Timothy Rainey

4/28/2024

CS 405

**8-2 Journal**

Learning about the importance of secure coding standards opened my eyes to how essential it is to embed security right from the start of software development. It makes sense now that waiting until the end to address security can lead to more vulnerabilities and increased costs in fixing them. The idea that "an ounce of prevention is worth a pound of cure" rings especially true here, as integrating practices like input validation and error handling early can prevent major security issues later on.

Understanding risk assessment and cost-benefit analysis in cybersecurity was particularly striking. It’s about not just identifying potential threats, but also efficiently prioritizing mitigation based on the impact and likelihood of these risks. This strategic approach helps in making informed decisions about where to invest in security measures.

The concept of zero trust was a real game-changer in how I view network security. The "no one is safe" mantra shifts the traditional security focus from merely defending perimeter breaches to ensuring that every access request is authenticated, authorized, and continuously validated, regardless of its origin. This approach seems crucial in today's environment where threats can be as likely to come from inside as from outside the network.

* **Unit Tests**: I learned that unit tests are not just about functionality but are crucial for security as well. They help catch vulnerabilities early in the development cycle, which can save a lot of trouble down the line.
* **Defense in Depth**: This layered approach to security, where multiple defenses are stacked to protect data, means that if one layer fails, others will still function as safeguards. It’s a robust method to ensure comprehensive security coverage.
* **Coding Standards**: Consistency in coding not only helps in maintaining the code but also in securing it. Standardized code is easier to review and less likely to contain security flaws.
* **KISS (Keep It Simple, Secure)**: The principle of keeping systems simple to reduce complexity and the potential for errors (including security vulnerabilities) was a revelation. It seems obvious now that simpler systems are easier to manage and secure.