Rezilion

EBOOK

Getting to DevSecOps Zen

Security is an essential element of the modern development environment. Learn how to bring security and developers together in a happy, harmonious way.

INTRODUCTION

1. Secure it—Ship It

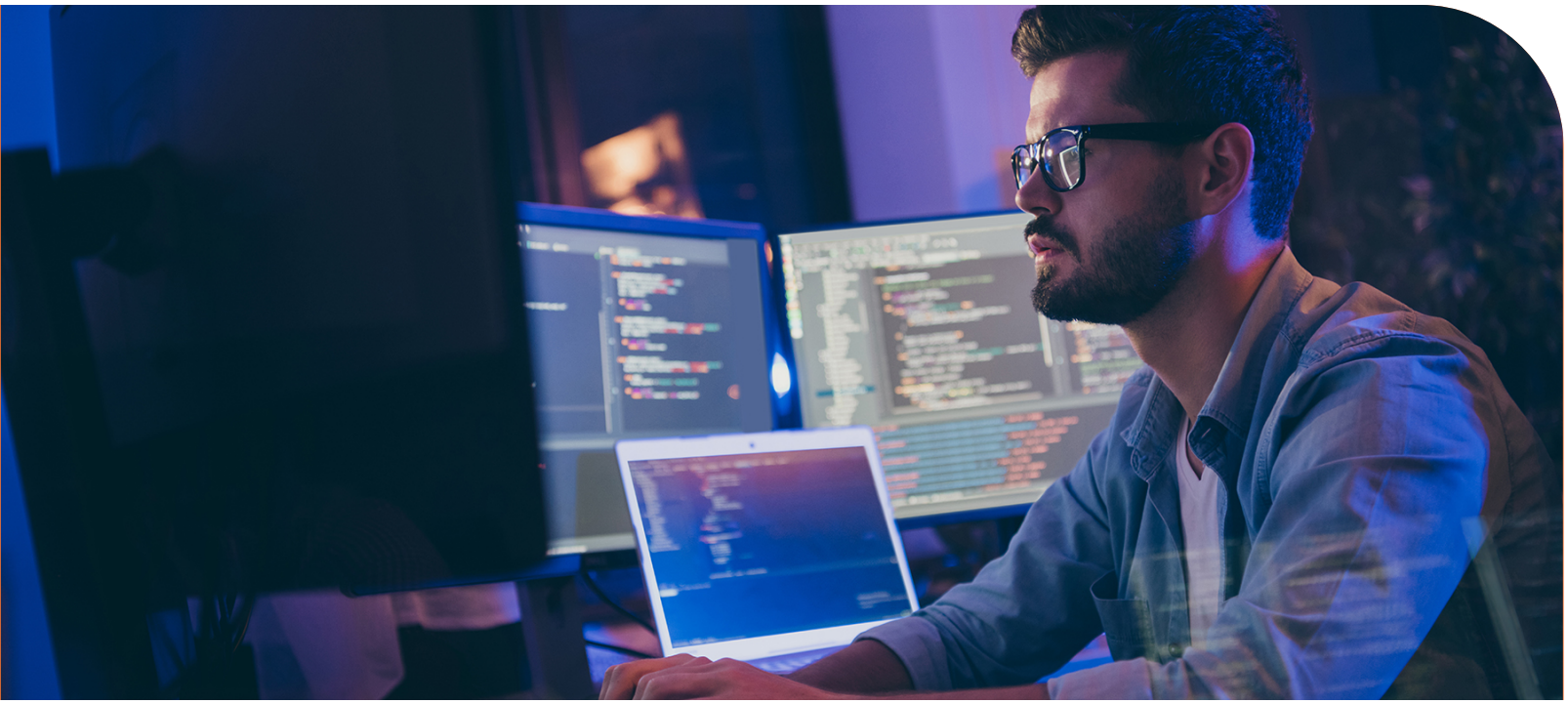
2. Shift Left and Build Security in Sooner

3. Workflow Integration: Dev + Sec = DevSecOps

4. Fight Friction Between Dev and Sec

5. DevSecOps: The Key to Secure Software

Conclusion Rezilion and GitLab Resolves DevSecOps Tension



INTRODUCTION

1. Secure it—Ship It

2. Shift Left and Build Security in Sooner

3. Workflow Integration: Dev + Sec = DevSecOps

4. Fight Friction Between Dev and Sec

5. DevSecOps: The Key to Secure Software

Rezilion and GitLab Resolves DevSecOps Tension

THE DAYS OF SECURITY SEEN AS THE BANE OF SOFTWARE DEVELOPERS ARE OVER—or at least they should be. Building strong security into products cannot be an afterthought at a time when vulnerabilities are leading to wide scale cyber attacks.

