Secure Programming Assignment 1 – Worth 50%
Deadline: 20th November @ 11.59pm

NOTE: This is a GROUP assignment. 2-3 (3 max.)

I will not accept individual submissions

The purpose of this assignment is to secure vulnerable source code in the web application provided and provide a report on how you achieved this.

You will be supplied with a simple Web application. The app is coded using Python Flask and connects to a SQLite backend database. There are 3 parts to the assignment:

i. Demonstrate the exploits (show using screenshots and a description how the application is vulnerable in the source code)
ii. Fix the code.
iii. Demonstrate the exploit is no longer possible (again use screenshots and a description).

For the first part of your assignment, you will need to analyse the code and look for possible weaknesses. You then need to try exploit any weakness you think you have identified. For example, if you think a form field is vulnerable to XSS, then perform some attack to show the weakness.

NOTE: Only one vulnerability and fix, per location, is allowed. ***In other words, you won't get any more marks for fixing the same SQL injection vulnerability in several places. However, different types of vulnerability count separately. For example, reflected XSS and stored XSS count as separate vulnerabilities.***

For the second part of the assignment, you must try to secure the code, by correcting any flaws you have found. Add comments where you have updated the code and write a brief explanation of this in your report. Lastly, you should re-run the exploit to show that the Web application is secure (use screenshots).

Setup

Please refer to the setup instructions included in the zipped application folder.

Section 1 (25%)

There are plenty of vulnerabilities in the code, not just the ones we have covered in the labs. You can find others via, e.g., OWASP and CWE Top 25 sites. You need to list each weakness that you think you have found and briefly mention what type of weakness it is. You should

also exploit each weakness, with some real-world hacking. You should highlight exactly what you did to exploit each weakness (use screen hots where necessary). If you find a possible weakness but fail to exploit it, then you should still include it, and mention anything you tried in your attempt to exploit it. Also, include some explanation of where the code is vulnerable and why.

Section 2 (25%)

The second part of your project is to correct the source code to fix as many of the weaknesses as you can. Your final corrected code must still run with all changes. You must correct the code I give you, not just hand me back a completely different app. If you tried to fix some code but it won't compile or gives errors, then include it in your source code (commented out), so I can see what you tried and where. In your report, you need to include each snippet of code you tried to correct and how your code fixes the problem.

NOTE: You should aim find/fix 5 vulnerabilities we covered in the labs and 5 we did not – see breakdown below.

Deliverables

A zipped file with your completed report and all the corrected source code uploaded to Moodle by the **20th of November @ 23:59**.

MAX word count for the report should be between 2,500 and 3,000 words, but reports can be considerably less. **I don't want a history or explanations of vulnerabilities** in your report. Just what vulnerabilities you found, how the app is vulnerable, why the code is vulnerable and how you fixed them in code.

Marking Rubric (100%):

Identifying and exploiting vulnerabilities (50%):

- Lab vulnerabilities - 20%
- Non-lab vulnerabilities - 30%

Fixing the vulnerabilities (50%)

- Lab vulnerabilities - 20%
- Non-lab vulnerabilities - 30%