CONTINUOUS PIPELINE SECURITY

HOW TO GET STARTED WITH DEVSECOPS IN YOUR CI/CD PIPELINE

Table of Contents

Introduction	2
Why Everyone Needs to Care About DevSecOps	3
Handling Open Source Software Vulnerabilities and Compliance with Software Composition Analysis (SCA)	4
What is Software Composition Analysis?	5
Software Composition Analysis is Only as Good as The Data Behind it	6
Key Challenges and Best Practices of DevSecOps and using Open Source Software	
-Developing Security Knowledge	7, 8
-Breaking Down the Silos	9
-Automating Security and Compliance	10
-Adopting of Security Processes	11
-Security Solutions for Containerized Environments	12
The Bottom Line	13
About JFrog	14

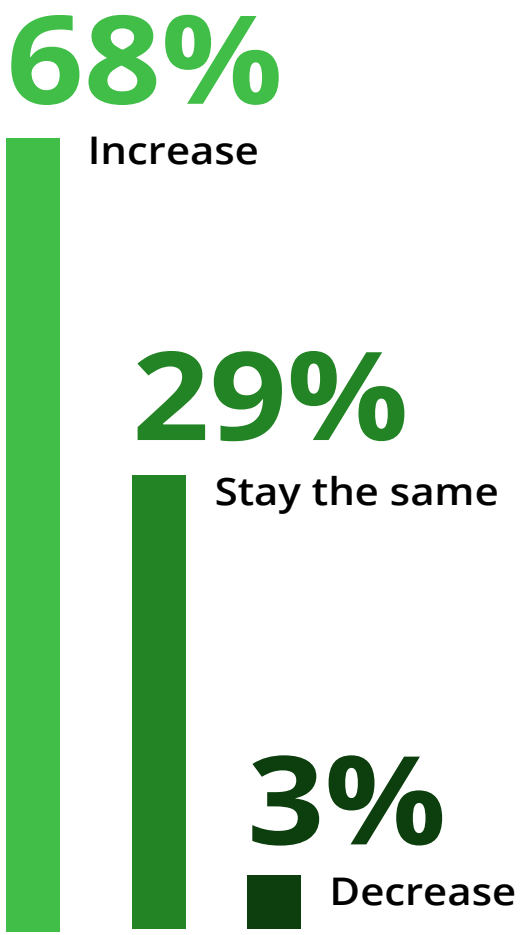
Introduction

If DevSecOps was easy, then we wouldn't even need to be talking about it, as we'd be protected from all vulnerabilities and we wouldn't even have any Cyber attacks. Implementing a true DevOps strategy is hard, but a successful DevSecOps one is even harder. We know this from bitter experience with breaches at companies like Equifax, Marriott, Facebook and Google - which highlight the importance of discovering software vulnerabilities early.

DevSecOps is the philosophy of integrating security practices within the DevOps process. With a methodical approach, organizations can create and deliver a secure software pipeline, ensuring they mitigate any known vulnerabilities early on in their software development lifecycle. One of the biggest mistakes companies who haven't adopted a holistic approach to DevSecOps make is to treat security as an afterthought. Security isn't an isolated matter, and identifying vulnerabilities is inseparable from your software development lifecycle.

The continous growth in use of open source software by enterprises, exposes code bases to potential vulnerabilities and license compliance violations hidden in open source components. The question is... How do we continuously secure our software development and delivery ecosystem to mitigate these risks, particularly as the frequency and intensity of the attacks are increasing?

Eliminating vulnerabilities and ensuring license compliance has to be tightly integrated into your CI/CD pipeline to respond to them as early as possible. Integrated and continuous pipeline security is possible, but it needs the cooperation of the IT, development, security and operations teams.



