

[Full-disclosure] The state of JavaScript Hacking

Source: <http://www.derkeiler.com/Mailing-Lists/Full-Disclosure/2006-11/msg00403.html>

- *From:* "pdp (architect)" <pdp.gnucitizen@xxxxxxxxxxxxxxxx>
 - *Date:* Mon, 27 Nov 2006 14:51:37 +0000
-

Please take my apologize if the following post has offended you in any way. The reason I posted it here is because I wanted to get this message heard by a wider audience. What better place to talk about this than the security mailing lists.

<http://www.gnucitizen.org/blog/the-state-of-javascript-hacking/>

In this post I would like to share a few thoughts with you about the importance of JavaScript and other under appreciate web technologies and their impact on the computer security industry and our lives in general. The purpose of this is to bring more light on the matter. Although this topic is becoming clearer now, I can still see quite a lot confused security professionals thriving to comprehend the core principles of these, relatively new, types of attacks. In this article I hope that I will be able to present my view as briefly and accurately as possible.

As you might already know JavaScript is becoming more and more popular among Web developers. The reason for this sudden growth is AJAX which among other technologies brought some quite useful and exciting features. Historically, AJAX is nothing new. This technology has been known for ages although as I said earlier it has started being implemented on a large scale just recently. On of the biggest AJAX evangelist up to day is Google which I believe is responsible for the AJAX hype in general. This is just a personal opinion.

AJAX and JavaScript are nothing new to security professionals, either. You can see that various attack vectors related to these technologies has been discovered in the past. I am not talking about browser vulnerabilities but pure design and implementation insecurities. Among them there are techniques such as XSS (a.k.a Cross-site scripting) and CSRF (a.k.a Cross-site request forgery). Both of them outline ways of performing information gathering, session hijacking and request forging. From the user prospective this is very serious but not that many companies have taken it seriously because they don't really understand them, I suppose.

You are probably aware of XSS and CSRF because the state of JavaScript hacking today is based around them. However, because of them the

security industry has never really understood their real potential. Simply put, performing session hijacking is not as interesting as sniffing the air and forging someone's requests is just not as fun as obtaining remote access. Will that change?

What security professionals must understand is that JavaScript is not about web pages anymore. It is a technology that is currently overtaking every WEB frontline and the desktop too. JavaScript is used on servers, web pages and desktop applications. It is a bridge technology. WEB designers use JavaScript to glue visual elements while browser vendors glue desktops and servers. The technology is the same all over the place which means less coding. That results into less money spent. Very utopic I must say.

If you have less overhead with developing desktop and web applications with JavaScript don't you think that attackers will have the same benefit? They can write cross-platformed viruses that can compromise desktop and web applications at the same time. Code once, destruct everywhere! Mozilla and Adobe are the biggest cheerleaders in this game. Microsoft is somewhere behind but they are quickly catching up.

Mozilla with their XUL makes attackers life so much easier. It is not that the Mozilla browser is vulnerable to any specific type of attack but the past has already proved many times that eventually someone will find an issue with the architecture. Then people will find the same mistake in other places. The Mozilla XUL is considered a true RIA (Rich Internet Application) platform that is currently the base of many open source products. All of them support JavaScript, CSS, Flash (if installed) and Java (if installed). If the developers of these applications don't have deep understandings of the security implications of the Mozilla platform the WEB will become suddenly very dangerous place for them.

Adobe on the other hand is making the process of creating a browser so transparent that everyone, I mean everyone, even your grandma will be able to create one in seconds. I am talking about Adobe's Apollo framework which is build on the top of Flex. If you haven't heard of it go and research now. Come back later. Don't get me wrong, I will probably use this platform to write a few security tools but just think about this for a second: developers will be able to write applications that will integrate the desktop with the WEB using already proven WEB technologies such as JavaScript, CSS, ActionScript, Flash and AJAX. I don't really know what Apollo's security model will be but apparently you can do whatever you want as long as the application is installed on the host environment. BTW, you install applications with a single click. Moreover, given the fact that Flash is so well spread, I am almost 90% sure that Adobe will want to push down their Apollo technology to every single computer on the this planet. And guess what, JavaScript has access to Apollo's runtime just like Firefox and Opera has access to Java via LiveConnect. If you develop applications with Apollo and you don't set the security model

[Full-disclosure] The state of JavaScript Hacking

properly the RSS feeds your are eating may start eating you.

Last but not least we have Microsoft with their XAML and WPF (Windows Presentation Foundation). I am sure that not that many people have heard of these technologies so let me explain what they are in brief. They are the Microsoft's way to do RIA. The only thing is that they relay on .NET3 which makes them explicitly for Windows. I am not sure what is the state of the MONO project though.

WPF will allow you to build Rich Internet Applications with XML, CSS and .NET. .NET supports many languages one of which is JavaScript. Try to do some coding in ASP and you will see that it feels the same as browser JavaScript. This is JavaScript on the server, the browser and the desktop. It enables web worms and future high-end attackers to a degree hardly imaginable by anyone today.

So what will be the state of JavaScript hacking in the future? WEB technologies will spread all over our lives. Your fridge and mobile will be powered by Flash Light and Java. Your desktop will be crowded with WPF games and Apollo goodies. Your website will run on AJAX, CSS and XHTML. Code once, destruct everywhere!

If you are still not convinced that this is not a joke I really don't know where to forward you to for more information. I guess you should wait until things start happening.

—

pdp (architect) | petko d. petkov
<http://www.gnucitizen.org>

Full-Disclosure – We believe in it.

Charter: <http://lists.grok.org.uk/full-disclosure-charter.html>

Hosted and sponsored by Secunia – <http://secunia.com/>