How DevSecOps culture can help agencies secure their software supply chain

Federal agencies everywhere are turning their attention to their software supply chains, trying to avoid getting caught up in the next big SolarWinds-esque incident. Best practices in the field are beginning to coalesce, but one of the easiest ways agencies can get a head start is by adopting another tried and true best practice: DevSecOps.

Continuous monitoring is a mainstay of a development, security and operations approach. Every time a developer makes a change in their code, it should run through an entire automated pipeline that scans not just the code, but also containers and artifacts produced in the build process. Not only does this save time, make the process repeatable and reduce the likelihood of human error, but it's also a key component in developing a trusted software supply chain, said Michael Radecker, senior specialist solutions architect at Red Hat.

> **With DevSecOps, what we really want is everyone to understand that security is part of their job. We want that to be part of their mindset. You have to really help them with buy-in and show that security from the beginning and shifting left.**
>
> — Michael Radecker,
> Senior Specialist Solutions Architect, Red Hat

Applying the DevSecOps methodology

"DevSecOps is more of a culture of integrating security in every phase of the software development lifecycle and collaborating with all teams involved. The trusted software supply chain, or a trusted or secure continuous integration, continuous delivery pipeline, these are more of the implementations which ensure security and trustworthiness of software components," Radecker said. "Security is everyone's job, not just the security team's. The security team is really there to put together artifacts, make sure that we're in compliance and maybe hold people accountable for security. The actual implementation of security, every step is everybody's job. And that's really what DevSecOps should accomplish."

In a trusted software supply chain, that starts with sources, repositories and images that are trusted and verified through vendors. For example, when dealing with containers, developers want to start with signed base images that have been scanned, and vulnerabilities have been remediated. Then as developers progress through the pipeline, they can sign new images with public or private keys to verify their status.

Similarly, a software bill of materials will list all the external dependencies, libraries and code used in the development of an application. But that's not immune to tampering; anyone can verify and sign off on those, Radecker said. But supply chain levels for software artifacts (SLSA) can help provide provenance to that SBOM.

SLSA lets developers follow that code all the way to the application deployment and view attestations, which is metadata that authenticates a software artifact or a collection of software artifacts. All of these elements allow developers to shift security left, apply continuous monitoring throughout the pipeline and enhance the supply chain security in an application.

DevSecOps is about culture, not just tools

But Radecker said DevSecOps goes beyond tools like SBOMs, SLSA and continuous monitoring. At its core, it's about breaking down the traditional silos between developers, operations and security teams.

"Having a culture of collaboration and integration is what I would say is probably most important to start off with DevSecOps, and then start talking about what kind of tools we're going to implement and how we're going to automate the software supply chain," he said. "With DevSecOps, what we really want is everyone to understand that security is part of their job. We want that to be part of their mindset. You have to really help them with buy-in and show that security from the beginning and shifting left."

One of the best ways to get that buy in, Radecker said, is to show the time savings each team can achieve. Integrating and automating security from the beginning means no more having to go back into a deployed application to figure out which piece of code or dependency broke when the security was applied and then remediating it.

"When we show them that we can save them time, we can save them headaches — and of course, reduce costs — then I think you can get the buy-in from pretty much everybody on the team," Radecker said.

Keep in mind that the goal of security is to minimize risk to an acceptable level. There is no such thing as 100% secure. You can't achieve that. ... But if we can minimize that to an acceptable level and you're using the tools that help you get to that point, then you're good. You've come up with a good software supply chain.

— Red Hat's Michael Radecker

No single path to DevSecOps

Because every team is different, each will take a slightly different path when implementing DevSecOps. Different tools will make better sense for different teams depending on what their goals and processes are. But in the end, every DevSecOps team is looking to accomplish the same four basic tasks: code, build, deploy and monitor. That's the application lifecycle.

From there, it's just a question of what makes the most sense for the organization. "There's no 100% right answer on that. The question is, what does your organization feel comfortable with? Keep in mind that the goal of security is to minimize risk to an acceptable level. There is no such thing as 100% secure. You can't achieve that. There's always going to be vulnerabilities in every step of your pipeline," Radecker said. "But if we can minimize that to an acceptable level and you're using the tools that help you get to that point, then you're good. You've come up with a good software supply chain."