Secure Programming Coursework

Robert Flood and David Aspinall

School of Informatics, University of Edinburgh

This is an **individual** assessed practical exercise. Provided you have attended the relevant lectures and lab sessions in the course, the work for both parts should take about 30 hours. The practical will be awarded a mark out of 100. The deadline for submission is **12pm, Fri 18th November 2022**. The final page summarises the submission instructions.

Obtaining the Files

The coursework files are stored in a zip file in the directory `/group/teaching/module-sp/2022/q3-code.tgz`. You can transfer this file to your local machine via scp:

```
scp [user]@student.ssh.inf.ed.ac.uk:/group/teaching/module-sp/2022/q3-code.tgz /local_dir
```

The files can then be extracted via:

```
tar -xpf q3-code.tgz
```

Docker

This coursework focuses on a vulnerable web server. We've provided a containerised version of the server based on Docker.

If you want to install Docker on your own machine, follow the intructions provided on the Docker website for your operating system.

Otherwise, the SEED Labs Virtual Machine contains the necessary software to run Docker pre-installed. Note that this VM is immutable so if you are using it (and anyway, for safety), **back up your work** by saving any work that you do inside the virtual machine (edited source files, etc) in your home directory. On the SEED Lab VM, Docker can be started by running `systemctl start docker`.

Note that the Docker containers are also immutable. Thus, restarting the container will reset the database and lose any changes.

Setting up

Prior to doing the coursework, you'll need to build the provided Docker image. You can build and launch the container with the correct networking set up by running `docker-compose up -d`.

If you make any changes to the web apps source code and you wish to test it, you can rebuild the image with `docker-compose build --no-cache`

Useful commands

- Restart Docker: `systemctl restart docker`
- List Docker Containers w/ IDs: `docker ps`
- Stop Container: `docker stop [Container ID]`
- Remove Container: `docker rm [Container ID]`
- Teardown Docker Compose: `docker-compose down`
- Run Command in Container: `docker exec -it [Container ID] [Command]`
- Check Running Processes in Container: `docker top [Container ID]` #

3. Web Security (28 marks)

The provided Dockerfile describes a very naive web app for image sharing and voting. You may access the web app through http://localhost:8080.
This server is poorly configured and is susceptible to a number of vulnerabilities. There are 10 users created for you in advance. They are user1 ~ user10. The passwords are the same as respective usernames. Feel free to register your own users as well.

1. Describe how the user field is vulnerable to an XSS based attack. Your description should include code to demonstrate the vulnerability (e.g., by injecting the javascript `alert("Hello World!")`) as well as *step by step* instructions. Provide a patch file to remedy this vulnerability (see Notes). (6 marks)

2. Describe how the signature field allows an attacker to mount a CSRF attack. The attack should cause a victim to vote for the attacker (another user) transparently. Describe the steps to perform the attack from the perspective of both the attacker and victim. Again, give your answer as a clear and unambiguous series of steps detailing the necessary inputs. Provide a patch file to remedy this vulnerability (see Notes). (6 marks)

3. Describe how the image upload field is vulnerable to a Remote File Inclusion attack. The attack should allow arbitary code to be run on the server. In particular, a successful attack obtains a reverse shell on the target system. Again, give your answer as a clear and unambiguous series of steps detailing the necessary inputs. Provide a patch file to remedy this vulnerability (see Notes). (7 marks)

   NB: This question requires that you run your own simple web server on your host machine. The most straightforward way to do this is the Python `SimpleHTTPServer`, which can be launched via `python3 -m http.server`.

4. The PHP initialisation file, php.ini, is a critical configuration file that governs many aspects of PHP's behaviour. The provided version contains a number of questionable configuration settings. Describe three of these misconfigured settings including how they can lead to exploitation. Provide a patched version of the PHP.ini file that fixes these problems (see Notes). (9 marks)

Notes

You should provide you description and answers in **answers.pdf** and submit the overall corrected version of the code in a single patch file called **question3.diff**. Remember to **backup** the files in */srv/http* before modifying them. You will need the original code to run the *diff* command.

Before making any changes to the code, first produce a back up:

```
cd code
cp -pr imageApp imageApp.orig
```

After completing all of the questions, you can produce a batch diff file using the command:

```
cd code
diff -cr imageApp.orig imageApp > question3.diff
```

Then submit question3.diff

Feel free to modify any source code and rebuild the containers, but be aware that any exploit code will be tested against the original Docker images running according to the original configuration.

Partial marks will be awarded if you cannot produce a working exploit, but have demonstrated that an attack vector is vulnerable.

Submission instructions (Part 2)

Go to the SP Learn course and select "Assessment" from the left hand menu. Select the "Assignment Submission" folder and then the "Coursework (parts 1 and 2) folder. Click on the link "Submit via Gradescope". This will take you to the Gradescope interface. For anyone who has sat an online exam over the last two years, this should look familiar to you. From here, you can drag and drop your file(s) to submit.

Please name your files as follows:

answers3.pdf A PDF document containing the answers to Question 3.

question3.diff The patch file generated for Question 3.

The PDF documents should be well-formatted printable A4 PDFs, you may generate them with whatever program you want. Text answers should be brief and to-the-point.

We will mark the most recent files and their submission timestamps must be before the deadline to avoid standard lateness penalties.

You must submit by the **final deadline 12 noon, Fri 18th November 2022**.

You're reminded that late coursework is only allowed if approved by the University's central ESC Team, see the Informatics advice page for more details and how to apply.[1]

[1] https://web.inf.ed.ac.uk/infweb/student-services/taught-students/information-for-students/information-for-all-students/your-studies/late-coursework-extension-requests