



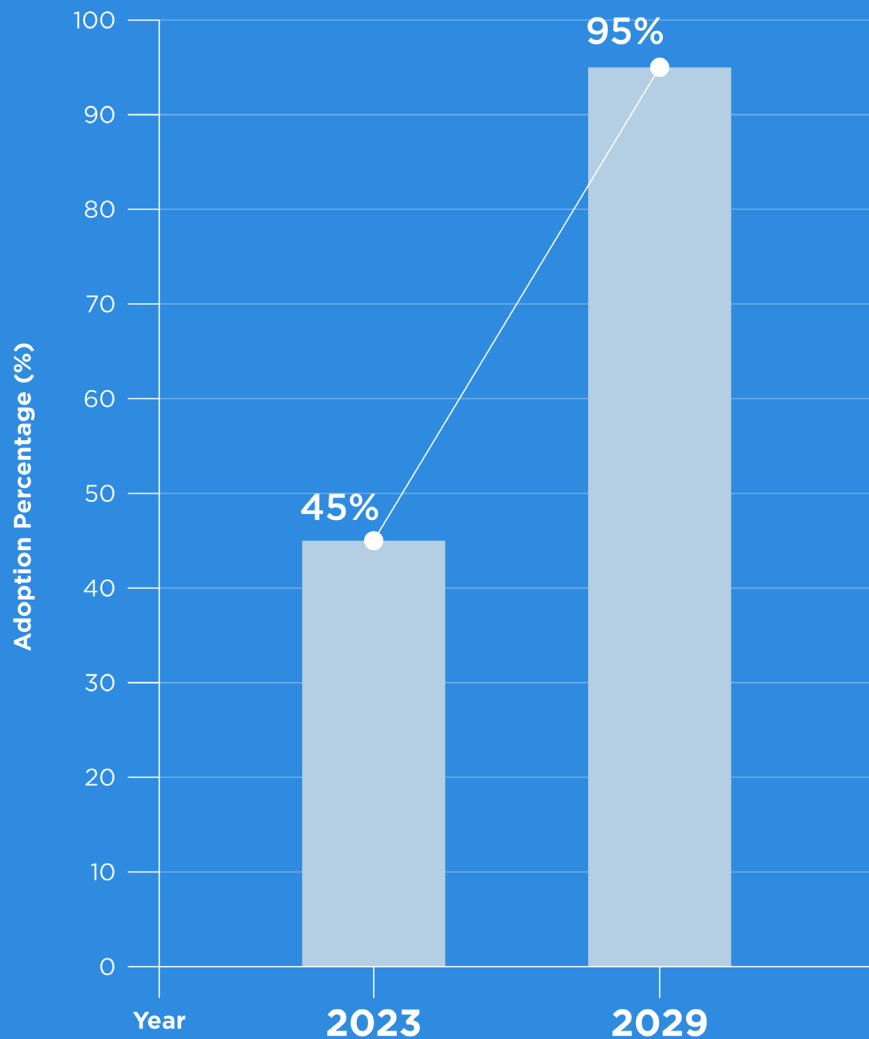
A Practical Guide to Kubernetes Misconfigurations and KSPM



Table of Contents

3	<u>The Rising Stakes of Kubernetes Misconfigurations</u>
4	<u>Understanding the Kubernetes Attack Surface</u>
8	<u>Real-World Data & Trends</u>
12	<u>What Is Kubernetes Security Posture Management (KSPM)?</u>
14	<u>4 Best Practices for Continuous KSPM</u>
17	<u>How Qualys Helps You Secure Kubernetes from the Inside Out</u>

Kubernetes Adoption Forecast: Significant Growth



Source: [Gartner](#)

Kubernetes is now the [driving force](#) behind modern cloud application delivery. In just the past few years, Kubernetes has gone from a niche open-source project to a widely adopted technology driving mission-critical workloads across the modern enterprise. [Gartner](#) predicts that by 2029, more than 95% of global organizations will be running containerized applications in production, up from less than 50% in 2023. That means millions more clusters—and millions more potential misconfigurations—may be open to exploitation.

Every cluster is a dynamic, multi-layered system with hundreds (sometimes thousands) of moving parts, from workloads and namespaces to role-based access control (RBAC) policies and application programming interface (API) configurations. When any of these elements are misconfigured, the consequences can be immediate and costly.

Kubernetes misconfigurations are one of the most dangerous and overlooked threat vectors in enterprise environments. A single overly permissive role or missing network policy can open the door to security breaches, downtime, and arduous recovery efforts. In 2023, researchers discovered [misconfigurations at SAP SE](#) that exposed credentials granting access to more than 95 million artifacts.

Although Kubernetes misconfigurations have emerged as one of the largest internal risks to uptime, security, and compliance, teams with the right tools and processes can evolve from reactive fixes to continuous posture management. Ultimately, this shift will stop issues before they cause lasting damage and allow Kubernetes to deliver the speed and flexibility it was designed for.

