anchore

EXPERT GUIDE FOR DEVOPS TO DEVSECOPS TRANSFORMATION

A GUIDE TO THE NEXT STEP IN YOUR ORGANIZATION'S DEVOPS JOURNEY

CONTENTS

Executive Summary

A move to DevSecOps is about the transformation of processes, tools, frameworks, job roles, and even culture. Some teams will transition from DevOps to the more security-focused DevSecOps bringing together formerly separate teams, roles, and processes to assert security and compliance goals and responsibilities through standard processes and technology. There are also teams transitioning from a legacy software development lifecycle (SDLC) to DevOps and DevSecOps at the same time.

Introduction

Companies and government agencies face continuous demands to speed the delivery of new software and features to customers, constituents, and employees. Speed, delivery velocity, and security were not necessarily mutually exclusive in a traditional waterfall software development life cycle (SDLC).

However, application security for the average commercial or public sector enterprise meant doing security scans and remediation at the end of a long, arduous waterfall SDLC. Security and development too often were strangers to each other or even natural enemies in the wild. This model often introduced long delays as developers scrambled to fix problems identified by security teams, and now, with the shift to DevOps approaches, it's showing cracks with age.

Prioritizing security as a design principle built into your development flow doesn't happen overnight — which is why it requires a DevOps to DevSecOps transformation. You need collaboration from your developers, cybersecurity experts, sysadmins, business stakeholders, and even your executives.

Digital transformation efforts and the new world of work during and after the pandemic will further spur DevOps to DevSecOps transformations. Fortunately, cloud computing, the rapid expansion of automation tools and DevOps teams, combined with the proliferation of open source software (OSS), now gives development teams the platforms, tools, processes, and frameworks to deliver software at higher velocity while maintaining quality and security.

DevOps versus DevSecOps

DevOps brings together cultural philosophies, practices, and tools that increase your organization's ability to deliver applications and technology services at a high velocity. It can enable you to shrink your quarterly or monthly release cycles to weekly or daily. You can also use DevOps to grow and improve your products faster than with a traditional waterfall software development process and siloed infrastructure management.

DevSecOps keeps all the goodness that's DevOps while baking security into each stage of the DevOps cycle. It knocks down the silos that traditionally stand between your development, security, and operations teams. Benefits of DevSecOps include:

- **Reduced risk of security incidents** because DevSecOps improves an organization's security posture by shifting security left and placing automated security scans inside the CI/CD toolchain, enabling your teams to discover and remediate issues before they hit production

- **Quick response to security issues** because DevSecOps increases your security focus through continuous assessments while giving you actionable data to make informed decisions about the security posture of apps that are in development and ready to enter production

- **Accelerated feature velocity** because DevSecOps teams have the data, tools, and automation to mitigate unexpected and unforeseen risks to avoid last-minute "fire drills" and accelerate remediation times

- **Lower remediation costs** because DevSecOps accelerates your development life cycle by design, enabling you to streamline resources, solutions, and processes

As security and compliance continue to become more critical for cloud applications, the definitions of DevOps and DevSecOps will merge. Rising concerns about security and compliance are driving a broad set of organizations, including some United States government agencies to implement DevSecOps to improve compliance. For example, the United States Department of Defense (DoD) has some major DevSecOps initiatives attracting DoD customers using legacy SDLC processes. Some are already using DevOps to build their software.

DevSecOps Maturity Model

It can help to benchmark your progress in a DevOps to DevSecOps transformation using a DevSecOps maturity model. DevSecOps is another step in your DevOps journey that embeds continuous security and compliance into your DevOps processes and toolchain.

STAGE	BEGINNER	INTERMEDIATE	ADVANCED	EXPERT
Automation	Minimal	Limited	Integrated	Continuous
Integration	» N/A	» Begin integration of security tools into the DevOps delivery cycle	» Broaden integration of security tools into multiple points in the DevOps delivery cycle	» Continuous automation across the DevOps lifecycle with a standardized platform
Key Focus Areas for Security	» Point-in-time security checks	» Improve AppSec » Initiate the move to security automation » Increase software delivery velocity	» Cultivate DevSecOps as an organization-wide strategy » Increase software delivery velocity	» Security » Automation tied to consistent
Tools	» Standard SW	» Individual DevOps and DevSecOps tools	» Integrated DevOps and DevSecOps toolchains	» Mature DevOps and DevSecOps platforms including SBOM generation as quality gate
Concerns	» Human error » Ad hoc security » No formal security training for developers	» Security » Process » Productivity	» Security » Process » Productivity	» Security » Process » Productivity

Beginner (Minimal Automation)

Security tests and checks are often point-in-time events that occur relatively late in the development lifecycle. Security testing may be limited to traditional penetration testing of the runtime environment and manual checklists. Vulnerability checks, if they occur, often generate significant false positives and take significant developers time to triage and address.

