Web 2.0 Security Position Paper:
"JavaScript Breaks Free"

Markus Jakobsson, Zulfikar Ramzan, Sid Stamm
{jakobsson, sstamm}@cs.indiana.edu, zulfikar_ramzan@symantec.com

Abstract. The web has become richer with content, and a host of technologies are in place to improve interactivity – whether between the web browser and web server or between the browser and other desktop applications and network devices. Consequently, there is a greater burden on Web scripting languages to not only support this flexibility, but to do so in a way that does not increase new security risks. While the web browser used to have the responsibility of interpreting web languages and displaying the results, we take the position that the environment with which the user interacts with the web is much more complex and the policies governing these boundaries needs to be better understood (and better enforced). There have been a host of powerful attack concepts that trespass the existing loosely protected boundary, and allow the attacker to infiltrate the user's home computer and network. These include drive-by pharming, overtaking Google Desktop, and universal cross-site scripting. While these types of attacks are not yet visible in the wild, given their simplicity, we believe it is only a matter of time before they are. In general, we expect this trend to continue and expect to see more powerful attack concepts along similar lines.

Overview and Motivation. The Internet, and the World Wide Web in particular, are becoming an increasingly important resource to people in modern society. According to the Pew Internet and American Life project, 42% of adult Americans (43 million people) have a broadband Internet connection at home, and 43% of those people are online for two or more hours per day [4]. Mostly, these people are browsing the web for news, shopping, blogging, researching, or just looking around (surfing), and the vast majority of Internet use is browsing the Web with one of many browsers: Internet Explorer, Firefox, Safari, Opera, etc.

Web content is becoming increasingly complex, especially with the "Web 2.0." movement. Web 2.0 involves the coupling of several concepts:

- Increased interactivity through dynamic technologies that heavily use JavaScript and Cascading Style Sheets (such as AJAX), whose popularity makes the Web seem more like a desktop;
- User-generated content including not only text-based blogs and wikis, but also more rich content like video;
- Richer media capabilities facilitated by browser extensions and plug-ins (e.g., Adobe Flash or PDF viewer).

In addition, the Web Browser is now capable of interacting with other desktop applications like Google Desktop or with a user's home network; these applications or devices often run a web server (though that may not always be apparent to the user). This increased interactivity provides rich functionality, but also poses dangers in that JavaScript (or any other scripting language running in the context of a user's web browser) can potentially escape past the browser as an emulator and cause serious security concerns.

Security Concerns. One interesting security concern along these lines is *Drive-by Pharming*, a concept co-developed by the authors [5]. In a drive-by pharming attack an attacker can write JavaScript code that when executed in the victim's browser will mount a cross-site request forgery on the victim's home broadband router (which hosts a web server) and change its DNS settings. From that point onward, the victim's DNS

requests are resolved by the attacker, who can control what sites the victim actually goes to. In other words, the attacker effectively owns the victim's connection. To be successful, the attacker only needs to know the password on the victim's router – and most of the time this password is set to a factory default which the victim is unlikely to change. The attack also leverages the fact that most home broadband routers have a web management interface, thereby permitting the possibility that configuration changes can be made through HTTP requests (which in turn stem from JavaScript execution). We successfully implemented proof-of-concept code for the the drive-by pharming attack on Linksys, Netgear, and DLINK routers. The attack is simple to mount, easy to fall for, and has potentially quite devastating consequences. This work builds on some prior work on JavaScript host scanning presented by Grossman [3].

Another security concern in this enriched environment is in desktop applications that interact closely with the web browser. The resulting policies governing the software boundaries are not just limited to the web browser, but must now also include any application that closely interacts with it. One example of note is the recent work showing how to overtake the Google desktop software application through a cross-site scripting vulnerability in Google.com [1]. Since the Google desktop application itself runs on the local machine, any compromise to it results in a compromise of the local machine rather than being limited to whatever sandboxing or emulating environment the browser itself provides. Along similar lines, researchers discovered a vulnerability with the Adobe PDF plug-in that made any web site hosting a PDF file susceptible to a type of cross-site scripting attack [2].

These concerns are exacerbated by the ability for signed or "trusted" code to execute with advanced privileges, such as access to the filesystem. Although such code is signed and presented to the user with a certificate allowing a user to reject code signed with an untrustworthy certificate, many people do not know to verify the origin of the certificate before clicking "yes," thus allowing the signed applet to execute with elevated privileges [6].

Conclusion. Along the lines of the mentioned attacks, we believe the future will be home to many new web security problems that live "between the lines," or in essence, within the rules set forth by policies such as the Same Origin Policy implemented in JavaScript. These new problems will be more elaborate instantiations of Cross-Site Scripting, Cross-Site Request Forgeries as well as new abuses in forthcoming technologies. These problems may not be bred out of flaws in specifications or technical vulnerabilities, but instead by crafty abuse of existing features.

References

[1] Yair Amit, Danny Allan, and Adi Sharabani. "Overtaking Google Desktop." Watchfire whitepaper. `http://download.watchfire.com/whitepapers/Overtaking-Google-Desktop.pdf`.

[2] Stefano Di Paola and Giorgio Fedon. "Subverting Ajax." 23rd CCC Conference, Dec. 2006. `http://events.ccc.de/congress/2006/Fahrplan/attachments/1158-Subverting_Ajax.pdf`.

[3] Jeremiah Grossman. "Hacking Intranet Web Sites from the Outside: JavaScript Malware." BlackHat 2006.

[4] John Horrigan, "Home Broadband Adoption 2006," Pew Internet and American Life Project. `http://www.pewinternet.org/PPF/r/184/report_display.asp`

[5] Sid Stamm, Zulfikar Ramzan, and Markus Jakobsson. "Drive-by Pharming." Indiana University Technical Report TR641. `http://www.cs.indiana.edu/pub/techreports/TR641.pdf`

[6] "Phishing and Countermeasures: Understanding the Increasing Problem of Electronic Identity Theft", Markus Jakobsson (Editor), Steven Myers (Editor). Pages 618–621. Wiley, November 2006. ISBN: 978-0-471-78245-2