NOVATEUR PUBLICATIONS
INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING RESEARCH AND TECHNOLOGY [IJIERT]
ISSN: 2394-3696
VOLUME 4, ISSUE 11, Nov.-2017

BEST PRACTICES FOR ENSURING SECURITY IN DEVOPS: A CASE STUDY APPROACH

Dhaya Sindhu Battina
Sr. DevOps Engineer & Department of Information Technology
USA

ABSTRACT
The main purpose of this paper is to explore the best practices that can be adopted to ensure security in DevOps. Among the paper's key topics are the definitions of both DevOps and DevSecOps. The day when developers created code and operations built and managed it is long gone. To put it another way, the DevOps approach has altered how firms build software. Companies benefit from speedier releases and staying on top of the ever-changing market thanks to the integration of both teams. Security teams, on the other hand, often fell behind, unable to keep up with the rapid development cycles [1]. To identify and address security issues early in the development cycle, DevSecOps incorporates security policies throughout the DevOps process.

Keywords: Artificial intelligence, AI-augmented automation, DevOps, cyber-physical systems, AIOps.

I. INTRODUCTION

DevOps security is the process of integrating information security policies and technology across the DevOps lifecycle and value stream. It becomes much more vital to have strong security with DevOps since it involves every level of the SDLC. Information security isn't anything new for the majority of companies. Security is a major issue for IT professionals when dealing with information technology. DevOps infrastructure, on the other hand, represents a considerable break from conventional IT paradigms. Software updates may be delivered up to 500 times per day in companies that employ DevOps procedures. Rapidly deployed software modifications are more likely to have security risks if the security team is not adequately involved [1]. Using experiences from implementing security principles in a DevOps setting, this article aims to help software practitioners better integrate cybersecurity with DevOps. As part of our research, we examined a sample of Internet artifacts and spoke with representatives from nine DevOps-using firms about their experiences with security procedures. In our research, we found that the majority of software developers and operations professionals believe that standard DevOps tasks like automated monitoring have the potential to increase system security [2]. The use of security settings and security needs analysis is common among firms that combine DevOps with their overall security strategy. The security team, development team, and operations team have all developed cooperation [2]. Most of the time, security modules and DevOps don't come together in a way that makes sense. Security is an important aspect of any company, but it has proven tough to integrate into the DevOps process at every stage [2,3]. Due to a widespread lack of security knowledge, security implementations are often imbalanced, slowing down the environment's speed and agility. The answer is in forming a partnership with the appropriate group to properly build up security measures [4].

II. PROBLEM STATEMENT

The main problem that this paper will address is to explore which best practices can help address DevOps security challenges. Moving to DevOps architecture or the cloud from legacy systems and rebuilding apps is perhaps the largest issue that keeps most companies from embracing DevOps [4,5]. In addition, the development team must be rebuilt, and internal procedures must be modified to meet the new model. Changes in team roles, new team members, and the adoption of new tools are all examples of this [6]. Incorporating

DevOps also presents a significant security concern since the organization is exposed to potential security threats. Mainly because the security team is not linked with DevOps and does not know what tools and procedures are utilized to improve the development processes. To expedite the process, DevOps often prioritize speed above security, resulting in missed scans and code inspections. This creates vulnerabilities and makes it difficult for the security team to keep track of all the security flaws that have been found.

III. LITERATURE REVIEW

A. DevOps

DevOps is a series of processes aimed at shortening the time between making a change to a system and putting that change into regular production while still maintaining high quality. Term A software engineering culture and practice are known as DevOps (short for "development" and "operations") try to bring together software development (Dev) and software operations (Ops) (Ops) [6]. Throughout the software development process, it aggressively encourages automation and monitoring. This covers everything from development and testing through integration, release, and deployment. They aim to create shorter development cycles, more frequent releases, and more trustworthy releases that are closely aligned with corporate goals [7,8]. For a long time, development and operations were done in separate silos. However, under a DevOps approach, both teams are combined with engineers working throughout the full development lifecycle. As part of the development process, quality assurance teams are also included.

