WHITE PAPER

The Importance of DevSecOps

WHITE PAPER

DevSecOps Overview

What Is DevSecOps?

DevSecOps refers to the integration of security practices into a DevOps software delivery model. Its foundation is a culture where development and operations are enabled through process and tooling to take part in a shared responsibility for delivering secure software.

The definition of DevSecOps Model, at a high-functioning level, is to integrate security objectives as early as possible in the lifecycle of software development. While security is "everyone's responsibility," DevOps teams are uniquely positioned at the intersection of development and operations, empowered to apply security in both breadth and depth.

The Importance of DevSecOps

WHY ARE DEVSECOPS PRACTICES IMPORTANT?

Digital transformation has become an existential requirement for almost all enterprises. Such transformation includes three significant motions: more software, cloud technologies and DevOps methodologies.

More software means more of the organization's risk becomes digital, raising the level of technical debt and therefore application security, making it increasingly challenging to secure digital assets.

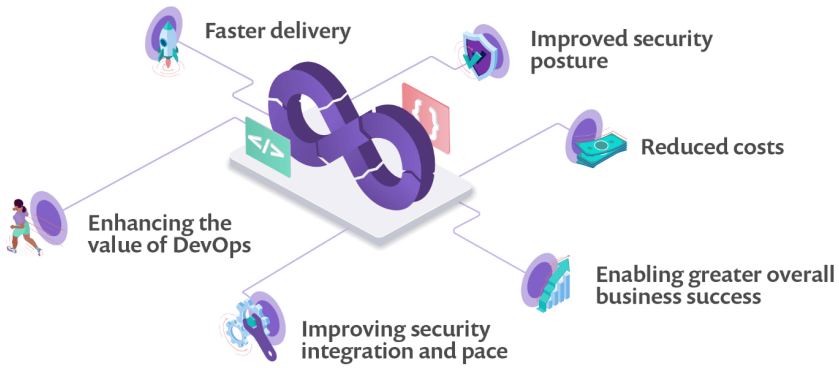
Cloud means use of newer technologies that introduce different risks, change faster, are more publicly accessible — eliminating or redefining the concept of a secure perimeter. It also means many of the IT and infrastructure risks are moved to the cloud, and others are becoming purely software defined, reducing many risks while highlighting the importance of permission and access management.

Lastly, DevOps means a change to how software is developed and delivered, accelerating the cycle from writing code to delivering customer value to learning from the market and adapting. Empowered development teams ship software continuously and faster than ever, making technology and implementation decisions autonomously and without intermediaries. The traditional slow feedback loops that bog down development are not tolerated as teams increasingly prioritize being self-sufficient — you write it, you run it.

As the rest of the organization evolves, security teams are faced with greater demands and often become more of a bottleneck. Legacy application security tools and practices, designed for the slower-paced pre-cloud era, put security teams in the critical path of delivering high quality applications. These teams, understaffed due to the severe security talent shortage, become a bottleneck and fail to keep up. As a result, dev teams ship insecure applications, security teams burn out, and security becomes a naysayer, negating the acceleration the business is seeking.

To deal with these challenges, people started changing their practices and this gave birth to DevSecOps. A DevSecOps culture brings security into the DevOps fold, enabling development teams to secure what they build at their pace, while also creating greater collaboration between development and security practitioners. It allows security teams to become a supporting organization, offering expertise and tooling to increase this developer autonomy while still providing the level of oversight the business demands.

6 BENEFITS OF THE DEVSECOPS MODEL

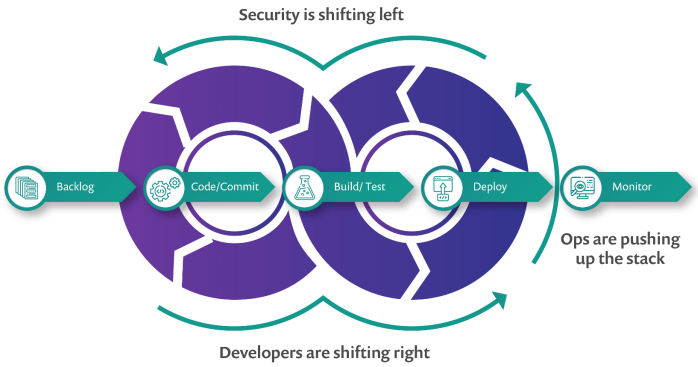


1. **Faster delivery:** The speed of software delivery is improved when security is integrated in the pipeline. Bugs are identified and fixed before deployment, allowing developers to focus on shipping features.
2. **Improved security posture:** Security is a feature from the design phase onwards. A shared responsibility model ensures security is tightly integrated—from building, deploying, to securing production workloads.
3. **Reduced costs:** Identifying vulnerabilities and bugs before deploying results in an exponential reduction in risk and operational cost.
4. **Enhancing the value of DevOps:** Improving overall security posture as a culture of shared responsibility is created by the integration of security practices into DevOps. The Snyk/Puppet 2020 DevSecOps Insights Report found this to be the case in mature DevSecOps organizations.
5. **Improving security integration and pace:** Cost and time of secure software delivery is reduced through eliminating the need to retrofit security controls post development.
6. **Enabling greater overall business success:** Greater trust in the security of developed software and embracing new technologies enables enhanced revenue growth and expanded business offerings.