Security at this stage is often ad hoc, with developers yet to receive any formal security training.

Intermediate (Limited Automation)

Development teams begin implementing DevOps practices and integrating a DevOps toolchain that includes CI/CD and infrastructure as code. Container-based development is growing. DevSecOps adoption is at the department or even just at the team level.

Signs of security automation begin to appear, such as the automation of container vulnerability scanning and some selected security tests. Automated security checks are likely occurring at only one or two points of the DevOps toolchain.

Advanced (Integrated Automation)

DevSecOps grows into an organization-wide strategy. With broader adoption, an automation strategy for application and infrastructure development and management takes form. Container-based development becomes the norm for new applications and those migrating to the cloud. DevOps teams can now embed security into their existing processes using containers, Kubernetes (K8s), and public cloud services.

United States federal government agencies and DoD elements reaching advanced maturity are taking advantage of self-service cloud brokerages, AWS GovCloud, and those doing highly classified work could even deploy applications to the intelligence community's C2S.

Bottom line, organizations in the advanced phase of DevSecOps maturity are seeking to improve how they deploy and secure applications at scale.

Expert (Continuous Automation)

Few organizations have yet to reach an expert state of DevSecOps maturity. When an organization achieves this maturity level, it often parallels cloud maturity. They are API and cloud-native first. They are embedding continuous security and compliance at each stage in their DevOps process. They are implementing emerging technologies such as microservices, serverless, and artificial intelligence/machine learning (AI/ML) to strengthen their application development and infrastructure security.

Inside a DevOps to DevSecOps Transformation

DevSecOps is another step in the DevOps journey for your organization. Here's a breakdown of a typical transformation:

PHASE	DESCRIPTION	ACTORS
1	Gain Stakeholder Buy-in and Consensus about DevSecOps Objectives	» Executive Stakeholders » Technology Leadership » Development Team » DevOps Team » Security Team » Operations Team
2	Identify and Integrate Security Requirements into Your Existing DevOps Processes	» Technology Leadership » Development Team » DevOps Team » Security Team » Operations Team
3	Begin to Integrate Security Tools into Your DevOps Toolchain	» Development Team » DevOps Team » Security Team » Operations Team
4	Mature Security from Independent Processes and Tools to Continuous Automation	» Development Team » DevOps Team » Security Team » Operations Team
5	Expand DevSecOps	» Executive Stakeholders » Technology Leadership » Development Team » DevOps Team » Security Team » Operations Team

Phase 1: Gain Stakeholder Buy-in and Consensus about DevSecOps Objectives

The first phase of a DevOps to DevSecOps transformation is to do the upfront preliminary work necessary to make this next step in your DevOps journey. If you are moving from a waterfall SDLC model, this phase is even more critical for your teams, as you will have to put more time and effort into DevOps training to bridge any knowledge gaps between your current processes and DevSecOps.

The crucial facets of Phase 1 are:

Analyze your Development Process Maturity

Whether DevSecOps is just the next step in your DevOps journey or you're making your initial foray into DevSecOps straight from a waterfall SDLC, a critical step in the first phase is to analyze the maturity of your software development process. Your analysis should include:

- Document any current state processes
- Gather any reporting data about your current development processes
- Interview key developers about what's working and not currently working in your development processes
- Interview key security team members about what's working well and what's not working in their processes and procedures that support your applications in development and production

Define DevSecOps for your Organization

DevOps and now DevSecOps can mean many things to people. Software vendor marketing and the OSS community each put their spin on the definition of DevSecOps. Therefore as part of your outreach it's important to define DevSecOps for your organization, including:

- What DevSecOps means to your organization
- The expected outcomes after moving to DevSecOps
- The tools and processes your organization is putting into place to ensure employee success

Spare your teams from any misunderstandings and document your DevSecOps definition. Post that definition in a place that's accessible to all your team members and stakeholders. It's not about creating a project charter for your DevOps to DevSecOps transformation, but defining your true north.

Foster a DevSecOps Culture

Like DevOps, you can't buy DevSecOps. Your managers and key technology team members need to work together to foster DevSecOps cultural philosophies that take your DevOps foundation to DevSecOps transformation.

