Oscar Rosa

CS 405

Southern New Hampshire University

06/26/2025

8-2 Journal

### **Portfolio Reflection**

This course has emphasized that security cannot be treated as an afterthought in software development. The principle of “don’t leave security to the end” has become central to how I approach coding. Instead of reacting to threats, I’ve learned the importance of proactively building in protections from the very beginning of the development process. This mindset ensures not only secure software but also reduces the cost of fixing issues that would otherwise emerge late in the development life cycle.

Adopting secure coding standards has shown me how consistent guidelines help reduce vulnerabilities and improve overall software quality. Coding standards ensure that every developer adheres to the same security expectations, reducing the risk of human error. These include practices such as input validation, memory protection, exception handling, and following least privilege principles. By implementing these rules early, we prevent issues from spreading into production environments.

One of the most valuable lessons was understanding how to evaluate and respond to risk. The NIST SP 800-30 guide helped me see how risks are measured by their severity and likelihood, and how remediation costs influence response decisions (National Institute of Standards and Technology, 2012). For example, mitigating a high-risk threat with a moderate fix is often more practical than delaying and facing a breach. Additionally, learning the financial aspect of risk assessments helped me understand their cost-effectiveness. Risk assessments may range from a few thousand to tens of thousands of dollars, depending on the organization’s size and complexity (Network Assured, 2024).

The zero-trust model changed the way I perceive security architecture. Traditionally, internal networks were trusted once accessed. However, zero trust treats every user and device as a potential threat until proven otherwise. This leads to the implementation of strong identity verification, continuous authentication, and minimal access permissions. Zero trust makes systems more resilient to insider threats and compromised credentials (Cloudflare, 2025). Although it can add more security checks, it significantly increases data safety in modern, distributed networks.

Lastly, writing and implementing security policies helped reinforce the value of structure and consistency in secure systems. Policies that govern encryption at rest, in transit, and in use ensure that sensitive data remains protected at all times. Similarly, frameworks like authentication, authorization, and accounting (AAA) provide strong identity management and track user actions, enabling visibility and control in any system. These strategies have reshaped how I approach coding projects. I now consider security a continuous effort integrated into each development phase rather than a final checkpoint. I plan to apply these best practices in all future software development efforts, whether academic or professional.

### **References**

Cloudflare. (2025). *What is Zero Trust security?* Cloudflare Learning. Retrieved June 27, 2025, from <https://www.cloudflare.com/learning/security/glossary/what-is-zero-trust/>

National Institute of Standards and Technology. (2012). *Guide for conducting risk assessments (SP 800-30)*. U.S. Department of Commerce. <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf>

Network Assured. (2024, August). *How much does a cyber security risk assessment cost today?* Retrieved June 27, 2025, from <https://networkassured.com/security/how-much-cyber-security-risk-assessment-cost/>