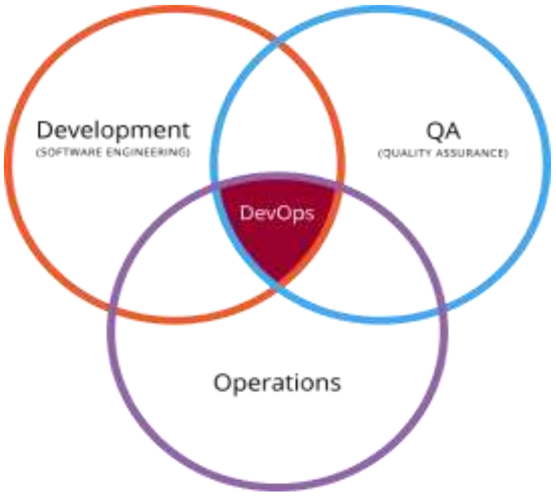

Fig i: DevOps practices

B. Security challenges in DevOps

DevOps' contemporary, agile emphasis on IT and software development has created new security concerns that need not just a change in security technology but also a change in culture, people, and procedures. DevOps and security have stayed distinct, despite the popular perception that the two are intertwined [9].

1. Rapid Pace of Change

One of the most noticeable implications of DevOps on security is its lightning-fast iteration cycle. Legacy environments often provided new infrastructure as bare-metal hardware, with a protracted cycle from a provisioning request to functional availability. Major releases of software are often spaced out over months or quarters, according to the waterfall model of software development [9,10]. Modern agile setups, on the other hand, may see many production deployments in a single day. Additional capacity may be added in minutes rather than hours or days by employing cloud-based infrastructure. As a result of all this, the pace of change in a particular environment has increased significantly [11]. Legacy systems and procedures were not designed to cope with such wide fluctuations in the environment.

2. Cloud Security

With cloud-first architecture becoming so prevalent, another difficulty has arisen. The cloud has a substantially larger attack surface than a typical on-premises deployment because of the fuzzier network boundaries. With only a few clicks or lines of code, almost any supplied resource may be made available to all types of internet traffic. A key assumption in traditional network safety was that networks would be tightly defined, with a small number of well-established entry and exit points [11].

3. Containerization of workloads

In addition, additional security variables are introduced as a result of this. However, the additional complexity of the underlying engine, orchestration, and networking means that there are more attack routes to be monitored and guarded.

4. Collaboration

These new use cases simply weren't taken into consideration when the original security solutions and processes were being developed. A DevOps-first culture will not scale with siloed security and engineering teams. By working in separate silos, security and engineering are duplicating operational effort and information flow that may simply be pooled. For teams to be in sync and get the same information from the same source, it's critical to have a unified pipeline [11]. It's not uncommon for organizations to have security and application teams each running their own Splunk agents on separate boxes, or for the fraud department to receive feeds from the security and infrastructure teams, each of which is running a completely separate and parallel event pipeline [11,12].

C. DevOps security

Application delivery teams using DevOps approaches face several security risks as a result of moving at a rapid pace. Failure to concentrate on security from the start is a major barrier to security in the DevOps context. Safety in the application's conception and development. Security is seen as a reactive activity, apart from the main development processes. It is a clean-up operation. As a consequence, apps are either delivered more slowly or deployed before they've had a chance to be completely vetted. Because traditional approaches to application and data security do not readily fit into DevOps procedures, they might slow down operations [10]. This is a significant challenge to overcome [12]. It is assumed by both the development and security teams that these two groups work separately and often without cooperating. Development and operations teams supply created applications while security teams look for potential security issues. If required, development teams may return developed applications to the development stage. When it comes to development and operations, they often work in unison [12,13]. However, only the development process may seem to be at odds with the steps necessary to keep an application secure. For some DevOps teams, the worry of slowing down application delivery creates cultural opposition to security. There is a lot of pressure on DevOps teams to provide new versions of the application often while also ensuring that operations are not slowed down. Traditionally, security concerns have been seen as a hindrance to progress because of the time and resources required to handle them.

Additionally, some companies do not assign specific duties for certain aspects of security. Especially when no security crew is in place. Furthermore, since certain DevOps members may lack adequate security knowledge, the likelihood of complex threats emerging is increased [14]. New and dangerous application technology may be used by a development team that does not fully address future risk issues. Unchecked container deployment or exposed secure API are examples of recently published open-source libraries. In addition, human mistakes or bad software development processes may introduce risks to the code. When there is no regular security review, these dangers may go unreported for a long time [14].