Here are some vital elements of DevSecOps culture that are important to foster during and after your DevOps to DevSecOps transformation:

Continuous Feedback and Interaction

Cross-functional DevSecOps teams may be collaborating remotely which can create challenges with continuous feedback. It's not about a manager delivering feedback on the DevSecOps team's performance. Instead, it's about enabling teams to collaborate more effectively. ChatOps tools such as Slack, Microsoft Teams, and MatterMost can now replace email for DevSecOps teams. As technology such as artificial intelligence (AI) improves, you can expect to see more automation through chatbots.

Container-Based Architectures

The shift to cloud-native applications is driving the adoption of container-based delivery models. DevSecOps plays a critical role in the move to container-based architectures, which can be a cultural change in and unto itself for DevOps teams. A proper and robust implementation of containers changes developer and operations cultures because it changes the development model of how architects design solutions, programmers create code, and how operations teams maintain production applications.

Team Autonomy

Like DevOps, DevSecOps is no place for micromanagers at any level of your organization. A standard part of DevSecOps culture is enabling your teams to choose their own tools and create their processes based on the way they work. DevSecOps also promotes distributed decision making to support greater innovation and delivery velocity.

Automation

DevSecOps extensively embeds automation for security checks and remediation workflows directly into DevOps processes and toolchains. An automation strategy that extends to security is a sign of a healthy DevSecOps culture.

DevSecOps Training for Developers

Another step to security becoming part of everyone's job is to provide security training for your developers. Training could take the form of in-house developer training in casual formats such as Lunch and Learns or more formal training classes conducted by your organization's training department. Another option is to send your developers to a third-party training provider to get the requisite security training. Depending on your security ambitions (and budget), there is always the option to send your DevOps team members to get a DevSecOps vendor certification such as the DevSecOps Foundation certification from the DevOps Institute the Certified DevSecOps Professional (CDP) from practical-devsecops.com.

Phase 2: Identify and Integrate Security Requirements into your Existing DevOps Processes

As your organization enters Phase 2 of your DevOps to DevSecOps transformation, it's time to integrate security processes and tools into your DevOps life cycle. If your enterprise is already using DevOps, this phase integrates security tools into your existing DevOps toolchains. It's also time to perform a security audit on your DevOps toolchains themselves to ensure they have been deployed and configured securely.

If your organization is taking the fast track to DevSecOps from a waterfall SDLC or other legacy development process, security needs to become a requirement of your CI/CD toolchain build.

Implement your Core Security Tools for DevSecOps

The following table defines some security tools you'll want to integrate into your DevSecOps toolchain:

SECURITY TOOL	PURPOSE
SCA	Software Composition Analysis (SCA) automates visibility into OSS for risk management, security, and license compliance.
SAST/DAST	Static Application Security Testing (SAST) is a white box method of security testing. Dynamic Application Security Testing (DAST) is a black box testing method that examines an application as it's running to find vulnerabilities that an attacker can exploit.
Container Security	For containerized applications, these tools can provide deep inspection of container images and OSS or third-party dependencies throughout the lifecycle of a container including the CI/CD pipeline, container registry, and runtime. These tools can track vulnerabilities, secrets, malicious code, and other policy violations.

Vulnerability Assessment	Vulnerability scanning detects errors in your codebase. You can also use vulnerability scans to search for weaknesses throughout the systems your developers are building. Look for a vulnerability scanner that integrates with your existing CI/CD toolchain. You can automate these tools to scan your application in development after each code modification and identify any code that contains vulnerabilities.
Infrastructure as Code (IaC) & Configuration Management Tools	Infrastructure as Code (IaC) and Configuration Management (CM) tools automate the application of configuration states. You can also use CM tools to support application scaling and deployments. CM tools are available for cloud environments or on-premises.
Threat Modeling & Risk Assessment	Determines which threats are most likely to affect your application and its deployment. These tools can also perform a risk assessment to prioritize risks and examine your in place controls. Additionally, these tools can deliver a report on which controls you still need to put in place.
Embedded Security Controls	A Runtime Application Self-Protection (RASP) solution monitors application security and blocks any API calls that appear unsafe. RASP solutions are available as web server plugins, servlet filters, and library replacements. A typical RASP solution includes APIs for scripting and runtime policies.
Log and Event Monitoring	Log and event monitoring tools are necessary in DevSecOps toolchains because developers and sysadmins have automated privileges more often than not.

Phase 3: Begin to Integrate Security Tools in your DevOps Toolchain

The automation phase includes analysis, outreach, and experimentation. Integration of security tools and software in the CI/CD pipeline to run vulnerability scans, configuration checks, container hardening, code analysis, compliance checks, and attestation artifacts ensures security issues are found early, when they are easier to fix, and simplifies the collaboration between security and DevOps to confirm security.

