Homework 4 – Client Side Attacks

Lab Information

Due Date:

Homework 4 Dropbox Deadline

Objectives/Goal:

In this homework we will be investigating part one of client side attacks. While you are free to experiment however you'd like – **note that this lab is not intended to be completed by leveraging XSS.** The environment which you are entering is also hostile so be aware and use private browsing as needed.

Deliverables:

- The access.log file
- The attribution script.
- The Matomo Docker image
- Your Armbook userid and photo evidence of Jon Doe having friended you.
- The version of Armbook that fixes the issue from 2.1
- The code that demonstrates the attack described in 3.1
- The writeup describing how to fix the attack in 3.1

Table of Contents:

Activity 1: Know Thy Enemy

On the backend we have setup a Selenium script to represent our vulnerable client. This script will click any link that you post to the following message board (http://csec380-core.csec.rit.edu:5004/). Your first goal is to demonstrate that this client is functioning.

Step 1: Setup a server

Setup a publicly facing webpage. This may be on EC2 or it may be on a RIT wireless, you just need a publicly routable external IP. Don't forget to disable your firewall as needed. Send our client a message with the link to your page. Use your access logs to verify that the client has connected to your site.

Make a file called access.log with just the access.log generated by your webserver.

Step 2: Find out about the client

After you have proved that the client will access your page, modify your site to gather some information about the client. You should gather the referer, IP, User-Agent, and inject JavaScript to determine what plugins are running. Additionally, create a Docker container that will use Matomo (https://matomo.org/), an open source analytics framework, to track similar content. Attach Matomo to your index page.

Provide your attribution script and a Docker container (docker-compose) that Matomo running and a beacon on the index page.

Activity 2: Attack Thy Enemy

Our first encounter with our enemy allowed us to get a better understanding of where the enemy's weaknesses may lie. This was our first foray into attacking a client. Now we will leverage some aspects of their environment to our benefit.

Step 1: Cross (not script) that client

Now we're going to use a weakness of the client (Jon Doe) against them. We know our client LOVES Armbook, he's ALWAYS online (http://csec380-core.csec.rit.edu:84/) and he'll also click on [message board] (http://csec380-core.csec.rit.edu:5004/) links. Make an account and figure out a way to get him to friend you. **Do not use Cross-Site-Scripting**.

Provide a writeup with your username and photo evidence that Jon Doe has friended you.

Step 2: Fix an Agent

The source code for the current version of Armbook is available online. Fix all variants of the underlying issue that you have encountered and provide a blurb explaining how you fixed the issue. Make sure to post your code in the dropbox.

Provide your zipped code with the underlying issue fixed and writeup.

Activity 3: Rebuff Thy Enemy

Step 1: A (click) Jack of all attacks

Your attacker has implemented the changes you suggested, but you still want to attack him. Using a different attack method, generate a page that if visited and interacted with would result in him becoming your friend once again (This is not the same attack as in the previous activity).

Create a demonstration page that would accomplish this task and provide it in the dropbox. Do not use Cross-Site-Scripting.

Note: Mycourses will modify uploaded HTML pages, do NOT upload raw.html files.

Step 2: A fix for the itch

Write-up a paragraph describing how the previous issue would be fixed.

Signoffs

Activity 1.1 – An access log demonstrating callback. (10%)

Activity 1.2 – A custom attribution script and Matomo docker-compose file. (20%)

Activity 2.1 – Provide evidence that Jon Doe friended you (20%)

Activity 2.2 – Provide and describe the fix for the issue from Activity 2.1 (20%)

Activity 3.1 – Provide code to demonstrate the vulnerability from Activity 3.1 (20%)

Activity 3.2 – Describe the fix for Activity 3.1 (10%)