ISSN: 2583-9888
DOI: doi.org/10.51219/JAIMLD/sri-kanth-mandru/194

URF PUBLISHERS
connect with research world

Journal of Artificial Intelligence, Machine Learning and Data Science

https://urfpublishers.com/journal/artificial-intelligence

Vol: 1 & Iss: 1

Research Article

PAM (Privileged Access Management) and DevOps: Secure Management of Privileged Accounts: Integrating PAM with DevOps Practices To Ensure Secure Development Processes

Sri Kanth Mandru*

Citation: Mandru SK. PAM (Privileged Access Management) and DevOps: Secure Management of Privileged Accounts: Integrating PAM with DevOps Practices To Ensure Secure Development Processes. *J Artif Intell Mach Learn & Data Sci 2022*, 1(1), 783-787. DOI: doi.org/10.51219/JAIMLD/sri-kanth-mandru/194

Received: 03 May, 2022; **Accepted:** 28 May, 2022; **Published:** 30 May, 2022

***Corresponding author:** Sri Kanth Mandru, USA, E-mail: mandrusrikanth9@gmail.com

ABSTRACT

Conventional organizations –before the widespread adoption of DevOps –were organized in functional silos. Teams operated independently and reported to their respective line managers. During deployment, the code transitioned to the operations teams to launch the new product feature. As the teams worked in isolation, communication became a challenge. Much time was wasted leading to slower development and deployment. The siloed-away environment made it difficult for the development team or the operations team to collaborate as each team was unaware of the challenges and goals of the other team. As a result, conflict always ensued between teams' goals leading to inefficiencies between them. Today advances in innovation have led to the designing of system architectures that evolve with new technologies. This evolution has yielded numerous advancements not only in applications but also in IT infrastructure. Organizations have leveraged DevOps teams to work in unison to ensure the applications developed are per the various elements of the architecture taking into account the diverse constraints of the application. Additionally, in this collaboration, security teams must provide specifications for implementation by the development team and software architects. The DevOps process uses privileged accounts containing critical information, a leak of any privileged account could lead to paralysis of an entire project or business if they fail in the hands of malicious users. In this paper, we shall explore how integrating Privileged Access Management (PAM) with DevOps can secure the development process.

Keywords: Privileged access management, DevOps, Continuous integration, Continuous deployment, Malicious user, DevSecOps

1. Introduction

Today's ever-evolving technologies and increasing competition have brought organizations a real challenge in designing and delivering products faster, while at the same time ensuring user satisfaction. Among the key solutions to addressing this challenge is introducing a custom of cooperation[1] between disparate departments such as development, operations, testing, and security teams. DevOps has created this collaboration culture where teams can adopt certain practices to minimize the time to market[2]. The adoption of DevOps has reduced application deployment timelines and brings real value to an enterprise's products and software applications[3].

DevOps entails a collection of principles and methodologies that foster the active involvement of both the development and operations teams throughout the entirety of the application development lifespan, software maintenance, and operations[4]. Mikael Krief[3] describes DevOps as a culture that necessitates a shift in perception, processes, and tools. Organizations manage various processes and tools that support the automation of the software delivery process enhancing speed and agility[5]. The integration of these processes minimizes code release cycle times through the implementation of continuous integration and delivery practices and effective monitoring of the applications running in production[6]. DevOps' approach encompasses creating better software quickly[4].

DevOps, a concept that combines Development (Dev) and Operations (Ops) has brought a dawn to a revolution that promotes collaboration between the development and operations teams[3]. The collaboration fosters economic benefits to users more swiftly boosting market competitiveness. DevOps reduces the barriers between developers whose goals are faster innovation and delivery time, and operations teams which focus on guaranteeing the solidity of production processes and the quality of the system modifications they create[6]. The DevOps culture extends from agile processes which allows reduced delivery times and entails development and business teams[2]. Integrating development and operations streamlines production deployment, leading to cost savings[3].

By fostering the cooperation between development and operations teams, the DevOps custom needs to employ usable and exploitable tools across the teams [7]. Developers should integrate with the monitoring and security apparatus utilized by operations teams to discover performance challenges and protect access to resources [3].

