PULSEMETER

Techstrong Research

Tackling CI/CD Security Anti-Patterns

SPONSORED BY

Red Hat

SOFTWARE SUPPLY CHAIN SECURITY has garnered increasing attention as developers, their code and the underlying technologies supporting their work are targeted by cyberattackers. As such, organizations must consider the security of DevOps and CI/CD workflow pipelines and toolchains much more intently. The risk associated with software supply chain security is arguably equal to, if not greater than, the security of a manufacturing line, a shipping distribution hub or the chain of custody of chips used in product assemblies.

CI/CD pipeline security has become a critical focus for organizations looking to increase the safety of their software delivery processes. With more frequent code updates and reliance on external software sources, including open source software, hardening CI/CD environments is now mission-critical. Pipeline security encompasses many areas, including poorly managed and secured source code repositories, poor secrets hygiene and CI/CD build environments that can be poisoned.

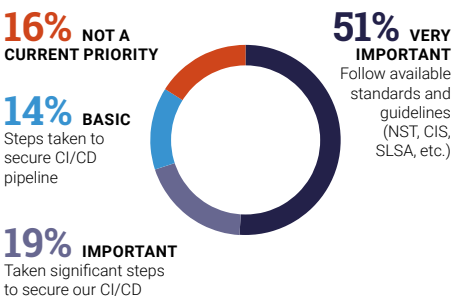
> ***Proper security designs, processes and controls must be implemented at each pipeline stage to prevent compromises.***

Ultimately, the software created through a software development life cycle (SDLC) is only as secure as the software supply chain that produced it.

In examining the security of our CI/CD implementations, we see a number of common security lapses–known as anti-

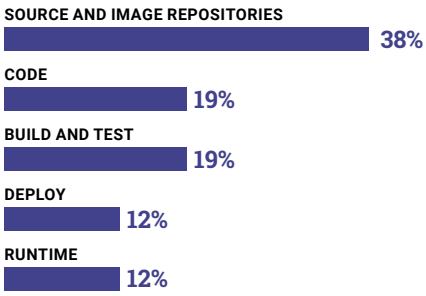
How important is the security of your CI/CD pipeline?

CI/CD pipeline security is a priority. A full 84.1% of respondents take security seriously and have taken steps toward securing the CI/CD pipeline.



What stage of the CI/CD pipeline do you believe is most vulnerable?

It's clear that particular stages of the CI/CD pipeline are more vulnerable than others. While improvements to security can be made at every stage of the CI/CD pipeline, ensuring source and image repository security is top of mind for many developers (38%).



patterns. CI/CD anti-patterns are security practices that open our development pipelines to attack, often broadening attack vectors malicious actors can use to compromise our software supply chain security. A few examples of CI/CD anti-patterns include introducing external code from non-vetted sources, including repositories, base images, package managers and a lack of moving beyond common sense security practices to those which add isolation, break apart overly complex build environments which assist to thwart pipeline poisoning.

To better understand practitioners' views on CI/CD pipeline security, Techstrong Research surveyed DevOps, cloud native and cybersecurity professionals in 2023 to assess the state of CI/CD pipeline security. The results make clear that pipeline security is now a top priority. A full 84% of respondents indicated they take it seriously, with securing source code repositories, managing secrets, and isolating build environments rising to the top as focus areas.

Unfortunately, this widespread prioritization comes in response to concerning incidents–in other words, it's not proactive. Over 20% of organizations suffered a pipeline security incident in the past year alone, leading to tangible impacts like delayed deployments, data loss and employee termination.

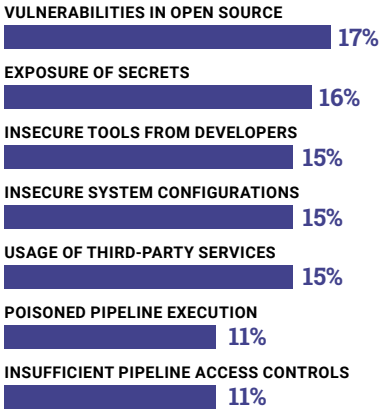
When examining the most vulnerable stages of the SDLC, source/image repositories and code were cited as higher-risk areas by a combined 57% of respondents. Meanwhile, top security concerns centered on open source vulnerabilities, exposure of secrets, insecure code and lack of pipeline controls. Companies are finally taking proactive steps to address these challenges like implementing secrets management, static application security testing (SAST) and policy enforcement. Though pipeline security presents obstacles, organizations remain committed to hardening these environments to optimize developer productivity and accelerate release velocity securely.

TECHSTRONG RESEARCH ANALYST VIEW

Today, addressing software supply chain security is critical. It is not optional or something to be addressed sometime in the future. Developers and SDLC, workflows and technologies

What attack surfaces are you concerned about for your CI/CD?

Vulnerabilities in open source, exposure of secrets, third-party services and insecure code run close to even as the attack surfaces of most concern. Pipeline security is close behind due to notable software supply chain attacks that compromised developers' pipelines.



