CASE STUDY · ROBINHOOD

Optimizing Kubernetes Security with Kyverno: A Deep Dive into Robinhood's Implementation

Robinhood's engineering team shared their journey of integrating Kyverno into their Kubernetes environment at KubeCon, explaining their decision-making process, migration from legacy systems, and how they tackled operational challenges along the way.

CUSTOMER	REPLACED	EVALUATED	ROLLOUT
Robinhood	Pod Security Policies and an in-house admission server	OPA and Kyverno	Audit mode, policy migration, then decommissioning

§01 CHALLENGE

The challenges before Kyverno

Before adopting Kyverno, Robinhood relied on Kubernetes' native Pod Security Policies (PSPs) and RBAC to manage security and enforce policies across their clusters. While these tools provided a solid foundation, they had limitations:

- **Deprecation of Pod Security Policies.** As Kubernetes deprecated PSPs, Robinhood faced the challenge of transitioning to a new solution.
- **In-house admission server.** Robinhood had developed a custom admission server to enforce more granular security policies, but it was difficult for other teams to adopt and maintain. Many infrastructure teams found it hard to contribute due to the complexity of the system.

§02 SOLUTION

Why Kyverno, and how they migrated

The team evaluated several options, with two leading contenders. OPA met all of Robinhood's security requirements, but it used the Rego policy language, which was unfamiliar to their engineers. Kyverno stood out for its flexibility, maturity, and ease of use: its policies are written in YAML, which was already familiar to Robinhood's engineers, and it provided a robust auditing tool to check existing resources and policies before enforcing them.

The migration was methodical and phased to ensure that policies were correctly enforced without disrupting operations:

01. **Evaluation and testing.** The team tested Kyverno in audit mode, running policies without enforcing them so they wouldn't accidentally break existing workloads.
02. **Policy migration.** They migrated their existing PSPs into Kyverno policies, continuously validating them with integration tests.
03. **Decommissioning legacy systems.** They phased out their in-house admission server and decommissioned Pod Security Policies, making Kyverno the primary policy management tool for their clusters.

They also implemented break glass scenarios, procedures for quickly disabling policies or reverting configurations if something went wrong, and used a combination of unit tests and end-to-end tests to validate their policies before and after deployment.

§03 RESULTS

Key benefits of Kyverno

- **Granular policy enforcement.** Fine-grained control over Kubernetes resources that was not achievable with native Pod Security Admission.
- **Easier policy writing.** Policies in YAML, a format familiar to many Kubernetes engineers.
- **Comprehensive testing framework.** A CLI tool for unit tests and integration testing to ensure that policies are properly enforced.
- **Active maintenance.** Responsive support from its community, so Robinhood could rely on a constantly improving tool.

Kyverno has proven to be a valuable tool for Robinhood, enabling better policy enforcement, simplifying policy writing, and improving scalability.

nirmata

Nirmata created Kyverno, the CNCF graduated policy engine for Kubernetes. We help platform and security teams govern clusters, workloads, and AI agents with policy as code.

WATCH THE TALK
Elevate Your Kubernetes Policy Game with Kyverno! - C. Breteche, L. Chiang, K. Tu
youtube.com/watch?v=CuW9Hszlnis

READ THE FULL STORY
Optimizing Kubernetes Security with Kyverno: A Deep Dive into Robinhood's Implementation
nirmata.com/2024/12/10/optimizing-kubernetes-security-with-kyverno-a-deep-dive-into-robinhoods-implementation

TALK TO THE KYVERNO MAINTAINERS
nirmata.com/request-a-demo

© 2026 Nirmata, Inc. Kyverno is a CNCF graduated project. Kubernetes is a registered trademark of The Linux Foundation.