Module Three Reflection – Anthony Schissler

As a developer, we stand at the forefront for finding security vulnerabilities within our code we work with, and trying to find the most effective way to mitigate them. As a first principle, secure coding is a developers first action they should be taking to mitigate these concerns. Developing robust error handling counteractrs the nasty snowball effect that can occur what unhandled exceptions or edge cases appear. This also involves utilizing encapsulation for data types and structures, and prevents openings into data from being opened through a lack of data protection. Developers also are involved with testing methodologies to test stress points and ensure that code performs as it should during its development cycle.

Security in traditional software development lifecycles is tested and conducted during specific planned phases of development. Usually this takes places later in development, as testing tends to occur broadly closer to release than closer to the beginning. As the monumental shift towards Agile occurs however, adopting a DevSecOps pipeline from a DevOps pipeline just makes more and more sense to reduce risk.

In order to transform into a DevSecOps pipeline, there should first be a greater emphasis placed on adopting security measures and testing them earlier in development. Securely coding means keeping these potential issues in mind while developing, and not putting it on the backburner to be dealt with later during some far off testing phase. Changing the organization’s culture on adopting a more security-based approach to development is probably the hardest part. Once a dialogue has started however, you can then move to bring all of the different stake holding teams together to work on developing products that are secure from the start. Start by influencing the people, then influence the process, and then finally the technology, and that security based mindset will turn a DevOps team into a DevSecOps team.

Following through with introducing security to the people, then the process, and then the technology helps to establish a baseline for securing the entire DevOps life cycle. The article recommends introducing slow and focused changes in order to not introduce unnecessary friction. As it becomes gradually more accepted in the pipeline, adding the cycle of effective continuous security to the DevOps cycle is a natural next stop for adoption. By adding this constant cycle into the natural DevOps cadence, it reinforces and fosters continuous attention towards security concerns that can be rectified.

Adopting the technological factor is the last but also potentially the longest step in the article’s plan. Because there’s so many steps for planning, building, testing, deploying, monitoring, and scaling software, a variety of tools might have to be used. Ideally, tools would be used that could allow for continuous integration with follow on tools, allowing security concerns to be addressed smoothly through the pipeline. Adding too much complexity can draw people back to their old ways however, and because of that I think adopting a streamlined approach to the technological aspect is important for slower-to-adopt organizations.

The article laid out an effective plan overall. It addressed the issues DevSecOps personnel might face when trying to change to a more security focused environment, or even just away from a traditional waterfall project management system. Integrating it within the individual will allow for a change to happen with the process which will then influence one of the hardest aspects in adopting the right technology to integrate a continuous security and continuous integration pipeline into a traditional DevOps environment.