Applying automation to everyday software development tasks such as quality assurance and security checks will require collaboration. Expect a push and pull between your security and development teams. Security teams are inclined to want to embed as many automated security checks as possible, even to the extreme. Developers and DevOps teams may be concerned that automated security checks will uncover false positives or low priority issues that will slow their velocity.

5 Tips for Introducing Automation into your DevOps Toolchain

Here are some tips for introducing security process automation into your DevOps toolchain:

1. **Prioritize automation:** Dispel the notion in your stakeholders that you'll be able to automate every task along your toolchain. Engage with your stakeholders to learn their automation priorities and take that feedback into an automation strategy for your development or DevOps teams.

2. **Consult developers:** Engage with your development teams — not just the team leads and managers — about how automation of security checks can help them perform their jobs. Listen to their concerns with empathy. Consult with them about priorities and how a first step towards automation won't affect their delivery velocity.

3. **Automate with a plan:** Create an automation implementation plan for your security tools — a sort of automation roadmap — that charts how you'll introduce automation into your toolchains. Start small and expand with automation across your toolchains. Seek a small project such as a patch or a feature update to test your implementation plan.

4. **Automate security checks as a start:** Start small and automate an initial set of security checks for one of your development teams as a proof-of-concept project. Document your findings from this small project, especially the lessons learned and any other feedback from the DevOps team members working on the project.

5. **Communicate often:** Communicate the successes, lessons learned, and yes, even the mistakes made on the pilot project to your stakeholders and internal DevOps community.

If your organization already has a DevOps center of excellence or is in the midst of establishing one, use it as an opportunity to bring in voices from across your organization to learn more about how they see automation affecting their jobs.

Phase 4: Mature Security from Independent Processes and Tools to Continuous Automation

There can be security changes that take place on the move to DevSecOps that may adversely affect operations and even security compliance. There could be changes to tools, process, and even staffing that can change the way teams develop and secure software.

Automation is a foundation of DevSecOps because it removes the prospect of human error from some common build tasks and security checks. If you're building and running cloud workloads, you need automation.

Here are some methods that can help your organization move to continuous automation:

- **Create a continuous automation roadmap for the project** if not your entire development organization that captures the skill sets and gaps for your teams with a plan to skill up team members as necessary. The roadmap should also include any automation tool standards that you may set for your organization and target dates for their implementation.

- **Establish an automation center of excellence (CoE)** for your organization that centralizes automation expertise from around your organization and gives a platform for information sharing, automation training, and the sharing of lessons learned between groups.

- **Bring in outside consultants or contractors** who specialize in automation to mentor and guide your teams through maturing your security. If you choose this option, be sure to capture the knowledge of these experts through documentation, training, and interviews so your organization doesn't lose that knowledge at the end of their contract.

It's important that your developers, operations, and security teams collaborate with each other during security reviews and other engagement points to set priorities. Security teams may prioritize security measures that adversely impact speed of delivery. Developers may overlook security holes that open source dependencies or system configurations may cause that will compromise the security and compliance of your systems.

When you conduct security reviews, a prime collaboration channel, during your DevOps to DevSecOps transformation, you give your developers and security staff a forum during which they can educate each other on their team's priorities and informed tradeoffs.

Phase 5: Expand on DevSecOps

As your organization crosses into Phase 5 of your DevOps to DevSecOps transformation, it's time to expand your activities to more teams. It's often best to not throw the switch on moving to Phase 5 as an entire organization. Rather, look for natural breaks in your project teams' schedules for them to move to a DevSecOps model. For example, let's say you have a DevOps team that just launched a new product release. After catching their collective breaths, they're working on bug fixes that come in from the field. This is the time to work with the team to move them to a DevSecOps model. Moving the team midstream during a release may prove to be more of a distraction than a help to the project team more focused on delivery than learning something new.

This phase also marks organization-wide standardization on a standard DevOps/DevSecOps toolchain or platform moving away from each team choosing their own tools. The security team will also be pushing out standard security tools across development teams to ensure baseline security and adherence to compliance policies. A software bill of material (SBOM) will serve as an entry gate to commercial, OSS, and custom-built code entering the DevSecOps toolchain.

Practice Continuous Learning and Iteration

There is no formal end in a properly executed DevOps to DevSecOps transformation. While your organization can take the step from DevOps to DevSecOps and adopt the principles and foundations, the learning and iteration need to continue on past the transformation.

As there seems to be no single accepted DevSecOps definition for the industry, you can expect to learn a lot as your DevSecOps moves gain momentum and your processes mature. You also need to prepare your organization for changes in DevOps and DevSecOps thinking that might benefit your internal efforts.

