Connor Bailey

CS-405: Secure Coding

Professor Battle

26 October 2025

Module Eight: Journal Portfolio Reflection

Adopting a secure coding standard helps to ensure that security is embedded into the SDLC (Software Development Lifecycle) from the beginning rather than treating it as an afterthought. Standards such as those developed by SEI CERT and OWSAP provide developers with concrete and enforceable rules to prevent commonly known vulnerabilities. Integrating these standards early in the process allows teams to identify and mitigate risks during development. This reduces the cost and complexity of fixing vulnerabilities later, saving time and resources which are always limited. By emphasizing security during all stages of development, developers can create software that is more resilient against evolving threats and cyberattacks. Additionally, maintaining consistent coding standards allows teams to collaborate more effectively and produce predictable and maintainable code. This not only helps to protect users, but also demonstrates commitment to long term risk management and software quality.

Evaluating risk is an important part of secure software development because it helps to prioritize which vulnerabilities pose the greatest threat and which mitigations are most cost-effective. Every security control carries a cost, whether it is time, performance, or resource utilization, and these costs must be weighed against the potential damages of an exploited vulnerability. Structured risk assessment frameworks allow developers to identify assets, assess threats, and determine the impact of each risk. By identifying these elements in a software project, software development teams can make informed decisions about where to allocate resources for the maximum-security benefit.

The Zero Trust security model operates on the principle of "Never Trust, Always Verify". The model assumes that no user or device should automatically trusted, whether inside or outside of the project’s network. Zero Trust safeguards digital infrastructure through several different components that overlap and work together to secure the system. These components can include Identity and Access Management, Network Security, Micro segmentation, End Point Security, and Data Security. In practice, Zero Trust segments networks and enforces the principle of least privilege across the scope of your project. Every request, every query, every command that is run must be authenticated against, and no user or developer is assumed safe. By verifying every connection and access control, organizations can ensure that sensitive data and systems are only ever accessible by verified authorized users.

Implementing effective security policies establishes the framework that will guide all aspects of a software development project. Security policies should define expectations for encryption, authentication, access control, and data handling procedures. For example, enforcing policies for encryption at rest, in transit, and in use helps protect sensitive data like PII (Personally Identifiable Information) throughout the data’s lifecycle. Additionally, the Triple-A principles of Authentication, Authorization, and Accounting ensure that users are properly identified, granted the correct level of access, and that all activity is logged for accountability and audit. Successful implementation of these policies requires regular staff training, clear communication, and continuous evaluation of these policies as threats continue to evolve. Organizations should also integrate automated tools (like Cppchecker) and regular audits to verify compliance with established standards and identify gaps where mitigation is needed.