The Rising Stakes of Kubernetes Misconfigurations

According to [Red Hat](#), over half (53%) of organizations have experienced project delays or disruptions due to Kubernetes or container security issues, while 46% have suffered revenue or customer loss, and 30% have faced fines, legal action, or lawsuits as a result of misconfiguration-related incidents. These risks are only intensifying as Kubernetes becomes the de facto platform for running AI workloads. Vendors like NVIDIA have introduced GPU operators to simplify these AI deployments, accelerating adoption while also increasing reliance on privileged containers and complex configurations.

Kubernetes' configurability and flexibility are a double-edged sword. Features like rapid scaling and multi-tenant design enable powerful orchestration, but they also make it easy to introduce risky settings. Without continuous visibility into how workloads, configurations, and permissions interact at scale, teams are left flying blind.

Even small misconfigurations can quickly snowball into issues such as:

Security breaches from exposed secrets, open APIs, or overly permissive RBAC

Downtime and instability caused by misconfigured workloads or network policies

Operational drag as engineering teams scramble to identify, patch, and document issues

Escalating audit costs when gaps are discovered late in the process

Many security leaders are already aware of the problem. In the same Red Hat report, 42% of engineering, security, and DevOps professionals ranked security as a top concern in container and Kubernetes strategies. Threat actors have also taken note, as the layered nature of Kubernetes makes it a prime target for privilege escalation and lateral movement.

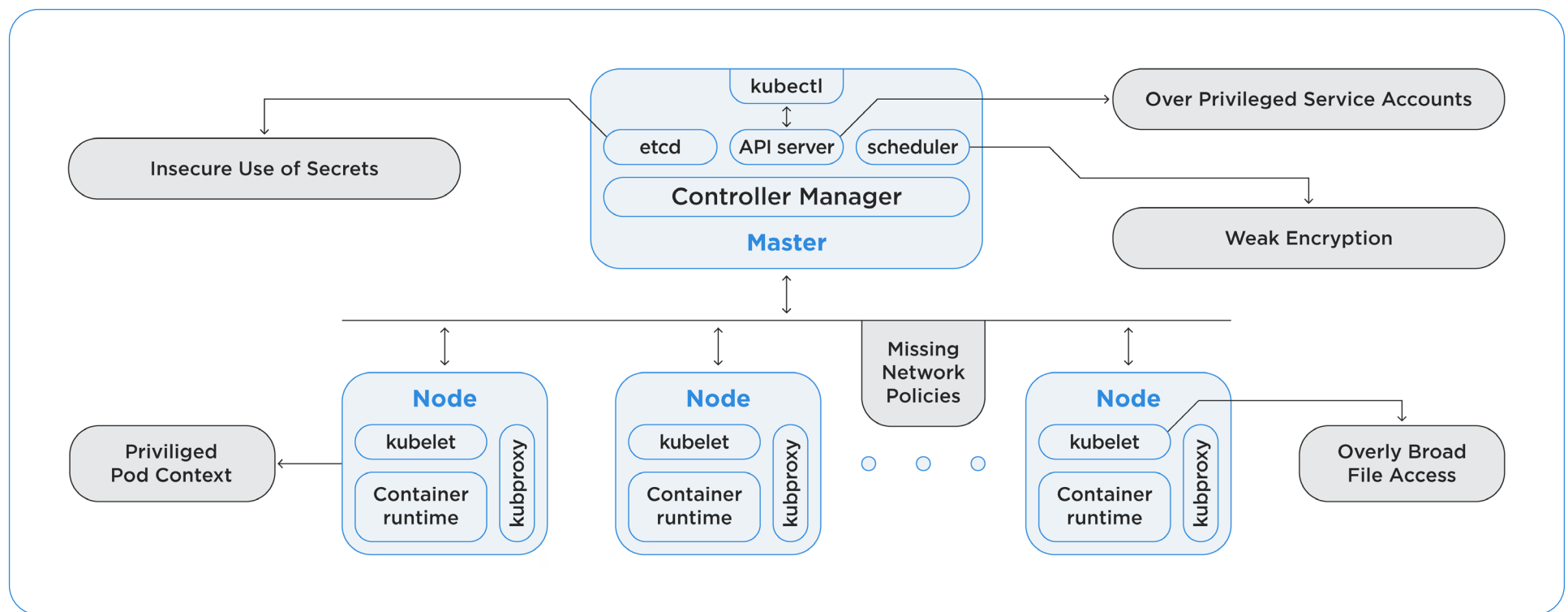
In short, Kubernetes misconfigurations represent a systemic risk to the business. The only effective defense is continuous, real-time posture management that makes secure configurations the default, not the exception.



Understanding the Kubernetes Attack Surface

Misconfigurations stem from the intricate architecture of Kubernetes itself. Each cluster is its own dynamic ecosystem, a “mini cloud” made up of a control plane, nodes, workloads, and APIs. With many potential entry points, each with layers of complexity, the attack surface is both wide and deep.