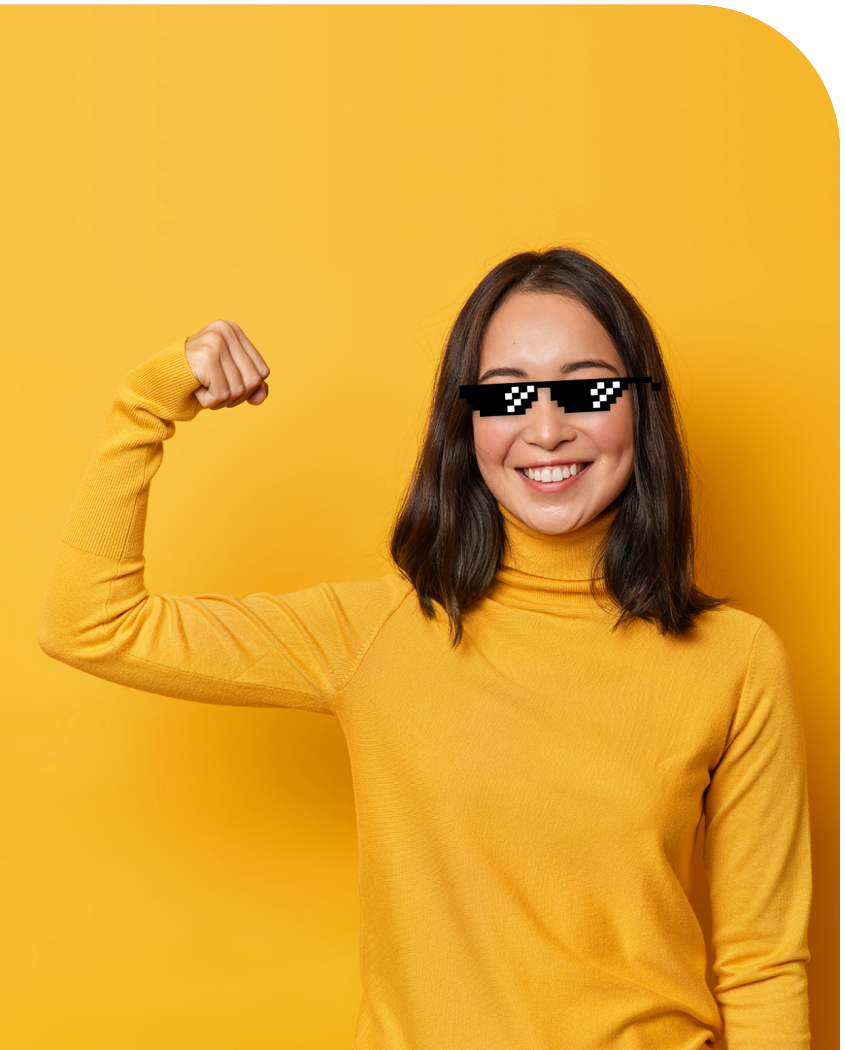
The fact is, security can and should be a competitive advantage for companies producing software. Recent, high-profile breaches and ransomware attacks have highlighted the importance of securing every aspect of the organization—including the software that runs so many critical business processes.

The mature, modern development environment must put security at the top of the list of priorities, along with quality. Yes, speed to market is still important. But not at the expense of delivering flawed software that can result in huge losses down the road, both for the software company and its customers.

This ebook examines essential steps organizations must embrace to create a modern development environment that makes security an integral part of the development lifecycle—without adding more work for developers.

A GOOD WORKING MOTTO FOR THE MODERN DEVELOPMENT ENVIRONMENT SHOULD BE "SECURE IT—SHIP IT". This takes into account the main goals of both the security and development teams.

On the development side, everyone is focused on building and shipping software products at maximum speed. Ship first, ask questions later is the line of thinking. As long as software gets into the hands of users as soon as possible, everything is good.



That's not at all how the security team thinks, however. Security professionals are all about making sure code is as free of vulnerabilities as it possibly can be before products ever reach internal or external users.

What development organizations need to do is figure out how to meld the two priorities into the singular goal of "secure it—ship it". The good news is this need not involve large costs in terms of infrastructure or personnel investment.

What it does require is a new way of thinking and acting among the leaders and members of the development and security teams. It also requires a willingness to work together harmoniously toward the common goal of delivering secure software as quickly as possible.