2. Problem Statement

In the past, before the adoption of DevOps, organizations had functional teams that reported to line managers and were isolated from the broader business and often from each other[6]. During deployment, the code was handed over to the operations teams for application deployment. However, these teams worked independently and in isolation, leading to time-consuming and repetitive activities. The results were always undesirable. This kind of setup had numerous challenges which led to inefficiencies. For instance, as a result of the siloed-away environment, the development team or the operations team were unaware of the difficulties and goals of the other team[8]. This often resulted in conflict with the other team's goals and led to inefficiencies between them. While the development team's main focus is to release new features to the product more quickly, the operations team is dedicated to ensuring the application's availability and high performance[6].

Additionally, as innovation advances, designing system architectures that evolve with new technologies has been a concern among IT personnel in organizations. The evolution has unquestionably led to significant improvements in both applications and their infrastructure[9]. This requires DevOps teams to collaborate effectively to ensure that the applications they develop align with the various components of the architecture, taking into account the different constraints of the applications. Furthermore, security teams need to provide specific requirements that developers and software architects will implement[3].

In many organizations, development teams often lack awareness of security rules, resulting in security measures being implemented too late in DevOps processes. To address this, developers need to be educated about application code security and the protection of CI/CD pipeline configuration to integrate security into their processes. Breaking down the barrier between DevOps and security can be achieved by involving security teams in meetups that bring together development and operational teams, promoting better collaboration and consistency[3].

The DevOps process involves the use of privileged accounts, which are sometimes hard-coded during development and may not be removed before being delivered to customers. This poses a significant risk of leakage and leads to a larger attack surface[10]. Since privileged accounts used across DevOps projects contain critical information, the leakage in any privileged account could lead to the paralysis of an entire project or business if it falls into the hands of malicious users[11].

3. Solutions

DevOps addressed these challenges by creating cross-functional teams, promoting collaboration, and fostering communication about each other's work and end outcomes[3,6]. This strategy improves the quality of feedback and addresses issues with existing automation. Most DevOps processes are continuous. Utilizing feedback cycles to enhance existing processes allows an organization to continually mature and evolve, learning from previous situations and becoming a more mature team[6]. Privileged access management (PAM) is a strategy through which an organization's critical data resource or infrastructure is protected against infringement through an array of practices[11]. The approach involves authorizing and monitoring all privileged users for all resources. The greater the user access to an organization's resources, the higher the risk the organization faces if the information is leaked to malicious users[12].

DevOps combines application development and operations through automated mechanisms that shorten release cycles throughout the software implementation lifecycle[2,5]. DevOps automation apparatuses utilize privileged credentials, such as application-to-application solutions, necessitating security integration into DevOps practices[11]. Extreme privileges or distributed secrets granted and accessible to DevOps team members can be leveraged by malicious insiders hence compromising code. Also, container misconfigurations and vulnerabilities can pave the way for security compromises. During continuous integration and deployment, the leveraged tools may contain scripts or vulnerabilities that may deploy malware or sabotage code[11]. In addition, hard-coded passwords, insecure code, and other privileges may expose the organization to external attacks. Automation for moving and testing code needs credentials to traverse network precincts. Compromising this process can enable threat actors to move laterally[11,13].

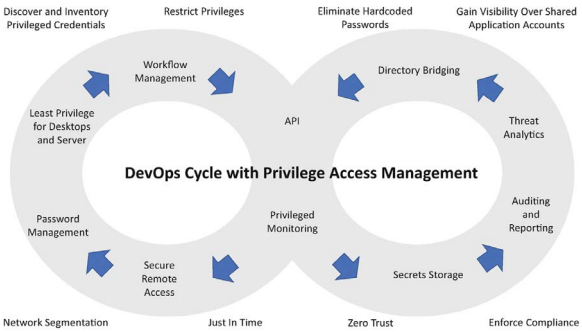

Figure 1: Typical SecDevOps (DevOps) life cycle with privileged access management[11].

