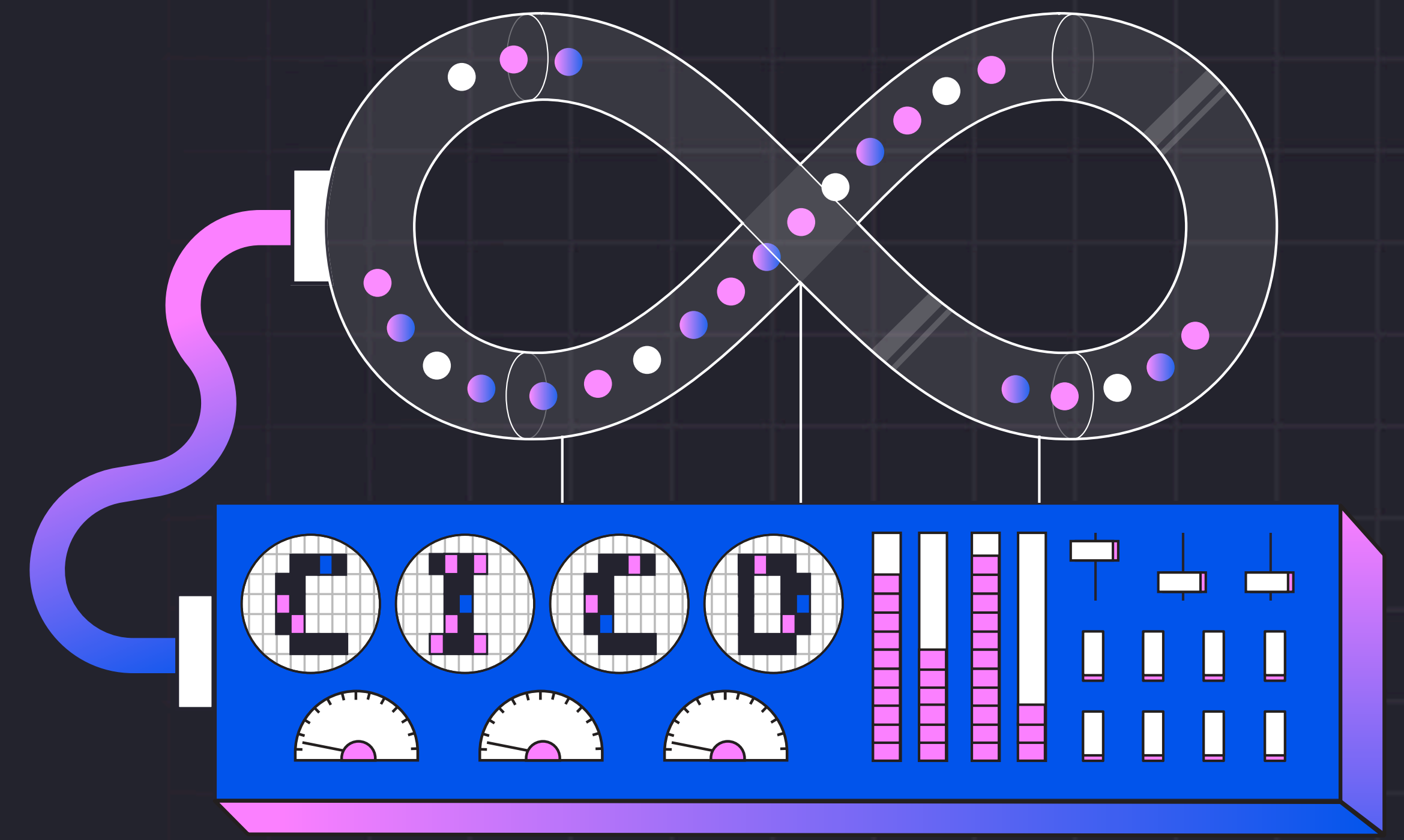


# CI/CD Pipeline Security Best Practices

Your CI/CD pipeline is the automated engine of modern software delivery, but its speed and consistency create a massive attack surface.



The last few years have demonstrated that CI/CD pipelines are a top target, with real-world breaches impacting major tech companies, including [Okta](#), [Sisense](#), [GitHub](#), and [JetBrains](#). Common vectors include dependency hijacking, compromised build tools, stolen credentials, and tampering with third-party scripts.