While it might not be a universal trait among developers, it's safe to say that for many, speed and quality outweigh security when it comes to creating software. Many development teams are probably fine with the idea of moving code along even though it's not perfect from a security standpoint.

This is understandable, because these teams are under lots of pressure to build and ship products quickly in what is becoming an increasingly competitive digital market. Development delays don't go over well with customers, and operating a slow and methodical development process doesn't work in today's hyper-competitive environment.

Security teams, on the other hand, consider ensuring the security of software to be the highest priority. Their thinking is that vulnerabilities can easily slip through and later be exploited by cybercriminals to launch attacks. Based on their worldview, no product should ever be delivered to users until it's declared to be as secure as possible.

By implementing tools that enable security to be built into software products efficiently and automatically at the earliest stages of development, organizations can have the best of both worlds—from both a security and development standpoint.

INTRODUCTION

1. Secure it—Ship It

2. Shift Left and Build Security in Sooner

3. Workflow Integration: Dev + Sec = DevSecOps

4. Fight Friction Between Dev and Sec

5. DevSecOps: The Key to Secure Software

Rezilion and GitLab Resolves DevSecOps Tension

INTRODUCTION

1. Secure it—Ship It

2. **Shift Left and Build Security in Sooner**

3. Workflow Integration: Dev + Sec = DevSecOps

4. Fight Friction Between Dev and Sec

5. DevSecOps: The Key to Secure Software

Rezilion and GitLab Resolves DevSecOps Tension

THE TERM "SHIFT LEFT" REFERS TO THE CONCEPT OF TAKING TASKS typically completed at later stages of a process and performing them in earlier stages. It's often applied to the testing of software code, but can also apply to security.

The further to the left of the Software Development Life Cycle (SDLC) security that is addressed, the better likelihood of shipping out secure products. By deploying and testing security controls early in the cycle, development teams can not only increase the security of finished products, but avoid additional work for developers at later phases.

Some developers view security as a burden, so the relationship between development and software teams can be contentious. The shift left approach can provide a solution that makes everyone happy because it doesn't increase the workload for developers and enhances the security of software products.

The problem is, developers might be dubious about shifting left for security because they will assume it means more work for them.

But there is a way security teams can deliver on shifting left and not add work for developers. They can do this by eliminating all the vulnerabilities that do not pose a risk to users. Of all the possible security risks that might exist with any given piece of code, most are likely not exploitable by bad actors and therefore not a concern from a security standpoint.

With technology that automatically filters out the non-threatening vulnerabilities, teams can focus on addressing exploitable vulnerabilities and avoid patching false-positives that aren't loaded into memory and therefore pose no risk.

In addition, they can get automated recommendations for the most efficient ways to fix the high-priority vulnerabilities based on aggregated and validated data, speeding up the process. This enables teams to make more informed decisions and take action more quickly.

The shift left approach for security is central to the DevSecOps model, which itself comes from the DevOps methodology.

INTRODUCTION

1. Secure it—Ship It

2. Shift Left and Build Security in Sooner

3. **Workflow Integration: Dev + Sec = DevSecOps**

4. Fight Friction Between Dev and Sec

5. DevSecOps: The Key to Secure Software

Rezilion and GitLab Resolves DevSecOps Tension

ONE OF THE COMPONENTS OF THE MODERN DEVELOPMENT ENVIRONMENT that makes the approaches of secure it—ship it and shift left possible is workflow integration.

Automated workflow is a key component of software development. It's what enables teams to complete tasks more quickly, increase efficiency, and deliver superior products. Automated workflow can also lead to cost savings from increased productivity; and greater transparency of development processes.

Within the context of security, the major benefit of workflow is when it's integrated with security controls. In a DevOps development environment, the level of security in software products is determined by how well security controls are part of the automated workflow.

For instance, consider a developer working on code for a new software product. If the code needs to be tested for a security issue, the developer typically has to stop working with one user interface and move to another one to do the testing. Then the developer has to go back to the initial environment to continue work. That creates additional work for the developer as well as increased complexity.

If the developer was using a security tool that integrates security into the development workflow, there would be no need to shift, which eases the process and eliminates complexity. And if teams can integrate security directly into the development workflow, it's less likely they will miss steps that could lead to vulnerabilities getting through.

With workflow integration security becomes an automated function, but it's necessary that cybersecurity tools be capable of such integration. Even though security is such an important component of the development process, it must not continually interrupt the development process. Any security tools that change DevOps workflows and processes are not the right ones to be using.

Security tools and processes should be considered in the same light as quality assurance testing tools. They are just part of the effort to create new software, rather than process inhibitors.