For example, securing an API isn't just a matter of enabling authentication; it involves managing access control, permissions, logging, rate-limiting, and integration with other systems. Risks quickly multiply when any component is left exposed.

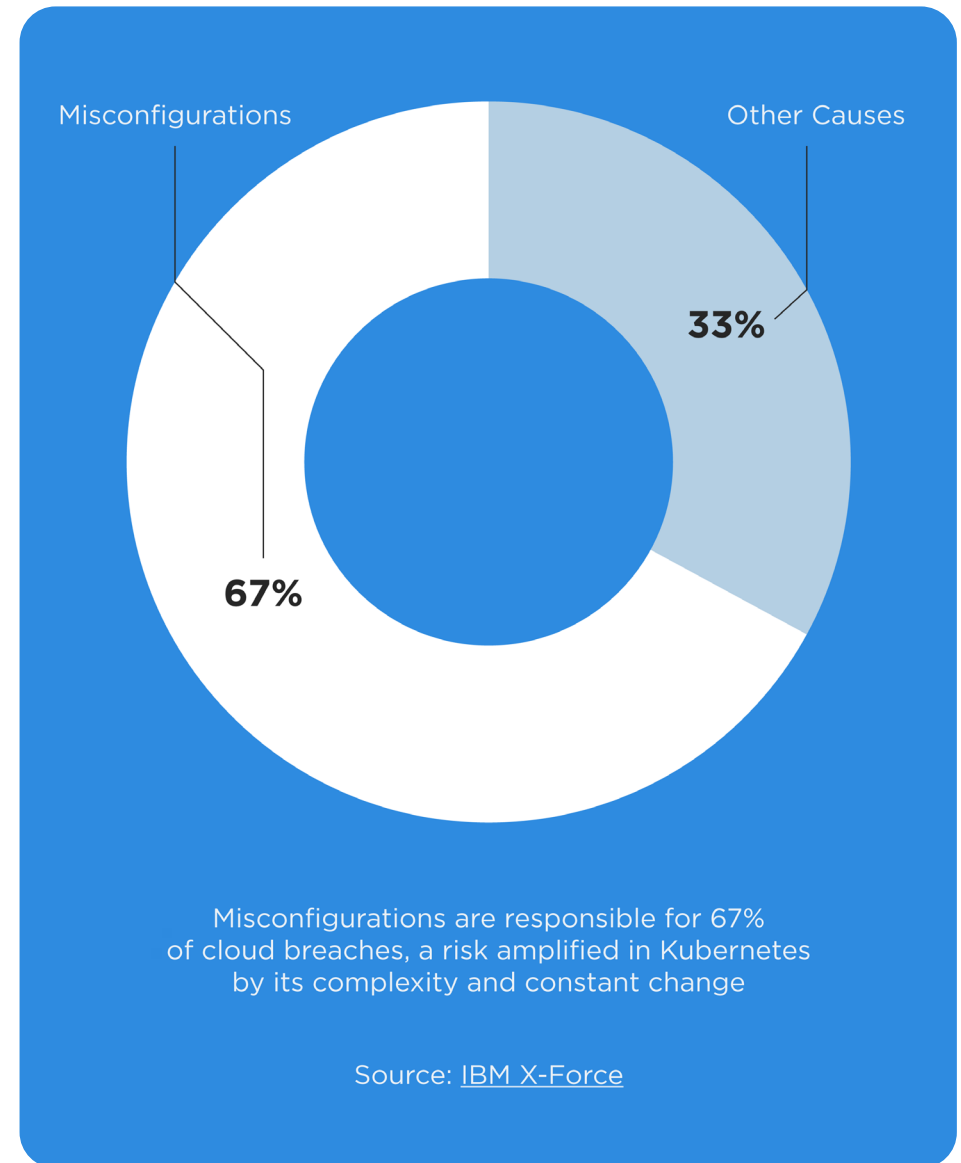


Kubernetes environments are inherently decentralized, ephemeral, and built for rapid scaling. This makes it incredibly easy to introduce insecure defaults or overly permissive configurations that static security tools struggle to catch in real time.

Here's what makes the Kubernetes attack surface so complex:

<p>Misconfigurations act as open doors:</p> <p>Overly permissive RBAC, insecure network policies, unauthenticated APIs, or improperly configured namespaces can grant attackers footholds or enable lateral movement within the cluster.</p>	<p>The control plane is a high-value target:</p> <p>Even when running managed Kubernetes services like Amazon EKS or Google GKE, misconfigured control plane integrations or role bindings can expose critical administrative paths.</p>
<p>Cloud-native environments expand exposure rapidly:</p> <p>Containers spin up and down by the second, and static audit tools can't keep up. This creates gaps that attackers can exploit in near real time.</p>	<p>Fresh clusters are under immediate threat:</p> <p>Newly deployed clusters in public cloud environments are scanned or probed by bad actors within minutes, demonstrating how little room there is for configuration errors.</p>

This dynamic attack surface demands continuous, Kubernetes-native visibility, especially into how cloud APIs, workloads, and role bindings interact at scale.



