**Craig Dillon** [x00205790@mytudublin.ie](mailto:x00205790@mytudublin.ie)

Technological University Dublin

MSc. DevOps

Enterprise Architecture Deployment

CA1 Report

Executive Summary

Company A is a global provider of product X, which offers financial transaction solutions to the banking sector. The organisation is experiencing considerable challenges with its current product development and delivery cycle with long release cycles, quality assurance issues, communication problems and is suffers from a culture of siloed workloads. Development and Security teams, primarily based in Dublin, exhibit severe personality clashes, resulting in poor product quality and inefficiencies.

The company is under the leadership of a new CEO who has set out a vision of quarterly releases and improved competitiveness in the sector. Since taking the position the CEO has had lengthy discussions with key stakeholders and has highlighted the need for strategic realignment. The feedback from customers has been poor.

Background

Company A develops and offers product X to the global banking sector, a highly regulated industry that requires stringent security standards. The product is designed to facilitate secure and efficient financial transactions. However, the current operational situation presents several challenges that require comprehensive strategic intervention.

**Operational Structure:**

* Development and security functions are largely concentrated in Dublin, forming the core of the company’s technical efforts.
* Quality Assurance (QA) operations are located in Bangalore, a city renowned for a large and well developed I.T sector.

**Identified Challenges:**

* A culture of siloed workloads in the company, with distinct personality and communication issues between the Development and Security teams. This has led to inefficiencies, poor quality of work and disruption in development and release cycles.
* The current release schedule is once per year, whereas the target is once per quarter. This discrepancy means the company is working at a severe disadvantage as competitors can respond much quicker to market demands and technological changes.
* The dynamic between the Development and Security teams is particularly concerning. The personality clashes are obvious and create a challenging work environment, impacting morale across the entire company and is detrimental to the collaborative culture required to work efficiently.
* Metrics highlight an alarming level of technical debt resulting in poor product quality which creates at least 10 critical field issues per quarter. This has a serious impact on the company’s ability to meet SLA agreements and the end user experience. As a result, teams are spending more time firefighting than improving the product.

**Leadership:**

* A new CEO has been appointed and has extensive technical experience. Having taken the position 2 months ago, the CEO has held in-depth discussions with all departments and customers, highlighting the need for a radical approach to address the challenges faced.
* The CFO is from a banking background and offers critical insights into the banking sector and this knowledge will be essential in aligning strategic decisions with market needs.

**Customer Feedback:**

* The CEO has met with customers and the feedback has been exceptionally poor. The CEO has emphasised that any solution proposed should have a customer-centric approach.

Proposal Options

Studies have shown that there is a direct correlation between working climate and the ability of software development teams to effectively collaborate and solve problems, where a strong foundation of trust and effective communication have a profoundly positive effect on these teams, *Purna Sudhakar et al (2011), Açıkgöz et al (2015).* Further to this, the company also needs to deal with the complexities of a global team, where a toxic environment in one site has an impact on the effectiveness of the overall business. In a study looking at factors affecting global team dispersion, *Lee et al (2013)*, concluded that rigorous process standardisation can help overcome challenges associated with managing global teams.

## Option 1: Streamline processes and integrating security functions into the development process.

Encourage collaboration between teams by creating a streamlined CI/CD process which includes DevSecOps methodologies and QA processes as part of the overall pipeline.

**Activities:**

1. **Automation of Security and QA Processes**

* Embed security checks and procedures into the development pipeline. Ensure proactive identification of security vulnerabilities by continuous monitoring.
* Automate QA tests in the development pipeline with regression testing, integration testing and performance testing, for example.

1. **Cross-Functional Development Team**

* Break down the silo between the Development and Security teams and create a cross functional team where smaller self-organised groups leverage their relevant expertise to overcome technical issues.

1. **Encourage DevSecOps Culture**

* Provide training and workshops for staff to understand the principles of DevSecOps. Arrange leadership training for senior members to empower them in decision making, collaborative efforts and fostering a culture of shared responsibility.

## Option 2: Focus on soft skills

Focus on developing soft skills within the teams to mitigate communication and relationship issues between teams. Fostering effective interactions between teams will allow them to create effective workflows and improve product quality.

**Activities:**

1. **Soft Skills Training**

* Hire consultants who specialise in delivering soft skills training to hold workshops focused on soft skill development like conflict resolution, active listening and effective communication, for example.

1. **Team Building Events**

* Organise a range of team building events off site, with the option of some remote based events to include staff in Bangalore. Activities must be group based and consideration should be made to ensure staff are grouped with others who they might otherwise avoid.

1. **Conflict Resolution Training**

* Provide training in conflict resolution techniques and establish a process for addressing conflicts and finding agreeable solutions.