IT'S NOT SURPRISING THAT THE SECURITY AND DEVELOPMENT teams within an organization can end up being adversaries. The security professionals emphasize the need to create software that is secure—whatever it takes to achieve that goal.

Developers, on the other hand, are laser focused on getting products to market as quickly as possible. If they're secure, fine. If not, that's the price to pay for the quick release of applications. Security teams expect developers to take on more of the burden of security. Developers can't understand why they're being asked to do additional work.

The result of these opposing viewpoints is friction—and there might be lots of it within development operations. Friction can lead the two factions to reduce communicating with each other, or stop communicating at all.

If the security and development teams are not working together harmoniously, that can result in software reaching the market with potentially dangerous vulnerabilities. Friction can also contribute to delays in product delivery, in some cases because products need to be revised constantly.

Research firm Ponemon Institute noted in 2020 that organizations are at risk when security and development teams do not have a common vision for delivering software in a secure manner. There needs to be a fundamental agreement that security should be integrated throughout the application development process, the firm said.

Friction does not have to be an accepted part of the security/development relationship. By deploying tools that automate the identity and remediation of vulnerabilities, teams can enhance product security without increasing the workload for developers.

Because security controls can be implemented from the beginning of the development lifecycle in a way that doesn't burden the development team, both sides can achieve their goals without alienating the other. If this is done effectively, the security of software goes up, and friction goes down—or even better disappears entirely.

INTRODUCTION

1. Secure it—Ship It

2. Shift Left and Build Security in Sooner

3. Workflow Integration: Dev + Sec = DevSecOps

4. **Fight Friction Between Dev and Sec**

5. DevSecOps: The Key to Secure Software

Rezilion and GitLab Resolves DevSecOps Tension

Friction can contribute to DELAYS in product delivery, in some cases because products need to be REVISED constantly.

ALL OF THESE ATTRIBUTES—SECURE IT—SHIP IT, SHIFTING SECURITY to the left, workflow integration, and reducing friction between the security and development teams—are vital components of DevSecOps.

Under the DevSecOps model, integrating security automatically is part of each phase of the software development process, including design, integration, testing, deployment, and delivery. This proactive approach to security anticipates and prepares for potential vulnerabilities and remediates them before they can become sources of attacks.

DevSecOps requires that teams review, audit, and test software code for security issues regularly, and address issues as soon as they arise.

Deploying the framework can lead to several benefits. For one, development teams can avoid costly and time-consuming code revisions late in the lifecycle, because they will have already addressed them earlier.

In addition, DevSecOps introduces repeatable processes that ensure cybersecurity controls are being applied consistently. Teams can help prevent security flaws that can result in costly breaches.

Another benefit of DevSecOps is that it promotes the concept that cybersecurity is everyone's responsibility. It becomes a part of the development process from the beginning, and development, operations, and security teams work together to deliver secure products.

Whereas the emergence of DevOps has helped transform the software development process by emphasizing shorter development lifecycles and continuous delivery of high quality software. DevSecOps ensures that security is a key part of the development effort.

In the process, DevSecOps supports the goals of secure it—ship it, shift left, workflow integration, and eliminating friction. All of this leads to a modern development environment that makes delivering secure products a priority.

