**8-2 Journal: Portfolio Reflection**

Astrid French

Department of Computer Science, Southern New Hampshire University

CS-405-12437-M01 Secure Coding

Prof. Kaan Esendemir

20 October 2024

Secure coding is an activity and a set of guidelines that make developers aware of the security measures that should be integrated from the start of the Software Development Life Cycle (SDLC), rather than leaving it to the end. This proactive approach helps prevent vulnerabilities and threats. Secure coding standards reference several established guidelines, such as:

1. **SQL Injection** – Protects against unauthorized database access by ensuring that SQL queries are utilized safely.
2. **Data Type** – Ensures data is used correctly to prevent errors and overflow.
3. **Data Value** – Ensures that input data meets expected formats to prevent processing harmful data.
4. **Preprocessor** – Organizes the code and reduces dependencies and compilation issues.
5. **Integers** – Ensures that improper arithmetic operations are avoided to prevent overflow and underflow.
6. **String Correctness** – Ensures that strings are validated to prevent overflows.
7. **Expressions** – Ensures the use of correct formulas and expressions to prevent bugs.
8. **Assertions** – Detects logical errors to prevent unexpected issues during runtime.
9. **Memory Protection** – Restricts access to specific memory locations to enhance security.
10. **Exceptions** – Provides a controlled way to handle errors, preventing application crashes and maintaining stability in case of unexpected conditions.

By incorporating coding standards and carefully applying them during development, software security can be significantly enhanced, reducing both the costs and time required for fixes later on. Additionally, it is important to use unit testing to verify that each standard is implemented correctly, ensuring the program functions as intended.

Mitigation is a method used to identify and address harmful elements in a system. It can be measured by assessing the impact of vulnerabilities and analyzing risks, including the likelihood of their occurrence. This involves prioritizing risks by ranking them from high to low, determining whether they need immediate attention or can be addressed later. Through mitigation, customers can train their employees on how to prevent incidents and how to respond if they occur. It also allows for cost estimation to resolve these issues. The benefit of this assessment is to reduce the risk of exploitation by hackers and to prepare for appropriate responses if an incident happens.

Zero trust is an important method for companies to adopt, emphasizing that no one should be trusted with sensitive data. By implementing zero trust, a company not only reduces the risk of its employees stealing data or accidentally leaking information to hackers through phishing emails, but it also builds trust with customers who use their products. This approach typically involves enterprise architects, security architects, IT staff, and IT security leadership. It ensures that each individual has limited access to data, with continuous monitoring in place. By implementing these measures, the company can establish stronger security controls.

My recommendations for security policies include first training employees on the importance of secure coding standards and unit testing during development. They should also be made aware that security measures should be applied from the start of the Software Development Life Cycle (SDLC). Additionally, after the software launch, it is essential to have a dedicated team to protect the system through the implementation of the AAA (Authentication, Authorization, and Accounting) model and layered security protections. Regular security checks should also be conducted, including assessments by white hat hackers. Furthermore, it is important to monitor software expiration to ensure safety, including maintaining encryption and decryption software, antivirus programs, and other security tools.