John Elbogen

CS-405

6/22/2021

**8-2 Journal: Portfolio Reflection**

This course has provided me insight on secure coding and security protocols that are relevant present day. Secure coding standards at a first glance can appear overwhelming. Often a new developer may look and wonder how they could match the coding standards and principles in their own work. After the work in this course, I take the approach that these are guidelines to create a standard for guidance. Rather than stating that there is one correct approach, the standards typically focus on avoiding dangerous or incorrect implementation of different concepts. For example, one standard discusses data sanitation when passing data into a sub-system. However, it does not state that there is one correct way to accomplish this. It more provides guidelines on incorrect implementation and the risks. Understanding this concept can allow an individual to use the guidelines and standards to improve their own code.

Security should not be left to the end. Consider a project in development. As the development continues the project becomes more complex. If one waits until the end of this process to being implementing security, the process is more difficult as there will be many areas to unpack. Also, if security implementations cannot coexist in the current setup, then backtracking must occur. Now consider early implementation of security. Security can be tested synchronically while testing the systems code. Also, the development process can consider what the program will need to accomplish to maintain security.

The cost benefit of mitigation and risk are the big contention points for secure coding. Looking at the cost of implementation, there is an increased time and expertise cost to add security early. This cost is outweighed by the mitigation gains. Depending on project size and value of sensitive data within a system, a product can prevent attacks which cost money and prevent damage that can be irreversible (like customer trust). Considering risk and evaluating risk is an important process still, since a project may have time constraints. In the real-world project, there may be deadlines and concessions made in the security department. Evaluating risk and assessing priorities can help ship the best possible product.

Zero-trust is a large component as cloud storage and computing becomes the mainstream go-to for applications. Using reverse proxies and confirming each device permissions is one implementation of zero-trust. I would recommend that security policy implementations follow an organized, clear, and simple format. When we consider, keep it simple, this can apply to all facets. When implementing the individuals involved in the implementation can contribute more if they can adopt the policies more easily.