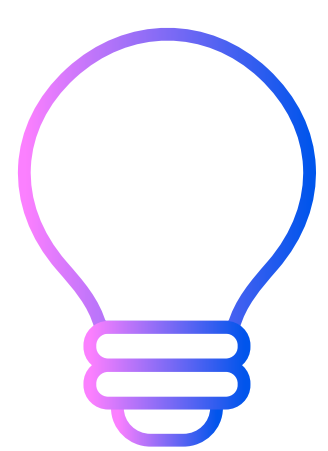
This cheat sheet is a practical guide to securing your environment, grounded in the [OWASP Top 10 CI/CD Security Risks](#). We provide actionable recommendations for defending the core of your development lifecycle.

## Top 10 CI/CD security risks and best practices

The OWASP Top 10 for CI/CD is an excellent framework for strengthening your pipelines. We'll break down each risk, offer practical tips to address it, and explain how Wiz can help.

### CICD-SEC-1 Insufficient flow control mechanisms

This risk covers unclear workflows that can allow unauthorized code into the pipeline. When workflows are unrestricted or poorly scoped, they create dangerous, unauthorized code paths.



#### Solution

Establish solid guardrails at the workflow level, utilizing policy engines to regulate the flow of code from commit to production.

#### Action items

- ☐ Use branch-protection rules, requiring [status checks](#) and [code owner reviews](#).
- ☐ Require manual approvals for deployments to critical environments.
- ☐ Restrict workflow triggers; avoid [pull request target](#) for untrusted code.
- ☐ Separate build and deploy workflows with explicit, auditable dependencies.



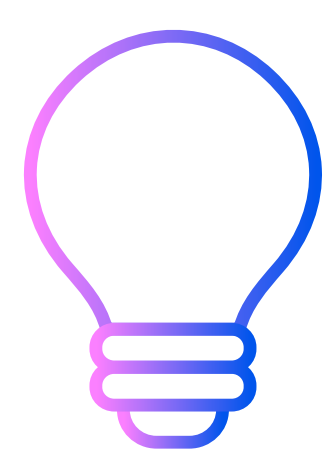


## How Wiz does it

Wiz identifies repositories and branches with weak flow control. It checks if default branches enforce re-approval for code changes or restrict who can push to them. The platform verifies that branch protection rules are active, disallowing force pushes and requiring code reviews to be completed. Wiz also flags risky configurations, like a repository that skips required merge checks while building containers with high privileges, helping you close these gaps before they're exploited.

## CICD-SEC-2 Inadequate identity and access management

This risk involves service accounts with excessive privileges or shared long-lasting credentials, which attackers can use to take over pipelines.



### Solution

Enforce a strict least-privilege model and use identity federation to eliminate static, over-privileged credentials.

### Action items

- ☐ Use OIDC-based ephemeral tokens instead of long-lived PATs.
- ☐ Assign unique, narrowly scoped service identities per pipeline.
- ☐ Automate credential rotation with a secrets manager.
- ☐ Regularly audit and revoke unused or expired credentials.

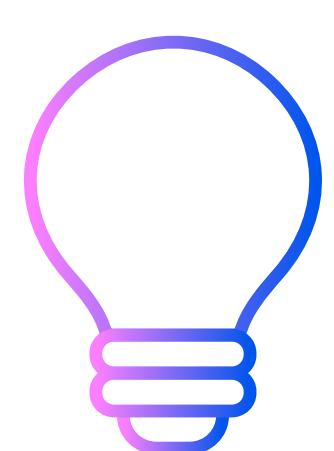


## How Wiz does it

Wiz detects weak identity and access controls across your CI/CD ecosystem. It finds container registries (ECR, ACR, GCR) that grant excessive privileges, allowing a compromised pipeline to push a malicious image. It also flags organizational-level misconfigurations, such as having too many owners or allowing non-administrators to create public repositories, helping you enforce the principle of least privilege. Wiz can also check whether secrets and encryption keys are regularly rotated and if the vaults they are stored inside are configured to automatically rotate them or not.

## CICD-SEC-3 Dependency chain abuse

This risk is about using unverified or malicious third-party packages that compromise your builds.



### Solution

Verify dependencies earlier in the development lifecycle (shifting left) and maintain a transparent supply chain by being critical of all third-party code.

### Action items

- ☐ Pin dependency versions using SHAs or content hashes.
- ☐ Integrate SCA scanning into every stage of the pipeline.



