strict Transport Security, HSTS, by Jeff]

J Hodges: Can we define a SiteSec directive that covers all these policy initiatives? For instance:

```
SiteSec: allow 'self'; img-src example.org; STS max-age 1200;
```

There are a number of challenges:
- CSP is per request whereas HSTS is cached
- CSP can apply to 3rd party domains whereas HSTS does not

There are some problematic details underneath that we need to address. Yes, PayPal would like to see something simpler but ...