How to automate DevSecOps in Red Hat OpenShift

Empower developers with security guardrails—while providing runtime protection
# Table of contents

- Introduction ......................................................... 01
- Start with the software ........................................... 02
  supply chain
- Go beyond the software ........................................... 03
  supply chain
- DevSecOps best practices ......................................... 04
  - Build stage .................................................... 04
  - Deploy stage .................................................. 05
  - Run stage ..................................................... 06
- How Red Hat OpenShift Platform Plus ......................... 07
  helps automate DevSecOps
- Learn more .......................................................... 09
How to automate DevSecOps in Red Hat OpenShift

Modern software delivery methods that embrace continuous deployments across hybrid environments require a new security approach—one that provides security guardrails earlier in the application development process, automates security assurance at each step, and transforms security into a business enabler.

Most organizations are embracing DevSecOps for their hybrid environments

Do you have a DevSecOps initiative in your organization?¹

<table>
<thead>
<tr>
<th>26%</th>
<th>49%</th>
<th>25%</th>
</tr>
</thead>
<tbody>
<tr>
<td>No—DevOps and security remain separate, with minimal collaboration</td>
<td>Yes—It’s in an early stage, with DevOps and security collaborating on joint polices</td>
<td>Yes—It’s in an advanced stage, where we’re integrating and automating security</td>
</tr>
</tbody>
</table>

Start with the software supply chain

Implementing DevSecOps starts with a “shift-left” approach to security that introduces security checks earlier in the development process to protect the software supply chain.

Content repository and storage

Private, internal registries often offer greater security capabilities.²

Choose a registry that offers advanced access control and built-in vulnerability scanning, such as Red Hat® Quay.

Trusted content

The base images used to build your containers are critical to security.

Look for a trusted and resilient, minimalist base image, such as Red Hat Universal Base Image.

CI build infrastructure

The security of your continuous integration (CI) build infrastructure is as important as your production environment.

Limit administrative access and allow only required network ingress.

Implement security as code using continuous integration/continuous delivery (CI/CD) tools, such as Red Hat OpenShift® Pipelines and GitOps.

Go beyond the software supply chain: DevSecOps best practices for the full life cycle

Tools such as image scanners and registries can provide governance and detect vulnerabilities. However, reducing risk from misconfigurations (such as when your containers run with root privileges) and runtime incidents requires processes and best practices that operationalize DevSecOps across the full application life cycle.

Organizations experience security incidents across the build, deploy, and runtime phases. Implementing and automating DevSecOps provides developer-friendly guardrails that can decrease user error at build and deploy stages and protect workloads at runtime.

In the past 12 months, what security incidents or issues related to containers and/or Kubernetes have you experienced?¹

- **59%** Detected misconfiguration
- **32%** Security incident during runtime
- **31%** Major vulnerability to remediate
- **20%** Failed audit
- **6%** None

Build stage

Catch issues early so they don’t become a blocker later in the development life cycle.

DevSecOps best practices

- **Use** a trusted, enterprise-grade private image registry and limit access.
- **Use** minimalist base images.
- **Update** outdated base images and their dependencies when new versions are available.
- **Remove** exploitable and nonessential software like:
  - Package managers (apt, yum, apk).
  - Network tools and clients (curl, wget, netcat).
  - Unix shells, compilers, and debuggers.
- **Scan** the image registry regularly for known operating system, application, and language vulnerabilities.
- **Integrate** image scanners with CI tools and build vulnerability awareness early, especially in the case of severe policy violations.
- **Don’t use** secrets in a risky manner:
  - Do not store secrets unencrypted.
  - Do not use secrets as environmental variables.
  - Understand where and how secrets are being used to remove unnecessary exposure.
- **Break** builds that exceed the security risk threshold, such as those that:
  - Contain high severity, fixable vulnerabilities.
  - Have not been scanned recently, or at all.
  - Contain misconfigurations missed by a scanner, such as identity configurations or environment variables.
Combine security-relevant data from the build stage with deployment configuration to determine the security risk of each deployment, including:

- Image vulnerabilities.
- Access to secrets, storage, etc.
- Privileges and capabilities.
- Workload isolation, network exposure, and blast radius.

Use a secrets management tool to protect sensitive data.

Assess the privileges used by containers to keep it to a minimum viable set of capabilities.

Avoid deployments without resource limits unless absolutely necessary.

Annotate deployments with name, email alias, or Slack channel of the team responsible for the application.

Block risky deployments and alert the correct team for automated and streamlined remediation.
Implement dynamic scanning to detect vulnerabilities in running containers.

Use behavioral baselining and process allow-listing to identify unusual runtime activity, such as:

- Privilege escalation.
- Unauthorized network flows.
- Cryptomining.
- Malicious process execution or other exploits.

Mitigate threats with Kubernetes-native controls:

- Scaling to zero.
- Killing pods and restarting.
How Red Hat OpenShift Platform Plus helps automate DevSecOps

Red Hat OpenShift Platform Plus is a single hybrid cloud platform that helps enterprises build, deploy, run, manage, and provide security for innovative applications at scale. Multiple layers of security, manageability, and automation work across infrastructures and clouds to provide consistency throughout the software supply chain.

### Key DevSecOps features

- **Enterprise-grade image registry**
  Store, build, and deploy your container images in a private registry that provides built-in vulnerability scanning and enterprise authorization and authentication.

- **Secure-by-default base images**
  Take advantage of the greater reliability, security, and performance with Red Hat Universal Base Image.

- **Full container life cycle vulnerability management**
  Protect your images and running containers against known vulnerabilities based on specific language, package, or image layer, with streamlined remediation during build, deploy, and runtime.

- **Developer-friendly security guardrails**
  Shift security left with earlier security checks, and support developers with automated security monitoring that does not slow development or break workflows.
**Key DevSecOps features**

**Enterprise-ready CI/CD pipelines with built-in security checks**
Enforce application security policies by integrating them into the CI/CD pipeline, protect the software delivery pipeline from unauthorized access, and deliver actionable feedback to improve application security posture.

**Risk-based prioritization**
Deliver risk-based security analysis that collects and synthesizes data across software components, declarative configurations, and runtime activity.

**Runtime threat detection and response**
Protect workloads at runtime using prebuilt threat profiles and policies that help prevent or detect common threat vectors, such as unauthorized access, lateral movement, persistence, or resource hijacking.
Learn how Red Hat OpenShift Platform Plus can protect, manage, and provide security for your applications—across infrastructures and clouds.

Learn more