Change in use of enterprise open source over the past 12 months

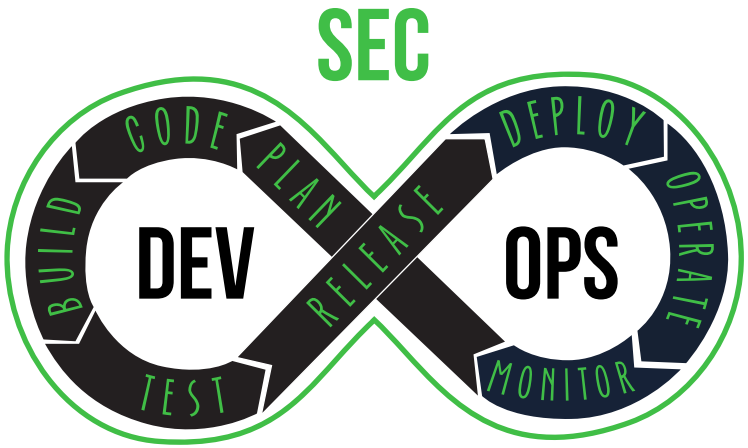
Source: Red Hat - The State Of Enterprise Open Source

Why Everyone Needs to Care About DevSecOps

The pace of innovation has driven up the use of open source software to create and build applications faster, rather than coding everything from scratch. Because of the relentless drive to innovate, developers are utilizing more and more open source software than ever before. Open source code has its advantages, the obvious one being, speeding up software development. But it also has downsides too. The code and its exploits become public information, allowing hackers to access this information, and devise clever and targeted attacks.

The need to secure your software earlier in the SDLC [shifting left], is now a must-do for any company. One of the major reasons for needing to adopt a DevSecOps approach is the fact that developers outnumber security professionals by 100:1 and you need to distribute the security knowledge in order to close the vulnerabilities gap faster.

DevSecOps is no longer a wish list item for a CIO, it is now a must-do IT strategy which needs to be an integral part of any software development lifecycle. In this ebook we will touch on DevSecOps as a whole, **but will focus mainly on how best to reduce the risk of using open source software (OSS) in your production code with Software Composition Analysis (SCA).**



DEV : OPS : SEC

100 : 10 : 1

Ratio of Developers to Operations and Security Engineers

Handling Open Source Software Vulnerabilities and Compliance with Software Composition Analysis

Modern applications are now comprised of up to 90% OSS components. This means the majority of the software we are building, deploying and consuming on a daily basis, is more likely to contain vulnerabilities than ever before. Because it's open it means that hackers have easy access to see the code and can look for vulnerabilities. Not only does open source software pose a risk through vulnerabilities, but it can also introduce complex licensing compliance issues for organizations. A potential complication could be that a license says "if you use this component, you need to make your code open source."

Ideally you want to scan and Identify license compliance and vulnerability issues on all of your OSS components as early in the development process as possible. Knowing what components you have across your entire application portfolio and keeping track of them is an absolute must and should ultimately be automated. This should be an integral part of your CI/CD pipeline, to keep your development and release velocity on track.

Historically securing across your SDLC and into production required running agents to do component scanning. Today there are different solutions that can achieve a greater level of security and compliance monitoring, that are integrated directly into your IDE, Repository Manager, CI/CD pipeline and can even scan your container images. For open source security and compliance monitoring, having a natively integrated SCA solution would work best.

Source: McAfee and the Center for Strategic and International Studies, 2018

What is Software Composition Analysis?

Using Static Code Analysis tools can help you in identifying vulnerabilities in your own proprietary developed code. However, it will not identify vulnerabilities in the components that you depend on. The need for Software Composition Analysis grew out of the increasing use of open source components, used by developers to keep up with the pace of innovation. Companies were struggling to manage and keep track of open source usage across teams and sites and needed a more automated way.

SCA encompasses managing and monitoring license compliance and security vulnerabilities in open source components. Knowing what OSS is being used and what their dependencies are is a primary concern. After identification of the open source components, the SCA tools will provide visibility of each of the components, providing information on the license, it's version number and whether there's any known security vulnerabilities associated with it.

Advanced SCA tools offer policy enforcement capabilities, enabling automated monitoring of open source components. These are configurable to enable different behaviours on identified Security or compliance violations, based on the context of what is being scanned. An example would be to fail a build of a highly sensitive application based on a vulnerability, while not failing the build of a test application with the same vulnerable component. The tool compares every open source component in your code against your policies, and will trigger different types of automated actions depending on the result.

