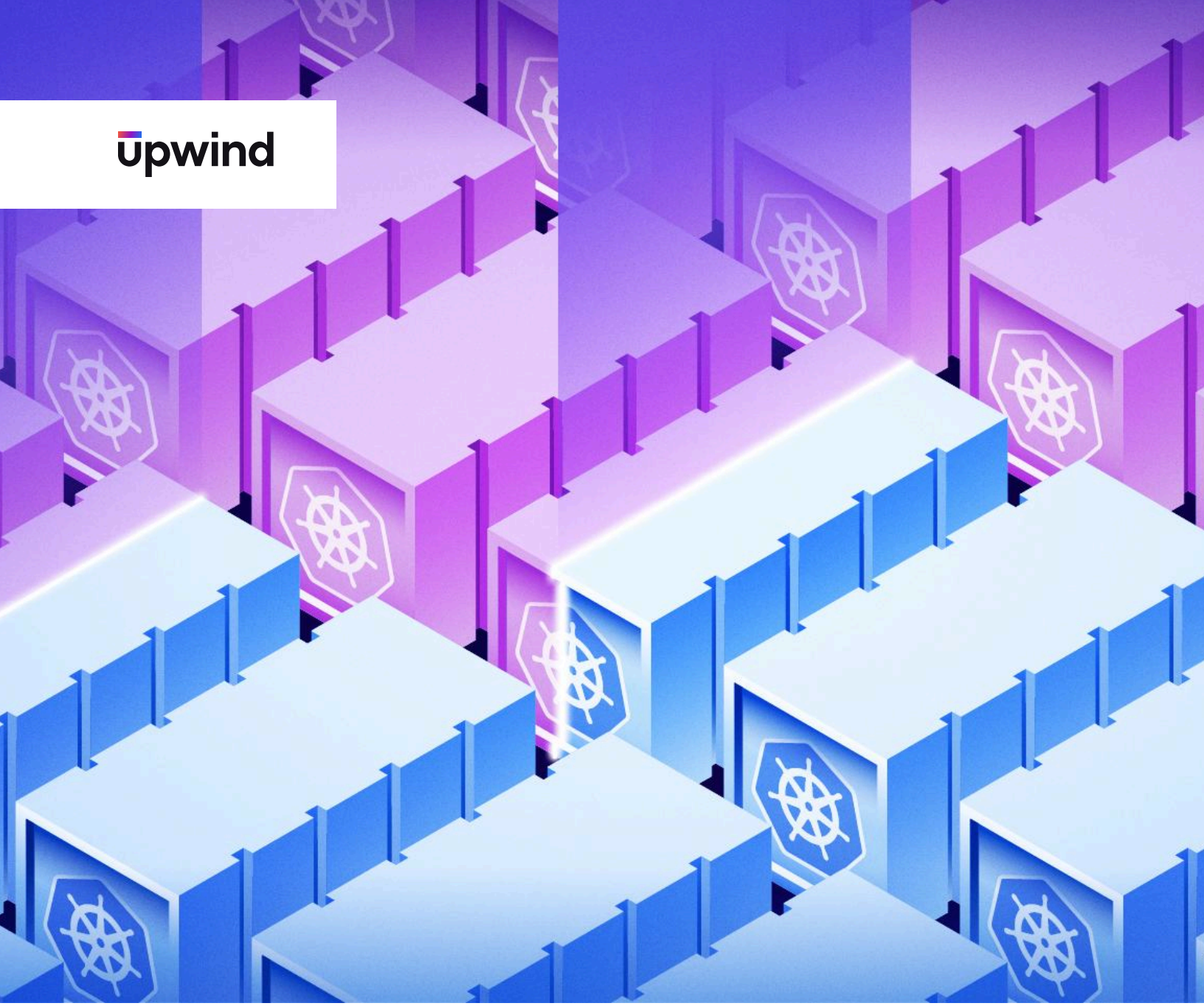
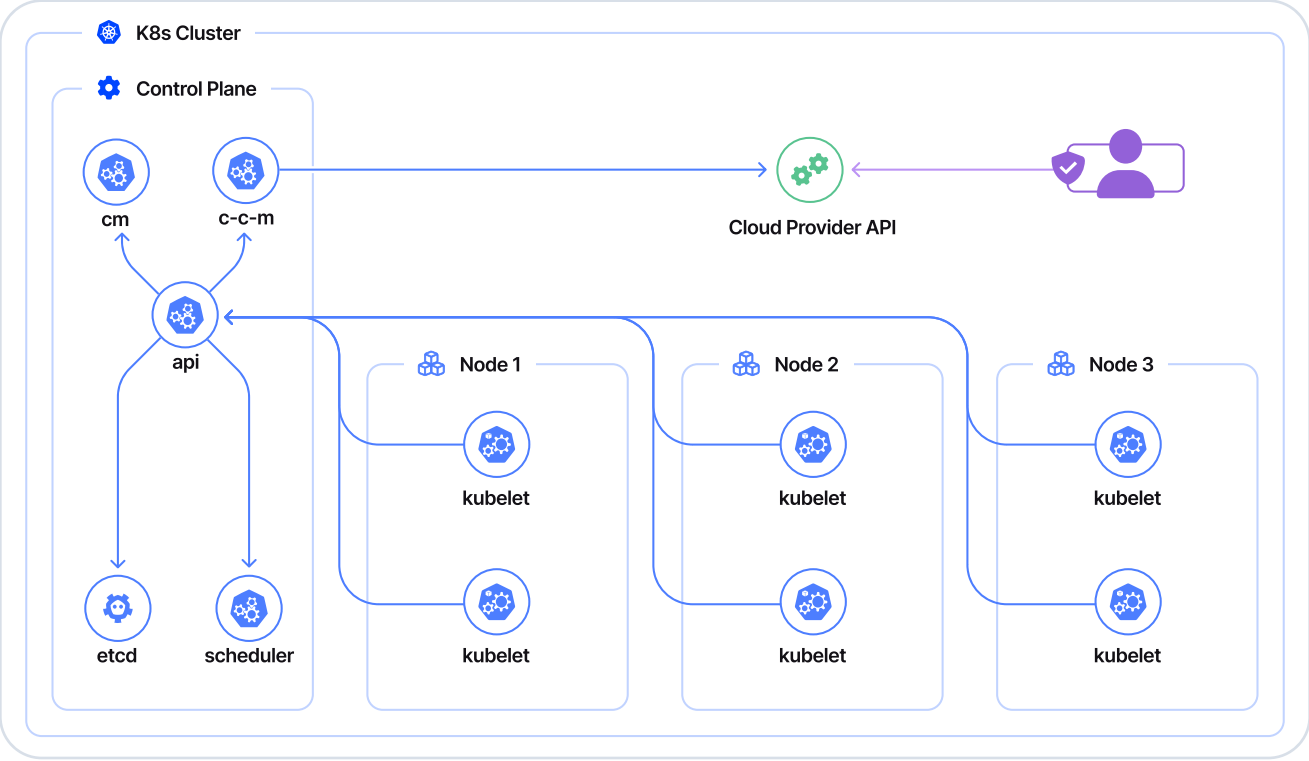
upwind