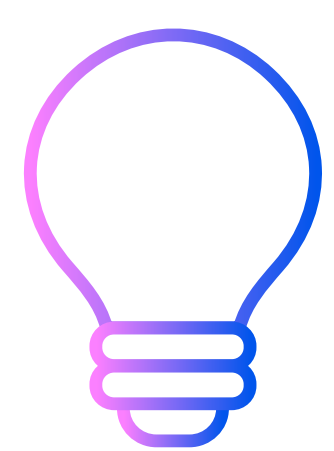
- ☐ Create automated alerts for upstream package updates.
- ☐ Maintain an internal, vetted registry for third-party dependencies.

#### **How Wiz does it**

While the specific risk is broad, Wiz addresses it by providing visibility into the software supply chain. By scanning code repositories, Wiz identifies vulnerable components in CI plugins like GitHub Actions and their sources of origin. Its analysis of high-profile attacks, like the [tj-actions/changed-files](#) compromise, informs its threat detection rules, allowing it to spot the patterns associated with dependency chain abuse in CI workflows.

### **CICD-SEC-4 Poisoned pipeline execution (PPE)**

PPE occurs when untrusted code runs in a privileged pipeline environment, allowing an attacker to move laterally, steal secrets, or tamper with the build.



#### **Solution**

Enforce zero trust in your pipeline execution environments and treat every job as potentially hostile by default.

#### Action items

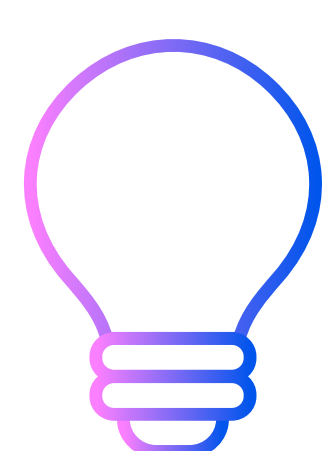
- ☐ Use dedicated, isolated, and ephemeral runners for untrusted builds.
- ☐ Limit runner permissions to only the resources required for the job.
- ☐ Sanitize all user-supplied inputs and variables in pipeline scripts.
- ☐ Employ container-based sandboxing with read-only mounts.

#### **How Wiz does it**

Wiz flags pipelines that use non-ephemeral self-hosted CI runners that are at risk of retaining sensitive data between builds. Wiz runs an agent to detect malicious activity on VMs hosting CI runners; this is a strong indicator of a Poisoned Pipeline Execution (PPE) attempt and can identify crypto-mining activity or malicious persistence mechanisms on runner hosts. Flagging a CI runner on a compromised VM is treated as a critical finding, as it can serve as a gateway for stealing deployment secrets and launching broader supply chain attacks.

### **CICD-SEC-5 Insufficient pipeline-based access control (PBAC)**

A single, monolithic pipeline that grants the same permissions to everyone is a significant risk. Without role-specific access controls, a single compromise can have a devastating impact.



#### **Solution**

Break down your CI/CD process into smaller, distinct services, each with a clearly defined and limited role.



## Action items

- ☐ Define granular RBAC policies per pipeline, job, and environment.
- ☐ Segregate runners and credentials for dev, staging, and prod.
- ☐ Enforce just-in-time (JIT) access for sensitive operations.
- ☐ Integrate policy-as-code (e.g., [Open Policy Agent](#)) to enforce rules programmatically.

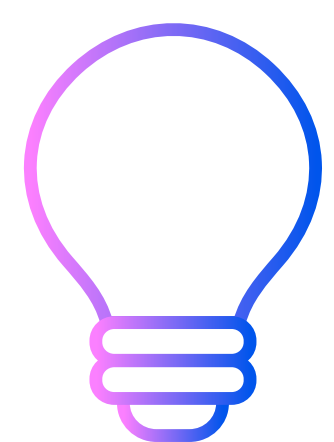


### How Wiz does it

Wiz checks for insufficient access controls tied to pipeline execution. It has policies that verify if runner groups are restricted to specific repositories, flagging groups visible to the entire organization as a risk. It also checks if the default workflow token permission is correctly set to read-only at the organization level and restricted from merging pull requests, preventing a common misconfiguration that grants excessive write access.

## CICD-SEC-6 Insufficient credential hygiene

This risk involves hardcoded secrets, long-lived tokens, and sensitive credentials left exposed in your codebase and CI environment.



### Solution

Treat secrets like cattle, not pets: Make them temporary, automated, and auto-rotated.