It's important to consider the following when looking at software composition analysis tools:

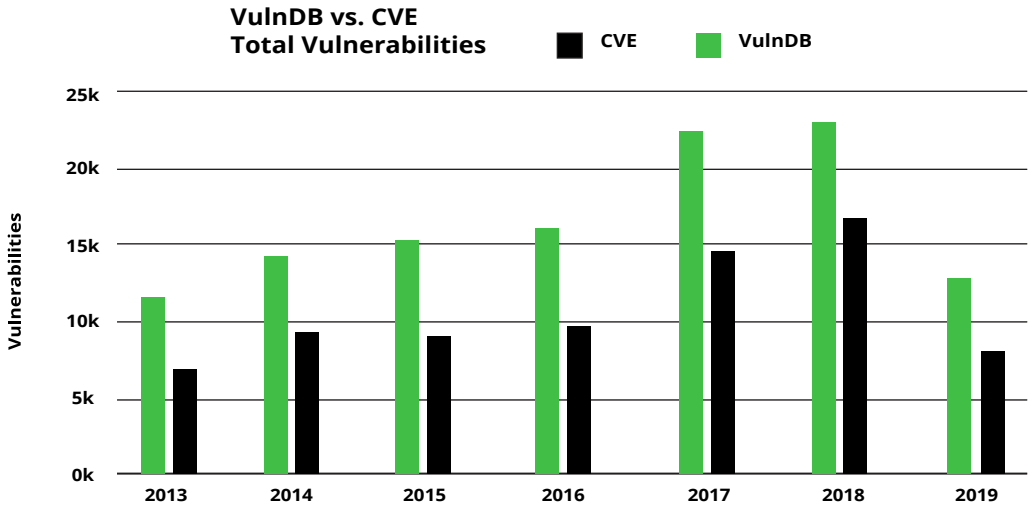
1. **Technology Support:** How universal is it and does it support all coding languages and packages used by the company
2. **Ecosystem Integration:** The SCA tool must integrate with repositories, IDEs, package managers, CI servers and more via out-of-the-box integrations as well as via rich open APIs
3. **Database:** A comprehensive license and vulnerability intelligence is a must have to provide you the best coverage

Software Composition Analysis is Only as Good as the Data Behind It

This all sounds great you may think… all you need to do is buy some great tools, integrate them into my tools ecosystem and voila, you're doing SCA like a master. While that may be true to a certain extent, your success at achieving the goal of truly mitigating vulnerabilities and license issues may well be limited. Here's why.

You can have the best integration of SCA tools ever, but a security scanning solution is only as good as the database of vulnerabilities that drives it. Using a database that isn't up-to-date with the latest vulnerabilities is like trying to find someone in a crowd without knowing what they look like. Some companies see the use of MITRE's Common Vulnerability Enumeration (CVE) in the National Vulnerability Database (NVD) as the gold standard to secure against. Many security experts are adamant that relying on CVE and NVD for vulnerability data is not sufficient anymore.

For example, Risk Based Security, who are known for having one of the most timely and comprehensive vulnerability intelligence services available, have consistently eclipsed the total number of vulnerabilities identified and cataloged by the CVE and NVD each year. To date they have Identified and cataloged over 75,000 vulnerabilities not found in the CVE/NVD databases (as of late 2019).



"Risk Based Security's VulnDB® team aggregated 11,092 newly-disclosed vulnerabilities during the first half of 2019. Risk Based Security's VulnDB published 4,332 more vulnerabilities than CVE/NVD in the first half of 2019."

Source: Risk Based Security - 2019 MidYear QuickView Data Breach Report - Issued August, 2019