WHITE PAPER

Top Kubernetes Security Risks & Best Practices

Kubernetes has emerged as the leading orchestration platform of containerized applications in the modern cloud ecosystem. This has coincided with a steep increase in container adoption – CNFC's 2022 annual survey showed that **76% of organizations** that employ cloud-native approaches use containers. However, this rise in popularity has also made Kubernetes a target for threat actors and its security an increasing concern for CISOs and security teams.



Deploying Kubernetes involves clusters that are made up of nodes – the worker machines. Each node is either a virtual or physical machine that runs one or more "pods," or running container groups. A Kubernetes cluster also includes the control plane, which is the direct orchestrator of the entire cluster. Every cluster needs at least one worker node to be included.

What is Kubernetes Security?

Kubernetes security refers to the practices, technologies, and processes designed to defend cloud-native applications running on Kubernetes against malicious cyberattacks and vulnerabilities. As an open-source platform, Kubernetes relies on containers and microservices operating in the cloud. That creates its own unique security considerations for defending teams.

Effective Kubernetes security protects this entire infrastructure – clusters, nodes, and pods – against unauthorized access, misconfigurations, malicious code, and vulnerabilities. Security teams should employ proactive measures to protect Kubernetes across resources including:

ReplicaSets

Used to maintain a stable set of replica Pods running at any given time.

Jobs

Create one or more Pods and will continue to retry execution of the Pods until a specified number of them successfully terminate.

CronJobs

Perform regularly scheduled actions such as backups, report generation, and so on.

StatefulSets

The workload API objects used to manage stateful applications. They manage the deployment and scaling of a set of Pods, and provide guarantees about the ordering and uniqueness of these Pods.

DaemonSets

Ensure that all (or some) Nodes run a copy of a Pod.

Pods

A group of one or more containers that share storage and network resources.

Deployments

Act as the manager for your pods, ensuring they run according to your specifications.

Nodes

Individual machines (physical or virtual) that make up a Kubernetes cluster.