DevSecOps Adoption: Integrating Security into the CI/CD Pipeline

Most modern DevOps organizations will depend on some combination of continuous integration and continuous deployment/delivery systems, in the form of a CI/CD pipeline. The pipeline is an excellent foundation from which a variety of automated security testing and validation can be performed, without requiring the manual toil of a human operator.

To integrate security objectives early in the development of an application, start before the first line of code is ever written. Security can integrate and begin effective threat modeling during the initial concept of the system, application, or individual user story. Static analysis, linters, and policy engines can be run any time a developer checks in code, ensuring that any low-hanging fruit is dealt with before the changes move further upstream.



DevSecOps Adoption: Integrating Security into the CI/CD Pipeline

Software composition analysis can be applied holistically to confirm that any open-source dependencies have compatible licenses and are free of vulnerabilities. A behavioral by-product of this is that developers feel a sense of ownership over the security of their applications, getting immediate feedback on the relative security of the code they've written.

Once the code is checked in and builds, you can start to employ security integration tests. Running the code in an isolated container sandbox allows for automated testing of things like network calls, input validation, and authorization. These tests generate fast feedback, enabling quick iteration and triage of any issues that are identified, causing minimal disruption to the overall stream. If things like unexplained network calls or unsanitized input occur, the tests fail, and the pipeline generates actionable feedback in the form of reporting and notifications to the relevant teams.

Once the deployment artifact passes the first battery of integration tests, it moves on to the next stage of integration testing. Now it will be deployed to a wider sandbox, a limited copy of the eventual production environment. At this stage, further security integration testing can be performed, albeit with a different objective.

Now, things like correct logging and access controls can be tested. Does the application log relevant security and performance metrics correctly? Is access limited to the correct subset of individuals (or prevented entirely)? Failure again results in action items to the relevant teams.

Finally, the application makes its way to production. However, the work of DevSecOps continues in earnest. Automated patching and configuration management ensure that the production environment is always running the latest and most secure versions of software dependencies. Ideally, immutable infrastructure means that the entire environment is frequently torn down and rebuilt, constantly subjected to the battery of tests along the breadth of the pipeline.

Utilizing a DevSecOps CI/CD pipeline helps integrate security objectives at each phase, without adding burdensome bureaucracy and gatekeeping, allowing the rapid delivery of business value to be maintained.

Empowering DevSecOps Culture

So how can an organization make the evolutionary climb from "DevOps" to "DevSecOps"? It's not as simple as just handing an already busy DevOps team a set of security KPIs and calling it a day. It needs to be a collaborative, shared culture of rapid iteration.

If integrating security objectives early is the goal, it needs to be as painless as possible to do so. The burden of integrating security teams and objectives into the value stream should not fall to the developers. Adding additional steps will only lengthen the time it takes to deliver features to customers. Security should be a nimble organization, with a pragmatic approach to applying security with minimal disruption.

During the planning process, particularly as it relates to infrastructure, security engineers should be involved in discussions, empowered to push back on poor/insecure choices, but knowledgeable enough to offer alternatives. Oftentimes, overburdened security teams simply say "no," and outsource the finding of alternatives to the DevOps teams. Again, this goes back to empowering security organizations with the right level of resources.

With security and DevOps collaborating early and often, security objectives have been tightly woven into the fabric of the infrastructure. Features and applications that are deployed to production will be the result of a comprehensive and effective collaboration between security, development, and operations. Security won't have to go ask for extra features or auditing from development teams after the fact; they will know these were built in from day one.

If your organization has evolved to practice DevSecOps, you know that not only are you iterating quickly, delighting your customers with new features and improved functionality, but that you are delivering that experience with a level of security to match.

