A guide to implementing DevSecOps

by Will Kelly

We are Opensource.com

Table of Contents

Will Kelly

Will Kelly is a product marketer and writer. His career has been spent writing bylined articles, white papers, marketing collateral, and technical content about the cloud and DevOps.

Opensource.com, TechTarget, InfoQ, and others have published his articles about DevOps and the cloud. He lives and works in the Northern Virginia area. Follow him on Twitter:@willkelly.

DevSecOps: An open source story

Recent supply chain breaches, plus President Biden's new Cybersecurity executive order, are bringing renewed attention to DevSecOps' value for the enterprise. DevSecOps brings culture changes, frameworks, and tools into open source software (OSS). To understand DevSecOps, you must understand its relationship with OSS.

What is DevSecOps?

In its purest form, DevOps (which is an amalgamation of development and operations) is a methodology for breaking down the traditional silos between programmers and system administrators during the software delivery lifecycle. Corporations and government agencies adopt DevOps for various reasons, including improving software delivery velocity to serve customers better.

DevSecOps adds security into DevOps, further refining the concept to address code quality, security, and reliability assurance through automation, enabling continuous security and compliance. Organizations seeking to comply with Sarbanes Oxley (SOX), Payment Card Industry Data Security Standard (PCI DSS), FedRAMP, and similar programs are candidates for implementing DevSecOps.

For example, a federal government agency seeking FedRAMP compliance should use DevSecOps, because it enables them to bake security automation into each stage of their software development process. Likewise, a healthcare institution entrusted with sensitive personal healthcare information (PHI) needs DevSecOps to ensure its cloud applications meet HIPAA compliance requirements.

The more you move security mitigation to the left to tackle these issues in development, the more money you save. You also avoid potential negative headlines because your teams don't have to respond to issues in production, where remediation costs can soar far higher than if you caught them in your development environment.

You can treat the move from DevOps to DevSecOps as another step in the DevOps journey. But it's more like a transformation for your development organization and your entire business. Here's a typical framework:

1. **Analyze, communicate, and educate:** This includes analyzing your development process maturity; defining DevSecOps for your organization; and fostering a DevSecOps culture with continuous feedback and interaction, team autonomy, and automation and architecture.
2. **Integrate security into your DevOps lifecycle:** Ensure your DevOps and security teams work together.
3. **Introduce automation into your DevOps lifecycle:** Start on small dev projects and gradually expand your automation strategy.
4. **Collaborate on security changes to your DevOps toolchains:** Get your development and security teams working jointly on projects to harden your DevOps toolchain.
5. **Execute on DevSecOps:** Get your teams fully engaged with your DevSecOps toolchains and new processes.
6. **Encourage continuous learning and iteration:** Offer your developers and sysadmins training and feedback mechanisms to support developer performance and the health of your toolchains.

We're at a unique point in the history of software development, where the need to increase security and speed software development velocity are at a crossroads. While DevOps has done a lot to increase velocity, there was always more to do.

Growth of DevSecOps

The growth of DevSecOps has been visible in compliance and security-conscious arenas. For example, it has a growing following inside the security-conscious US Department of Defense. Projects such as Platform One are setting an example of how DevSecOps practices can protect open source and cloud technologies in the most security-conscious government missions.

DevSecOps has a 20% to 50% penetration within industry, according to Gartner's Hype Cycle for Agile and DevOps, 2020. The pandemic has acted as a catalyst for DevSecOps as organizations have moved application development to the cloud.

Challenges of DevSecOps

Even if you treat DevSecOps as another step in your DevOps journey, you can expect changes to your toolchain, roles on your DevOps and security teams, and how your groups interact. Over 60% of the respondents to GitLab's 2021 Global DevSecOps Survey report new roles and responsibilities because of DevOps, so prepare your people upfront and keep surprises to a minimum.

There is a variety of open source DevSecOps tools you can adopt to build out your DevOps pipeline, including:

- Alerta consolidates and deduplicates alerts from multiple sources to provide quick visualizations. It integrates with Prometheus, Riemann, Nagios, and other monitoring tools and services for developers. You can use Alerta to customize alerts to meet your requirements.
- StackStorm offers event-driven automation providing scripted remediations and responses. Some users affectionately call it the "IFTTT for ops."
- Grafana allows you to create custom dashboards that aggregate all relevant data to visualize and query security data.
- OWASP Threat Dragon is a web-based tool that offers system diagramming and a rules engine for modeling and mitigating threats automatically. Threat Dragon touts an easy-to-use interface and seamless integration with other software development tools.

