Jeremiah Kellam

CSD380 Module 12.2 Assignment

7/22/2024

Compliance: Case Studies

Two case studies were assigned to expand on from our reading. “Providing Compliance in Regulated Environments” and “Relying on Production Telemetry for ATM Systems” (Kim, et al.). With the first case study, the challenge is that servers appear and disappear which is not feasible to audit like the standard physical audits would be performed. Security measures and required documentation clash for the purpose of DevOps and its nature. The way that Shinn stated they worked around the issue is to automate the process so that information like screenshots and CSV files were compiled automatically and contained in a system that auditors could log in to and go through for audit evidence required in a specified time range (Kim, et al.).

Engineering requirements were derived from the forty-five CFR Part 160 legislation to find the specifics needed for what is required. Those requirements would determine what activities must be logged and retained for audit review. The DevOps Audit Defense Toolkit gives an end-to-end description of compliance and audits. The point of the case study is to show that building documentation allows Dev and Ops practices to meet in the middle and handle auditor requirements and improve risk assessments and mitigation (Kim, et al.). By taking advantage of automation and allow the tools to take care of the burden of collection, developers are able to engineer compliance into the build so that real-time monitoring can occur, and lower risk of non-compliance or possible penalties could occur.

The second case study was very short, but value added. An observation was made that code reviews were relied on to heavily to detect fraud. The head of the DevOps initiative for a consumer banking property determined production monitoring controls and automated testing paired with code reviews and approvals mitigate risks for both errors and fraud. A case of fraud was found from a backdoor put in by a developer in ATM machines that wasn’t found in code review, but in product monitoring and operations review. The main point of this case study is that separation of Dev and Ops can lead to vulnerabilities and that overreliance on code reviews shouldn’t continue as telemetry would allow visibility to detect and address both instances of errors and fraud. This case study gave a clear picture of the capabilities to be proactive rather than reactive in the processes that DevOps take to design, build, and maintain a service.

The main takeaway of both case studies is that Dev and Ops should not be separate. Automation and innovation will allow compliance and requirements to be met. Early integration would be required in order for compliance to be melded into a build and real-time monitoring of telemetry can allow for many things to be discovered from the data for both Dev and Ops. There will always be a need to trust but verify all things so it cannot be abused by developers or others with malicious intent.

Reference(s):

Kim, G., Humble, J., Debois, P., Willis, J., & Forsgren, N. (2021). *The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations*. IT Revolution.