1. **Mentorship Program**

* Create a mentorship program where senior members of each team mentor and guide junior members of another team. Encourage knowledge sharing, mutual support and dialogue.

## Option 3: Implement accountability and performance monitoring frameworks

Enforce strict and regular performance reviews where staff and teams are held accountable for any shortcomings. The goal here is to apply pressure and attempt to break any deadlocks that exist between teams as a result of poor interpersonal skills or unwillingness to address underlying issues.

**Activities:**

1. **Performance Metrics**

* Create clear and measurable performance metrics for all teams. Match team performance and individual member performance to collaboration metrics and communication benchmarks. (For example, response times and closing times for issues opened)

1. **Mandatory Cross-Team Training Events**

* Arrange cross-team training sessions that encourage collaboration, communication and conflict resolution. Enforce participation for all team members.

1. **Leadership Empowerment**
   * Foster a culture of empowerment for senior team members to actively address poor communication or poor interpersonal interactions by providing leadership training and having teams perform regular check-ins to identify challenges.
   * Hold team leaders accountable for maintaining an open and collaborative environment.
2. **Conflict Resolution Process**
   * Implement a strict conflict resolution process where there is a clearly defined escalation path to resolve communication problems and disagreements.

Conclusion

After analysis of the current situation, it is clear the issues are a result of strained relationships between the Development and Security teams and siloed processes resulting in poor product quality with slow development cycles. Having evaluated several potential strategies, implementing DevSecOps principles and practices is the recommended solution. This will encourage a cultural shift that emphasises collaboration, communication and shared responsibility for security throughout the development cycle.

The proposed solution follows the DevSecOps principles of “shift left” where security tests and implementation happen much earlier in the development lifecycle by automating testing and codifying security policies which ensures an integrated approach to implementing security functions. Continuous monitoring and feedback will contribute to a resilient security framework that is continuously improving.

Along with the technical shift to bring both development and security functions together, there is a cultural shift that brings a shared responsibility between groups who have so far been polarised. By moving security into the earlier stages of the development pipeline, the members of the Security team can effectively develop security solutions in an integrated manner. By implementing DevSecOps principles and practices, the organisation can improve its security posture, streamline development and follow a common industry approach to security, removing bottlenecks and enabling innovation and rapid product delivery.

What is DevSecOps?

DevSecOps is an expansion of the DevOps methodology that “places security at the forefront of requirements to avoid the costly mistakes that come from treating security as an after-thought”, *Carter (2017).* Since DevSecOps builds on DevOps methodologies, they share many of the same principles. According to *Morales et al. (2020)* the principles are;

* **Collaboration** - Developers, operators, engineers, end users, customers, and other relevant stakeholders are allowed to be part of decision making and work progress. This allows transition from development to operations to occur with fewer or no blockers.
* **Infrastructure as Code** – Deployments are stored as code in a version controlled repository, guaranteeing standardisation and reproducibility and allowing for environment parity (test, staging and production being the same, for example).
* **Continuous Integration** – Enables automated testing, early detection of security vulnerabilities and consistent application of security standards for a streamlined and secure development environment.
* **Continuous Delivery** – CD (in this example) is automated up until the staging environment. While it could be automated into the production environment, manually verifying and confirming the change to production is the most common form of CI/CD in highly regulated environments.
* **Continuous Deployment** – CD, involves the automated transfer of software into a production environment. This approach relies on rigorous testing but still carries a high risk of defects entering the production environment, this form of CI/CD is not common in highly regulated environments.
* **Environment Parity** – Parity between environments, Test, Staging and Production reduces potential problems in the production environment, although integration testing should still be carried out.
* **Automation** – Automation allows for consistent and repeatable security testing and removes the potential for human error. This also removes manual tasks from developers and allows them to focus on writing code.
* **Monitoring** – Continuous monitoring improves the pipeline and software under development. Monitoring based on performance metrics and testing results in faster alerting and response times.

While there may be an initial slowdown in development speed when implementing DevSecOps, the end goal is to streamline the process and speed up the development cycle by shifting security left in the pipeline and integrating it as a core component in the development process.

DevSecOps has seen great success in the Banking Industry, for example Maverix implemented DevSecOps processes for a large European bank which brought many benefits including, greater transparency, reduced time for security analysis, widespread adoption of security practices, reduced manpower required for many development processes and timely detection and remediation of security vulnerabilities, *Maverix (2021).*

Recommendation

The recommendation is to adopt DevSecOps methodologies to not only enhance the current development pipeline but to address the identified problems of strained relationships, siloed processes and slow development cycles.

**Action Plan:**

1. **Implement DevSecOps Culture**

* Begin a range of DevSecOps training sessions covering the core principles, methodologies and processes.
* Cross-function training programs to ensure a greater understanding of the goals and practices of different teams.
* Leadership training for senior team members to encourage a culture of open communication.

