Browser Security Track @ OWASP Summit 2011 (14:00 - 15:15 Thursday Feb 10)

EcmaScript 5 Security

M Heiderich == Mario Heiderich, Uni Bochum
M Samuel == Mike Samuel, Google
J Schuh == Justin Schuh, Google
I Fette == Ian Fette, Google
J Hodges == Jeff Hodges, PayPal
J Nagra == Jasvir Nagra, Google
L Adamski == Lucas Adamski, Mozilla
J Steven == John Steven, Cigital
J Wilander == John Wilander, OWASP

[About this document] These are the raw notes from the Summit discussion on EcmaScript 5 Security. 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 – Site Security Policy, DOM sandboxing, and HTML5 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.

M Heiderich: ECMA Script 5 provides a novel feature allowing to set a final state for a property. Final in terms of: cannot be changed afterwards - he who deploys first will be able to set the final state.

M Samuel: Behavior of getters and setters on host objects not defined in EcmaScript 5. Hopefully browser vendors will implement the expected behavior anyway and we get it specified in EcmaScript 6.

J Nagra: EcmaScript 5 Strict-mode allows for static analysis - the strict subset removes from the language the sharp corners introduced by dynamic scope.

M Samuel: In ES5, if it looks like a local variable it is a local variable. If it looks like a global it is a global.

M Heiderich: XSS has to be fixed on the client. ES5 allows this. What do the browser vendors think?

L Adamski: Sure, lets ask Brendan Eich.

J Schuh: Looks kind of complicated. New layers to fix things are good. But as an XSS silver bullet – that I don't see. The sandboxing property is good. But you're adding necessary complexity to client-side developers.

J Steven: But this isn't sandboxing. We should look at what the Java applet people did wrong back in the days. I'm not seeing secure by default which is what we should aim for in ES5.

L Adamski: Security by default would require a library implementation to do the heavy lifting.

M Samuel: We need the authority of the running script. What it can reference. If it should not see cookies we remove them.

M Heiderich: TechCrunch are including 12-15 foreign scripts. In such a situation you cannot trust the DOM anymore. The document is an important property and you cannot trust it anymore. An attacker can make it look real but be compromised.

J Nagra: Well they could never trust the DOM previously. In the new regime, scripts that load earlier are responsible for deciding what authority subsequent scripts get. Now TechCrunch need to only pass only the authority each script requires.

I Fette: You just passed on a lot of responsibility to developers. If they have a Twitter applet or Facebook “like” button they need to know what to pass on. How will they know?

M Samuel: Well, now they at least have a way to do that.

M Heiderich: Solution is defineProperty

J Nagra: Unfortunately the properties defined by “defineProperty” are not enforceable on host objects like the DOM. The specification allows host objects to refuse to be proxied, to silently ignore freezing and to fail to invoke getters and setters. In practice, browsers may not do this - but they according to the ES5-strict spec, they are within their rights to do so.

M Samuel: Harmony is the name for EcmaScript 6. They want to clean up the language around host objects. There will be some host functions left such as alert() but they might do away with host objects and go for proxying over host functions. The source of undefined behavior is calling and construction of host objects.

J Nagra: The language constructs that strict mode removes mostly coincide with the constructs that have semantics that few developers understand or are aware of. The smaller language has most of the constructs existing developers rely on and fewer of the sharp edges from semantics they either don’t use or do not understand but which attackers can take advantage of.

M Samuel: Many hard to find errors come from confusion around the meaning of `this`. Strict mode cause programs misusing `this` to fail early. `this` will be undefined when you invoke a constructor without `new` of a function’s method that doesn't properly bind. In essence a JavaScript `NullPointerException`.

J Wilander: So developers switching to strict mode are better off and basically get the “Crockford Good Parts” while the attacker doesn’t get to use the bad parts?

M Heiderich: Strict mode usage as security effort has to start with JavaScript libraries and

frameworks.

J Nagra: The Caja input language is very close to ES5 strict mode - in essence it is ES5-strict mode sans eval. When we tried to run JQuery under Caja, most of the tests passed immediately. That gives you a good sense of how close a library like JQuery is to running under strict mode. Unfortunately it isn't necessarily easy to spot the places which need to change - in part because some of failures caused by strict-mode can only be detected at runtime. For developers wanting to get started using strict mode today - a good way to test is to run your program through a linter like JSLint or the Caja linter and then to run under FF4 which today is the browser leading the adoption of ES5-strict mode.

M Samuel: Beware, a lot of sites are assuming you can concat two pieces of JS with a ';' between. ES5 Strict mode is opt-in. If you concat a strict with a non-strict program it won't work. The preamble `"use strict";` should be placed either at the top of the script or of a function (strict mode applied to all nested functions). Each body of the script tag is a "program".

J Hodges: You need to be very clear on the definition of "program".