Zach Steele

05/23/2025

Practicum Project Proposal Version 2

Summary: The goal of my project is to create a penetration testing course for developers that will teach the viewer how to test their application for a myriad of security risks. This is an important endeavour as a recent study shows that 33% of developers do not know what makes their code vulnerable. (Secure Code Warrior, 2022) After scouring the internet, I have realized that almost all penetration testing courses are either aimed at people who want to switch careers into penetration testing or are too high level to provide useful takeaways. What about the developers that want to poke around their application looking for some of the more basic vulnerabilities? This course will help developers understand common vulnerabilities and help them exploit these vulnerabilities in their environments with developer tools that are already in use. The reality is, most developers do not want to watch 20+ hours of content just to find out that their organization does not allow dev teams to have access to penetration testing tools or scanners. My YouTube series will use common developer tools while explaining how to exploit and remediate common risks.

What exactly will you work on for your practicum project?

I will be filling the gap in web application penetration testing courses in an effort to reduce the increasing rate of insecure systems found on the open internet(Bassett et al., 2021, p.44) by building a web application penetration testing crash course designed for developers. This is unlike the many courses on YouTube, Medium, or various hacking sites, such as HackTricks, PortSwigger, and OWASP. My course is designed for application teams that would like to test their own applications and new code before deploying to production without the resource overload that is built into most courses. This project will include a list of payloads, a checklist outlining every item that should be reviewed on your application and YouTube videos walking through risk, exploitation, and remediation. These videos will also show how the app teams can test for each item using developer tools, such as Postman, Insomnia, or their own browser. The previously mentioned material will have the goal of catching the most common vulnerabilities and low hanging fruit in a short amount of time, while providing the option to dig deeper with a handful of third party resources. We will be using the OWASP top 10 and Verizon DBIR results to guide the majority of our focus. According to OWASP, security misconfigurations and sensitive data exposure have become more prevalent since their

2021 research.(OWASP, 2025) The Verizon DBIR shows similar results in their 2024 to 2025 trends. Basic web application attacks have grown 3% while 42% of all vulnerability exploitation takes place in web applications, as opposed to VPN, edge device and other vectors.(Hylender et al., 2025, p.21) It is important to note that we will not be attempting to teach all the edge cases associated with these attack types, but we will instead give an overview of the attack, a handful of payloads that most commonly work, then remediation advice. There is no need for a developer to get full exploitation or demonstrate risk when testing their own applications, that is only a requirement for true penetration test reports. For this reason, we will show the developers what behavior to look for during testing to identify vulnerable endpoints, even if the payloads were not completely successful.

Why do you think it is important and has real-world relevance?

This course is important because it will bridge the knowledge gap between developers and cyber security professionals. As a full-time penetration tester, I often find myself answering the same set of questions from app teams because most dev teams do not put an emphasis on security. While I understand the bias associated with a r/cybersecurity subreddit survey, it does show similar results to my conclusion. In this survey, 61 of the respondents stated that they have not learned about secure coding and only 32 of the 166 respondents learned secure coding practices on the job.(Reddit, 2024) The truth is, a developer's job is to complete the user stories that are given to them for that sprint. Most of the time, these stories say things like “add a search function.” They do not say “add a search function that is not vulnerable to SQL injection.” Studies show that 86% of developers do not view application security as a top priority. 30% of those developers state that this is because meeting deadlines takes priority over secure development. (Secure Code Warrior, 2022) This leaves us in a situation where the app team first hears about SQL injection after an attacker has exploited the vulnerable endpoint or the penetration testing team has reported this finding. PreEmptive explains that security is paramount in the Systems development lifecycle (SDLC), but it is an after thought because most developers are not using software security tools, using source code analysis tools, or doing security testing during all phases of the SDLC.(PreEmptive, 2024) Perry et al. (2023) believes this problem is only getting worse as developers use AI assistance when writing code, stating “Overall, we find that participants who had access to an AI assistant wrote significantly less secure code than those without access to an assistant. Participants with access to an AI assistant were also more likely to believe they wrote secure code, suggesting that such tools may lead users to be overconfident about security flaws in their code.”(Perry et al., 2023, p.1)

It can be difficult to point app teams in the right direction when testing or remediating these issues because they are not allowed to use penetration testing tools on the company network to recreate the exploit, and linking to hacking resources often requires the reader to have a passion for penetration testing because of the length and complexity of most reads. As an example, pointing the application teams to PortSwigger can be a bad idea because the target audience is people who want to become penetration testers, not people who want to quickly understand, test, and remediate a specific vulnerability. There is not a great resource online that can help teams test for vulnerabilities within their applications without dedicating massive amounts of time to learning the craft. From my experience as a developer, we were always told to write secure code, but didn't really understand what that meant because we didn't have a security background or useful training. Additionally, we would do “security reviews” of our code by looking through a word document containing vulnerable code examples. This was not a great way to ensure the code was secure and it never grew long term secure coding skills.

What do you expect to complete by the time you are done?

By the end of this project, I will have a checklist with 20+ checks ranging from simple response header review to SQL injection exploitation using a list of tools that developers already have at their disposal. Additionally, I plan on having a list of payloads to test for the vulnerabilities outlined in the checklist, a list of third party resources, and longer payload sections if the developer would like to learn more about the attacks. This material will be used in a YouTube video series where app teams can review the whole course, or skip to the videos most relevant to their current situation. I believe this is the perfect amount of work for the semester, but if I do have more time at the end, I would like to add a handful of less common exploits that the app teams can test for depending on their application’s architecture/tooling. To be more specific, a stretch goal would be to add material for NoSQL injection, XXE, and a few other attack types.

Lastly,I will evaluate the quality of my work by engaging other developers and asking them to review the material. My goal is to find multiple developers and security professionals with varying levels of experience and cyber security training to get honest feedback on how well this material could be utilized by a wider audience. Additionally, since my course will be focused on higher frequency vulnerabilities and easy to identify issues, the course will already follow the most important penetration testing topic based on frequency of exploitation. This document covered some examples, but more information will be provided throughout the course.

References:
Bassett, G., Hylender, D., Langlois, P., Pinto, A., & Widup, S. (2021). *Verizon 2021 Data breach investigations report (DBIR).* Verizon.
https://www.verizon.com/business/resources/reports/2021-data-breach-investigations-report.pdf

Owasp Top Ten. OWASP Top Ten | OWASP Foundation. (2025).
https://owasp.org/www-project-top-ten/

Hylender, D., Langlois, P., Pinto, A., & Widup, S. (2025). *2025 data breach investigations report.* Verizon Business. https://www.verizon.com/business/resources/reports/dbir/

Developers, where did you learn about secure coding practices?. Reddit. (2024).
https://www.reddit.com/r/cybersecurity/comments/1bbak2w/developers_where_did_you_learn_about_secure/

Secure code warrior survey finds 86% of developers do not view application security as a top priority. Secure Code Warrior. (Apr 05, 2022).
https://www.securecodewarrior.com/press-releases/secure-code-warrior-survey-finds-86-of-developers-do-not-view-application-security-as-a-top-priority

3 common security mistakes developers make – preemptive. PreEmptive. (2024, November 4).
https://www.preemptive.com/blog/3-common-security-mistakes-developers-make-in-their-sdlc/

Perry, N., Srivastava, M., Kumar, D., & Boneh, D. (2023, December 18). *Do users write more insecure code with ai assistants?* https://arxiv.org/html/2211.03622v3