CS-405-10861-M01 Secure Coding 2024 C-3

8-2 Journal: Portfolio Reflection

Computer Science Department

Southern New Hampshire University

Eric Farkas

6/30/2024

**Adoption of a secure coding standard, and not leaving security to the end**

Creating a secure coding standard is essential in the digital age. Cyber-attacks are sophisticated and coordinated, using well-established and ingenious new methods. The development community must be responsible for creating software and systems containing strong security measures, which should be established during planning and development. Putting them off until development is nearly finished will lead to oversights and significant rework. Establishing a secure coding standard creates a development culture that prioritizes secure coding and keeps security at the forefront of developers' minds. This culture results in safer and more robust releases.

**Evaluation and assessment of risk and cost benefit of mitigation**

When planning a secure coding standard, consider the STRIDE threat model. STRIDE is a mnemonic for Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. This model arms developers with an easy-to-recall representation of the common risks. The elements of STRIDE will directly correlate with the secure coding standards. This alignment helps to identify all associated risks and the remediation strategies quickly. Developers can immediately address the risk, reducing future rework and refactoring. This proactive approach also reduces risks identified in testing. This approach redistributes both monetary and resource-based development costs. All rework and manual testing efforts focus on functionality and features, as security has already been addressed. The finished system will be more robust with minimal vulnerabilities at a similar cost and timeframe. Measuring the cost of reputation damage is difficult, but launching software or a system that leaks data due to a vulnerability can significantly impact customer confidence and future business.

**Zero trust**

Zero trust overcomes the false pretense that everything behind the corporate firewall is safe. This model assumes breach instead of trust and verifies each request as though it originated from an unknown source. Zero trust teaches developers to "never trust, always verify." This approach embraces remote users and distributed architectures required in modern organizations. This method extends to user accounts, devices, applications, and data anywhere within the organization.

**Implementation and recommendations of security policies**

Implementing security policies should involve cross-functional discussion and planning. It is important to understand and prioritize an organization’s resources for protection. The implementation should also include a continuous improvement plan to ensure the policy keeps up with the latest threats and vulnerabilities. These policies should be formally documented and stored in a private repository available only to teams involved in organization security and provided to customers when requested with an agreement of non-disclosure. Training should be established for any developers with deficiencies and to keep competent developers abreast of the latest exploits and techniques to mitigate them. This implementation method creates the foundation for an organizational culture that prioritizes security. Peer review of code helps the development team by placing a second set of eyes on everything. The same should apply to any third-party code supporting the system. Third-party code and methods should not be implicitly trusted in environments where security is essential. Code obfuscation should be considered for any applications residing on end-user equipment. This practice makes it more difficult to reverse engineer applications and discover opportunities for exploitation. Threat modeling methods further improve the effectiveness of reviews throughout the development and testing process. It also allows the targeting of specific threat profiles that are specific to your organization. This is one of many automated tools that can be added to your CI/CD pipeline to improve the thoroughness and efficiency of code review.