CSSS 3510 Writing Secure Code

Spring 2 2020

Author: Jose Pablo Murillo

Hands-on lab report

1.) Lab settings:
    a. PC Hardware settings and Software configuration
        i. See phpinfo() output in Annex 1

    b. Virtual machine settings
        i. Operating system: Ubuntu 64-bit
        ii. Base memory: 4096 MB
        iii. Chipset: PIIX3
        iv. Number processors: 2
        v. Network: Bridged

2.) Lab objectives: This lab is a demonstration of session hijacking where the attacker is able to obtain the current cookie session ID and then be able to access the system. Recommended corrections to vulnerabilities will be presented.

3.) Lab description:

1. A simple html form is designed with input for a username and password. The form sends the input via a post request to the vulnerable main.php file. A session is created and the username is stored in the session cookie.

2. The information stored in the cookie is then used to load a user's information regarding notes that have been hardcoded for the lab. The information is stored in an array where the key is the user and the value is an array of strings for each note.

3. **Attack under MITRE ATT&CK Framework**

    a. **Enterprise - Steal web session cookie**
        i. The attacker can steal the application's session cookie in order to gain access to the system. This also allow him to enter as a legit user without the need for credentials. The attack exposes a system's information, files, and services to be exploited and used at the attacker's will. In this demo I logged in to the web application through one web browser. Since the application does not prohibit the use of JavaScript from the client side, I used in the console **let a = document.cookie; alerta);** in order to see the cookie name and the value for the session id. I then used the session id to enter the notes section and changed the sessionId to that of the first browser. This allowed me to access as the user and see their notes which contained sensitive data.

4. The secure version of the main.php script can be found in main2.php. Main2.php avoids session hijacking through the previous method using **session_set_cookie_params()** function from PHP. The function is used to alter the behavior of the cookie and then is configured so that JavaScript cannot be entered from the attacker.

5. Code can be found in Annex 2. In order to run the simulation, edit the index.html in the form action parameter to go to either main.php or main2.php

4.) Conclusion: Server-side scripts should not allow the possibility for session hijacking. Such bad practice can let the attacker access the system using legit credentials and access the system's services, information and files.

Resources:

https://www.php.net/manual/en/function.session-set-cookie-params.php

https://attack.mitre.org/techniques/T1539/

https://www.php.net/manual/en/language.types.array.php

https://www.w3schools.com/php/php_sessions.asp