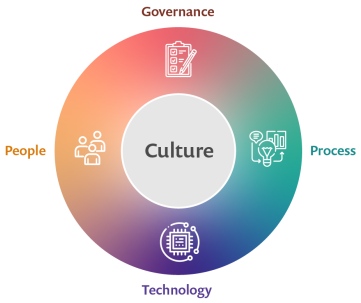
DevSecOps Culture

Online media and marketing are filled with terms like DevSecOps methodology, DevSecOps model, or DevSecOps techniques. However, in order to be successful, organizations must understand that DevSecOps is first and foremost a culture. DevSecOps culture focuses on uniting the normally siloed roles of Development, Security, and Operations into a collaborative shared-responsibility paradigm. It seeks to break down barriers of finger pointing and deflection. Instead, it aims to build empathy and common goals among various disciplines within the organization.

4 pillars of DevSecOps Culture

There are four key pillars that must be considered when looking to shift the DevSecOps culture of an organization:

- **People**
- **Processes**
- **Technologies**
- **Governance**



DevSecOps principles build on these four intersecting parts, by eliminating the silos and creating a collective focus. This environment of shared responsibility and mutual empathy requires breaking down barriers between teams. Consequently, people are the starting point and the foundation of any DevSecOps implementation. Restructuring DevOps and Security teams to establish efficient cooperation between them, as well as offering good quality and targeted training to the wider organization will ensure that security becomes a frame of mind rather than a hindrance.

The next step is to introduce supporting processes, with the aim to further improve collaboration between people as well as achieving more secure development processes as a whole. These process changes are designed to span the three functional areas of development, security, and operations providing cohesion and uniformity between them. They establish a common goal of secure and stable software developed at scale.

Additionally, the DevSecOps approach requires having the right technologies in place to enable employees to execute these processes as well as automate them. This, ultimately, reduces the organization's attack surface and enables effective management of the technical security debt. Technology, the tooling that supports a DevSecOps pipeline, is often the area most organizations think of first.

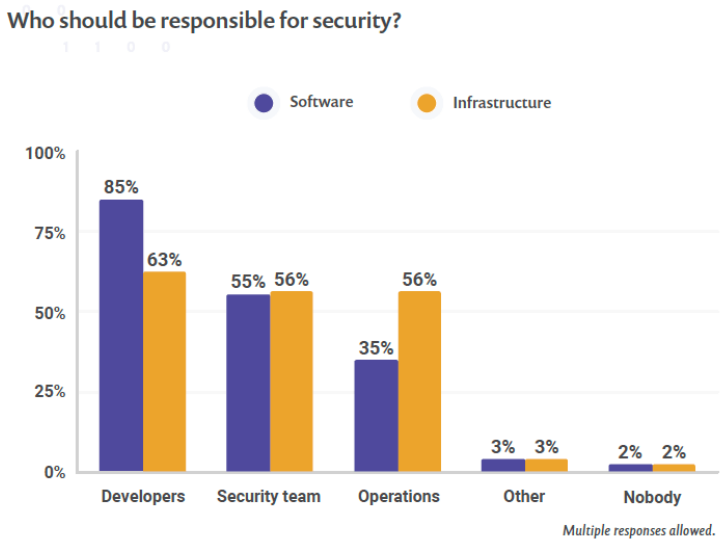
And finally, one of the least thought of elements of a true DevSecOps culture is governance. While the people, processes, and technologies come together to support each other, governance also plays a key role. It measures the performance of the other elements and can point out where more focus is needed to ensure all parts of the culture form together.

As best practice, before starting on the journey to DevSecOps, organizations should assess their current development, security, and operations teams. The objective of this assessment is to plan for how DevSecOps approaches can be integrated into the organization. Visibility of the organization's overall readiness to adopt a DevSecOps paradigm should be established with clear action items for addressing any deficiencies.

People: empowering the team

Rather than following the habit of calling humans ‘the weakest link’ when looking at security-related factors, we can empower them to be the strongest link and an important part of a company’s defenses. A modern security culture and mechanisms that work for, rather than against, people are crucial to making security work. Moving to DevSecOps starts by challenging the way traditional security teams integrate with the wider business. But the focus needs to be wide-ranging and not forget operations. Strong links between development, security, and operations teams ensure earlier feedback on the quality, from a security point of view, of the code, software or application, and in turn reduce the costs of implementing fixes.

Traditionally development was responsible for fast delivery, security was responsible for application security, and operations was responsible for stability. DevSecOps destroys those silos, eliminates finger pointing and unites all three roles in a common goal of quickly delivering software that is both secure and stable. Everyone has equal stake in all three objectives and uses their own expertise to support the others. Accountability, empathy, enablement become crucial characteristics of successful teams. To support this, underlying processes must change as well.



As more organizations adopt DevSecOps delivery models, attitudes about who is responsible for security are shifting. Source: 2020 State of Open Source Security report

Curious how Snyk can help?

Snyk is a developer-first platform for building software securely. Learn more about how Snyk can help you secure cloud native applications across your IDEs, repos, containers, and pipelines.

Learn More Here

snyk