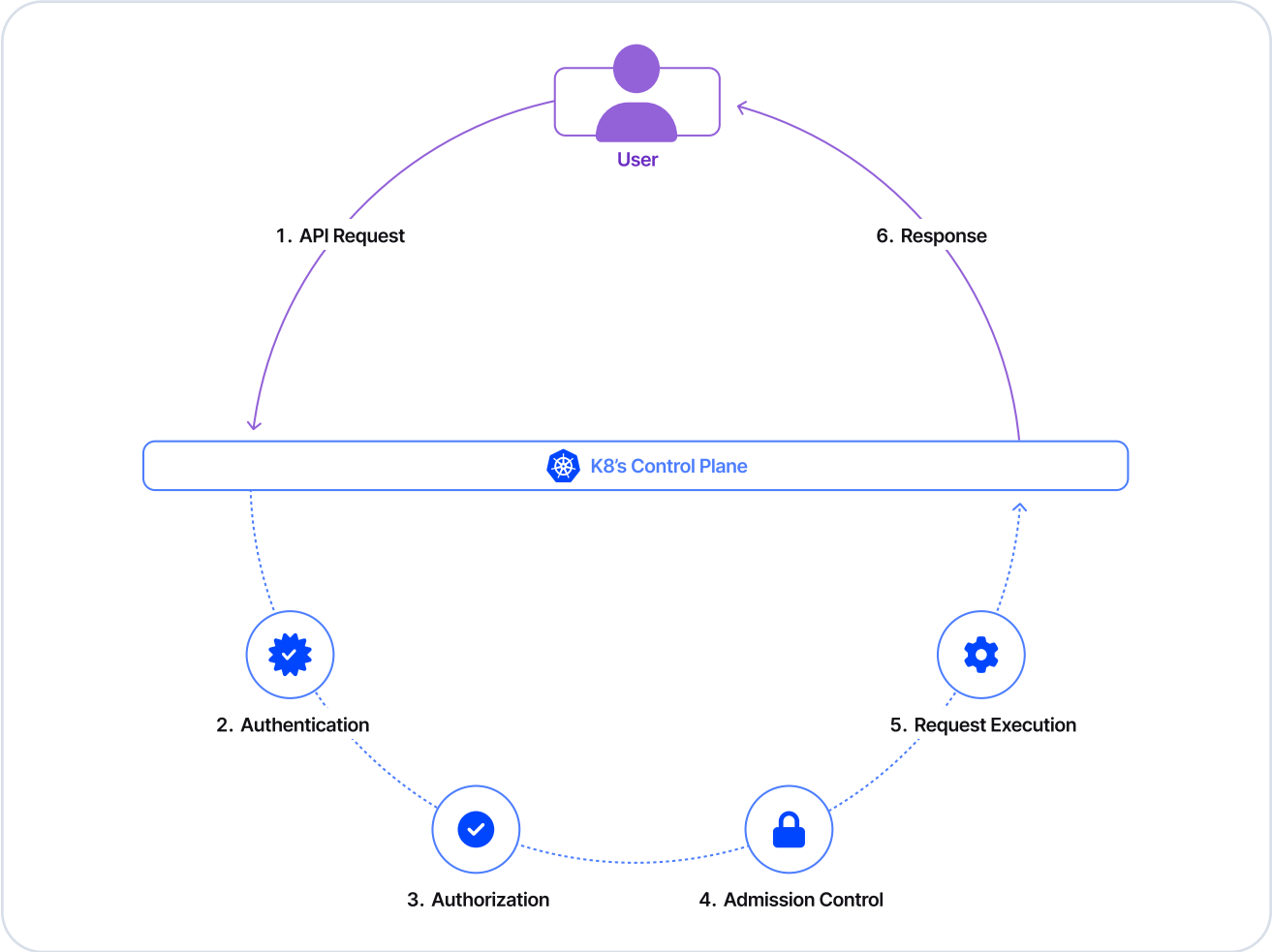
Modern Challenges in Kubernetes Security

The decentralized nature and open-source foundation of Kubernetes introduce challenges in governance, visibility, and threat detection. While its large contributor base accelerates innovation, it also increases the risk of malicious code inclusion or misconfigurations that may go unnoticed. Below, we dive into the top Kubernetes risk vectors.

