**8-2 Journal: Portfolio Reflection**

Corey Sampson

Southern New Hampshire University

CS405: Secure Coding

Professor Yi Wang

April 25, 2025

**8-2 Journal: Portfolio Reflection**

**Adoption of a Secure Coding Standard and Avoiding End-Stage Security**

One of the most critical lessons from this course is the importance of integrating security from the beginning of the software development lifecycle, rather than treating it as an afterthought. Adopting a secure coding standard, such as the SEI CERT C++ standard, ensures that developers follow consistent, security-focused practices that mitigate common vulnerabilities like buffer overflows, integer overflows, and race conditions. In our coursework, we examined how early adoption of secure practices reduces the technical debt and saves time and cost compared to retroactively fixing vulnerabilities discovered later during testing or post-deployment.

Secure coding should be a proactive practice embedded into DevSecOps pipelines. As illustrated in our resources and throughout our projects, tools like static analysis (i.e., SonarQube) and secure code reviews at each commit or merge help maintain this standard continuously. By shifting security left, teams are not only able to detect issues early but also reinforce a culture of security awareness among developers in the future.

**Evaluation and Assessment of Risk and Cost-Benefit of Mitigation**

Effective cybersecurity also involves making informed decisions about which risks to prioritize. Our exploration of risk management frameworks emphasized both qualitative and quantitative approaches, such as the NIST Risk Management Framework (RMF), which evaluates threats based on their likelihood and impact. One case study we reviewed highlighted that not all risks are worth mitigating and that some are acceptable given the business context and available resources.

**8-2 Journal: Portfolio Reflection**

A strong takeaway was the use of cost-benefit analysis when deciding on mitigation strategies. For instance, implementing two-factor authentication might carry initial overhead, but the reduction in risk from credential theft justifies the investment. Organizations must weigh the potential cost of a data breach, including legal, reputational, and operational damages, against the cost of proactive security controls.

**Zero Trust Architecture**

This course also covered the evolving model of Zero Trust, which assumes that threats can exist both outside and inside the network perimeter. Unlike traditional perimeter-based security, Zero Trust requires strict identity verification for every person and device attempting to access resources, regardless of their location within the network. The core principle, “never trust, always verify”, has become especially critical with the rise of remote work and cloud-based infrastructures.

Implementing Zero Trust involves practices like least privilege access, continuous monitoring, and micro-segmentation. Zero Trust is not a single product but a comprehensive strategy requiring a layered defense approach. For example, implementing role-based access controls (RBAC) ensures users can only access resources essential to their duties, thereby limiting the attack surface.

**8-2 Journal: Portfolio Reflection**

**Implementation and Recommendations of Security Policies**

Implementing a strong security policy is foundational to managing risk across an organization. Policies must address authentication, access control, data protection, and incident response, among other areas. Our security policy for Green Pace highlights how policies should be tailored to organizational goals and risk profiles while aligning with compliance standards like ISO/IEC 27001 or HIPAA. An effective policy should also be dynamic, updated regularly based on new threats and technologies, and supported by ongoing employee training. One key insight from this course was that even the best-written policy fails without organizational buy-in and proper enforcement mechanisms.