# 5 Keys to a Secure DevOps Workflow

**As container and cloud adoption accelerates, most enterprises struggle with visibility into their container and cloud environment. According to Gartner "Nearly all successful attacks on cloud services are the result of customer misconfiguration and mistakes." They also predict that through 2023, at least 99% of cloud security failures will be the customer's fault.[1]**

**In addition, containers are essentially black boxes. It's hard to see what's going on inside, and the lifespan of a container is very short. In fact, 49% of containers now live less than five minutes[2], according to our research. Traditional security tools can't see inside containers, handle the dynamic nature of Kubernetes, or scale across multi-cloud deployments. Proprietary security tools can't keep up with the standardization and speed of innovation possible with open-source software.**

How can you automate security and compliance controls to implement an efficient and secure DevOps workflow? While challenging, it's not impossible. With the right set of integrated tools, you can efficiently manage cloud and container security risks.

It is important to continuously scan for containers, hosts and cloud vulnerabilities, detect abnormal activity, reduce risks from cloud misconfigurations, and prioritize threats to ensure your applications are secure across their entire lifecycle. These five key workflows will enable you to cover the most critical security and visibility requirements so you can confidently run containers, Kubernetes, and cloud.

[1] Gartner: Innovation Insight for Cloud Security Posture Management

[2] 2021 Sysdig Container Security and Usage Report https://dig.sysdig.com/c/pf-2021-container-security-and-usage-report?x=u_WFRi

1

5 KEYS TO A SECURE DEVOPS WORKFLOW

# Continuous Cloud Security

**Continuous cloud security is required to immediately identify configuration errors and suspicious behavior. The following steps can help you validate your cloud security posture:**

Automatically discover and inventory your cloud assets including systems, applications, and services, running in your environment.

Flag misconfigurations based on open source Cloud Custodian rules. Check your cloud configuration periodically against CIS benchmarks to identify risky configuration settings (i.e. public storage buckets, exposed security groups and access controls etc) and take steps to remediate violations.

Detect unexpected changes and suspicious activity across all cloud accounts, users, and services by parsing cloud activity logs using open source Falco rules.

2

5 KEYS TO A SECURE DEVOPS WORKFLOW

# Manage Vulnerabilities: Scan container images and hosts

**As the number of container images, versions, and builds proliferates, you lose control of what software is being used and whether software updates are applied. Embedding security into your delivery pipeline as you build applications helps you identify and address vulnerabilities faster, and keeps your developers productive. Here are steps you can take to get control.**

Embed scanning into CI/CD pipelines and registries to prevent risky images from being deployed.

Adopt in-line scanning and maintain full control of your images.

Identify new vulnerabilities that impact the image once the container has been deployed.

Validate the build configuration (Dockerfile instructions) and image attributes (like size and labels).

Create different policies for each workflow, including images from public repositories and images built in-house. Consider different checks for each app.

Alert the right team for each issue (notify the owner of each image and integrate with your CI/CD tool to show the scan results directly in that context).

Automatically scan physical, virtual and cloud based host instances to identify vulnerabilities in OS and non-OS packages.

By integrating security analysis and compliance validation into this process, you can address issues earlier so you don't slow down deployment. This is known as "shifting security left."

3

5 KEYS TO A SECURE DEVOPS WORKFLOW

# Detect and respond to runtime threats

**You can reduce runtime risk by configuring applications with minimum privilege and access permissions. Your policies should also monitor for anomalous behavior and configuration drift. Creating policies that can prevent attacks without breaking the applications is challenging. Be sure to capture a detailed record for incident response. Consider these steps for reducing your runtime risk.**

Leverage Kubernetes-native controls for runtime protection of cloud-native workloads.

- → Use Admission Controllers to allow or block specific configurations and determine whether the container can be run on the cluster.
- → Prevent attacks by enforcing “least privilege” on containers through Pod Security Policy (PSP). PSPs control what permissions pods get at runtime (e.g., which user is running in privileged mode, whether they have access to the host network or filesystem, etc.).