Vulnerable Container Images

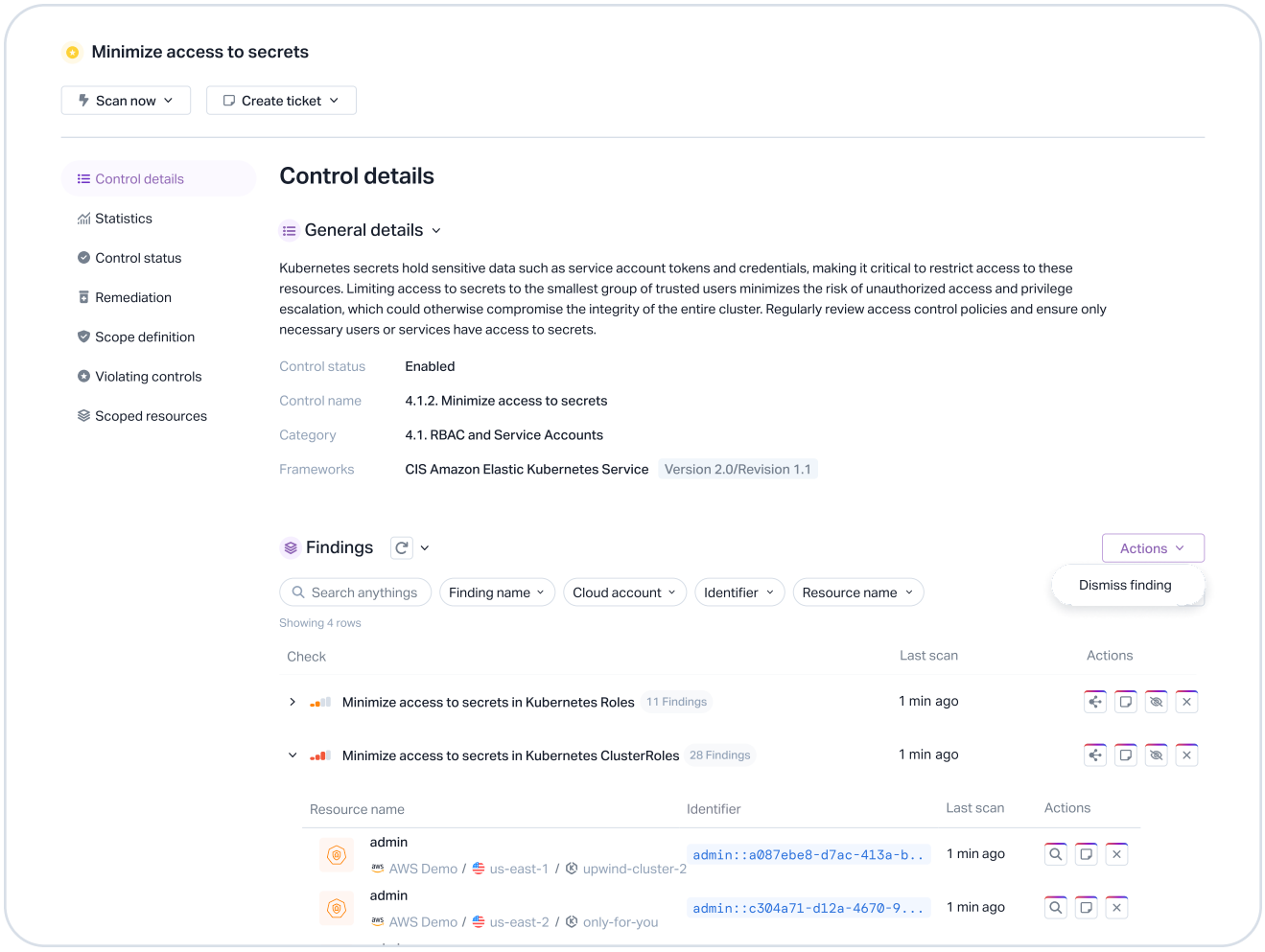
Kubernetes does not validate container images for vulnerabilities, making unverified or outdated images a critical threat vector. Security tools must be capable of automatically discovering, analyzing, and prioritizing vulnerabilities throughout the CI/CD lifecycle and across the runtime environment to maintain integrity.