D. DevSecOps

Fast-feedback software delivery, company culture, and information security procedures are all included within DevSecOps. The DevSecOps approach, as opposed to DevOps security, focuses on integrating security and engineering goals with a "shift-left" perspective before implementing them as part of the DevOps cycle rather than after the fact [14,15].

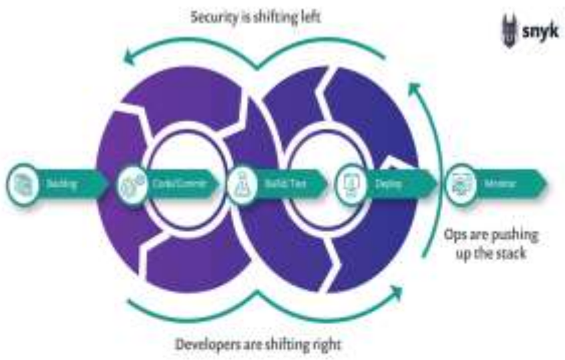
Security is shifting left
snyk
Building
Code/Commit
Build/Test
Deploy
Monitor
Ops are pushing up the stack
Developers are shifting right

Figure 2: Security objectives are being shifted left

For the most part, DevOps security begins with a broad attitude or posture. Tools are an important part of DevOps security, but they aren't the only consideration. The goal of implementing DevSecOps techniques is not to cross items off a to-do list. It will be very difficult to establish a safe platform if the team does not have a security attitude. Using tools and methods given by DevSecOps, security goals are applied to various phases of the life cycle [15]. Although there may be a more collaborative atmosphere, security teams may still be functioning in separate silos from engineering and operations teams. DevOps' shared responsibility model has been a basic principle from the beginning, but security generally required engagement with a separate security team or organization outside of the company [15,16]. Secure design and development are at the core of DevSecOps, which strives to incorporate security goals across the whole DevOps life cycle, especially during the design and development phases.

E. What Is DevSecOps?

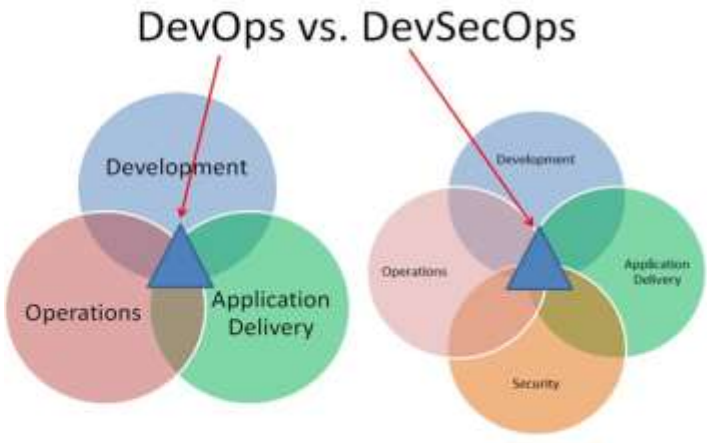

Fig ii: DevOps vs.DevSecOps

Since this, many businesses have been unable to keep up with the rapid adoption of the DevOps paradigm because their security models haven't kept pace. The DevSecOps concept integrates security into the DevOps process, and it's here to stay. Because security and development teams often fight, this may be more difficult to do than it first seems to be. In the end, it seems a little paradoxical that the code must be safe yet the process must be accelerated. The synergy between release engineers and security teams may be achieved by creating a work culture that stresses "security as a code."

F. Best Security Practices for DevOps

Secure practices should be included in every stage of the DevOps lifecycle, from conception to maintenance, to produce a successful and secure DevOps process. Security practices and culture are integrated throughout the whole development and operations process using a method dubbed Devsec, which is short for "DevSecOps," which stands for "DevOps Security and Cooperation" [16]. There are frequently difficulties with implementing DevOps security, such as a lack of manpower or a dispute in cooperation. These methods support DevOps's shorter development cycles while yet maintaining high levels of security. To solve these issues, let's look at the most effective strategies.

