Vanh Somchaleun

Module 12 – Case Study Review Report

<https://github.com/VanhSom/csd-380.git>

Companies must meet strict rules and regulations while staying efficient and agile in today’s rapidly changing tech world. Old-school auditing methods, designed for static, physical systems, often do not work well in modern, cloud-based environments where servers come and go automatically. This report explores two case studies from The DevOps Handbook that highlight the challenges of proving compliance in regulated environments and the critical role of production telemetry in detecting and mitigating risks. These examples highlight the necessity for updated auditing practices, improved collaboration, and the integration of advanced monitory tools to maintain compliance and security effectively.

The first case study, Proving Compliance in Regulated Environments, focuses on Bill Shinn, a security expert at Amazon Web Services (AWS), who helps big companies prove they follow rules in highly regulated industries. Shinn points out a big problem: traditional auditing methods, like checking a sample of physical servers and using screenshots or spreadsheets, do not work well in DevOps environments. These old methods cannot keep up in cloud systems, where servers are constantly being created and deleted.

To tackle this issue, Shinn proposes an iterative approach to control design, where auditors and DevOps teams collaborate to define audit requirements during each sprint. Companies can use telemetry system tools like Plunk and Kibana to give auditors access to real-time data to find the needed evidence without asking for manual reports. This makes the audit process faster and more transparent while reducing errors and security risks.

Shinn stresses transforming regulatory requirements into practical engineering tasks to address these challenges. For instance, to comply with HIPAA regulations, teams must identify which activities must be monitored, implement the required controls, and provide clear documentation on how these controls function. Tools like AWS CloudWatch make it easy to test controls and link audit evidence directly to the rules. The DevOps Audit Defense Toolkit also helps by providing examples of how the design controls and proves they work. This case study shows how modern tools and teamwork can help companies stay compliant without slowing down.

The second case study, Relying on Production Telemetry for ATM Systems, examines a DevOps project led by Mary Smith (a fake name) at a large US bank. Smith noticed that relying too much on code reviews and separating duties between teams is not enough to catch fraud. In one case, a developer added a hidden back door to the code used in ATMs, allowing them to steal cash. The fraud was detected not through code reviews but during a routine operations review, where telemetry data revealed unscheduled ATM maintenance modes.

This example shows how important production telemetry is for detecting problems. While code reviews and approvals are helpful, they are not enough on their own. Real Todd mongering gives teams the visibility they need to spot issues quickly. Smith argues that good monitoring can reduce the need for strict separation of duties or extra layers of approval, making operations smoother without sacrificing security.

Both cases show why companies need to update their compliance and security practices for modern DevOps environments. Old auditing methods designed for static systems do not work well in today's fast-moving, cloud-based world. By leveraging telemetry systems, companies can give auditors the data they need quickly and easily, helping them stay compliant without slowing down. These examples highlight that modernizing auditing and monitoring processes is a technical requirement and a strategic priority in today's regulated environment.

Reference:

Kim, G., Humble, J., Debois, P., & Willis, J. (2021). The DevOps Handbook (2nd ed., p. 312 - 313). IT Revolution.