## Action items

- ☐ Centralize secrets in a vault and fetch them at runtime.
- ☐ Issue secrets with short expiry times.
- ☐ Continuously scan code, logs, and artifacts for exposed secrets.
- ☐ Rotate CI tokens on every build or a frequent schedule.



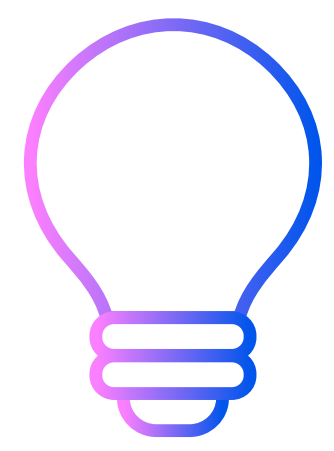
### How Wiz does it

Wiz excels at finding exposed credentials where they shouldn't be. It scans code repositories, CI workflows logs, and container images for cleartext cloud keys and SaaS API tokens. It contextualizes these findings, highlighting when an exposed key provides a direct path for lateral movement, which can turn a simple secret leak into a critical, environment-wide threat.

## CICD-SEC-7 Insecure system configuration

This risk covers misconfigured runners, outdated CI/CD platforms, and insecure plugins that create easy entry points for attackers.





### Solution

Harden your build hosts and use strict controls to prevent configuration drift, ensuring your runners maintain a secure, known-good state.

### Action items

- ☐ Regularly patch and update self-hosted runners and their base images.
- ☐ Remove unused tools, plugins, and services from runner images.
- ☐ Run build jobs as a non-root user with minimal privileges.
- ☐ Use configuration-as-code to define and monitor runner baselines.

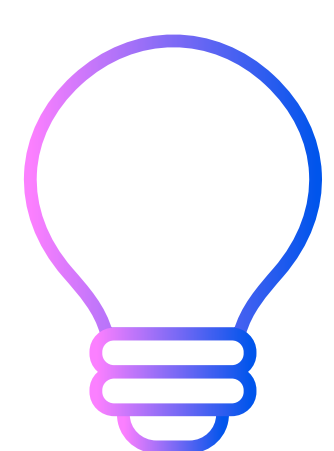


### How Wiz does it

Wiz identifies insecure configurations across your CI/CD toolchain. It checks if repository webhooks are configured with a secret and use SSL, flagging those that don't. It also finds abandoned CI runners on VMs, which could be signs of a past compromise, and detects critical vulnerabilities on publicly exposed CI servers, like the [Jenkins RCE \(CVE-2024-23897\)](#). Wiz even identifies malware hosted in private code repositories, stopping it before it spreads.

## CICD-SEC-8 Ungoverned usage of 3rd-party services

Unmanaged connections to external APIs and services create significant security risks for your system.



### Solution

Maintain a risk-based allowlist for all external connections, ensuring you know and approve every third-party service your pipeline uses.

### Action items

- ☐ Maintain an approved registry of third-party services.
- ☐ Proxy external API calls through a secure layer for policy enforcement.
- ☐ Monitor and alert on unusual outbound traffic from CI runners.
- ☐ Use private networking for sensitive service interactions.



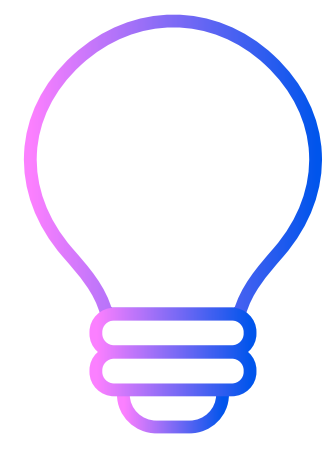
### How Wiz does it

Wiz helps you govern the use of third-party services by checking organizational settings in platforms like GitHub. It verifies if your organization restricts GitHub Actions and Apps to a list of verified or trusted creators. This control is crucial for preventing developers from pulling in a compromised or malicious GitHub Action or GitHub App from the public marketplace. Wiz can also detect inactive GitHub Apps with read or write access to the codebase, directly mitigating the risk of ungoverned or stale integrations.



## CICD-SEC-9 Improper artifact integrity validation

How do you know the artifact you're deploying is the one you built? Without proper checks, unsigned or unverified artifacts can be swapped or tampered with before deployment, potentially compromising the integrity of your system.



