Brad Jackson
CS405
Feb 26, 2023

Portfolio Reflection

Adoption of a secure coding standard is important to ensure any organization's information security. A policy to combat insecure coding standards begins with identifying gaps in the underlying architecture of the DevOps lifecycle. Adding security into this loop to standardize security vulnerabilities in code and policy in systems architecture creates a DevSecOps lifecycle. It is often the cost of standing up a SOC or application security team which deters companies from adopting secure coding standards; however, neither is needed to integrate static code analysis into a DevSecOps lifecycle. GitOps can be taken advantage of in order to add SAST tools like Codegate, Cppcheck, etc. into the build process. This will add time to the build process; however, it is a fundamental way to assure compliance with common coding standards These tools would scan the source code of an application and report any known malpractice and vulnerabilities. Automated gatekeeping processes should prevent applications with known vulnerabilities from being deployed in production until the findings are remediated.

In coding standards should be clearly defined based upon the programming language of the application. They should be presented in a document with examples of compliant and non-compliant code. They should also be presented with a rating of severity, likelihood, remediation cost, and a priority determined by the aforementioned metrics. Defining coding standards is a great baseline for a security policy but more is required for a complete policy. Security policy should include considerations for audit controls, encryption, and Triple-A

(authentication, authorization, accounting) to name a few. It should be cyclically re-evaluated and distributed to developers.

Information security requires multiple layers. This approach is called defense in depth. One consideration when implementing defense in depth is the concept of zero trust. Zero trust means the fewest possible number of subjects can access an object or resource. A process should be in place to evaluate user accounts and access at any given time. Ideally, this evaluation process would be recurring and automated. This would ensure that accounts that were transferred to other roles or of employees who left the company would have their access revoked.

It is important to protect data not just for the security of the data but to mitigate potential costs of falling out of compliance. Audit drives a lot of data governance practices within any company and is often the sole motivator for such practices. For this reason, organizations must also consider their reputation when implementing security policies. If a breach leads to millions of users' private data becoming public, the compliance costs may not break the company but their ill repute will.

At the lowest level, preventing threats begins at securing business-critical data. This data ranges from financial data, user data, or even intellectual property. It is important not to leave security to the end. Security can be overwhelming for organizations to implement but it is even more of an undertaking after infrastructure has already been established. Humans are responsible for weighing the options when it comes to implementing security policies and are also the most common point of failure when it comes to InfoSec. If we are not considering security when architecting systems or coding, we are destined to fail.