i. Adopt a DevSecOps model

Effective DevSecOps integrates governance and cybersecurity activities like identity and access management (IAM), code review, configuration, and vulnerability management across the DevOps process. In this way, security may be integrated into the DevOps process, enabling the delivery of safe goods and reducing the need for post-release repairs.

ii. Update the governance policies

Governance rules and IT procedures should be updated as the organization grows. As a result, the norms of behavior will be adhered to, preventing data leaks. Employees may feel confident sharing concerns when there are signals of suspected internal threat behavior when there is an open governance framework in place.

iii. Implement vulnerability management

Performing regular scans and monitoring on systems may help maintain security adherence throughout the development and integration processes. Penetration testing, for example, might detect possible flaws or failures so that the team has time to fix them before going live.

iv. Implement Automation

When security operations like privilege management and vulnerability testing are done manually, the likelihood of human mistakes increases dramatically. In the latter phases of development, automating security processes may save time and free up resources for security testing. It may also minimize expenses by lowering labor hours and vulnerabilities [16].

v. Remember to test the hardware

Hardware is often ignored, and as a result, its efficiency and security are not checked. This may result in inaccurate test findings, or even worse, security vulnerabilities. Constant validation should be performed following security regulations to guarantee that your top-tier software is operating correctly.

vi. Use network segmentation to your advantage

To make an attacker's job more difficult, use the divide and conquer method. Limiting the number of people who may use your application resource server can help you avoid network congestion. Securing the network by segmenting it helps to maintain the environment error-free while also preventing unauthorized access to all network data.

vii. Reduce the number of people with administrative rights

Keep in mind that the greater the number of individuals who have access to data, the greater the possibility of an inside attack or an error being apparent. Reduce the danger of security concerns by not granting administrative powers to everyone. You may restrict access to private and sensitive data by storing it on a few designated local workstations [16].

IV. FUTURE IN THE U.S

DevOps in the US has a bright future because more firms are adopting rules and practices that merge software development with IT operations. SaaS platform Azure DevOps is offered by Microsoft and is also known as Microsoft DevOps. A complete DevOps toolchain is available for software development and deployment. Customization and connection with industry tools are included. DevOps has moved beyond product delivery as it enters its second decade. With an emphasis on providing not just new features and products, but value as well, it's no longer simply about development and operations [17, 18]. The future of DevOps seems promising. The number of DevOps-related practical applications is increasing regularly. Knowing what the future holds for DevOps developers will help you improve the effectiveness and quality of your mobile app development efforts. In the following years, expect to see some significant shifts. To increase the effectiveness of your internal operations, you must keep one step ahead of your rivals. Several benefits have accrued to several firms that have used DevOps in their business processes. Because of the prevalence of bespoke app development, data science teams may embrace DevOps development approaches in the future [17]. As long as data scientists make use of DevOps, they may reuse production models they've previously generated during testing. A growing number of data scientists and bespoke app development teams are collaborating to maintain track of several applications.

V. ECONOMIC BENEFITS IN THE UNITED STATES

Many American companies are realizing that implementing DevOps best practices is a necessity. In the end, DevSecOps is all about improving and standardizing security concerns in software development. Compliance is a key factor to keep in mind while working in this field. This helps preserve customer data while also helping corporations avoid costly penalties and public criticism. Companies in the United States benefit from system security since it lowers the risk of data loss [18,19]. AWS's continuous integration and deployment strategy, for example, assists enterprises hosting services in the Amazon Web Services (AWS) cloud by enhancing preventative and investigative security measures. To avoid expensive downtimes, enterprises increasingly rely on cloud apps to keep operations going. This necessitates security measures apart from those provided by AWS. DevSecOps automation helps companies meet security objectives while reducing the reliance on humans (that is to say, less manpower). Code may be released considerably quicker since security checks cause fewer delays [19]. As a result, businesses can keep up with their rivals while also meeting changing customer demands faster. Security operations are cheaper with DevSecOps, and the risk of financial fines is lower as well. Security's efficacy as a value producer is aided by its quickness.

