Adam Koenig

CS-405

8/25/2024

8-2 Journal: Portfolio Reflection

Adopting a secure coding standard at the beginning of development is important. Code should be developed with security in mind, not left until the end. Refactoring code that hasn’t been designed with security in mind is time consuming and expensive. Ensuring that all code written follows the secure coding standard before it is put into production reduces the need to refactor the code. The secure coding standard shows developers examples of secure code, such as how to write queries to prevent SQL injection and how to utilize strings without buffer overflow.

When developing security measures, a good strategy is to consider the approach and motivations of potential attackers. Knowing what motivates attackers to focus on your business can help determine what areas they are likely to target. That in turn lets developers know where to prioritize security resources. The financial risk of different types of attacks needs to also be weighed with the cost to mitigate the attack.

Zero trust is the idea that no one is trusted, even once they are authenticated on the network. Under a zero trust policy, requests made inside the network should be authenticated and the network itself should be segmented. Data moving between the segments of the network should be encrypted. A triple-a security policy should be employed, where every employee should be authenticated, authorized, and their activity should be audited. Multifactor authentication should be employed to prevent attackers from gaining access with only username and password, and critical functions should be restricted to registered devices.

Cybersecurity goes beyond just creating and sticking to a secure coding standard. A comprehensive security policy is needed to maintain security. A defense in depth approach creates multiple redundant layers of security that ensure data protection even if a single layer is breached. Automation should be employed to facilitate security maintenance. Many applications today utilize third-party software. Static testing can be run periodically to identify known vulnerabilities in third-party APIs. Once a vulnerability is discovered, steps can be taken to mitigate the vulnerability. Unit testing can also be employed to preserve functionality and security. When a new class or function is created, a unit test should be created along with it. Then, every time that code is modified, the unit test can be run to ensure that the modifications preserve the functionality and standards of the original code.