Technologies that streamline application development and deployment through integration and automation do not require identities as they are automated. These credentials are sometimes stored in configuration files and shared as scripts[11]. Storing and intermittently updating credentials to automate DevOps procedures can render them vulnerable to misuse and hacking, particularly if they are not encrypted and are in plain text. To minimize these risks, companies should enhance their privileged

access management methods and incorporate a design phase that eliminates hard-coded credentials in codebases[11]. To promote secure development, it is essential to put in place a remote access solution with session monitoring to effectively regulate developers' access to production servers[11]. Haber[11] notes that this approach allows developers to safely and easily carry out critical tasks without having direct access to the system. This upholds the concept of least privilege across the environment, which means developers and development tools should not be given administrative or root access to minimize the risk of privilege exploitation and abuse. Haber further notes that introducing application governance into the DevOps procedures can help prevent the execution of malware by denying access to programs with questionable reputations[11].

Moreover, errors in the code base can lead to software vulnerabilities, which malicious actors can exploit to access an organization's critical data. This challenge can be tackled by implementing DevOps practices that accentuate continuous integration and continuous delivery (CI/CD). Continuous integration is a software development practice in which teams frequently integrate their work and verify each integration with an automated test to detect errors quickly[3]. In continuous integration, the team can test the code's integrity whenever a team member makes a change. It's important to perform this verification quickly. The continuous integration as envisioned in DevOps advocates a spirit of collaboration and communication[1]. This collaboration ensures teams are at par with one another, and that any feedback on code integration is acted upon quickly thus allowing delivery of new features more quickly to users[3]. The automated code review and modifications to code ensure that no bug, vulnerability, or misbehaving of an application's code can be introduced into an organization's IT system. This helps in reducing challenges that are associated with privileged accounts which are among the weak links in the fight against cybersecurity protection as they can introduce malicious code into an organization's system.

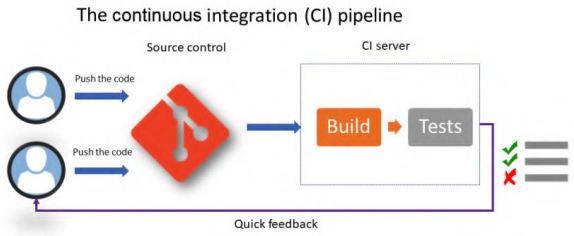

Figure 2: The continuous integration workflow[3].

The continuous integration process must be optimized within the least time possible to ensure fast execution and that developers can receive prompt feedback on their code integration. Archived codebases that do not compile may hamper test execution hence impacting and blocking the entire team. With an automated continuous integration process, the development team can quickly address issues, enhance their code, or discuss it internally before committing their code for a new integration[3].

The next phase after continuous integration is continuous delivery where the application is deployed in a non-production environment. During this stage, functional and acceptance tests are conducted to evaluate the entire application, including all of its dependencies[3]. This staging environment makes it possible for developers to test and validate the entire application before deploying it into a production environment. All functionalities are tested, and in case of any bug fixes or code modifications following the verification process, the code will have to go through a continuous integration process again. If the software complies with all security practices, the validation personnel will ensure proper functionality before triggering the deployment of the software or new features into the production environment[3].

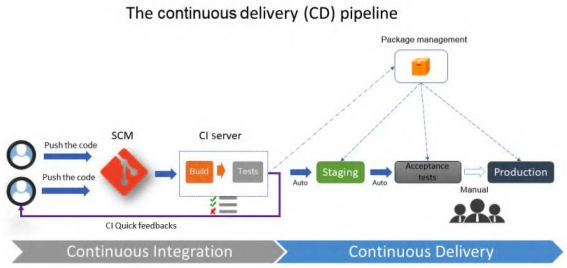

Figure 3: The continuous delivery workflow[3].

Continuous deployment, an extension of continuous delivery, is crucial for automating the entire CI/CD pipeline from code commit to production deployment and verification[7]. The continuous deployment process involves all the necessary steps to restore the application in case of a production issue[3]. Continuous deployment involves packaging the application's functionalities into features and activating them on demand directly in production, without needing to redeploy the application's code[3].

