Shawn Ciaurro

Professor Spencer

CS 405

24 August 2025

Secure Coding Reflection

This term I my views on security and secure coding practices had a significant shift. The assignments and concepts of the course moved me to a “security in all stages of production” rather than incorporating security measures after production. Martin Fowler (2019) emphasizes that web security should be incorporated in the design because of the fact that many vulnerabilities arise from simple security oversights like poor input validation or inadequate session management. I learned that it is important to develop with strict coding standards in mind and that meeting these standards, as well as adhering to best practices and regulations, is essential for a legally compliant and secure software.

I learned to evaluate risk and cost-benefit of mitigation through analysis of the cost of mitigation tools and techniques and the risk that some security concerns pose. For example, SQL injection attacks pose a very high risk to a system’s security, are highly likely, and can be mitigated with some simple techniques such as input validation and sanitation and avoid direct system calls. The benefit of acting early on security risk is saving a system from more costly repairs later and the possibility of a tarnished reputation in terms of user trust. IBM (2024) reports that the average cost of a data breach in 2024 was $4.88M. This fact paints a picture as to how early intervention can save a lot of financial burden.

I was convinced that the Zero Trust approach to security was the best policy to adopt to avoid internal attacks and make external attacks much more difficult. A zero trust policy would have prevented internal breaches like the Capital One data breach in 2019 where a former Amazon Web Services employee was able to exploit misconfigured permissions to access over 100 million accounts containing sensitive financial user data. External breaches would be made a lot more difficult with road blocks on each step of the way rather than free access after bypassing perimeter security. Key concepts in the Zero Trust policy are to abide by the principle of least privilege, prioritize authentication, authorization, accounting, and treat DB access by services as untrusted until authenticated and authorized. Through the course I applied these concepts and adopted them to my personal security policy that I can bring with me in all of my future coding endeavors.

Clearly defined security policy is where a trusted and secure system begins. I would recommend that any developers or organizations should have a clear set of guidelines that are understood and referenceable while any software development takes place. I recommend policy that incorporates automation and testing through production with the use of audits, reviews, tools and static testing. Encryption in all three states of data (in flight, in use, and at rest) should be applied to secure data in the appropriate encryption level for the system’s use and context. Clearing sensitive data from memory and proper file management should be followed as a rule. Adopting these policies and following the rest of the recommendations of risk analysis and secure coding through the SDLC will make for a successful policy and produce secure software.

References

Fowler, M. (2019, May 1). Web security basics. Martin Fowler. <https://martinfowler.com/articles/web-security-basics.html>

IBM. (2024, July 24). Cost of a data breach 2024: Financial industry insights. IBM. https://www.ibm.com/think/insights/cost-of-a-data-breach-2024-financial-industry

Wired. (2019, July 30). The alleged Capital One hacker didn't cover her tracks. Wired. Retrieved from [https://www.wired.com/story/capital‑one‑hack‑credit‑card‑application‑data](https://www.wired.com/story/capital%E2%80%91one%E2%80%91hack%E2%80%91credit%E2%80%91card%E2%80%91application%E2%80%91data)