Reflective Piece (Word count: 985)

This module was quite possibly the most interesting, stimulating and enjoyable so far. As a culmination of all the theory we've learnt so far in other modules, this module provided a practical assignment to "put it all together". Aside from smaller tasks cleverly designed to provide exposure to a range of security-based concepts e.g. buffer overflows, linters etc., the module had a project that required us to take a real-world problem and develop a secure web application for it.

The project was very cleverly divided into two parts. The first part required us to create a design document in which we laid out the design of the proposed application. There were two very useful aspects to this. First, this required us to carefully investigate the problem domain and consider the possibilities as regards vulnerabilities and security issues. Second, I really appreciated that, while we were placed into teams, the module lecturer required each team member to prepare a separate document. This meant that I was able to give the problem comprehensive attention, and not just focus on a few things, which was invaluable learning time. One key aspect that I'm focusing on is to try and get over various learning curves which are usually the most difficult part of learning something new.

Our application was a web application for reporting vulnerabilities to the National Cyber Security Centre (NCSC) of the government of the Netherlands. When it came to the design, and given a lack of access to the NCSC itself, I decided to gather information from the NCSC website (NCSC, n.d.) to determine how the process of collecting information on vulnerabilities work, as well as what kind of information is gathered. Once I had this information and had summarised it, it was about using the knowledge that I gained from previous modules around security analysis techniques and frameworks such as STRIDE (Kaufman et al., 1998) and the Open Web

Application Security Project's OWASP Top 10 security vulnerability list (OWASP, 2021), to analyse the problem domain. One significant challenge was the word limit. There was so much to say and figuring out how to say it concisely was a good part of the challenge, and this was obviously intended; it's important to be able to take complex ideas and express them concisely.

I appreciated the individualised feedback that we received from the lecturer (also appreciating just how much work this would have been on her part!) The team work aspect was also valuable as it required us to take three viewpoints (with some overlap) and deliberate and negotiate what stays and what goes, and what goes where. This was all valuable learning time.

When it came to coding, I must admit that it was a challenging experience of "learning curves upon learning curves within learning curves towards learning curves…" There were so many technologies to put together. At the most basic, it was about using Python, web servers, javascript, HTML, CSS, databases and SQL; but then it was about specific frameworks such as Flask (Grinberg, 2018), SQLAlchemy (Bayer and O'Meara, 2006), the SQLite "dialect" of SQL (Owens, 2010), Bootstrap (Kent et al., 2016), jQuery (Chaffer and Swedberg, 2013), among others… I obviously had to prioritize what I would learn more comprehensively (Flask, SQLite and Bootstrap) and what I would just learn enough to get things done in the short time available (jQuery, CSS).

I was really glad that my team mate came out with a very basic non-functional Flask and Bootstrap template that we could use, which I used as a basis on which to build on. At first, and for a while, the going was very slow. I found myself spending a lot of time just trying to make very basic things work, such as defining a simple route,

passing parameters, displaying those parameters. I was probably doing Internet searches by the hundreds, if not thousands, on a daily basis. As time went by, and as I got over the initial tough learning curve with Flask and SQLAlchemy, I found myself really falling in love with these super powerful yet clean, modular and simple frameworks. I started by implementing user registration, followed by user login. I then progressed to report creation. Figuring out how to encrypt the data was really interesting; I decided to assign each user a secure key of their own so that if one user's key leaked, other user's reports/information would still be safe.

Once I got it working, (it is an understatement to say that) it was **deeply** satisfying to view the SQLite database using the small nifty lightweight application called "DB Browser for SQLite" (Zhou and Wei, 2019) and see nothing but a bunch of gibberish in the Report table, but then logging in and viewing a nice readable report through the app. A few weeks and 28 commits later, I had a fully functional app and it honestly felt (and still feels) absolutely amazing seeing this full-functional "living breathing" application. I especially still feel elated when I access the account page of a specific user that has posted many reports and messages, and then click on a message in the messages list on that page and watch as the app jumps to the messaging page and highlights the message. This was a feature that really frustrated me to the point of wanting to abandon it, but eventually got it working.

I also found myself feeling a deep sense of appreciation for all the developers out there that have volunteered so much of their time creating amazing and secure frameworks like Flask, SQLAlchemy, Bootstrap and others, which can readily and freely be used to create amazing applications in a fraction of the time it would have taken without them.

All in all, at the end of this amazing module, I feel a real sense of accomplishment, growth and satisfaction, and am thankful for the experience that this has given me.

References

Bayer, R., & O'Meara, T. (2006). SQLAlchemy: Database access using Python. Pearson Education, Inc.

Chaffer, J., & Swedberg, K. (2013). Learning jQuery: Better Interaction Design and Web Development with Simple JavaScript Techniques. O'Reilly Media, Inc.

Grinberg, M. (2018). Flask web development: Developing web applications with Python. O'Reilly Media, Inc.

Kent, T. P., Johnson, L. C. and Lindsay, K. A., "Improving Usability with Bootstrap: A Guide for Designers, Developers, and Students," Journal of Usability Studies, vol. 11, no. 4, pp. 165-179, 2016.

Kaufman, L., Zimmerman, P. R. and Levy, Y., "Security Considerations for IP Protocol," Request for Comments (RFC) 2401, November 1998.

National Cyber Security Centre (NCSC) (n.d.). About NCSC. National Cyber Security Centre (NCSC). Government of the Netherlands. Retrieved March 17, 2023, from https://www.ncsc.nl/english/about-ncsc

OWASP (2021). OWASP Top Ten. [online] Owasp.org. Available at: https://owasp.org/www-project-top-ten/.

Owens, R. (2010). SQLite: Up and Running: A Guide for Database Developers. O'Reilly Media, Inc.

Zhou, Y., & Wei, Q. (2019). Database management with DB Browser for SQLite. Journal of Chemical Information and Modeling, 59(9), 3802-3806. doi: 10.1021/acs.jcim.9b00516