5 Key Challenges and Best Practices with DevSecOps and Open Source Software

1. Developing Security Knowledge

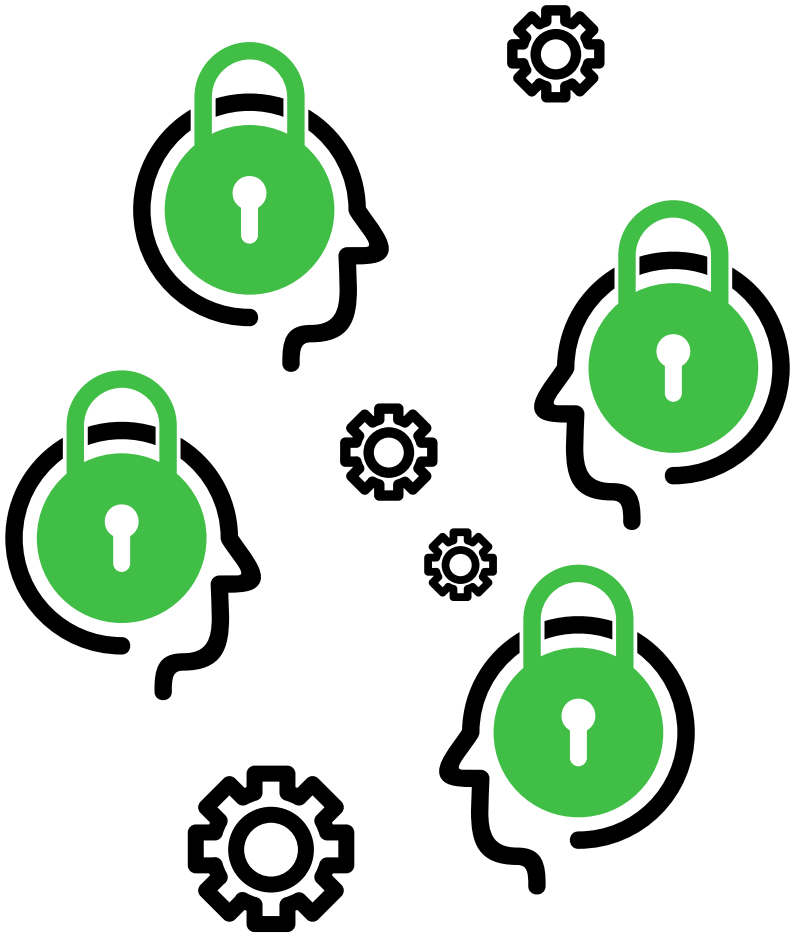
In the crazy world of continuous integration and continuous delivery, developers are mostly concerned with speed of execution and how quickly they can get the application coded. This means they're not thinking about how 'secure' their code is or whether the open source software components they're including, have any vulnerabilities or whether they align with their companies license compliance policies. Developers and operations engineers can no longer ignore application security. If you don't have correctly trained or knowledgable developers, it's more likely they will write vulnerable code and include unsafe OSS components into their applications. This is a recipe for disaster - especially once you're in production! Mitigating friction in your process can also be a challenge as you look to increase your adoption of security, while maintaining release velocity.

As you start integrating security into your software development lifecycle, people can be the differentiator between success and failure. For individuals and teams to truly adopt new security processes, concepts and technology, they need to understand why. Knowing why they're being asked to do something new, helps them be comfortable to adopt those new ideas. A primary focus should be to have a well prepared and trained workforce. That means having developers and operations staff trained in security best practices, which could require the hiring of an experienced DevSecOps or Security Engineer.

Education starts with the teams writing the software. There is immense value in training developers to follow better coding practices and knowing how to write secure code, and identify and fix vulnerable code. The more developers know about the risks of poor coding practices the better, which builds a solid coding foundation of your DevSecOps process. It will also be imperative to train developers and operations teams on any new technology or solutions that they will be expected to use.

The goal of DevSecOps is to incorporate security into all stages of the software development workflow, and has to start with an “I’m responsible for security” mindset for developers, operations and security teams alike. This may mean certain team members will have new job functions and responsibilities, which can be challenging. You may need to hire new staff with specific expertise to help with security threat analysis and help to setup security best practices. Security education is in fact a never ending process and should be considered a constant effort.

Software composition analysis enables your developers to have visibility and insights into all of the OSS components and their dependencies they are planning to use. It provides license and dependency information and will show whether there are any vulnerabilities or license violations. This can be done as early as during development in the IDE, in your repository manager and can span across your build tools and deployments. This gives the developer the knowledge to find violations sooner and mitigate expensive security or compliance costs later on.