1. **Cross-Functional Team**

* Form a cross-functional team that enables collaborative decision making to ensure security related decisions are discussed and implemented as part of the design process.

1. **Automation of Security Processes**

* Evaluate and implement automated security testing tools that fit with the current development pipeline.
* Include continuous monitoring systems and automated feedback systems to provide timely information on security vulnerabilities and issues.

**KPIs**

In order to measure how successful these changes are, key performance metrics must be monitored.

1. **Decrease in Development Cycle Time**

* Track development and release cycles and compare to cycles before implementing the DevSecOps solution.

1. **Reduction in Security Vulnerabilities**

* Measure for reduction in identified security vulnerabilities compared to historical records.
* Monitor detection rate, response and fix times in comparison to previous data.

1. **Team Collaboration Statistics**

* Create a collaboration metric based on meeting attendance, collaborative effectiveness and communication (for example issue opened, response and resolution times).
* Regularly check in with senior team members for updates on progress in addressing interpersonal issues and effectiveness of team communication.

**Implementation Timeline**

The implementation is planned over a 12 month period, providing a phased approach and reducing disruption to ongoing development and support.

**Months 1-2: Preparation and Kickoff**

* Weeks 1-2: Kickoff
  + Arrange a kickoff meeting to introduce the DevSecOps initiative.
  + Communicate the vision, goals and benefits of DevSecOps to the company.
* Weeks 3-8: Assessment and Awareness
  + Identify key stakeholders and develop a plan to form the new cross-functional team.
  + Hold awareness sessions for all teams, highlighting the need for change.
  + Perform a comprehensive assessment of current development and security processes and requirements.

**Months 3-6: Tool Evaluation and Culture Shift**

* Weeks 1-4: Begin Embedding the Culture Shift
  + Regularly update the company on the progress of the project.
  + Establish lines of communication for feedback and updates from any interested parties.
  + Begin cross-functional training sessions to promote DevSecOps understanding.
* Weeks 5-12: Tool Evaluation and Selection
  + Shortlist, test and evaluate tooling for the new workflow, this may involve also changing existing tooling to meet the needs of DevSecOps.
  + Agree on and begin planning implementing the chosen toolset.
  + Arrange training to cover all aspects of the new pipeline.
  + Begin developing procedures and guidelines for security checks within the new automated process.

**Months 8-9: Implementation of New Tooling and Development Process**

* Weeks 1-4: Implementation
  + Integrate new tooling into existing pipeline.
  + Deploy monitoring systems for tracking security or development issues.

**Months 9-10: Formalise Cross-Functional Team**

* Weeks 1-4: Formally Create the Cross-Functional Team
  + Schedule regular meetings with the new team to discuss requirements, challenges and solutions.
  + Encourage small self-organising groups of both developers and security engineers to take ownership of development processes, issues and solutions.

**Months 10-12: Continuous Improvement**

* Ongoing Tasks
  + Conduct regular evaluations of the effectiveness of the implemented changes via monitoring metrics.
  + Ensure KPIs are in place and track closely.
  + Collect regular feedback from the teams and implement improvements when identified.
  + Refine processes and procedures to meet changing needs.
  + Implement on a continuous learning program where team members are regularly trained on latest tooling, security techniques and best practices.
  + Employ a mentorship program where senior members guide and transfer knowledge to junior members.
  + Continue to evaluate the effectiveness of implementing DevSecOps principles, both on a technical level and cultural level.

References

Açıkgöz, A., Günsel, A. and Kuzey, C., 2015. Climate and product quality in software development teams: assessing the mediating role of problem solving and learning. *Yönetim Bilimleri Dergisi*.

Carter, K. (2017), ‘Francois raynaud on devsecops’, IEEE Software 34(5), 93–96.

Lee, G., Espinosa, J.A. and DeLone, W.H., 2013. Task environment complexity, global team dispersion, process capabilities, and coordination in software development. *IEEE Transactions on Software Engineering*, *39*(12), pp.1753-1771.

Maverix. (2021). *How Maverix helps to easily transform DevOps in to DevSecOps in Financial organizations*. [online] Available at: https://maverix.ai/how-maverix-helps-to-easily-transform-devops-in-to-devsecops-in-financial-organizations/ [Accessed 1 Mar. 2024].

Morales, J., Turner, R., Miller, S., Capell, P., Place, P. & Shepard, D. (2020), Guide to implementing devsecops for a system of systems in highly regulated environments, Technical Report CMU/SEI-2020-TR-002.

Purna Sudhakar, G., Farooq, A. and Patnaik, S., 2011. Soft factors affecting the performance of software development teams. *Team Performance Management: An International Journal*, *17*(3/4), pp.187-205.