Jonathan Wolanyk

Professor Prasad

CS-405: Secure Coding

April 23, 2023

Module 8 Journal

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

Leaving security to the end of a sprint or project expands the scope of a project farther than necessary along with an increased likelihood for breaches and vulnerabilities to be overlooked. Secure coding practices should be implemented early and alongside automated tools to identify vulnerabilities and weak points in a system. Identifying and tracking down issues in an efficient manner is difficult to do in any scenario, but the longer a team waits, the more difficult damage control becomes.

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

Vulnerabilities should be assessed in a way that determines the likelihood for attack from a specific vector along with the damage such an attack would cause on a team's reputation, finances, and legal culpability. Sensitive information must be protected, but it is important to prioritize the implementation of mitigation techniques based on the likelihood and severity of the vulnerability.

**Zero trust**

From a developer's view, no user should be inherently trusted when accessing or using a system. The responsibility to prove that a user should be able to request and change information lies on the user, not the server. This is the core principle of zero trust in software.

**Implementation and recommendations of security policies**

Implementing, maintaining, and following a security policy is the first step to coding a secure manner. A development team needs to commit itself to coding in a secure way, and the methods in which the team does so should be consistent across team members. As a result, security policies are used to provide consistency and best practices for developers in the DevSecOps lifecycle.