Monitoring CPU and other resource usage is relevant for security, as they are typically exploited in DoS and crypto-mining attacks. Monitoring network connections gives you information about the attack, runtime behavior, and spread vectors. Some attacks are first detected as monitoring alerts rather than security violations.

Create and maintain a runtime policy that observes workload behavior, cloud activity, and identifies anomalous events.

- → Leverage tools to automatically build and customize policies or use out of the box Falco rules.
- → Implement least-privilege and compliant network policies with K8s and app metadata.
- → Simulate the effects of runtime policies before applying them in production to avoid breaking application functionality.
- → Visualize network communication in and out of a particular pod/service/app/tag over time with topology maps.
- → Apply the right security policy based on container role and Kubernetes context.
- → Automate use of events in cloud logs to detect threats and configuration changes on cloud services.

Streamline incident response and quickly respond to container and cloud security threats with a detailed activity record. Use capture files based on syscall data enriched with Kubernetes and cloud context to quickly answer the questions of “when”, “what”, “who” and “why” for your container security incidents. This detailed record allows you to conduct post-mortem analysis and determine root cause, even after containers are gone.

4

# Continuously validate compliance

**Implement compliance checks to meet regulatory compliance standards (CIS, SOC2, PCI, NIST 800-53, etc.) across containers, hosts, Kubernetes, and cloud. Monitor cloud services continually for configuration drift that can impact compliance. Measure compliance progress with scheduled assessments and detailed reports.**

Check your container and platform configuration against CIS benchmarks for Docker and Kubernetes.

Validate compliance during the build, mapping container image scanning policies to standards (e.g., NIST, PCI, SOC2, or HIPAA) or internal compliance policies (e.g., blacklisted images, packages, or licenses).

Implement File integrity Monitoring (FIM) to detect tampering of critical system files, directories, and unauthorized changes. FIM is a core regulatory requirement for a number of compliance standards.

Manage compliance at runtime. Check for best practices (e.g., don't run privileged containers and don't run containers as root) and look for known adversary tactics and  techniques. Achieve and maintain compliance with security frameworks mapping through a rich set of Falco rules for security standards and benchmarks, like NIST 800-53, PCI DSS, SOC 2, MITRE ATT&CK®, CIS AWS, and AWS Foundational Security Best Practices.

Provide proof of compliance with capture files that incorporate detailed forensics data and host scanning reports. It's important to record configuration and policy changes, including an audit of runtime changes for compliance audits.

5

5 KEYS TO A SECURE DEVOPS WORKFLOW

# Monitor and troubleshoot containers, K8s, and cloud

**Containers are short-lived, dynamic, and churn constantly. Once a container dies, everything inside is gone. You cannot Secure Shell(SSH) or look at logs, and most of the traditional tools used for monolithic applications don't help when something goes wrong!**

Monitoring the dynamic nature of container-based applications is critical for the high availability and performance of cloud services. Microservices-based applications can be distributed across multiple instances, and containers can move across multi-cloud infrastructure. Monitoring the Kubernetes orchestration state is crucial to understanding if Kubernetes is keeping all of the service instances running.

→ Monitor health and performance with deep visibility into infrastructure, services, and applications. Get the operational status of your cluster with Kubernetes orchestration monitoring.

→ Immediately identify owners for issue resolution using container and cloud context.

→ Identify pods consuming excessive resources and monitor capacity limits. Control unexpected billing and application rollouts and rollbacks of deployment by monitoring auto-scaling behavior.

→ Reduce cost by optimizing capacity across clusters and cloud.

Improve application performance and rapidly solve issues with deep container visibility and granular metrics enriched with Kubernetes and cloud context. You can monitor the impact of a given security incident on service availability.

Get productive quickly by using Promcat.io, a resource catalog of Prometheus integrations with curated, documented, and supported monitoring integrations for Kubernetes platform and cloud-native services.

**Confidently secure containers, Kubernetes, and cloud with the Sysdig Secure DevOps Platform. Scan images, detect and respond to threats, validate cloud posture and compliance, monitor, and troubleshoot.**

www.sysdig.com

**START YOUR FREE TRIAL**

**REQUEST PERSONALIZED DEMO**

sysdig