Journal 8-2 [ Portfolio Reflection ]

Reflecting on this course and the projects I’ve completed, I've come to understand just how crucial it is to adopt a secure coding standard from the very beginning of the development process. Initially, I might have thought that security was something to be addressed towards the end of a project, almost as a final polish. However, the readings and lessons, especially from the SEI CERT C++ Coding Standard and the OWASP Secure Coding Practices, have really driven home the point that security must be woven into every stage of development. By adopting these standards early, I'm not just protecting the code I write; I’m building stronger, more resilient applications from the ground up.

One of the things I did particularly well in these projects was to integrate security practices consistently throughout the coding process. I made a conscious effort to assess risks and consider the cost-benefit of various mitigation strategies. This approach aligns closely with the concept of "shifting left" on security, which we discussed in class. It’s about tackling security issues early on, which ultimately saves time and resources that would otherwise be spent fixing vulnerabilities later. The idea of Zero Trust, which was a significant part of our discussions, also influenced my approach. By adopting a mindset where nothing is trusted by default, I ensured that every aspect of the system was validated and secure, enhancing the overall security posture of my projects.

However, I recognize there are areas where I could improve. While I’ve been diligent about integrating security into my coding practices, I realize that I could be more thorough in my risk assessments. Sometimes, I focus too much on the immediate risks and not enough on understanding the broader implications of those risks over time. Additionally, while I’ve used some static analysis tools, I could benefit from diving deeper into more advanced tools like Cppcheck or the Clang Static Analyzer. These tools could help me catch subtler issues that might slip through manual reviews, ultimately leading to even more secure code.

To bolster my ongoing efforts, I’m adding a few more tools and resources to my support network. I’ve learned the value of real-time monitoring through tools like SIEM systems and Intrusion Detection Systems (IDS). These will help me stay on top of any potential security incidents as they happen, rather than after the fact. Resources like the "Practical Guide to Zero-Trust Security" will also remain crucial as I continue to refine my approach to ensuring that security is always front and center in my work.

The skills I’ve developed through this course—like implementing secure coding practices, conducting comprehensive risk assessments, and applying Zero Trust principles—are incredibly valuable and transferable to future projects. Whether I’m working on a new application or revisiting existing code, these practices will help me ensure that security is never an afterthought but a core component of my work.

To make these projects maintainable, readable, and adaptable, I focused on writing clean, modular code and maintaining consistent documentation. I ensured that each function was well-documented and easy to understand, making it easier for anyone else who might work on the project in the future. This approach not only made the code more accessible but also ensured that it could be easily adapted or extended if needed. By structuring the code in this way, I’ve set up the projects for long-term success, making them easier to maintain and adapt as requirements evolve.

In summary, this course has been a valuable experience that has deepened my understanding of the importance of integrating security into every stage of development. The practices and tools I’ve adopted will undoubtedly influence my approach to future projects, ensuring that security remains a top priority in all my work.

References

OWASP Foundation. (n.d.). *OWASP secure coding practices quick reference guide*. OWASP Foundation. Retrieved from <https://owasp.org/www-pdf-archive/OWASP_SCP_Quick_Reference_Guide_v2.pdf>

SEI CERT. (n.d.). *CERT C++ coding standard*. Software Engineering Institute. Retrieved from <https://wiki.sei.cmu.edu/confluence/display/cplusplus>

Zero Trust. (2018). *What is zero trust security?* [Video]. YouTube. Retrieved from <https://www.youtube.com/watch?v=PrU4EvYP7Rs>

Shifting Left on Security. (n.d.). *Shifting left on security*. Retrieved from <https://www.redhat.com/en/topics/devops/what-is-shifting-left>

Microsoft. (n.d.). *Security best practices for C++*. Microsoft Docs. Retrieved from <https://docs.microsoft.com/en-us/cpp/security/security-best-practices-cpp>

MITRE. (2021). *A practical guide to zero-trust security*. Retrieved from https://www.mitre.org/sites/default/files/publications/practical-guide-to-zero-trust.pdf