Lecture 28: Web Security: Cross-Site Scripting and Other Browser-Side Exploits

Lecture Notes on "Computer and Network Security"

by Avi Kak (kak@purdue.edu)

April 18, 2017

11:01pm

Goals:

- JavaScript for handling cookies in your browser
- Server-side cross-site scripting vs. client-side cross-site scripting
- Client-side cross-site scripting attacks
- Heap spray attacks
- The `w3af` framework for testing web applications

CONTENTS

28.1: Cross-Site Scripting — Once Again

- Earlier in Section 27.3 of Lecture 27 you saw an example of a **server-side** cross-site scripting attack through server-side injection of malicious code. In this section here, I will now give an example of a **client-side** cross-site scripting attack.

- As mentioned in Lecture 27, a cross-site scripting attack, abbreviated as **XSS**, commonly involve three parties. For the server-side XSS, the three parties are the attacker, a web-hosting service, and an innocent victim whose web browser is being exploited.

- For the client-side XSS, we again have three parties: an attacker whose goal is to get an innocent victim to click on a JavaScript bearing URL in order to cause the victim's browser to exfiltrate the cookies to a third party or to download malicious browser exploiting code from third parties. A client-side XSS is an example of UXSS, which stands for Universal XSS. [See the paper "Subverting Ajax" by Stefano Di Paola and Giorgio Fedon for other examples of UXSS. You can get to the paper by googling the author names.]

- That client-side XSS continues to be very important to web security can be judged from the fact that the 43 patches in the mid-July 2015 update of Google Chrome for Android included those for fixing XSS vulnerabilities. Googling CVE-2015-1286 and CVE-2015-1285 will take you to further information related to the vulnerabilities fixed by these patches.

- Since the client-side XSS attacks typically involve getting a victim's browser to execute a fragment of JavaScript, we will start in the next section with a brief review of this language. [Client-side XSS attacks also involve other client-side scripting languages for web applications. These include VBScript, Flash, etc.]

28.2: JavaScript: SOME QUICK HIGHLIGHTS

- JavaScript is meant specifically for browser-side computing.

- JavaScript is not allowed to interact with the local file system. [However, it can interact with the plugins for the browser and that can become a vulnerability, especially if the plugins have their own vulnerabilities.]

- JavaScript started out as a scripting language that consisted of commands that would be executed on the browser's computer for what is generally called "browser detection" and for form verification. To ensure that a web page is optimized separately for both the Internet Explorer and Firefox, a web server may deliver a page that contains both ways of displaying an HTML object optimally — with the expectation that JavaScript would first figure out which browser was being used and then execute only those commands that are appropriate to that browser.

- In addition to the duties mentioned above, JavaScript is now widely used for producing mouse-rollover, animation, and other effects in web pages.