**Module 8 Journal**

Zachary Mohler

Southern New Hampshire University

CS405

Aaron Demory

December 15, 2022

When I took my very first programming class at my local community college, I was very blessed to have a teacher that not only taught us secure coding principles, but also ensured we knew how to follow them. He would also test our programs very rigorously to ensure they met the standards of best practice. Not only do I credit this teacher for my love of programming, but also for my appreciation and dedication to secure coding.

Throughout this course I have furthered my knowledge of secure coding principles and standards of best practice. For me, the class simply gave me some new perspectives and techniques to my already strong desire to code securely. I have always been a stickler for ensuring even the simplest of programs are coded securely—handling all known errors and validating all input. If I had to choose one concept that this course really cemented even further, it would be “***DO NOT*** leave security to the end”. You would be hard pressed to find any situation where security was left to the end and the product turned out to have no major flaws or vulnerabilities that needed patching immediately upon release.

This course also gave me some very valuable skills using tools to evaluate and assess the risks associated with any particular code base. Where my previous approach to risk assessment would rely on manually reviewing code and my own limited knowledge of the deep inner workings of any said language—I will definitely be using analysis tools moving forward, especially in cases where the code base is far too large to manually review. I do strongly believe in manual review in addition though.

The concept of *zero trust* seems elementary at the end of a course like this. It is fundamental to the field of software security, and therefore seems like such a simple concept. From requiring users to have two-factor-authentication to validating all input no matter how small—all are examples of a zero trust mindset.

But all of these ideas are build-phase related ideas—do we just expect all developers to just know what to do, and what security standards to adopt? In the design phase, we can create a security policy to ensure that everyone involved during the build phase knows exactly what standards to adopt. I believe that security policies are an incredibly valuable resource during development. I can also see why a security policy may not be invested in though. It isn’t “fun” perse. Not many developers I know would be excited to create a security policy. I genuinely believe however, that this ‘chore’ will do nothing but benefit the project and the development team.