Kubernetes API Exploits



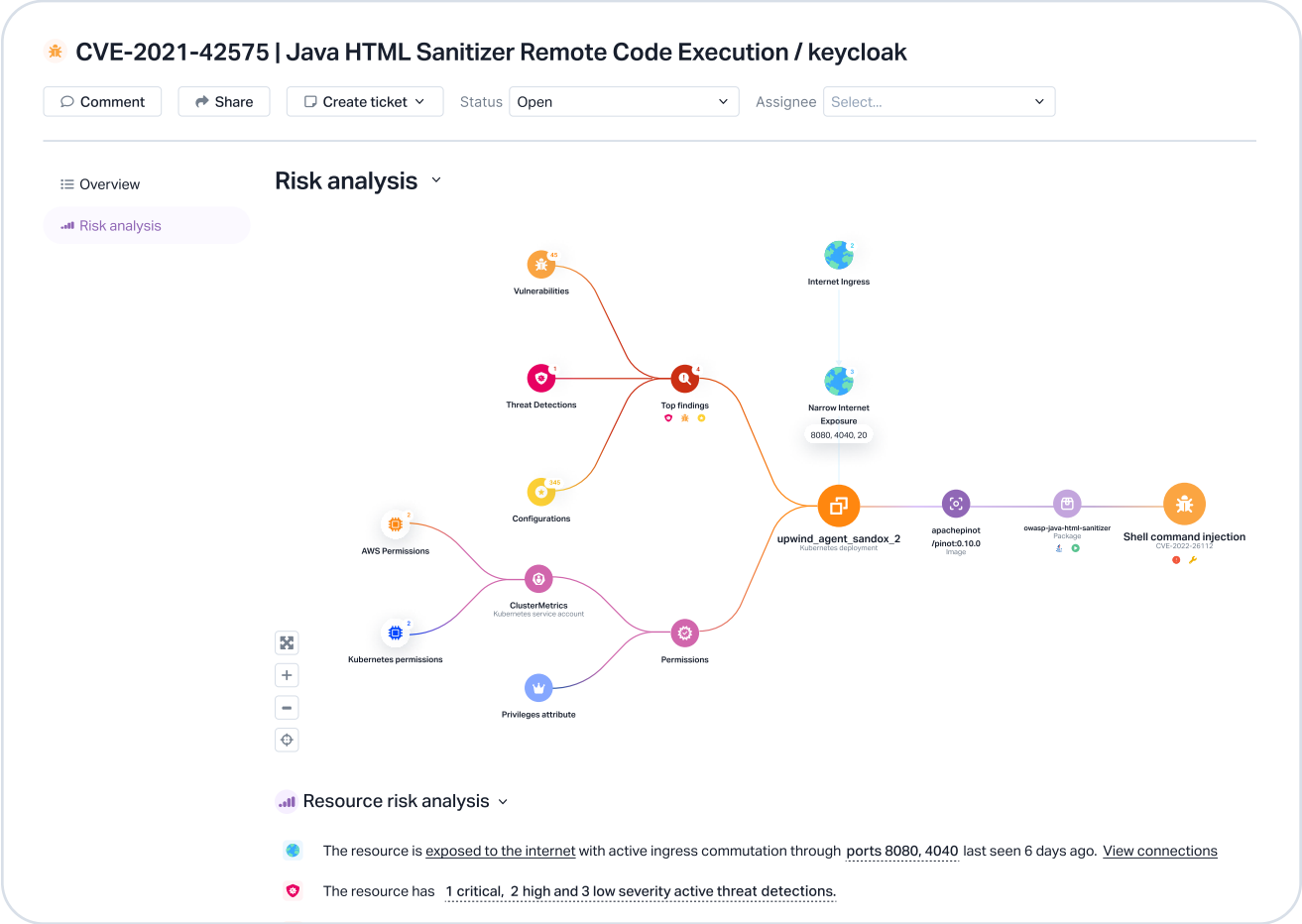
The Kubernetes API server is a primary interface and an attractive attack target. Improperly secured APIs can allow unauthorized access, lateral movement, or denial-of-service (DoS) attacks. Enterprise-grade solutions should offer centralized visibility, continuous monitoring, and runtime detection of anomalous API behaviors, including privilege escalations and misconfigurations, even in cloud-managed environments.

Cluster Misconfigurations and Default Settings



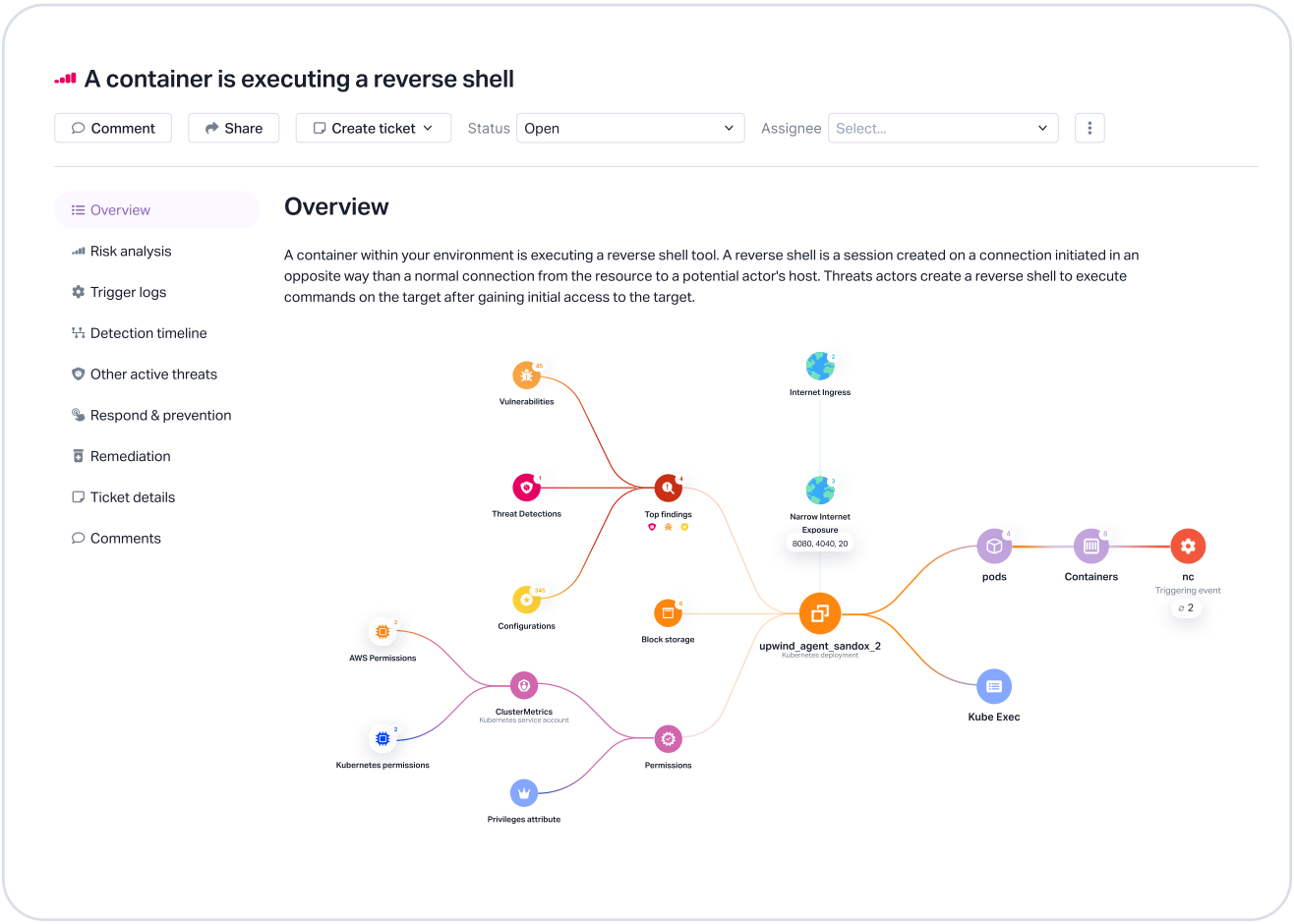
Misconfigured clusters often result from overreliance on default configurations that prioritize usability over security. These weaknesses expose workloads to data breaches and compliance violations. Automated auditing and real-time detection of configuration drift are essential to proactively remediate risks.