VI. CONCLUSION

This paper explored how machine learning can be applied in clinical research. The findings from this research show that the integration of Security and DevOp drives a productive synergy. When security standards are integrated throughout the whole process, a secure result is ensured. By including DevOps security from the beginning of the lifecycle, security risks may be addressed throughout the whole development process of an application or system. The use of DevOps Security principles will lead to more rapid and secure deployment of a product. A productive intersystem is the result of combining Security and DevOps strategically. Error reduction principles include identifying problems and their extent, restricting network access, assuring limited access, and managing vulnerabilities. As opposed to mistake correction, the primary goal of DevOps should be error prevention. You'll be able to accomplish your goal thanks to the advice provided above. Even though many see DevOps and agile approaches as mutually exclusive and incompatible, the truth is that they may occasionally mitigate one another's flaws, resulting in a more harmonious working environment. The future of corporate DevOps may see it complementing agile rather than replacing it in many workplaces.

REFERENCES

1) R. Branson and D. Armstrong, "General practitioners' perceptions of sharing workload in group practices: qualitative study", BMJ, vol. 329, no. 7462, p. 381, 2004.
2) S. Conger, "Software Development Life Cycles and Methodologies", International Journal of Information Technologies and Systems Approach, vol. 4, no. 1, pp. 1-22, 2011.
3) J. Grünenfelder, "Adopting a 'bifocal approach' in Swiss academic mentoring programmes - challenges and opportunities", Zeitschrift für Hochschulentwicklung, vol. 9, no. 1, 2014.
4) Hajjdiab and Al Shaima Taleb, "Adopting Agile Software Development: Issues and Challenges", International Journal of Managing Value and Supply Chains, vol. 2, no. 3, pp. 1-10, 2011.
5) W. Jiao and H. Mei, "Supporting high interoperability of components by adopting an agent-based approach", Software Quality Journal, vol. 15, no. 3, pp. 283-307, 2007.
6) J. Roche, "Adopting DevOps Practices in Quality Assurance", Queue, vol. 11, no. 9, pp. 20-27, 2013.
7) U. Rahman and L. Williams, "Software security in DevOps", Proceedings of the International Workshop on Continuous Software Evolution and Delivery, 2016.
8) M. Workman, "How perceptions of justice affect security attitudes: suggestions for practitioners and researchers", Information Management & Computer Security, vol. 17, no. 4, pp. 341-353, 2009.
9) Yasar and K. Kontostathis, "Where to Integrate Security Practices on DevOps Platform", International Journal of Secure Software Engineering, vol. 7, no. 4, pp. 39-50, 2016.
10) L. Zhu, L. Bass and G. Champlin-Scharff, "DevOps and Its Practices", IEEE Software, vol. 33, no. 3, pp. 32-34, 2016.
11) M. Airaj, "Enable cloud DevOps approach for industry and higher education", Concurrency and Computation: Practice and Experience, vol. 29, no. 5, p. e3937, 2016.
12) Ceccarelli, L. Montecchi, F. Brancati, P. Lollini, A. Marguglio and A. Bondavalli, "Continuous and Transparent User Identity Verification for Secure Internet Services", IEEE Transactions on Dependable and Secure Computing, vol. 12, no. 3, pp. 270-283, 2015.
13) S. Gutta, S. Prasad and J. Angara, "Devops product line engineering (DPLE): Where devops meets software product lines", PONTE International Scientific Researchs Journal, vol. 72, no. 11, 2016.
14) Hancock, "CIO's Get Serious About Best Security Practices", Computers & Security, vol. 19, no. 5, p. 388, 2000.
15) G. Linkevics, "Adopting to Agile Software Development", Applied Computer Systems, vol. 16, no. 1, pp. 64-70, 2014.
16) S. Mudrakartha, "Ensuring Water Security through Rainwater Harvesting: A Case Study of Sargasan, Gujarat", Water Nepal, vol. 11, no. 1, 2004.

17) C. Papagianni, A. Leivadeas and S. Papavassiliou, "A Cloud-Oriented Content Delivery Network Paradigm: Modeling and Assessment", IEEE Transactions on Dependable and Secure Computing, vol. 10, no. 5, pp. 287-300, 2013.
18) S. Turnbull, "Case Study on the Irrelevance of Best Practices in Corporate Governance", SSRN Electronic Journal, 2007.
19) Yasar and K. Kontostathis, "Where to Integrate Security Practices on DevOps Platform", International Journal of Secure Software Engineering, vol. 7, no. 4, pp. 39-50, 2016.