DevSecOps brings a culture, much in the same way that DevOps does. Fostering a DevSecOps culture is about putting security first and making it everybody's job. DevSecOps organizations need to go beyond the mandatory corporate-wide online security training with canned dialogue and bring security into development and business processes.

DevSecOps and open source risk mitigation

Businesses and even government agencies use as much as 90% open source code. That sometimes accounts for hundreds of discrete libraries in a single application. There's no doubt that OSS saves DevOps teams time and money, but it may take a DevSecOps security model to mitigate OSS risk and licensing complexities.

Forty-six percent of respondents to Synopsys' DevSecOps Practices and Open Source Management in 2020 survey said that media coverage of open source issues affects how they implement controls in their OSS projects. Continuing coverage of the recent supply chain breaches are amping up tech leaders' concerns about the stringency of their controls.

OSS risk mitigation strategies and DevSecOps go together in many ways, such as:

- Begin generating a software bill of materials (SBOM) as a quality gate before OSS enters your software supply chain.
- Give OSS procurement the same attention as you do the vetting, purchase, and intake of enterprise software by bringing in talent from your development, security, and

corporate back-office teams. You can adapt your DevSecOps lifecycle to factor in your OSS procurement strategy.

Final thoughts

DevSecOps is a noisy topic right now. Plenty of marketers are trying to put their spin on defining it to sell more products to commercial and public-sector enterprises. Even so, the relationship between OSS and DevSecOps remains clean because DevSecOps tools and strategies offer a security gate to bring OSS into the software supply chain and your DevSecOps pipeline while maintaining security and compliance from the first step in the process.

Launching a DevOps to DevSecOps transformation

Widespread adoption of DevSecOps is inevitable. Security and delivery velocity are unrealistic expectations as part of a waterfall software development life cycle (SDLC). Businesses and government agencies are under constant pressure to deliver new features and functionality to their customers, constituents, and employees. Recent high-profile software supply chain breaches and President Biden's Executive Order to improve the nation's cybersecurity also increase the urgency for businesses and governments to move to DevSecOps.

All of that means, sooner or later, your enterprise will need to integrate security with its DevOps process.

Historically, cybersecurity teams focused on app security only at the end of a long, laborious waterfall SDLC, after scanning and remediating security issues. This model has shown cracks with age. Customer and market demands for new features, security, and compliance are at the top of executives' minds. Digital transformation efforts aimed at adjusting to the new world of work during and after the pandemic have made software security a higher priority. A DevOps process that makes security an afterthought is out of step with software users and consumers.

What's needed is a DevOps-to-DevSecOps transformation. Fortunately, cloud computing in the commercial and public sectors, combined with the influence of open source software (OSS), now gives development teams the tools, processes, and frameworks to deliver software at higher velocity while maintaining quality and security.

DevSecOps brings your security and DevOps teams to work together during the development life cycle. To make that transition, you will need collaboration from your developers, cybersecurity experts, sysadmins, business stakeholders, and even your executives.

Assessing DevOps and DevSecOps

DevOps combines cultural philosophies, best practices, and tools that allow your organization to deliver applications and services more rapidly. Shifting to daily and weekly releases enables you to reduce your quarterly or monthly releases. Using DevOps can also help you grow and improve your products more rapidly than traditional waterfall software development processes and siloed infrastructure management.

While preserving the best qualities of DevOps, DevSecOps incorporates security in every stage of the cycle. It knocks down the silos standing between your development, security, and operations teams. Benefits of DevSecOps include:

- **Prevention of security incidents before they happen:** By integrating DevSecOps within your CI/CD toolchain, you're helping your teams by detecting and resolving issues before they occur in production.
- **Faster response to security issues:** DevSecOps increases your security focus through continuous assessments while giving you actionable data to make informed decisions about the security posture of apps in development and ready to enter production.
- **Accelerated feature velocity:** DevSecOps teams have the data and tools to mitigate unforeseen risks better.
- **Lower security budget:** DevSecOps enables streamlined resources, solutions, and processes, allowing you to simplify your development lifecycle by design.

We're at peak Ops in many industries. Rest assured, the definitions of DevOps and DevSecOps will merge in the months and years to come, if only for the sake of enterprise sanity and management.