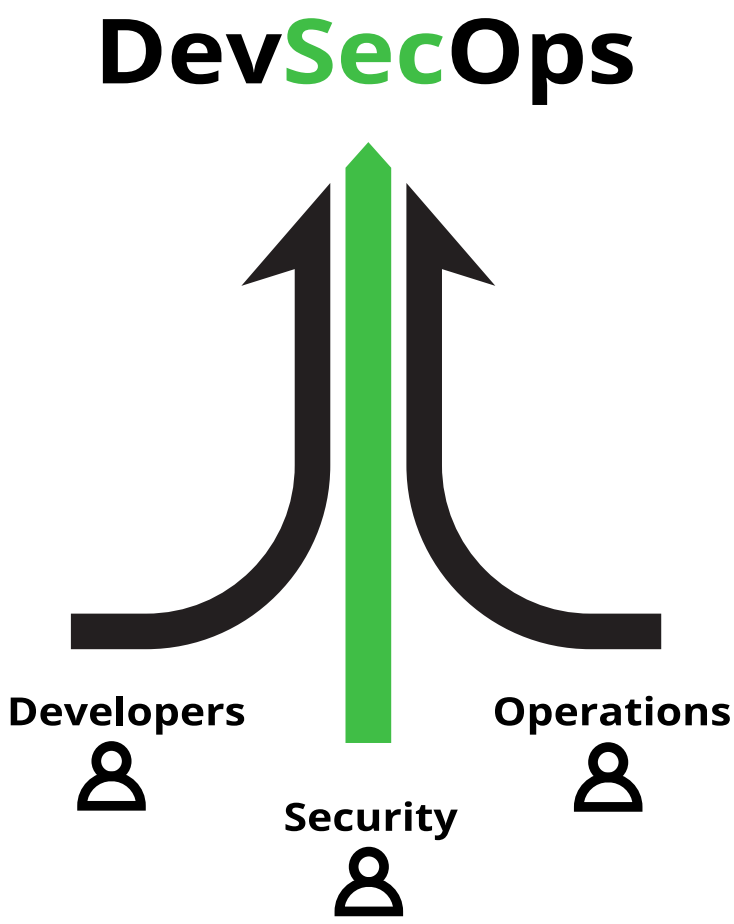


2. Breaking Down the Silos

Successful DevSecOps requires a fundamental shift from traditional approaches to product and software deployment. To make a DevSecOps process succeed requires not only changes in processes, organizations and education, it also needs teams that would normally operate in their own silos (IT, developers, operations and security) to now work together. They must evolve to be cooperative and supportive, challenging people, culture, processes and organization structures.There has to be consistent cooperation across departments, and teams need to work together to bridge the gaps and focus on a common security conscious goal. Each team will have to adopt new ways of thinking, working, and communicating to be united.

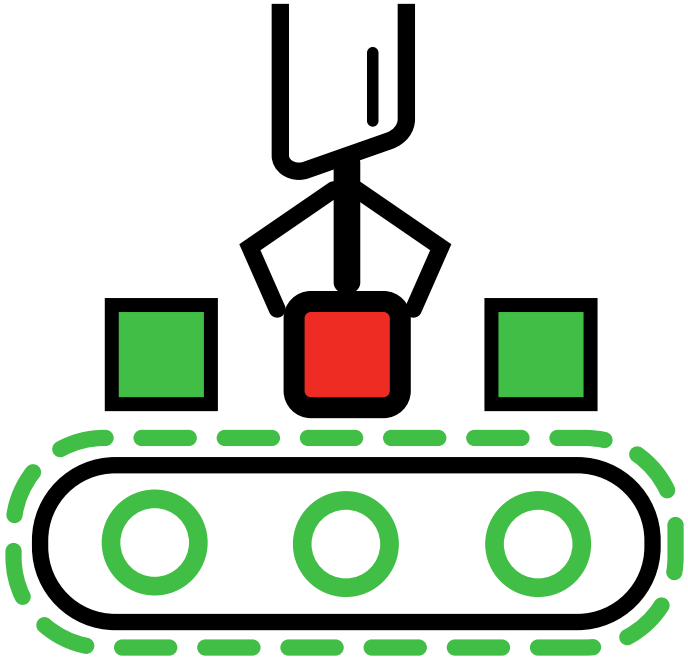
A DevSecOps process is more easily achieved if you already have an established DevOps process. The goal with DevSecOps is to introduce a “shift left” mentality across the different teams. This means a practice in which development and operations teams focus on quality, security and problem prevention (by testing for security and violations) earlier. A solution would be to include security team members into developer teams, so they can be involved in their education, see code earlier in the life cycle, and be present in code reviews and daily standups or scrums.

An comprehensive and integrated SCA tool can enable developers, operations and security teams to see all of the OSS metadata for all their artifacts, builds and container images. They all see and share the same data, helping them work together to find and mitigate vulnerabilities and non-compliant licenses early on. This bridges the gap between the siloed teams and they become unified by the responsibility and aim to release vulnerability free and compliant code.