Why Misconfigurations in Kubernetes Happen

Kubernetes has transformed how organizations deploy and scale modern applications—but its flexibility and complexity also make it easy to get wrong. Misconfigurations no longer arise only from manual errors; they increasingly result from automation, scale, and the speed at which cloud environments evolve. Several factors commonly contribute to this challenge.

Human Error and Kubernetes Complexity

Fragile YAML configurations:

Dense and interdependent YAML files are prone to small formatting errors, leading to unintended privileges or exposed workloads.

Scaling multiplies mistakes:

As Kubernetes environments grow, human errors become more frequent and harder to detect.

High stakes of missteps:

A single misplaced value or indentation can compromise security across the entire cluster.

Insecure Defaults in Cloud and Platform Services

Ease over security:

Managed Kubernetes services often ship with permissive defaults, prioritizing quick setup over robust protection.

Silent vulnerabilities:

Defaults like overly privileged service accounts or public APIs create hidden entry points for attackers.

Hardening is essential:

Without explicit security measures, these insecure defaults leave environments exposed.

Lack of Continuous Visibility

Gaps in detection:

Periodic scans and manual reviews leave long windows where risks go unnoticed.

Dynamic scaling challenges:

As clusters grow and shrink, blind spots expand, increasing exposure to threats.

Real-world consequences:

Attackers exploit these gaps, as seen in a 2024 cryptojacking campaign targeting Kubernetes APIs.

Configuration Drift from Automation

Automation pitfalls:

IaC templates, Helm charts, and GitOps pipelines can unintentionally deploy insecure defaults.

Breaking GitOps promises:

Direct cluster modifications bypass version control, leading to untracked and insecure changes.

Drift amplifies risk:

Misconfigurations worsen over time, undermining the consistency automation aims to provide.

AI Workloads as a Risk Multiplier

Privileged containers:

AI pipelines often require elevated permissions, increasing the attack surface.

Data and compute exposure:

Misconfigurations can leak sensitive datasets or grant attackers access to vast compute resources.

Rapid provisioning risks:

The speed of AI workload deployment leaves little room for thorough security checks.

Other Contributing Factors

Complex dependencies:

Interconnected cloud services and third-party tools create intricate security challenges.

Ownership gaps:

Misaligned responsibilities between DevOps and security teams exacerbate misconfiguration risks.

Persistent vulnerabilities:

These factors make misconfiguration a long-standing threat in Kubernetes environments.

Real-World Data & Trends



Real-World Misconfiguration Trends

According to Qualys' analysis of hundreds of real-world cluster scans, some misconfiguration risks have become more pervasive than others. Our research found that access control misconfigurations are the most frequently observed category, followed by workload-related issues, secrets mismanagement, control plane weaknesses, and network misconfigurations.

Control Group	Number of Misconfiguration Occurrences
Access Control	16,615
Workload	9,519
Secrets	1,295
Control Plane	1,161
Network	629

Examples of Some of the Most Prevalent Misconfigurations for Each Control Group Include:

Access control: Misconfigurations here can grant users or services more privileges than needed, opening doors for privilege escalation and unauthorized access.

Access Kubernetes dashboard: Leaving the dashboard exposed or unauthenticated gives attackers direct administrative access

Prevent containers from allowing command execution: Without this control, adversaries can run arbitrary commands inside containers, leading to compromise or data theft

List Kubernetes secrets: Misconfigured permissions allow attackers to enumerate and exfiltrate sensitive credentials stored as secrets

Administrative roles: Assigning cluster-admin or overly broad roles unnecessarily creates high-value targets for takeover

Workload: Insecure workload configurations can make applications vulnerable to container breakout, privilege misuse, or lateral movement

Ensure that the default namespace is not used: Running workloads in the default namespace makes them harder to isolate, increasing the chance of privilege abuse

Ensure that Service Account Tokens are only mounted where necessary: Unnecessary token mounts allow attackers to steal credentials and access other cluster resources

Ensure that Security Context is applied to Pods and Containers: Missing security context can enable containers to run with dangerous privileges or root access

Ensure that default service accounts are not actively used: Default accounts make it easier for attackers to predict permissions and exploit them for lateral movement

Secrets: Inadequate secrets management can result in credential exposure and unauthorized access to cloud resources or sensitive systems

Automatic mapping of service account: Mapping secrets by default increases the risk of accidental leakage or theft of sensitive credentials

Control Plane: Weaknesses in control plane settings can compromise the “brain” of the cluster, exposing critical orchestration and administrative functions

system:authenticated user has elevated roles:

