Proving Compliance in Regulated Environments and Relying on Production Telemetry for ATM Systems: Lessons from DevOps in Practice

The evolution of DevOps in highly regulated environments introduces both immense opportunities and significant compliance challenges. The two case studies presented in Chapter 23 of *The DevOps Handbook*—"Proving Compliance in Regulated Environments" and "Relying on Production Telemetry for ATM Systems"—highlight how modern organizations are addressing regulatory requirements while maintaining the speed and flexibility that DevOps promotes.

Case Study 1: Proving Compliance in Regulated Environments

In the first case study, Bill Shinn, a principal security solutions architect at Amazon Web Services (AWS), discusses the hurdles enterprise customers face when proving compliance in dynamic cloud environments. Traditional auditing methods are rooted in legacy infrastructures, where static systems and manual reviews dominated. For example, auditors have been trained to ask for screenshots and CSV files for compliance verification, even in production environments with thousands of servers and ephemeral resources that auto-scale.

Shinn underscores the disconnect between regulatory expectations and DevOps realities. In DevOps environments, where infrastructure is code and systems can appear or disappear in seconds, traditional sampling methods become outdated. To bridge this gap, AWS works with auditors in the control design phase using an iterative process. Each sprint focuses on one control, defining exactly what evidence will be needed to prove that the control is working. This approach ensures that, when the system is in production, evidence is readily available.

One of the key solutions Shinn emphasizes is leveraging modern telemetry systems like Kibana or Splunk. By funneling logs, metrics, and configuration states into these systems, organizations can enable real-time, self-service access for auditors. Instead of requesting screenshots, auditors can directly query the telemetry data, which provides clearer, more consistent visibility into operational controls.

Moreover, Shinn discusses how regulations such as HIPAA must be thoroughly examined to distill actionable engineering requirements. The regulation's abstract requirements—such as ensuring audit trails for healthcare data—must be translated into specific technical controls. These might include AWS CloudWatch logs that track access to sensitive data, all of which must be integrated into a cohesive logging and monitoring framework that can produce audit evidence on demand.

The major lesson from this case study is that compliance cannot be treated as an afterthought or an external burden. It must be integrated into the development and deployment process. Working collaboratively with auditors, transforming compliance requirements into automated controls, and using telemetry as an audit tool all allow organizations to demonstrate compliance in real-time, even in the most dynamic environments.

Case Study 2: Relying on Production Telemetry for ATM Systems

The second case study, presented by “Mary Smith,” a pseudonymous leader of DevOps at a major U.S. bank, emphasizes the importance of operational monitoring and telemetry over reliance on static code reviews. Her story centers on a real-world incident where a developer inserted a backdoor into code deployed to ATM machines. This backdoor allowed the perpetrator to put ATMs into maintenance mode at specific times, enabling unauthorized withdrawals.

Despite existing safeguards—such as change approval processes and separation of duties between development and operations—the fraud was not caught through traditional code review or static analysis. Instead, it was uncovered by a team member during a routine operational review, who noticed an anomaly: ATMs in a particular city were being placed into maintenance mode at unexpected times. This real-time detection occurred even before a scheduled cash audit, preventing greater damage.

The lesson from this example is that code reviews and approval processes, while essential, are not foolproof against insider threats or subtle manipulations. Production telemetry—continuous monitoring of system behavior and patterns- is often more effective in catching real-time anomalies and potential security breaches. It reflects the principle that observability and feedback loops are critical components of a secure and resilient system.

Moreover, the case illustrates that attackers can bypass technical controls if they have enough motivation, opportunity, and access. Thus, relying solely on preventive mechanisms is not enough. Detection and correction capabilities must be integrated into live operations. By focusing on patterns in production data, organizations can uncover misuse more quickly and with less reliance on manual reviews.

Synthesis and Broader Implications

Both case studies emphasize a paradigm shift: compliance and security in the DevOps era depend heavily on visibility, telemetry, and collaboration. In the AWS case, telemetry systems serve as a reliable and scalable source of audit evidence. In the ATM case, they serve as the front line of defense for detecting and responding to fraud.

Additionally, both scenarios highlight the importance of cross-functional collaboration. Whether it involves involving auditors in sprint planning or training operations teams to identify anomalies, successful DevOps compliance strategies require a shared responsibility across roles.

Ultimately, the key takeaway is that automation, observability, and iterative collaboration are not just DevOps best practices—they are essential enablers for maintaining trust, integrity, and compliance in modern, fast-moving environments. Organizations that adopt these approaches will be better equipped to meet the challenges of regulation, security, and operational excellence simultaneously.