DevSecOps and OSS

DevSecOps can also play a vital role in the integration of OSS into enterprise applications. OSS and DevSecOps are becoming increasingly intertwined, especially as enterprises seek to improve the security of their software supply chains. DevSecOps can serve as an OSS remediation tool because it permits scanning automation throughout each pipeline phase. OSS is also foundational for adopting and security software containers and Kubernetes.

Final thoughts

Before your organization embarks on a DevOps to DevSecOps transformation, take a step back and define DevSecOps for your teams. Cut through the marketing. Talk about the results you hope your teams will achieve. Instill a culture of openness and collaboration, and be sure to listen to the positive and negative vantage points of your development, operations, and Quality Assurance (QA) teams.

4 steps to make DevSecOps adoption a team effort

Perhaps your organization is already experimenting with DevOps tools or considering how to move towards DevOps. Maybe you're still relying on ad hoc processes. Then suddenly your C-suite or auditors raise the need to standardize on a secure and agile development process. Enter DevSecOps.

To mitigate the challenges that come with DevSecOps adoption, you'll need to make it a team effort. Here's what you need to do.

Start small

It's vital to start with a small proof-of-concept project, apply your lessons learned, and then build upon your successes. Choosing a small project is best done by involving a business stakeholder open to moving one of their smaller projects to a DevSecOps development model. Application migration to the cloud is an opportune time to conduct such a proof-of-concept project.

Foster DevSecOps advocates across your organization

Just like DevOps, moving to DevSecOps is ultimately about people and culture.

A move to DevSecOps requires that you build internal DevSecOps advocates to spread the good word. Here are some everyday advocates and champions you should consider:

- An individual contributor and early adopter angling to build more secure applications: Think of the person to whom the other developers go with their questions.
- A business stakeholder who will benefit either by increasing security or by boosting sales with the move to DevSecOps: Think about the salesperson or business developer who can better serve their customers if your company can securely deliver additional features and versions. A government agency manager (with budget control) whose division is migrating their legacy applications to the cloud to meet FedRAMP compliance could be another potential advocate.

- Development team project leads who manage teams that deliver code but get stuck in endless rounds of security updates to mitigate security issues that made it into production: DevSecOps offers them automation and frameworks that can take some work off their team's shoulders, so they can give their attention to more strategic tasks.

Implement automation incrementally with team support

Automating security checks sounds appealing to some executives and financial staff; they hear "automation" and think "saving dollars by cutting the staffing headcount." This perspective is counterproductive for garnering support from developers.

As you move from DevOps to DevSecOps, there is a delicate balance between automation and security. Analyze what automation tools are already in place in your DevOps toolchain and determine if they support security integration. You can then build out your automation strategy from there. Collaborate with your teams who will benefit from automation at each step of the implementation. You want to communicate a way forward for your teams and quell any work-from-home anxiety about losing their jobs because of DevSecOps.

Prevent developer overload

You run the risk of creating developer overload when you add security responsibilities to their existing workload. After all, we're just entering an era where developer-friendly application security tools are entering the market.

Too many of today's 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 remediation. It's up to your leadership to put the tools, training, and frameworks in place that help prevent developer overload.

One last thing

Getting on the fast track to DevSecOps means taking care of your people and being proactive when they worry about automation. Teams benefit most from the DevOps to DevSecOps transformation when they play active roles in making the transformation happen.

Following a DevSecOps maturity model

DevSecOps is in many ways another level of DevOps maturity for an enterprise. Executive management and other stakeholders understand the concept of a maturity model, making it a helpful way to explain the value of this shift. Following a maturity model also helps you tell a story that includes the people, process, and technology changes that come with a DevOps-to-DevSecOps transformation.

Here are four typical levels of DevSecOps maturity:

Level 1: pre-DevOps (no automation)

At this level, developers perform every task manually, including creating and testing applications and systems. Team management, processes, and application security are still at a very ad hoc level.

Take the extra step to capture your lessons learned and the challenges of your pre-DevOps development era. You need to know your history, so you don't repeat it in the future.

Level 2: early DevOps/DevSecOps (lightweight automation)

Development teams standardize on some form of a DevOps toolchain to implement Infrastructure-as-Code and Compliance-as-Code. DevSecOps adoption is at the department or even just at the team level.

