CS-405

Zach Beachy

8-2 Journal

To begin, I think it is very important to adopt a secure coding standard and not leave security to the end. In both my professional career and previous classes I’ve taken, it has become abundantly clear that the pitfalls of avoiding security methodology far outweigh the perceived time benefit from avoiding it until later. Incorporating secure coding standards throughout the development cycle produce nothing but benefits for both the application in question and the development team in their productivity and efficiency. Part of the secure coding standards that should be adopted are rigid practices to evaluate and assess risk and the cost benefit of mitigation. While evaluation and assessment of risk is important to ensure the mitigation of threats and in protecting a company’s wallet , the world of software development is often a design process of trying to adequately balance time and quality of product. This is why many companies can fall into a trap by implementing “feature-creep”, where the addition of more and more features to an application can delay the release of the product further and further away, sometimes to the point of never seeing a production release. A portion of these features should be security features, where the team must assess the vulnerabilities in the code-base and determine the cost in both time and in traditional means to secure or correct that vulnerability.

Outside of mitigation and evaluation, another consideration that is extremely necessary in the secure coding sector is the concept of zero trust. Using methods such as multi-factor authentication every single time a user tries to access a system ensure that by never assuming access and forcing the user to authenticate themselves, the system can remain far more secure than previous methodologies.

All of these factors: Adoption of a secure coding standard, not leaving security to the end, evaluation and assessment of risk, and zero trust, are extremely important to implement in any application’s life cycle. Combining all of these methods together is another example of practicing defense in depth, which throws a blanket of security over a system rather than leaving a few sole endpoints to be exploited. My recommendations on security policies would always be to push for the most secure methods applicable without sacrificing usability of a system. If the user of the system is given restrictions to the point of impacting their day-to-day operations, it can become a pain point for the user, which is only a net negative for the company. It is important to find a balance between having a system that is secure enough to protect from threats while not over-securing to the point of causing another set of issues in the operation of the sytem.