Angelica Hayes

CS-405 Secure Coding

Professor Olga Mill

August 12, 2021

# 8-2 Journal: Portfolio Reflection

Developing secure code is important. A simple statement, but the concise nature of the statement carries with it just how important it is. In today’s world, everything is connected, and this means everything is at risk in some way. Limiting the number of vulnerabilities within code limits the avenues of attack for malicious threat agents. A great way to look at it is an analogy. You are trying to cross a lake and there are two boats at the shore. One boat is full of holes, and the other only has one hole. Which boat would you logically think has a chance of making it across the lake? The boat with only one hole would be best as you would have a much easier time preventing water from entering through one hole versus multiple. This concept of limiting vulnerabilities by adopting a secure coding standard will be remembered and fully adopted going forward.

Evaluate and assess the risk of a codebase that is full of buffer overflows or script injections. These are huge risks to the code base and need to be rectified immediately. Being able to effectively evaluate and assess the risk of a system based upon its codebase is crucial to developing effective secure code. Furthermore, if this evaluation and assessment are done during the development process you can mitigate many common mistakes. This mitigation provides huge cost benefits to the development project as it limits the time fixing mistakes as fewer mistakes are made because of this process.

Zero Trust is a simple concept. No one is trusted, zero zip nada. As cloud computing has become more commonplace, security teams are finding it harder to “trust” who should or should not have access to a network. Zero Trust Security is based upon the concept that every single user and device is not trusted by the network. Each user must verify their identity and validate their device before they can gain access to the system. Furthermore, their access can be intelligently controlled to limit access to critical systems if suspicious activity is detected.

Security policies are the backbone of any security system. These policies outline specific guidance for employees and customers on the steps to practicing secure use computing. Normally, these include guidance on things such as passwords, device usage, how information should be classified, proper internet use, and even physical security recommendations. These written policies provide users with a playbook on how to remain safe as they work. Additionally, these policies have the added benefit of setting up patterns of behavior that can be tracked with intrusion detection software and hardware. If a user’s “normal” behavior starts to become erratic and stray from the norm, the intrusion detection software or hardware would flag this behavior.

Each of these topics holds significance to what secure coding is. Each topic is only one piece of the puzzle, and only when all of them are utilized and brought together does the whole picture become clear. Over the past eight weeks, I have become fully immersed in the practice of secure coding. Each week I learned a new concept or strategy that could be added to my list of development best practices. I think one of the most important projects for me was the case study. I focused on the attack at the Marriot International Starwood resort breach. This case opened my eyes to the reality of just how compromising information can be in the wrong hands. Every day we experience the world through computers in some measure. It is my job as a developer to ensure that the code on those computers is as secure as possible because one day my code could be the difference between business as usual or it could be responsible for the next big breach.