CS-405 Secure Coding

8-2 Journal: Portfolio Reflection

Vincent Yanez

4/24/25

**Reflection on Secure Coding and Policies**

One of the biggest takeaways from this course is how important it is to use secure coding from the start. Waiting until the end to add security can leave systems open to attacks. As explained in *Secure Coding in C and C++* by Robert Seacord, following a secure coding standard early on helps developers avoid common vulnerabilities and build stronger software from the beginning.

When it comes to risk, it’s smart to look at both the chances of something going wrong and how much it would cost to fix. According to the *NIST Risk Management Framework*, evaluating risk and performing cost-benefit analysis helps prioritize which security issues should be addressed first. Not all risks need immediate fixes, but it's important to consider long-term impacts and the cost of doing nothing.

The idea of “zero trust” really stood out to me. It means “no one is safe” — everyone and everything has to be verified, even if they’re already inside the system. The Department of Homeland Security and NIST both promote the zero trust model as a modern and proactive security approach to stop both outside and inside threats.

Lastly, putting clear security policies in place helps guide everyone on what to do. These policies should be easy to understand and kept up to date. As emphasized in OWASP’s best practices, strong policies support secure development and help teams stay consistent when it comes to security.

**References:**

NIST. (2018). *Risk Management Framework for Information Systems and Organizations (SP 800-37 Rev. 2)*. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-37r2

OWASP. (n.d.). *Secure coding practices - quick reference guide*. Open Worldwide Application Security Project. https://owasp.org/www-project-go-secure-coding-practices-guide/

Seacord, R. C. (2013). *Secure coding in C and C++* (2nd ed.). Addison-Wesley.