3. Automating Security and Compliance

One challenge of delivering secure and compliant code, can be 'human errors' that enable vulnerabilities to creep into production releases. Manual security processes for the amount of code being produced today is simply impossible. Each developer would need a dedicated security engineer to review and identify security issues. In addition you will also need to look into existing code to identify new vulnerabilities.



Any security critical functions should therefore be automated and integrated into your existing tools ecosystem. This will help mitigate some of the 'human errors' that enable vulnerabilities to creep into production releases. This means that some tasks previously done manually will need to be automated with a specific tool and process. An example here would be 'security gating' which would be an automated process to fail builds or prevent promotion if certain security conditions aren't met.

If you aren't using a Software Composition Analysis tool today, you are probably struggling to manage and track the use of open source software across your teams. As you try to increase the velocity of your releases, developer use of open source software will steadily rise. It's also tough for your developers to know whether the OSS components they are using, have any vulnerabilities or license violations included in them, which can be problematic down the road. You may have an inventory of open source components used in your binary repository manager, but what you really need is comprehensive knowledge of the components, their dependencies and the associated vulnerabilities and license violations. Integrating SCA tools into your tool chain will help you automate the identification and mitigation of vulnerabilities and license violations.

"Organizations that had not deployed security automation experienced breach costs that were 95 percent higher than breaches at organizations with fully-deployed automation ($5.16 million average total cost of a breach without automation vs. $2.65 million for fully-deployed automation)."

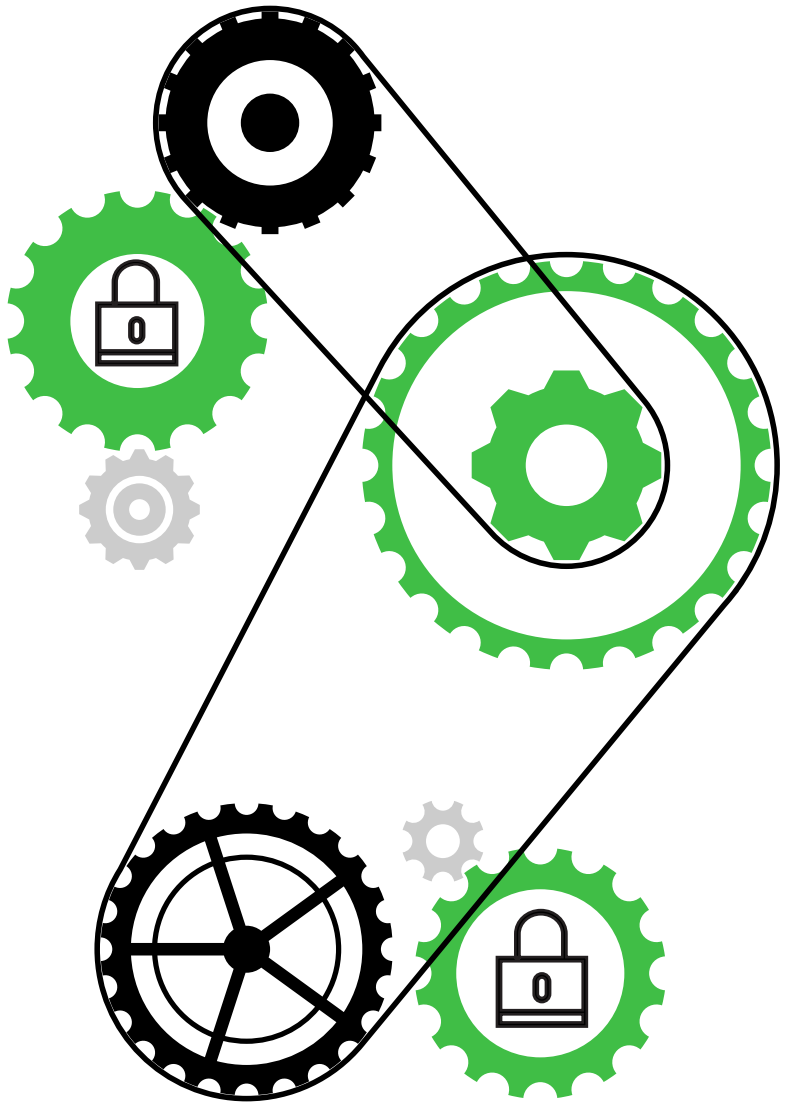
Source: Cost of a Data Breach Report 2019, IBM Security

4. Adoption of Security Processes

DevOps companies have an existing tool chain in which they do their day to day work. Asking them to adopt yet another system or tool to check if there are security issues with their code, would cause them more context switches and will require them to proactively look for such violations. The result is less productivity and potentially less adoption.