Excessive privileges for this user group can let anyone with basic cluster access escalate to admin

Consider external secret storage:

Storing secrets directly in etcd without encryption or external management exposes them if the control plane is compromised

Anonymous user had

RoleBinding: Allowing anonymous bindings can give unauthenticated users unintended access to cluster resources

Linux hardening: Lack of hardened configurations leaves control plane nodes exposed to OS-level exploits

Network: Misconfigured network policies can allow unrestricted traffic between pods or external exposure, undermining segmentation and isolation

Ingress and Egress blocked:

Failing to restrict traffic allows attackers to move data in and out of the cluster without controls

Missing network policy:

No policies in place means any pod can talk to any other, enabling easy east-west lateral movement

Cluster internal networking:

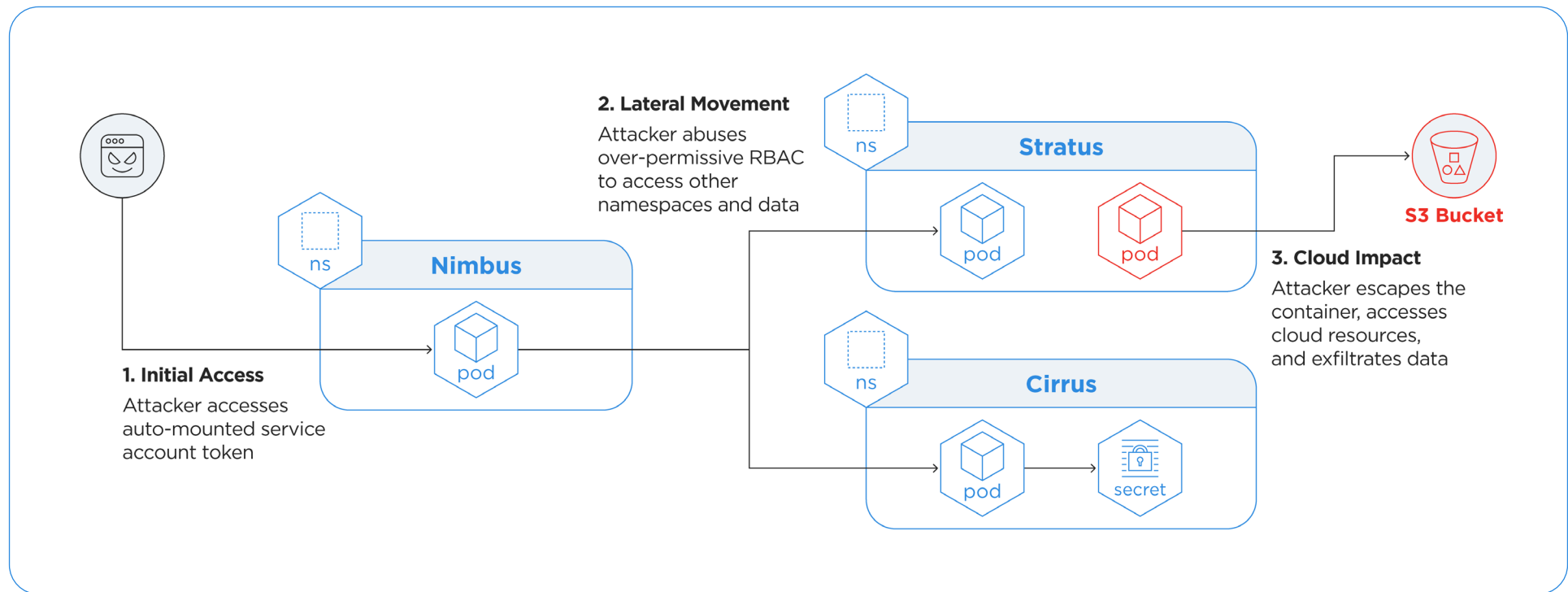
Insecure internal routing can allow attackers to intercept or reroute traffic between services

Network mapping:

Lack of restrictions enables reconnaissance and enumeration of the entire cluster network

When Misconfigurations Form Toxic Risk Combinations

Not all misconfigurations are dangerous on their own. A single privileged pod or overly permissive role might look like a minor issue in isolation. But when these weaknesses are chained together, they create a combination that attackers can exploit to move laterally and escalate privileges across the cluster.



For example, an attacker might start with an automounted service account token in one namespace, then leverage an overly permissive role to gain access to other namespaces. From there, they could create a privileged pod to escape into the underlying node and reach sensitive cloud resources such as S3 buckets. What began as a small oversight has now turned into a full-blown attack path to critical data.

That's why misconfigurations must be understood in the context of overall risk posture, not just as isolated findings. Organizations must be able to tie configuration data to broader risk signals (e.g., exposure, privilege levels, exploitability) so teams can identify and break these attack chains before they are abused.

What is Kubernetes Security Posture Management (KSPM)?