Have you experienced a security incident in your CI/CD pipeline in the last 12 months?

The question is not if but when you will experience a security incident. The good news is that 59.7% noted no security incident in their CI/CD pipelines, but 21.5% reported they had experienced an incident in the previous 12 months.

they work with, are the new attack vector. Inputs into the development pipeline now receive greater scrutiny and added controls and processes to verify the security of code introduced from container and software images, package managers, open source and artifact repositories and own developers' created code. Protection of secrets is essential through the use of commercial, open source and hosted secrets management options which are integrated across the development, build and test environments.

> ***Trusted base images, vetted package managers, security scans and secrets protection are table stakes, that still leave workflow pipelines open to pipeline poisoning.***

While these commonly accepted practices are vitally important, they are genuinely the low-hanging fruit of software supply chain security. If security efforts stop there, development teams and their leadership create a false sense of security, essentially their own CI/CD anti-pattern, by not taking further steps to secure their software supply chains and workflow pipelines.

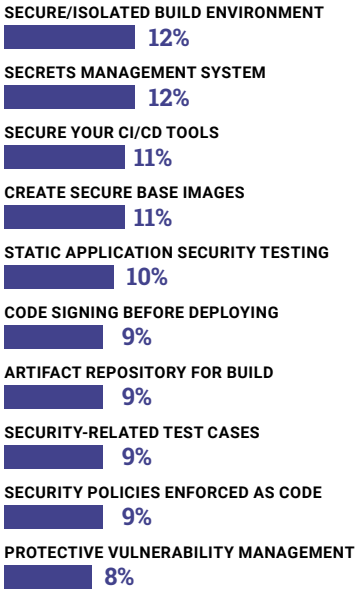
In August 2023, NIST issued a public draft of the NIST SP 800-204D ipd, Strategies for the Integration of Software Supply Chain Security in DevSecOps CI/CD Pipelines. SP 800-204D lays out two security goals in the application of SSC security measures or practices in CI/CD pipelines:

1. **Incorporate a range of defensive measures to ensure that attackers cannot tamper with software production processes or introduce malicious software updates (e.g., secure platform for build process).**
2. **Ensure the integrity of the CI/CD pipeline artifacts (e.g., repositories) and activities through role definitions and authorizations for actors.**

As a public first draft, SP 800-204D provides a baseline for securing CI/CD pipelines, repositories, images and build processes. It's a start, and we expect many more good things from NIST's work.

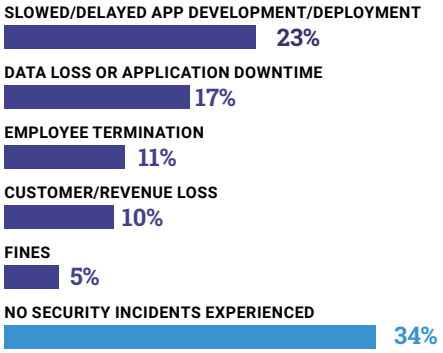
What steps are you taking to harden your CI/CD pipeline?

Hardening the CI/CD pipeline frequently starts with the build environment, the core of CI/CD. But this data shows there are many other areas required to consistently secure what goes into and comes out of the CI/CD pipeline.



In the last 12 months, have you experienced any of the consequences of a CI/CD security issue or incident listed below?

Cyberattacks on CI/CD pipelines can significantly disrupt software development and operations. Among organizations that experienced CI/CD pipeline security incidents in the past year, the most common consequences were slowed or delayed development (23%) and data loss or application runtime issues (17%).



Software Composition Analysis (SCA), when automated throughout the pipeline, provides trackable metadata about individual software elements as they are introduced into the development and platforms. SCA aids in identifying dependencies and errant or suspicious software, including where it was introduced into the pipeline, as a regular course of development and during incident response triage.

In further addressing the challenges traditionally faced in implementing secure and scalable CI/CD pipelines, some organizations are turning to cloud-native continuous integration approaches. With a Kubernetes-native approach, organizations can standardize and automate builds across applications using declarative pipelines. These pipelines allow the execution of builds and tests in isolated environments such as pods on a cluster, avoiding conflicts between teams. Containerization offers it's own level of protection as standardized software packaging, easy transportation between environments, a content "bill of materials", defined ingress and egress points, and software traceable from developer to production environments. As cloud-native CI/CD applications continue to gain wider adoption, this approach can help bring advanced pipelines within practical reach.

As CI/CD pipeline security continues to grow and mature, approaches that provide strong isolation of elements within the CI/CD pipeline address more challenging security issues, such as poisoning the CI/CD pipeline. Every software and security vendor needs to watch the CI/CD pipeline security space in the industry as we expect new innovative approaches to emerge.

Who defines and updates the build pipeline(s) for your code repositories?

DevOps teams play an important role in CI/CD pipelines with 58.9% of respondents noting that they're responsible for defining and updating build pipelines for code repos in their organizations. This being said, a sizable number either rely entirely on developers (23.1%) or do not have a rulebook for build pipeline responsibilities (17.9%).