“Generally, JavaScript codes are obfuscated and dense by software designers who are apprehensive about guarding against intellectual property or even sustaining data bandwidth.”

Hi Albert,

Having done a lot of Perl programming at the workplace, I’ve seen a lot of obfuscated code. I think though, that most of it was written without that intent, at least where I work. Purposefully obfuscating one’s code for the purpose of trying to keep it proprietary is an interesting concept, and one I haven’t heard about until you brought it up. However, if one studied the code, can they merely just reverse-engineer it, obfuscate it in another way, and claim it as their own?

Regards,

Emanuel

Hi Rony,

Thanks for your comments. I think that the pictures you provided aid in the description of these concepts. I’m finding myself applying the same habits from “regular” programming to web programming. Start with the bare minimum and simplest unit first, and make sure that it compiles and does what it was intended to do. Then proceed by adding another unit or functionality, and recompile. I find this preferable to debugging large units of code. Since I’m still kind of getting the hang of web programming, I’m applying a similar style, and using the W3C validator in place of the compiler. I think as I get better at this, I can probably write out bigger portions of code confident that there are no bugs before submitting to the validator.

Regards,

Emanuel

Hi Albert,

Thank you for the additional information on code obfuscation. So it looks like code obfuscation methods such as the one presented by Schrittwieser and Katzenbeisser can help thwart efforts to reverse engineer using automated deobfuscators. Though, I supposed that if one were to be totally committed to the goal, then most likely it will eventually be deciphered. However, if I were really serious about protecting my code, I would probably just put it on the server side. Still, from a risk mitigation perspective, code obfuscation does definitely have a role.

Regards,

Emanuel

“Flash is on the decline even for video sharing sites and full flash sites are a dime and a dozen now because once you implement it your viewers will need the plugin and a lot of persons won’t know how to install it, are prevented from installing it (On some local networks) and Apple devices don’t support it (C, 2013).”

Hi Adrian,

Thank you for bringing this topic up. I’ve always heard that Apple used the popularity of their mobile devices to kill Adobe Flash. However, I never knew the real story, so I decided to research it a bit. In 2010, Apple published an essay written by Steve Jobs titled “Thoughts on Flash” (Jobs, 2010). In it Steve Jobs cited the following reasons:

* Adobe Flash, being entirely owned and controlled by Adobe, is a closed system, and therefore should not be a web “standard”. Web standards should be open (E.g., HTML5, CSS, JavaScript).
* Most existing Flash video on the web is available in other formats.
* According to Symantec, Flash is one of the least secure software platforms.
* Flash does not perform well on mobile devices. Playback of Flash video requires decoding in software, which requires more battery power. The preferred way to decode in hardware.
* Because web pages using Flash rely on mice hovering over specific spots, not suitable for touchscreens.

Regards,

Emanuel

Reference:

Jobs, S. (2010) *Thoughts on Flash* [Online]. Available from: <http://www.apple.com/hotnews/thoughts-on-flash/> (Accessed: 10 June 2014)