For developers to adopt new tools, the new pipeline needs to be as similar to their previous DevOps one as possible. There are lots of newer security tools, including many that integrate directly or 'plug-in' to existing ecosystems. Developers won't jump through hoops to scan software or components. With integrated security tools, code will be automatically scanned for them. The developer will simply need to decide how to handle the findings as they do with other types of bugs today.

Ideally you should choose a Software Composition Analysis tool that can integrate with your IDE, Repository Manager, and Build tools. It would be used to identify any vulnerabilities or license violations throughout the whole software development lifecycle. An important aspect to consider is which programming languages and package types your developers are using, and therefore these will need to be supported by the SCA solution.



5. Security Solutions for Containerized Environments

Containers are becoming a common practice for cloud based systems. It helps with providing scalable systems and increasing the speed that companies can deliver new services, and ultimately release software much faster than before. But as always with new technologies, they come with new security and compliance risks.

There are multiple security technologies available to mitigate these new risks such as SCA tools for the CI/CD, and container runtime tools for runtime protection.

Look for an SCA tool that can recursively scan through all the layers of a container and identify security vulnerabilities and license compliance issues. The tool should be able to identify violations in an existing container, as well as while you build one, and should also be able to fail a build or prevent the usage of a container. It's important to select a solution that integrates directly into your repository manager, build platform, and container infrastructure, to ensure adoption and directly help mitigate expensive software rework or worse an insecure production release.

Security is the top challenge for Kubernetes users as 46% of of surveyed respondents cited Security as their #1 challenge."

Source: The New Stack Analysis of Cloud Native Computing Foundation survey (Fall 2017)

The Bottom Line

Adopting a DevSecOps “Security Everywhere” mindset is imperative for any company today. The drive to innovate at speed will go hand-in-hand with an increase in open source software usage. This means the need to have a solid OSS security and license compliance process, will be even more pressing in the future. Why? Because of the increased usage of OSS in applications, makes it a huge and appealing target for cyber attacks and hackers.

So what do we take from this?

- Establish DevSecOps as a cornerstone of your software development lifecycle
- Instill security knowledge and ownership across your developer and operations teams
- Utilize security and compliance best practices and adopt continuous improvement tactics
- Use an integrated suite of DevSecOps tools that can automate security and governance
- Ensure your toolsuite includes a Universal Software Composition Analysis solution
- Utilize the most comprehensive and timely vulnerability intelligence database

DevSecOps isn’t an exact science, there’s more than one way to ensure your software deployments are secure and compliant. To be successful at DevSecOps and to safely use open source software, requires a strong foundation of knowledge, cooperation, automation and ownership. Without DevSecOps adoption, you’re exposing your company to a risk of being breached, which can be directly translated into hundreds of millions of dollars of potential compensation payments, lost customers, lost revenue and software re-work expense.

About JFrog

JFrog is on a mission to enable continuous updates through liquid software, empowering developers to deliver high-quality applications that securely flow to end-users with zero downtime. Our solutions meet your business model needs, and support on-prem, cloud, hybrid and multi-cloud configurations. More than 6,000 customers depend on JFrog to manage their binaries for their mission critical applications including more than 70% of the Fortune 100 – companies such as Amazon, Facebook, Google, Netflix, Uber, VMware, and Spotify trust JFrog.

To learn more about JFrog solutions, check out these webinars:

Introduction to JFrog Artifactory

DevSecOps with JFrog Xray - Universal Component & Impact Analysis

Hands-on Lab with JFrog Xray

Try it for yourself - JFrog Xray FREE Trial

References:
http://info.nowsecure.com/rs/201-XEW-873/images/NowSecure-WhiteHat-2018-Application-Security-Report.pdf
Logz.io, “The 2018 DevOps pulse,” 2018
https://www2.deloitte.com/content/dam/Deloitte/uk/Documents/technology/deloitte-uk-tech-trends-2019-chapter7-devsecops.pdf
https://techbeacon.com/security/6-devsecops-best-practices-automate-early-often