Brian Engel

Module Three Journal

As a software developer there are many things that should be done to solve security concerns and not just rely on adding them in later down the line. Such things include validating user input, encrypting sensitive data, updating dependencies, frequent and thorough testing for bugs, and incorporating validation of user credentials throughout the application. By designing these types of security into your coding, you have a huge jumpstart on security.

Traditionally security has been more of an afterthought in the software development lifecycle and has just been sort of thrown on top of an application instead of being designed into it. Since SDLC is leaning towards a more agile methodology lately, it is easier to incorporate and test security throughout the process. Security should be on everyone's mind in every stage, from planning through deployment. It should also be as layered as possible, since every layer has its own security concerns.

For transforming a DevOps pipeline into a DevSecOps pipeline, I would change a few things in the process of the SDLC. I would address high level security concerns in the initial planning stages. I would ensure that the coding was secure with encryption, input validation, and other good coding practices. In the testing phase the article brought up a really good idea of having red and blue teams with the red team focusing on offensive testing and the blue team defending against them. Once the application is done, I would have a team to monitor the application and check for updates on dependencies that might be at risk.

The suggested plan included in the article is a start to finish planning of the SDLC and sounds like the ideal thing to do. In the planning phase start with a high-level security plan and set up the infrastructure of the project and require user authentication and authorization depending on role in the project. In the coding phase require secure coding practices, industry standards, peer reviews, and unit testing. In the build phase splitting the repository and having access control so only authorized people can merge onto the main branch. In the testing phase instead of just functionality, test for penetration as well. In the release and deploy phase ensure access controls, authentication and authorization, and pushing and pulling containers from storage is secure. In the operate phase have continuous monitoring, analysis, and protection of the runtime environment. The plan sounds like a good all-around plan, although there might be areas that need more or less attention depending on the application, and I would use the plan as a starting point.

Jeganathan, S. (2019). DevSecOps: A Systemic Approach for Secure Software Development. ISSA Journal, 17(11), 20-27.