Lack of Network Visibility



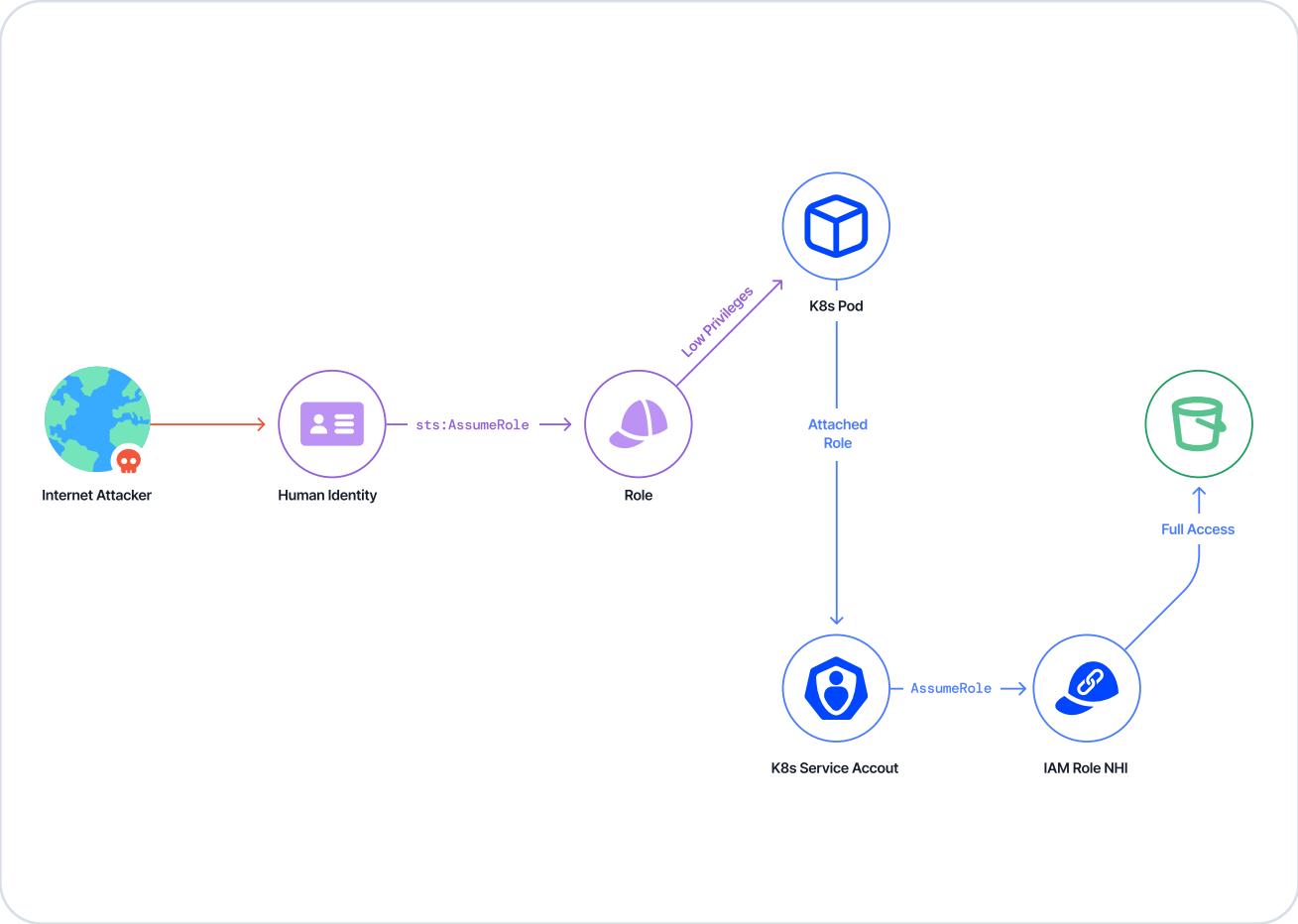
Kubernetes defaults allow unrestricted pod-to-pod communication, increasing the risk of MitM attacks and lateral movement. Lack of visibility into Kubernetes ingress communication between clusters compounds the risk. Modern platforms must deliver fine-grained network visibility showing continuous traffic flows to proactively identify risky behaviors and detect anomalies.

