Michael Sinclair

Professor Davis

CSCI U375

26 *March 2023*

SecDevOps: A New Approach to DevOps

DevOps is a streamlined approach to software development that aims to be very quick and efficient while joining development and operations teams. Traditionally, developers and IT operators were completely separate with those who deploy the code having little to do with those who wrote it. DevOps allowed teams to center activity around a central objective by focusing on continuous integration/continuous delivery (CI/CD) that delivers software and applications in rapid succession to meet target business metrics (Aqua Security). However, the speedy nature of DevOps often causes security vulnerabilities and risks to go overlooked since there is little time to discover them. With the modern age having such complex software and technology alongside the meteoric rise in cybercrime, security has become more important than ever (Bhandari).

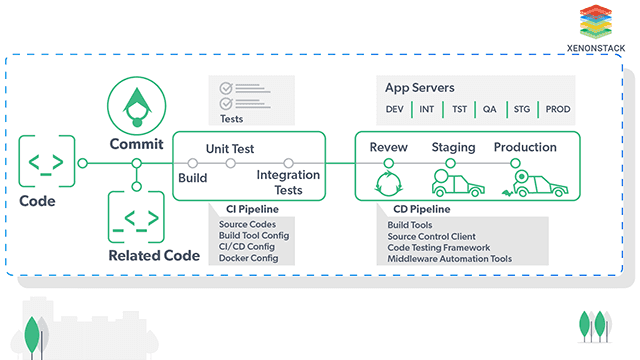


Fig. 1. Pictured above is the DevOps Assembly Line with the CI/CD outlined. This method focuses on connecting many of the different groups involved in the software development cycle and automating much of the process (Gill).

The initial movement towards integrating security practices more into the software development process was DevSecOps over DevOps. The traditional approach to DevOps had security remaining as an afterthought, with vulnerabilities, mistakes, and misconfigurations often being overlooked and passed along to meet rapid delivery deadlines. This practice may appear to be quicker on paper, however, it ends up being very expensive as soon as attacks occur and the organization has both financial losses and a loss of goodwill (Reynolds).

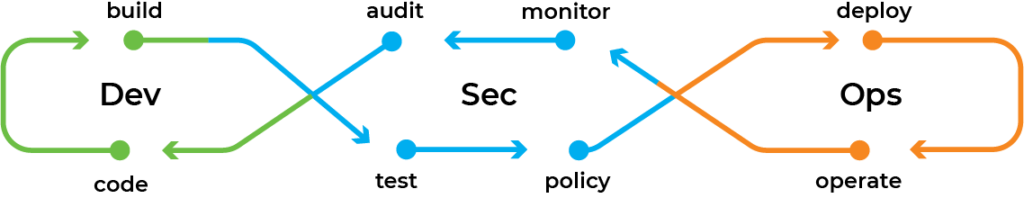


Fig. 2. Pictured is a typical DevSecOps model for the software development cycle, with security having been shifted from the end of the cycle to the center (Reynolds).

Within DevSecOps, security is intended to be “shifted left” in the software development cycle, though oftentimes the results are not much different than with DevOps. This shift left developers alone carrying the burden of security and running security tests themselves in addition to production, leading many developers to continue to disregard security (Reynolds). Additionally, many DevSecOps models are organized inefficiently such that they have separate security teams that are not developers or IT operators. New code is compiled and tested (not for security), then sent to a separate security team meaning that vulnerabilities and issues are found late in production. If a security issue is found developers must go back to address the issue, and in the worst-case scenario, the team may not even have time to fix it (Nidecki). Additionally, within DevSecOps the three separate teams (developers, security professionals, and IT operators) are instructed to collaborate on security but are still left separate, resulting in silos and bottlenecks (Aqua Security). While DevSecOps is a step in the right direction, SecDevOps has proven to be a much better approach.

Chart, sunburst chart

Description automatically generated

Fig. 3. Pictured is where security is integrated into the software development cycle between SecDevOps, DevSecOps, and traditional DevOps (DevOpsSec) (Greenfield).

SecDevOps shifts security even further left so that security is now the first step in the software development cycle. With security taking a focus very early in the cycle DevOps professionals are still able to keep up with quick releases. The objective is to create software that can be well-maintained, defensive, and upgradeable in the modern landscape. SecDevOps utilizes two main processes: security as code (SaC) and infrastructure as code (IaC). SaC applies security to the DevOps technology to automate manual tasks by using tools such as Dynamic Application Security Testing (DAST) and Static Application Security Testing (SAST) (Reynolds). IaC uses DevOps tools such as Ansible (an open-source configuration management tool (Gill)), Helm (a package manager for Kubernetes, which is a platform used to deploy and manage containerized applications (Singh)), and Puppet (an automated tool used for continuous configuration management (Singh)) to update, maintain, standardize, simplify, and configure infrastructure (Bhandari).