4. Uses

DevOps involves integrating people, processes, and products to enable continuous value delivery to end users. Organizations can reap numerous benefits from its implementation. For example, the DevOps culture fosters improved collaboration and communication within teams, thereby impacting the organization socially and operationally. DevOps also leads to shorter production times, improving performance, and increasing end-user satisfaction[3]. It reduces infrastructure costs with Infrastructure-as-Code (IaC)[7] and saves considerable time with repetitive cycles, minimizing software errors. Automation tools minimize manual tasks, allowing teams to concentrate on creating new features that add business value[3,4].

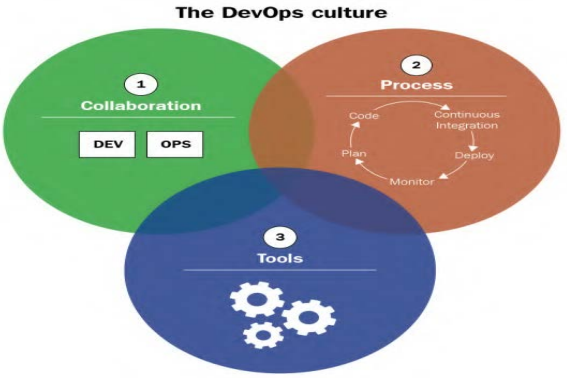

Figure 4: The DevOps culture union[3].

The IT security team is dedicated to providing authorization, digital identity management, and authentication services to enhance cybersecurity through Identity and Access Management and Privileged Access Management[14]. The business drivers for the IT security team include security risk management, user experience, and operational excellence, with security risk management being the primary business driver. The IT security

team enhances an organization's security risk management by overseeing users' access and activity, conducting access reviews, removing outdated or archived accounts, managing accesses and rights, and ensuring segregation of duty[15]. The security team aims to achieve operational excellence by streamlining and automating processes, reducing service desk calls, and consolidating IAM and PAM infrastructure and software[14]. These efforts allow the business to deliver a smoother and faster user experience.

5. Impact

The fast-paced and complex business world, coupled with the rapid evolution of digitalization, creates a dynamic landscape that calls for organizations to rethink their strategy to adapt to survive or risk becoming obsolete. The hard reality is that change is inevitable and enterprises should harness the power of DevOps if they wish to cultivate continued business growth and expand their market opportunities[9,16]. Therefore, whether companies are ready or not prepared to transform with the market trend, they will soon have to adapt to the new normal in the applications economy[9]. DevOps encompasses all aspects of software and management, representing the subsequent business environment.

DevOps practices have been adopted in various industries such as the banking, retail, and automobile sectors[1] to provide cutting-edge competitive advantage thus promoting user experience, consumer satisfaction, and overall organization performance[9,16]. Traditional approaches to software and operations do not adequately address the changing customer needs. However, with the adoption of DevOps, the development and operations teams, as well as the IT security teams collaborate to automate the testing and execution of codebases and new product functionalities. DevOps culture of continuous delivery has been leveraged in online and mobile banking applications to suit emerging customer preferences. New challenges within the banking sector have been swiftly identified and addressed, resulting in faster code delivery to end users and improved quality[17]. DevOps plays a crucial role in bridging the gap between development and operations by introducing higher levels of automation, thus accelerating the pace of application releases[9].

6. Scope

An important aspect that has been omitted by many organizations in the DevOps culture is security. It makes sense to incorporate security into DevOps as early as possible, given the rapid development and deployment involved in the practice[3,18]. The DevSecOps security approach involves automating conformity and security validation procedures in CI/CD practices to ensure continuous security without decelerating application deployment[3,6]. Today, integrating security teams into the DevOps culture not only improves application quality but also enhances security. According to Gaurav Agarwal, the implementation of modern DevOps requires the adoption of a DevSecOps approach[4]. Agarwal notes that though organizations have security policies, most of them only adhere to such policies just to comply with rules and regulations. Instead, there should be association and effective communication between development, operations, and security teams, and employees should upskill and cross-skill themselves to leverage a DevSecOps strategy and embed security early during development processes[4,15]. In this paper, we realize that embedding security within an organization's culture is vital. We understand that the security aspect should be integrated with DevOps and PAM concept alone will not address the far-reaching risks in the cybersecurity landscape. This paper is limited only to PAM and DevOps, and not to DevSecOps, through being a critical integration in the development process.