Taking the Fast Track to DevSecOps

There might also be cases where your organization is already experimenting with DevOps tools or contemplating how to move to DevOps. Still, for reasons of security and compliance, you need to fast forward to DevSecOps.

Much of the advice in this white paper still applies if your organization wants to leave behind the limitations of waterfall SDLC and move to DevSecOps. Your team will choose DevOps and security tools as part of your toolchain build-out. Other readers of this white paper may already have DevOps toolchains in place and thus only need to integrate new security tools to integrate into their existing toolchains.

Mitigating DevSecOps Adoption Challenges

DevSecOps adoption brings with it certain challenges that require a full team effort to mitigate. Here are some ways to mitigate DevSecOps adoption challenges:

Start Small and Build Upon Successes

Whether your organization is taking the next step from successful DevOps projects to DevSecOps or taking a giant step from waterfall SDLC to DevSecOps, the key is to start with a small proof-of-concept project, take your lessons learned, and then build upon your successes.

Choosing a small project is best done involving a business stakeholder who's open to having one of their smaller projects moved to a DevSecOps development model. Application migration to the cloud is an opportune time to conduct such a proof-of-concept project.

Foster Internal DevSecOps Advocates and Security Champions

A move to DevSecOps takes more than just the CIO or the CISO to make happen. Rather, it requires that you build internal DevSecOps advocates and security champions to spread the good word.

Here are some common advocates and champions you should eye:

- The individual contributor who's well respected for their technical chops and is an early adopter. Think of the person to whom the other developers go with their questions.
- The business stakeholder who has something to gain with your move to DevSecOps whether it's improving their security posture or reducing risk of a security breach. Think about the sales or business development leader that can better serve their customers if your company can deliver additional features and versions securely. Another example would be a government agency manager (with budget control) whose division is migrating their legacy applications to the cloud and must meet FedRAMP compliance.
- The development team project lead that handles teams delivering code. DevSecOps offers this person automation and frameworks that can take some work off their team's shoulders, letting them focus on more strategic tasks.
- The security champion — a key to taking DevSecOps to the next level — is a developer, engineer, architect, or engineer who takes the lead on security objectives within their project team.

Implement Automation Incrementally

Automating security checks is an important element in the adoption of DevSecOps, but it can cause issues if it generates too many false positives or alerts for non-critical issues. Irrelevant alerts and warnings can waste the time of developers and cause resistance to further automation of security checks.

Security teams will need to define automated and checks to ensure they are relevant and important for the organization. Collaborate with development teams to tune security policies, provide automated remediation recommendations, and integrate tools with their existing tools and workflows.

Prevent Developer Overload

When you add security duties to a developer's existing workload, you instantly risk developer overload. After all, we're just entering an era where developer-friendly application security tools are entering the market. Too many application security tools are designed for security teams and often require hours to days until the security team delivers their findings to the development team for their remediation.

It's up to your leadership to put the tools, training, and frameworks in place that help prevent developer overload. As this white paper goes to publication, we're still amid a global pandemic, further putting a strain on developers and other team members working on your projects. Such a major change in job duties can always benefit from developer input at each stage of the decision and implementation process.

Next Steps: DevOps to DevSecOps Transformation

Here are some immediate steps you can take to start your organization's DevOps to DevSecOps transformation:

- Start with an analysis of the current state of your development lifecycle whether your organization is already relying on DevOps or another methodology.
- Define DevOps and DevSecOps for your organization because definitions vary across industry and government. You need to own your own DevSecOps definition to help set expectations for what you expect to get from your transformation.
- Create a DevSecOps culture that's not beholden to your tool choices. Every level of your organization has a role to play. You want to build a cadre of internal champions amongst your developers, and sysadmins, not just your stakeholders and management.
- Select security tools that integrate into your existing DevOps tools and processes. Make sure you don't overload developers as you implement automation.

About Anchore

Anchore accelerates the development of secure and compliant cloud-native applications. Our suite of container security solutions seamlessly embeds in the DevOps lifecycle with continuous security and compliance checks early in the software development process. From sourcing to CI/CD pipelines to production, Anchore's solutions protect the software supply chain and prevent container security risks from reaching production. Using Anchore as part of the DevSecOps toolchain creates a reliable way to detect issues earlier, save developers time and lower the cost to fix vulnerabilities. Built with an open source foundation, Anchore solutions provide transparency into source code and the benefit of peer reviews.

Headquartered in California with offices in Virginia and the UK, Anchore customers include large enterprises and government agencies that require secure and compliant cloud-native applications. To learn more about Anchore's solutions, visit **www.Anchore.com**.

anchore

info@anchore.com

anchore.com