Matt Tranchina 11/12/2023

CS 305 Software Security 3-2 Journal: Reflection

It is my firm belief that software is only as good as how secure it is. With the world moving towards a remote access future, security should be among the highest concerns. Many development teams view security as an afterthought or attempt to implement security measures in the later stages of the software development lifecycle. Some even use the approach of “deal with it when it becomes a problem”. The reality of it is that as technology grows, so do the threats, and if a developer is not ahead of the curve, their application can fall victim to these threats.

There are many different software development lifecycles, such as Waterfall, Agile, and DevOps. The DevOps approach is gaining speed in the technology industry. “DevOps integrates and automates the work of software development (*Dev*) and IT operations (*Ops*) as a means for improving and shortening the systems development lifecycle.” ("DevOps", 2023). While this method speeds up production, it leaves the security factor on the back burner. DevSecOps (DevOps + Security = DevSecOps) is an alternate version of DevOps that “shifts security left”. This allows each team to implement security measures into their software delivery. This allows for the software lifecycle to still be shorter than other methods without compromising security.

To add security measures to transform DevOps pipeline into a DevSecOps pipeline, one must implement security measures at every stage of the lifecycle continuously. This is referred to as a continuous security pipeline. This allows for security to be apart of the design and engineering from the beginning stages, rather than at the end. Continuous Security (CS) implements security design and engineering, security testing and monitoring, and security risk management. All of which work with Continuous Integration (CI) and Continuous Delivery (CD). This process model allows for security to be included in all aspects of the lifecycle, including development and deployment. Having the development team and the security team working cohesively and continuously testing the applications’ integrity and security will ensure safer software, and this can be accomplished by following a set plan.

The article by Seetharaman Jeganathan illustrates a plan once the stage where security requirements and designs are done for the project. An organized plan is crucial when multiple teams are involved in one project. It is also important to know that the plan might deviate or change as the project progresses, as unforeseen problems might unfold. The plan illustrated by Jeganathan begins with starting with a “high-level rapid risk assessment for the new release and quantify the risks by evaluating the threat models.” (Jeganathan, 2019). Next would be to plan and secure the life cycle tools such as GitLab, Azure DevOps, etc. The next step would be to “Ensure user access keys, privileged service accounts, API keys, etc. are protected properly with privileged account security tools if exist, otherwise with a compensating control to meet the requirements.”, and finally, “define infrastructure protection controls and enforce segregation of duties.” (Jeganathan, 2019). Once all the criteria have been met, the rest of the plan can be implemented, such as code, build, test, secure and deploy. All while security is being implemented at each stage continuously by developers, ensuring a safer and more secure application.

**References**

DevOps. (2023, November 8). In Wikipedia. <https://en.wikipedia.org/wiki/DevOps>

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