Mentioning DevOps and DevSecOps interchangeably in this phase is deliberate. Some organizations will fast-forward from traditional waterfall development straight to a DevSecOps model. At level 2, DevOps/DevSecOps and lightweight automation is the domain of innovative and more forward-thinking development teams. Developers are driven to find a better way to do things, either as a result of their own initiative or because a customer is asking for a DevOps approach.

Making it from level 2 to level 3 depends upon communicating and selling the successes of your early adopters of DevSecOps to the rest of your organization. Be sure to keep in touch with your early adopters and encourage them to share their DevOps and DevSecOps wins with the rest of their peers. Early win stories resonate much better than managerial mandates.

Level 3: DevOps to DevSecOps transition (advanced automation)

DevSecOps grows into a corporate or agency-wide strategy. With organization-wide support, an automation strategy for application and infrastructure development and management takes form. DevOps teams can now improve their existing processes using containers, Kubernetes, and public cloud services.

Bottom line: Organizations at this advanced phase of DevSecOps maturity are deploying applications at scale.

Level 4: full DevSecOps (full automation)

Such an expert state of DevSecOps maturity will be elusive for all but the most prominent and well-funded enterprises, those who must routinely meet the most strict cybersecurity and compliance demands. An organization that reaches this level of maturity is API and cloud-native first. These organizations are also implementing emerging technologies such as microservices, serverless, and artificial intelligence/machine learning (AI/ML) to strengthen their application development and infrastructure security.

Final thoughts

Only when you track the maturity of your processes, team culture, and tooling do you get the best current and future-state views of your organization's progress to DevSecOps. The pandemic pushed many teams to remote work in the past 18 months. As a result, teams had to mature their processes and mature them quickly to ensure their organization could still deliver to their customers. DevSecOps brings together the very cultural, collaboration, and toolchain improvements that development teams require to deliver secure and compliant software in their new world of work.

3 phases to start a DevSecOps transformation

DevSecOps is another step in the DevOps journey for your organization. Breaking down your transformation into phases facilitates working directly with developers and other team members. A phased approach also allows you to get feedback from those affected by the change and iterate as necessary.

Here are the first three phases of a DevSecOps transformation:

Phase 1: analysis, education, and training

In phase 1, you do the preliminary work necessary to make DevSecOps the next step in your DevOps journey.

This phase is even more critical for your teams if you're moving from a waterfall software development lifecycle (SDLC) model. Making that leap may require you to put more time and effort into DevOps training to bridge any knowledge gaps between your current processes and DevSecOps.

Analyze your development process maturity

Whether DevSecOps is just the next step in your DevSecOps journey or you're making your initial foray into DevSecOps straight from a waterfall SDLC, analyzing the maturity of your software development process is a critical step. An effective analysis includes:

- Documenting the current state of any processes
- Gathering any reporting data about your current development processes
- Identifying what's working and not working in your development processes by interviewing key developers

Define DevSecOps for your organization

DevOps and now DevSecOps can mean many things to people. Software vendor marketing and the open source software (OSS) community each put their spin on the definition of

DevSecOps. Spare your teams from any misunderstandings and document your definition of DevSecOps. A clear definition includes:

- 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

Writing a definition is not merely creating a project charter for your DevOps to DevSecOps transformation; it identifies your true north.

Foster a DevSecOps culture

You can't *buy* DevSecOps. Your managers and key technology team members need to work together to foster DevSecOps cultural philosophies to set a foundation for your DevOps to DevSecOps transformation.

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

Continuous feedback

Remote DevSecOps teams have their advantages and disadvantages with continuous feedback. The manager's role is not simply to deliver feedback on the DevSecOps team's performance. Instead, the purpose of feedback is to enable teams to collaborate more effectively. Open source chat tools provide the instant communication necessary for DevSecOps teams to collaborate in real time.

Container-based architectures

DevSecOps sets the stage for moving to container-based architectures that can be another cultural change for DevOps teams. A proper and robust implementation of containers changes developer and operations cultures because it changes how architects design solutions, programmers create code, and operations teams maintain production applications.

Team autonomy

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 tools and create processes based on their work. DevSecOps also promotes distributed decision making that supports greater agility and innovation.

DevSecOps training

Providing security training to your developers is another step towards making security part of everyone's job. Training could take the form of in-house developer training in casual formats such as lunch-and-learns, or it could include more formal training classes conducted by your organization's training department.

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 or the Certified DevSecOps Professional (CDP) from Practical DevSecOps.