Runtime Threats to Containers



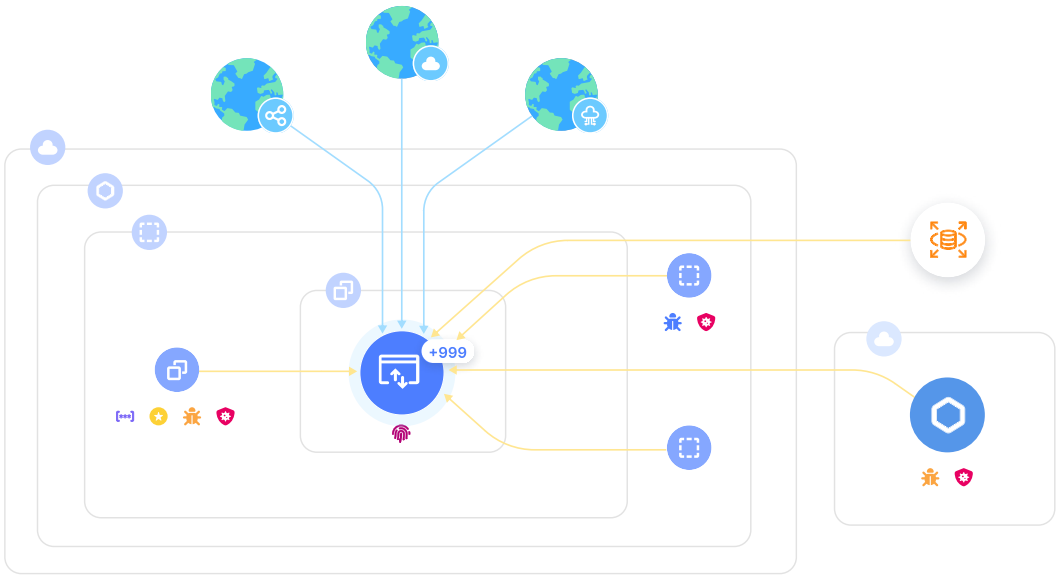
Containers can be compromised post-deployment due to permissive runtime policies or vulnerable libraries. Runtime protection must include OS-level monitoring, behavioral detection, and real-time threat response capabilities to guard against active exploitation.

Secrets Management & RBAC and K8s Permissions Oversight



Kubernetes Secrets are designed to manage sensitive data, but they require encryption and strict access controls to prevent leakage. Security teams should use tools that enforce best practices around secret encryption, usage auditing, and scope-limited access.

Overly permissive role-based access control (RBAC) increases the attack surface and may inadvertently expose sensitive operations. Regular RBAC audits and least-privilege enforcement mechanisms should be a core part of Kubernetes governance.



How Upwind Protects Kubernetes

Upwind leverages real-time, runtime insights and correlates them with CI/CD and DevOps context, giving you end-to-end visibility and protection for Kubernetes and associated workloads. By using Upwind, security teams can:

- **Prioritize Real Risk**

  Upwind leverages runtime insights to identify which packages are in use, internet-facing and exploitable, helping you focus on real risk.

- **Unify DevSecOps**

  Receive built-in DevOps context with every finding, including image version details and insights into CVE diffs.

- **Reduce Time to Remediation**

  Identify packages within your environment and their dependencies with our runtime software bill of materials (SBOM). Streamline remediation efforts and easily search for packages by framework, package manager and how many resources use each package.

- **Streamline Investigations**

  Integrates with your CI/CD pipelines, including Jenkins, GitHub Actions, Circle CI and GitLab, to automatically receive information on developer actions that led to code changes and resulting vulnerabilities. Streamline your investigations and identify the root cause of problems with every finding.