### Solution

Embed verification steps and digital signatures throughout your pipeline to create a transparent, trustworthy chain of custody for every artifact.

### Action items

- ☐ Digitally sign all build artifacts, such as images or binaries, at build time.
- ☐ Verify artifact signatures before any promotion or deployment.
- ☐ Generate and store an SBOM alongside every build artifact.
- ☐ Use immutable artifact repositories to prevent overwrites.

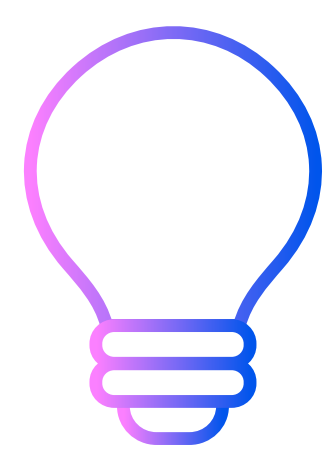


### How Wiz does it

Wiz checks for configurations that are fundamental to artifact integrity. It verifies that your default branches require all commits to be signed, authenticating the contributor and preventing code spoofing. It also detects critical misconfigurations that enable tampering, such as an ECR repository that allows image overwrites by any user and is consumed by a container with high privileges; this prevents an attacker from swapping a clean image with a malicious one. Wiz can also help security teams sign container images with the WizCLI and verify their integrity before deployment with the Wiz Admission Controller. Only trusted images are deployed in this case.

## CICD-SEC-10 Insufficient logging and visibility

Without comprehensive logging, you can't spot attacks. This risk stems from a lack of secure, centralized logging, which allows malicious activity to go unnoticed.



### Solution

Apply security information and event management (SIEM) and endpoint detection and response (EDR) principles to your CI/CD monitoring, ensuring that every pipeline event is transparent and auditable.

### Action items

- ☐ Enable and centralize audit logs from your VCS, CI platform, and runners.
- ☐ Use a SIEM with tamper-proof log retention policies.
- ☐ Define alerts for anomalous patterns and privilege escalations.
- ☐ Incorporate build telemetry into security dashboards.





## How Wiz does it

Wiz connects to your VCS and CI/CD platforms to verify that audit logging is enabled and configured for comprehensive event capture. It cross-references this data with runner host telemetry, flagging gaps where logs aren't being collected or forwarded to a central SIEM. The platform includes threat detection rules that analyze CI/CD audit logs for suspicious activities, such as a user rapidly escalating permissions or changes to critical pipeline configurations made outside of a standard Git-based workflow. This provides a unified view, turning scattered logs into actionable security signals.

## Beyond the top 10: Advanced defenses

The OWASP Top 10 is a great place to start beefing up defenses, but new threats are constantly popping up. As your CI/CD security matures, focus on these advanced defenses:

- **Runtime threat detection:** Adopt agent-based telemetry in build environments to achieve EDR-like capabilities, detecting threats such as reverse shells in real-time.
- **SBOMs and attestations:** Build a chain of custody with attestations to create a verifiable cryptographic record for artifacts.
- **Policy-as-code:** Use tools like OPA to enforce security rules directly in CI/CD pipelines, failing builds that violate policy.
- **Zero trust CI/CD:** Authenticate and authorize every action with context-based, just-in-time access and ephemeral credentials.

## Comprehensive CI/CD protection with Wiz

Dealing with security risks can be challenging. Wiz offers a simple platform to find, prioritize, and fix security issues across your entire CI/CD system:

- **Unified asset inventory and configuration drift detection:** Get a comprehensive inventory of your repositories, CI/CD platforms, and runners, simplifying detection of configuration drift. Wiz continuously monitors for the misconfigurations we've discussed.
- **Secrets and identity posture management:** Wiz doesn't just find hardcoded secrets; it maps out who and what can access them, showing you the full blast radius of a potential leak.
- **Pre-built threat detection rules:** Wiz comes pre-loaded with rules that detect active threats and emerging attack patterns, including those targeting popular GitHub Actions, like the [tj-actions/changed-files compromise](#).
- **Compliance posture reporting and audit trails:** Benefit from a unified view of your compliance posture against frameworks like SOC 2, PCI, and ISO 27001, with evidence automatically collected from your CI/CD environment.

Ready to see Wiz in action? [Request a personalized demo](#) today!

Schedule a demo with Wiz to get visibility into exposed and unprotected APIs.



Get a Demo