Through the use of SaC and IaC as well as their associated tools, teams can be proactive with security rather than react to threats and vulnerabilities as they appear. With security first and at every step, systems are ensured to be reliable and safe while carrying out their intended functions and delivering secure software (Bhandari). Other beneficial security practices found in SecDevOps models include red/black deployments (two identical production environments with only one online so that the offline environment may be used to test new code and changes without affecting the live environment) and version control which allows security issues and vulnerabilities to be audited and investigated efficiently (Aqua Security).

SecDevOps is not without its challenges and shortcomings, however, the largest of which is that there are simply not enough security professionals or team members to review code. Teams may also be resistant to the transition to SecDevOps since it requires the very quick release schedule of DevOps to slow down some to focus on security. Another challenge is that the implementation of security can look very diverse in different environments but the businesses will need to have secure access to it all the same (Bhandari).

Ways to overcome these challenges involve changing tools, processes, and culture. The use of automated tools can highlight security risks and identify them as early as possible while keeping infrastructure in mind. As for processes, security needs to be reviewed like any other code review with negative feedback being recognized and accepted. Documentation must also be excellent so that when issues arise they are more easily dealt with. A culture of accountability and shared responsibility must be created with a clear promotion of security (Bhandari). This shared responsibility may include actions such as quality assurance being taught how to automate and perform security tests and given the tools to do so, operation teams making security scans a regular occurrence, and members of every team receiving education to incorporate valuable security practices into their everyday work meaning that every individual does their best with security, not just a separate security team (Nidecki).

The education and training that developers and operators receive are vital to keeping both teams from overcompensating and focusing too heavily on less important issues. Given that not every vulnerability is equal, some require more attention than others, and both developers and operators must understand this distinction. Security in SecDevOps is not about eliminating any risks and vulnerabilities, but managing them such that deployment speed is upheld and data is well-protected. As for developers that are resistant to the speed reduction that proper security practices require, it is important to highlight that overall SecDevOps significantly reduces how much time the team has to spend fixing security issues. This not only saves time spent fixing the issues but also money and goodwill since attacks and exploits are avoided in the first place. Having time to resolution for security issues as a key performance indicator is an effective way that SecDevOps can visualize improvement and have a better understanding of where teams may be falling short while bridging the gap between security professionals and developers (Gudepu).

In summary, while transitioning from traditional DevOps, or even DevSecOps, to SecDevOps requires a significant time, energy, and financial investment, its benefits cannot be overstated. In an age where major cyberattacks reach headlines and lose companies millions daily, security has never been more important. Above simply shifting security into the software development cycle, security must instead become the very first step and the very first thought. Through SecDevOps teams can deliver secure software, applications, and infrastructure while still maintaining a relatively quick release schedule.

Works Cited

Aqua Security. “SecDevOps in Your Organization: A Practical Guide.” *Cloud Native Wiki*, 28 Nov. 2022, https://www.aquasec.com/cloud-native-academy/devsecops/secdevops/.

Bhandari, Parveen. “SecDevOps: What Is It, and Do You Need It in Your Organization?” *XenonStack*, Xenonstack Inc, 19 Aug. 2022, https://www.xenonstack.com/insights/secdevops#:~:text=in%20the%20SDLC.-,What%20is%20SecDevOps%3F,than%20being%20held%20by%20them.

Gill, Navdeep Singh. “Complete Guide to Ansible Configuration Management and Its Features.” *XenonStack*, Xenonstack Inc, 31 July 2022, https://www.xenonstack.com/blog/ansible.

Gill, Navdeep Singh. “DevOps Assembly Lines and Continuous Integration Pipelines.” *XenonStack*, Xenonstack Inc, 14 Mar. 2023, https://www.xenonstack.com/blog/devops-assembly-line.

Greenfield, Brittany. “What Is SecDevOps?” *Wabbi*, 1 Nov. 2019, https://wabbisoft.com/what-is-secdevops/.

Gudepu, Raja. “SecDevOps vs DevSecOps: A Distinction with a Difference.” *Security Magazine RSS*, Security Magazine, 29 Dec. 2021, https://www.securitymagazine.com/articles/96811-secdevops-vs-devsecops-a-distinction-with-a-difference.

Nidecki, Tomasz Andrzej. “DevSecOps vs. SecDevOps.” *Acunetix*, 28 Feb. 2022, https://www.acunetix.com/blog/web-security-zone/devsecops-vs-secdevops/.

Reynolds, Justin. “SecDevOps: A Practical Guide to the What and the Why.” *Plutora*, 4 May 2022, https://www.plutora.com/blog/secdevops-a-practical-guide-to-the-what-and-the-why.

Singh, Gursimran. “Introduction to Helm - Package Manager for Kubernetes.” *Starter's Guide to Helm - Package Manager for Kubernetes*, Xenonstack Inc, 25 July 2022, https://www.xenonstack.com/blog/helm-kubernetes/.

Singh, Gursimran. “Puppet: The Ultimate Solution to Configuration Management.” *XenonStack*, Xenonstack Inc, 16 Aug. 2022, https://www.xenonstack.com/blog/puppet.