Andrew Ebert

SE 4930

1/10/12

Dr. Schilling

Automated Analysis of Security-Critical JavaScript APIs

This paper is from the IEEE Symposium on Security and Privacy. It is called "Automated Analysis of Security-Critical JavaScript APIs" written by Ankur Taly, Ulfar Erlingsson, John C. Mitchell, Mark S. Miller, and Jasvir Nagra. The main point of this article is to show how JavaScript needs to be fixed in order to make it security safe. It also shows how the JavaScript standard has been improved to incorporate security measures against exploits from untrusted code.

The article begins by talking about how JavaScript in its current format is not secure and untrusted code could compromise the security of a website. Even though the JavaScript can be encapsulated or sandboxed there are ways around it and still gain access to code that is hidden. To help with the sandboxing ECMA committee who manages JavaScript released a new standard that introduced a "strict mode". This mode removes access to calls that allow access to hidden code and it prevents untrusted from running

Another problem with JavaScript was that every method in JavaScript has write access to built-in objects so untrusted code could modify other code by accessing these built-in objects to access other methods. To fix this the built-in objects have their properties set so they cannot be extended with additional classes or properties. Also it helps to make sure the evaluation method is restricted to only the variables handed in.

These were just a few of the vulnerabilities found by this study. The article also talks about different ways to verify the API's confinement. This was done in the paper by running a procedure that they set up to analysis for all of the different vulnerabilities that they listed previously in the paper. From this procedure they try to see if they can access hidden variables or if any function leaks certain objects that should not be leaked. During the study the writers of the paper tested three benchmark examples. These were the Yahoo! ADsafe library, the Sealer-Unsealer mechanism, and the Mint mechanism. All of these libraries were made with sandboxing and security in mind.

From their tests the group found that one of the three was vunerabile to security breaches. This was the ADsafe library. The library had a vulnerability. It was leaking the document object in lib and go methods. This allowed for editing outside of the normal access allowed.

Overall I learned that when I create webpages that use JavaScript that I need to be careful in implementing it since I could leave my website open to attach. I can apply this knowledge by sandboxing my code in such a way that it is only used for what it is intended to do. This requires that I do not hand any global variables back to the webpage and that no one can see how to access these variables. Also I would need to make sure that certain functions only use the variables handed in.

This paper is about the vulnerabilities in JavaScript that is being used now and how to go about fixing them while making sure that the new code is not leaking global variables and other parts of unsecure JavaScript. At this point I do not have any questions as the document is pretty self-explaining with examples of how they exploited the code and how the code could be exploited.

Works Cited

Taly, A., Erlingsson, U., Mitchell, J. C., Miller, M. S., & Nagra, J. (2011). Automated Analysis of Security-Critical JavaScript APIs. *IEEE Symposium on Security and Privacy*, (pp. 363-378).