7. Conclusion

The technologies that automate application development and deployment require credentials, which are often stored in configuration files and shared as scripts. DevOps automation tools use privileged credentials, highlighting the need for security integration in DevOps practices. The DevOps process often involves the use of privileged accounts. Sometimes, these accounts are hardcoded during development and aren't removed before being sent to customers. This creates a higher risk of information leakage and increases the potential for security breaches. Because these privileged accounts contain important data, if they were to be leaked, it could severely disrupt an entire project or business, especially if they were to fall into the wrong hands. As a result, security should be maintained to protect the organization's application and infrastructure from being infected with malware.

Traditional software development processes took more time to deploy due to the siloed nature of organizations. In modern times, DevOps seamlessly integrates software development and operations through automated practices, significantly reducing release cycles throughout the software development lifecycle. By focusing on the synergy of people, processes, and products to ensure a continuous delivery of value to consumers, organizations stand to gain tremendous advantages from its adoption. For example, fostering a DevOps culture encourages enhanced collaboration and communication among teams, thereby fostering positive human and social dynamics within the organization. Additionally, DevOps facilitates quicker production times, leading to improved performance and heightened end-user satisfaction.

8. References

1. Gonzalez D. Implementing Modern DevOps: Enabling IT organizations to deliver faster and smarter, Packt Publishing 2017.
2. Sandu AK. DevSecOps: Integrating Security into the DevOps Lifecycle for Enhanced Resilience. Technology & Management Review 2021;6: 1-19.
3. Krief M. Learning DevOps: A comprehensive guide to accelerating DevOps culture adoption with Terraform, Azure DevOps, Kubernetes, and Jenkins. Packt 2022.
4. Agarwal G, Modern DevOps Practices: Implement and secure DevOps in the public cloud with cutting-edge tools, tips, tricks, and techniques. Packt 2021.
5. Yasar H, Kiriakos K. Where to Integrate Security Practices on DevOps Platform. Int J Sec Soft Eng 2016;7: 39-50.
6. Coupland M. DevOps Adoption Strategies: Principles, processes, tools, and trends: Embracing DevOps through effective culture, people, and processes. Packt Publishing 2021.
7. Kantsev V. Implementing DevOps on AWS Bring the best out of DevOps and build, deploy, and maintain applications on AWS. Packt 2017.

8. Glazemakers K. Opening the network to DevOps without letting threats inside. Network Security 2021;2021: 7-9.

9. Ravichandran A, Taylor K, Waterhouse P. DevOps for Digital Leaders: Reignite business with a modern devops-enabled software factory. Apress 2016.

10. Pompon R. It security risk control management : An audit preparation plan. Apress 2016.

11. Haber MJ, Privileged Attack Vectors: Building effective cyber-defense strategies to protect organizations, Second edition. Apress 2020.

12. Anton P, Soetomo M, Assessing Privileged Access Management (PAM) using ISO 27001: 2013 Control. ACMIT Proceedings 2018;5.

13. Haber MJ, Rolls D. Identity attack vectors: Implementing an effective identity and access management solution. Apress 2019.

14. Blomberg V. Adopting DevOps Principles, Practices and Tools. Case: Identity & access management. in practice 2019;29: 1-14.

15. Hsu T. Hands-On Security in DevOps: Ensure continuous security, deployment, and delivery with DevSecOps. Packt 2018.

16. Sharma S. The DevOps Adoption Playbook: A Guide to Adopting DevOps in a Multi-Speed IT Enterprise. John Wiley & Sons, Inc 2017.

17. Walls M. Building a DevOps Culture: DevOps is a much about culture as it is about tools. O'Reilly Media 2012.

18. Battina DS. Best practices for ensuring security in Devops: A case study approach. Int J Innovations Engineering Research and Technology 2017;4: 38-45.