A single Kubernetes misconfiguration can cost [thousands](#) of dollars to fix in production. To avoid such costly mistakes, organizations need to take a more proactive approach to managing Kubernetes security and compliance, one that's tailored to the platform's unique complexity and risks.

KSPM is the continuous assessment and enforcement of security and compliance best practices across Kubernetes clusters, at both the control plane and the node level.

Much like Cloud Security Posture Management (CSPM) helps secure cloud infrastructure, KSPM focuses specifically on the unique risks and configurations inside Kubernetes environments. These include

overprivileged service accounts, insecure defaults, exposed secrets, and misconfigured workloads that could violate compliance.

However, whereas CSPM focuses on cloud resource configurations (e.g., S3 buckets, identity and access management roles), KSPM dives into cluster internals such as namespaces, workloads, networking, and policies.

KSPM is designed to secure Kubernetes from the inside out, ensuring resilience within each cluster's architecture to secure the control plane, reduce audit complexity, and prevent misconfiguration-driven exploits before they happen.



KSPM is CSPM for your Kubernetes clusters.

Gain Control Plane Visibility

CSPM Checks	What You're Missing Without KSPM	Why It Matters (Attack Path)
IAM role is scoped correctly in AWS	Kubernetes ServiceAccount mapped via IRSA to IAM role, but is bound to over-permissive RBAC in cluster	Attacker exploits pod with that SA → uses RBAC to escalate → steals cloud credentials
S3 buckets are private	Cluster lacks NetworkPolicy , allowing lateral pod access	Compromised pod can move laterally → reach app with S3 access → exfiltrate data
API Gateway is locked down	Kubernetes API service is exposed to 0.0.0.0/0	Attacker discovers endpoint → uses brute force or misconfigured token → gains cluster control
Root user in AWS is disabled	Kube-admin still used with static token	CSPM sees IAM secured, but attacker can escalate via K8s backdoor
EBS volume encrypted	etcd unencrypted and accessible	Secrets in etcd (DB creds, tokens) are visible in plaintext to attacker with low-level access

4 Best Practices for Continuous KSPM

Continuous KSPM demands a proactive, multi-layered approach. Here are 4 best practices for catching misconfigurations early, locking down access, and ensuring audit readiness at all times.

1. Gain Control Plane Visibility

First and foremost, security teams need to establish continuous visibility into the Kubernetes control plane, the “brain” of the cluster. This layer governs all orchestration, so weaknesses here have an outsized impact. Even in managed services like EKS, GKE, or AKS, misconfigurations can introduce high-risk attack vectors that traditional cloud security tools often overlook.

Strong visibility includes:

Monitoring API server access
for exposed or unauthenticated
endpoints

Identifying risky role bindings
or elevated permissions

Detecting control plane
changes introduced via automation
(IaC, Helm, GitOps)

Many traditional security tools monitor workloads but fall short in critical ways. They miss the control plane, require node-by-node configuration that doesn't scale (e.g., making 100 calls on a 100-node cluster instead of one API call), and often rely on open-source components that lack fine-grained reporting. Some also fail to provide exception management or consistent coverage across both on-premises and cloud environments.

2. Use Gold-Standard Guides and Benchmarks

Once visibility is established, the next step is to measure what you see against proven standards. Industry benchmarks translate high-level security goals into concrete, testable configurations. They also provide a shared language for security, DevOps, and compliance teams, ensuring everyone is aligned on what “secure” looks like before scaling protections across environments.

Essential guides and benchmarks include:

[CIS Kubernetes Benchmark:](#)

Prescriptive checks across RBAC, kubelet configurations, control plane auditing, secrets encryption, and network segmentation

[NSA/CISA Kubernetes Hardening Guide:](#)

Security-first recommendations to reduce the Kubernetes attack surface, enforce strong authentication, and harden the control plane and nodes

[OWASP Kubernetes Top 10:](#)

A risk-focused view of the most common and impactful misconfigurations seen in the wild, helping teams focus on the most exploitable weaknesses

3. Enforce Least Privilege and Other Low-Hanging Fruit

Excessive permissions and weak baseline controls create some of the fastest, easiest paths for attackers to escalate privileges or move laterally within a cluster. Addressing these issues as soon as possible reduces security risk and compliance exposure while building a strong foundation for more advanced posture management.

RBAC misconfigurations remain the most frequent and dangerous finding in Qualys scans, so start by tightening permissions:

- Remove wildcard access (*)
- Avoid default cluster-admin roles
- Segment access using Namespace and RoleBinding

From there, you can address other low-hanging fruit that will immediately reduce risk:

- Enforce default network policies to restrict pod communication
- Block workloads that run as root using admission controls
- Use [Pod Security Admission \(PSA\)](#) controls to enforce baseline pod protections

4. Leverage Automation to Improve Audit Readiness

Audits are too costly and complex to prepare for at the last minute, which is why audit readiness must start long before auditors arrive. Mature KSPM programs build automation into daily operations to collect evidence and enforce security by default.