INTRODUCTION

1. Secure it—Ship It

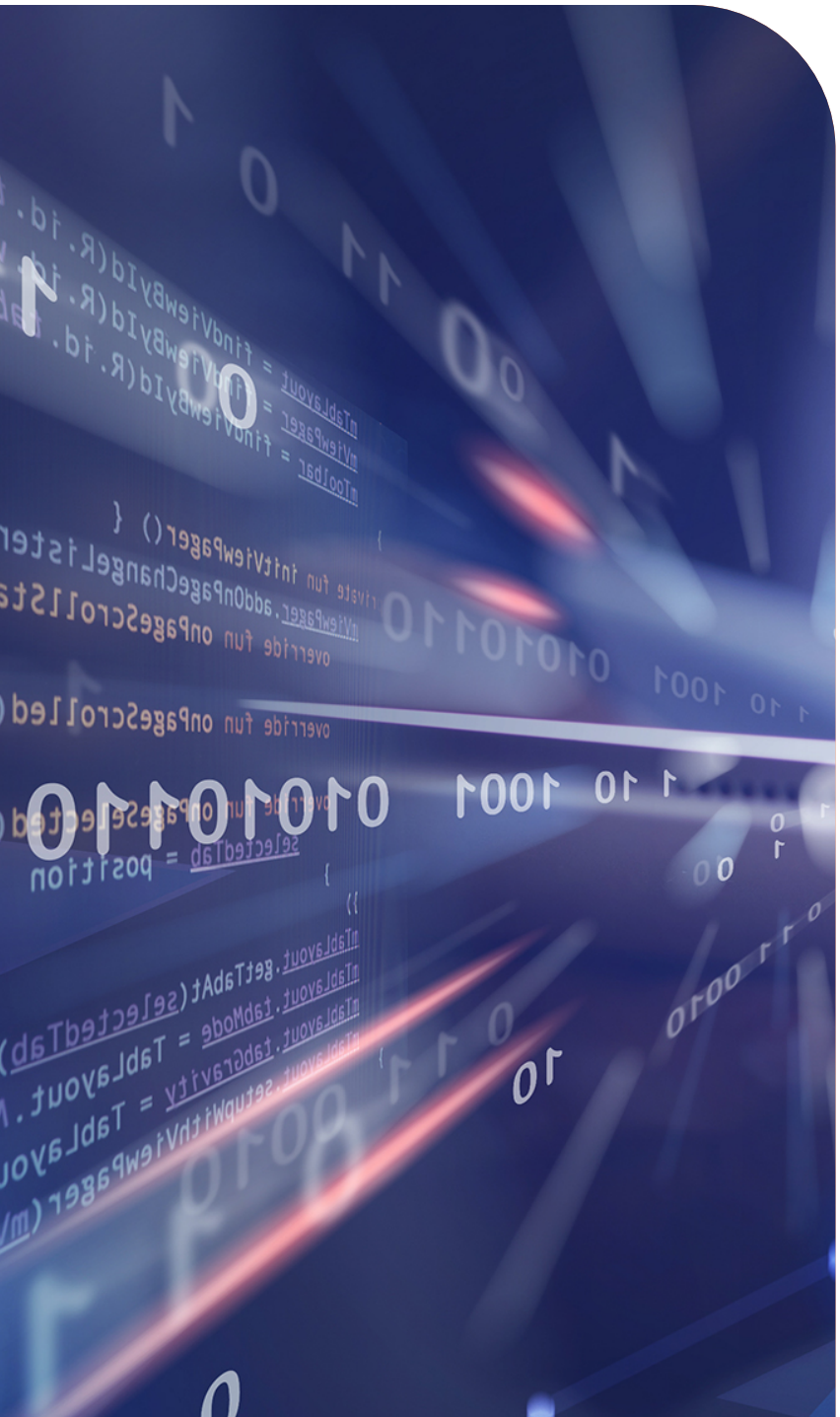
2. Shift Left and Build Security in Sooner

3. Workflow Integration: Dev + Sec = DevSecOps

4. Fight Friction Between Dev and Sec

5. **DevSecOps: The Key to Secure Software**

Rezilion and GitLab Resolves DevSecOps Tension



REZILION RECENTLY ANNOUNCED AN INTEGRATION WITH DEVOPS PLATFORM GITLAB that enhances developers' ability to release secure software products more quickly. Rezilion's native integration with GitLab CI, deployable within minutes, eliminates an organization's vulnerability backlog by as much as 85% and reduces remediation from months to days while addressing 100% of exploitable risk.

By using Rezilion in GitLab CI, organizations can identify which vulnerabilities are loaded to memory and executed in runtime. Having this capability means they can focus on the actual risks rather than spending time addressing vulnerabilities that are not exploitable.

They can reduce the time needed to address vulnerabilities in the SDLC. As they test and scan code for vulnerabilities, developers can see within the Gitlab user interface which vulnerabilities require attention. Non-exploitable vulnerabilities are marked as "false positives" that shouldn't hold back releases.

Teams can get a clear view of all the software components in use with Rezilion's dynamic Software Bill of Materials (SBOM), to understand which components are vulnerable in the specific runtime context of their environment.

The GitLab integration extends the ability of DevSecOps and DevOps teams to have a continuous view of the actual attack surface and allows them to prioritize remediation efforts on the vulnerabilities and weaknesses that present the most risk. By being able to make security a part of software development from the very beginning, organizations can support the new objective of secure it—ship it.

Visit Learn more about how our partnership is transforming DevSecOps and start your free 30-day trial today by visiting https://www.rezilion.com/sign-up-for-30day-free-trial/.

INTRODUCTION

1. Secure it—Ship It

2. Shift Left and Build Security in Sooner

3. Workflow Integration: Dev + Sec = DevSecOps

4. Fight Friction Between Dev and Sec

5. DevSecOps: The Key to Secure Software

Rezilion and GitLab Resolves DevSecOps Tension