# CS 405 Module Eight Journal

Malik Spruill

Professor Tam

CS-405

02 March 2025

Throughout the semester, I’ve had an opportunity to learn the significance and impact in adoption of secure coding standards as well as what it means to not leave security to the end. Integrating security measures from the beginning of software development, this is needed to prevent critical and costly issues later. Adopting secure coding standards, which provide guidelines for safe implementation of functions like memory management and data representation, is a key aspect of this proactive approach as I’ve learned throughout the semester. For example, implementing role-based access control ensures that employees have permissions strictly aligned with their job requirements, restricting access to sensitive information.

Additionally, employing strong Multi-Factor Authentication (MFA) mechanisms in ways of what a user knows (password), what a user has (secure token), and what a user is (biometric fingerprint) can significantly deter unauthorized access. Also, adopting continuous monitoring of networks in the beginning, to detect unusual activities, enables early identification of potential threats, allowing organizations to respond quickly and maintain control over their security. By adhering to these practices throughout the development process, organizations can prevent leaving security to the end.

When evaluating and assessing risk and cost benefit of mitigation, there were a few key points we’ve learned throughout the semester as well. For instance, I’ve learned that security issues require evaluating both the risks of inaction and the cost-benefit of proactive measures. An example of this, failing to address SQL injection vulnerabilities can lead to data breaches, exposing sensitive user information and more. The consequences of these effects are reputational damage, financial degradation, and more. Conversely, investing in secure coding practices by incorporating trainings, audits, and layers of security in various components of the software architecture, this may require additional development time but ultimately prevents costly breaches, reduces downtime, and improves system reliability.

Whether the concern is whether internal systems are vulnerable from the inside or the outside from external malicious users, the correct answer is both. As in, it’s imperative to develop a zero-trust framework to better secure systems entirely. Learning more about zero trust throughout this semester, zero trust means that no user, device, or application should be implicitly trusted.  For example, we wouldn't want all employees to have an entire breadth of access to services like ones that are outside the scope of what’s required of them.  This concept has changed how I view access. Continuing, I initially assumed all employees having access was fine for development, debugging, or being able to assist someone with needing access eventually. Now I know this is not conventional. Authenticate, then authorize the individual based on what is explicitly intended for them to access.

Overall, implementing security best-practices and policies establish clear guidelines for data protection, access control, and threat reduction to reduce vulnerabilities. I now look to include enforcing role-based access control, using encryption for sensitive data, and developing with best-practices in mind to prevent introducing exploits. That way, companies can continue with business as usual.