A key capability here is the [Kubernetes Admission Controller](#), which, when deployed correctly, acts as a gatekeeper to block insecure or non-compliant deployments before they ever reach production. Combined with automated scanning, remediation playbooks, and compliance mapping, this ensures clusters stay secure and audit-ready.

KSPM automation allows teams to:

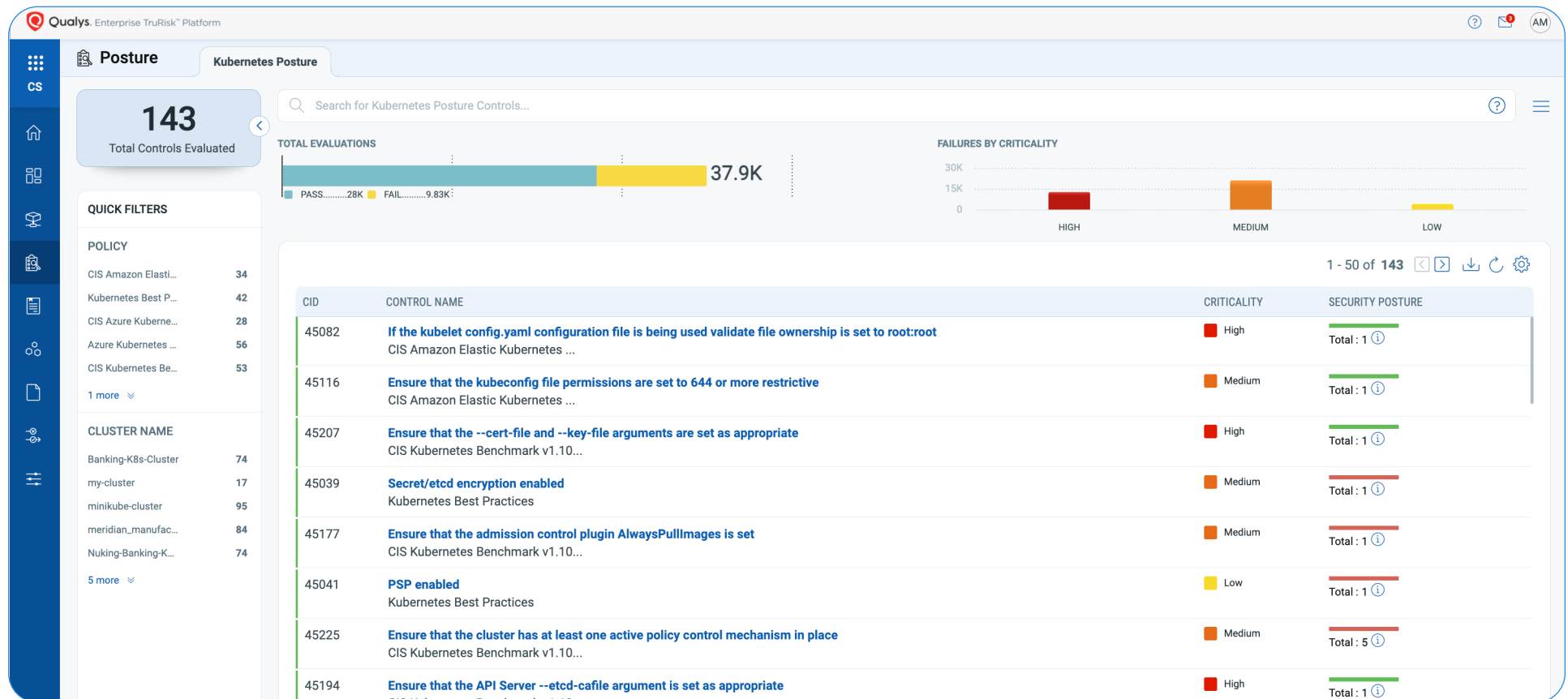
- Continuously scan for misconfigurations
- Map configurations to compliance frameworks (e.g., CIS, PCI, HIPAA, NIST)
- Automatically run remediation playbooks
- Collect evidence and produce reports

Manual Audit	KSPM Automation
Reactive (triggered by audit cycles)	Continuous, preventive auditing
Weeks to prepare reports	Instant, auditor-ready export
Prone to human error	Policy-based, consistent results
High cost per audit	Lower total cost of ownership

How Qualys Helps You Secure Kubernetes from the Inside Out

Kubernetes is just one piece of your broader attack surface, and managing it in isolation only makes things more difficult. That's why KSPM built into TotalCloud Kubernetes & Containers Security (KCS), enables continuous hardening of your Kubernetes clusters both on

premises and in the cloud. With KCS, you get unified visibility and reporting across any cluster in any cloud, streamlined audit reporting that proves compliance, and findings that tie back to risk prioritization and attack paths for a truly integrated security posture.