Phase 2: integrate security into your DevOps lifecycle

During phase 2 of your DevOps to DevSecOps transformation, you integrate security processes and tools into your DevOps life cycle. If your enterprise is already using DevOps toolchains, this phase integrates security tools into your existing DevOps toolchains. This phase is also the time to perform a security audit on your continuous integration and continuous delivery/deployment (CI/CD) toolchains to ensure security.

Suppose your organization takes the fast track to DevSecOps from a waterfall SDLC or other legacy development process. In that case, security needs to become a requirement of your CI/CD toolchain build.

Phase 3: introduce automation into your DevOps lifecycle

The automation phase includes analysis, outreach, and experimentation. Applying automation to everyday software development tasks such as quality assurance and security checks isn't an exact science. Expect a push and pull between your executives and development teams. Executives often want to automate as much as possible, even to the extreme. Developers and sysadmins are going to approach automation more cautiously.

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

How well the automation tools are implemented determines how effectively you can enforce security practices and facilitate security sign-offs.

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

- Dispel the notion in your management and stakeholders that you'll be able to automate every task along with your toolchain. Engage with your stakeholders to learn their automation priorities and take that feedback into an automation strategy for your DevOps teams.
- Engage with your development teams — not just the team leads and managers — about how automation can help them perform their jobs. Listen to their concerns with empathy and answer their questions with definitive answers.

- Create an 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.
- Automate one build, quality assurance, or security check for one of your DevOps 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.
- Communicate the successes, lessons learned, and, yes, even the mistakes made on the pilot project to your stakeholders and internal DevOps community.

You can use your existing DevOps center of excellence or DevSecOps center of excellence as an opportunity to gather input from employees from across your organization about how automation affects their work. Otherwise, look for formal and informal channels in your development and operations organizations to gain the input. For example, informal lunch and learns, group chat channels, or team meetings can be ideal for gathering input depending on your corporate culture.

3 more phases of DevSecOps transformation

Making a major operations transition must be a long-term and well-planned process. Because DevSecOps is an important step in the DevOps journey for your organization, you are more likely to find success if you introduce and implement your transformation in phases.

In my previous article, I explained the first three phases of making this change. This article presents three additional phases of DevSecOps transformation you must work through to achieve your goals. Finishing these phases requires that you foster team collaboration to carry your organization through security changes, going live with DevSecOps, and putting the tools in place for continuous learning and iteration of your DevSecOps toolchain and processes.

Phase 4: collaborate on security changes to your DevOps toolchains

Some security changes on the move to DevSecOps may adversely affect operations and even security compliance. Changes to tools, processes, and even staffing sometimes change the way teams work.

Your development, operations, and security teams must collaborate before deployment and at other touchpoints to set priorities. Security teams sometimes prioritize a security measure that adversely impacts operations. Likewise, your developers probably overlook some holes caused by system configurations that could compromise the security and compliance of your systems.

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

Phase 5: execute on DevSecOps

As your organization crosses into phase 5 of your DevOps to DevSecOps transformation, it's time to execute your plans with one or more teams. Don't move to Phase 5 as an entire organization. Instead, look for natural breaks in your project teams' schedules for them to move to a DevSecOps model. For example, say that one of your DevOps teams has just launched a new product release. After catching their collective breath, they're working on bug fixes that come in from the field. Don't interrupt their flow with a full-on move to DevSecOps during an in-progress project.

Look for new project opportunities to begin executing on DevSecOps. Such an approach offers the following advantages:

- Providing teams a clean slate to learn a new process from the beginning, not midstream during a project
- Enabling you to include process and tools training as part of the project kickoff process
- Affording the chance to bring your developers, operations, and security teams together to discuss mutual expectations for the project
- Giving teams a chance to learn to work together better during the new workflows that DevSecOps brings to an organization

Phase 6: pursue continuous learning and iteration

There is no formal end to an adequately executed shift from DevOps to DevSecOps. After your organization moves to DevSecOps and adopts the principles and foundations, the learning and iteration need to continue past the transformation.

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

Final thoughts

The phases I outline in this series are general guidelines for a path toward achieving your DevSecOps transformation. The emphasis on collaboration is deliberate because your enterprise's particular circumstances could require that you modify these phases to achieve your transformation. Even if you need to make substantial changes to these phases, having a graduated implementation roadmap will get you much closer to success.