As attackers increasingly target cloud-native environments, organizations must invest in tools that go beyond static analysis and offer deep, runtime-aware protection. Upwind solves this problem by visualizing Kubernetes topologies, monitoring runtime behavior, and integrating with CI/CD pipelines to empower security teams to act with precision, reduce noise, and eliminate blind spots.

About Upwind

Upwind holistically secures containers & Kubernetes throughout the development lifecycle from runtime to build time.

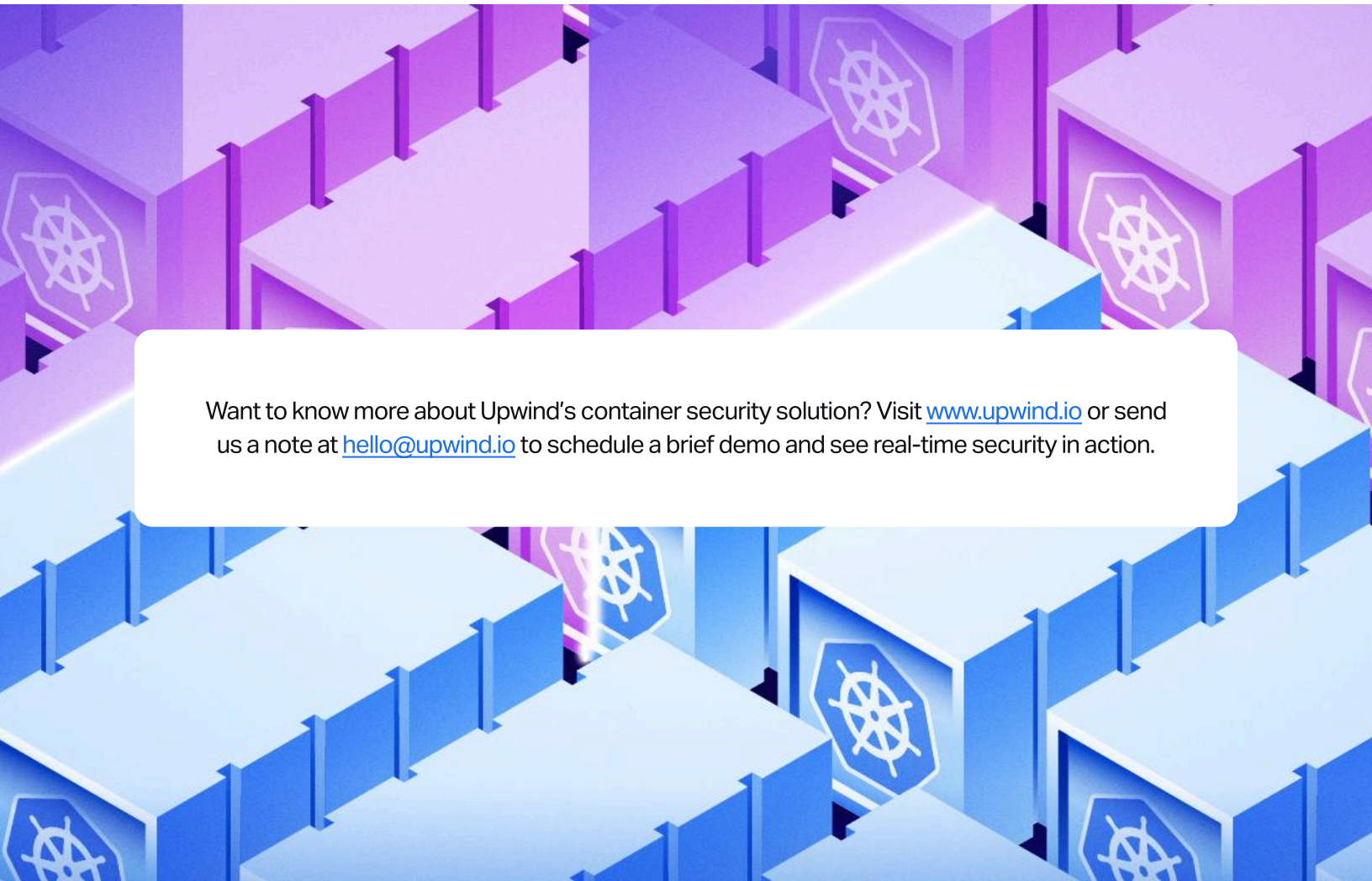
Container Security from Runtime to Build Time

Upwind gives you complete visibility and protection for all types of containers across Kubernetes, Amazon ECS and Fargate. Correlate runtime insights with build time context, enable end-to-end visibility and proactively stop threats with automated response.

Context-Rich Network Topology Mapping

Protect Containers in Multi-Cloud & Hybrid-Cloud Environments

Identify & Understand K8s Risks & Threats



Want to know more about Upwind's container security solution? Visit www.upwind.io or send us a note at hello@upwind.io to schedule a brief demo and see real-time security in action.