With TotalCloud KCS and integrated KSPM, you get:

Unified, Real-Time Cluster Inventory and Scanning:

Gain comprehensive visibility across all clusters—on-premises and in the cloud—through a single Cluster API Collector that continuously discovers assets, configurations, and vulnerabilities in real time.

Continuous and Scheduled KSPM Reporting Across Environments:

Generate on-demand or automated compliance reports for Kubernetes posture across EKS, AKS, GKE, OKE, Red Hat OpenShift and any Self-Managed or on-prem clusters, ensuring consistent security and audit readiness at scale.

Customizable, Audit-Ready Reporting and Insights:

Create tailored compliance and governance reports combining real-time and scheduled data, helping teams demonstrate adherence to CIS, NIST, or internal security frameworks with minimal effort.

Admission Control for Secure Deployments:

Prevent drift and insecure configurations before they reach production with policy-based admission controls that validate workloads and block deployments violating posture or compliance requirements.

A critical part of the TotalCloud advantage is the platform's scalable control plane visibility. With a single API call to your Kubernetes control plane, Qualys delivers complete, cluster-wide visibility without deploying extra software such as kubectl on each node. This reduces operational overhead while surfacing high-risk misconfigurations.

“Qualys plays a key role in helping us to protect all our virtual machines, workstations, and cloud services, and we look forward to working with Qualys to further strengthen our security posture.”

– Miroslav Sklansky, Senior Director and Head of Information Security at EPAM Systems

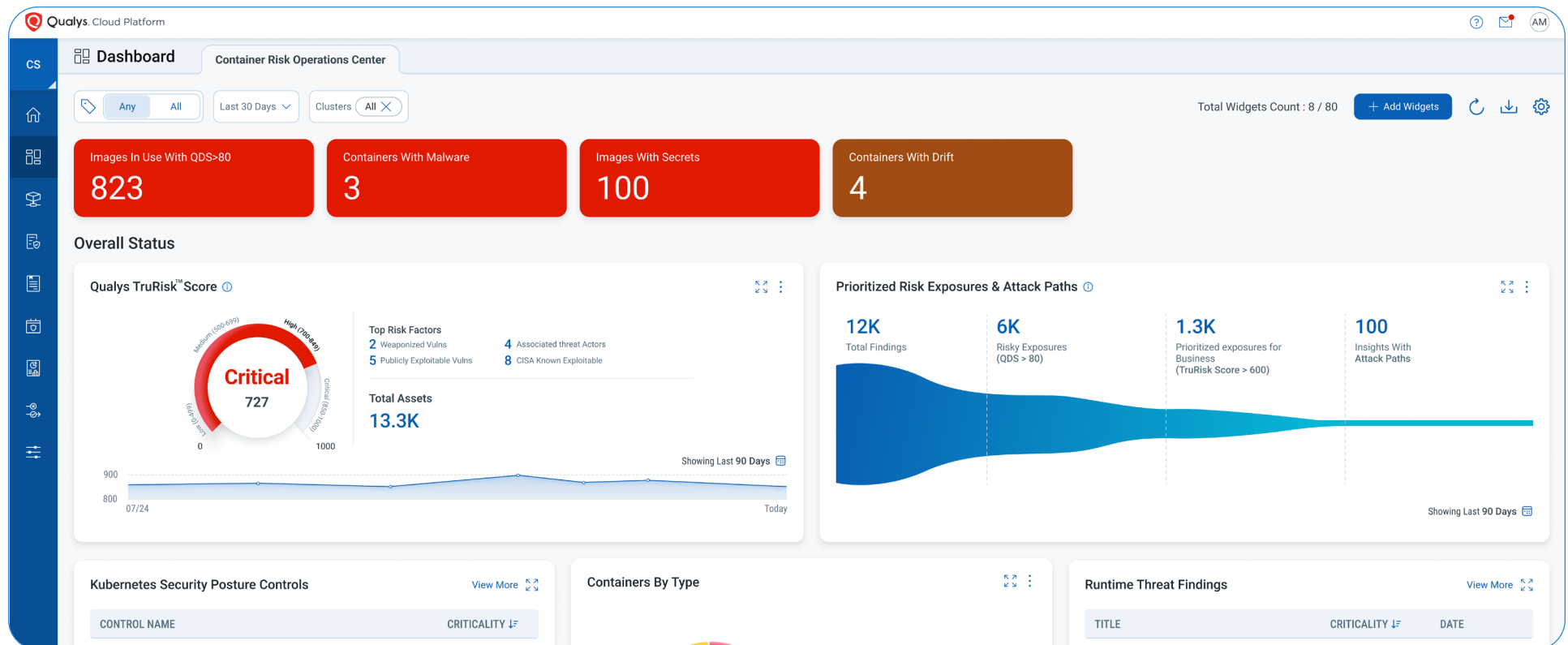
Closing the Misconfiguration Gap Starts with KSPM

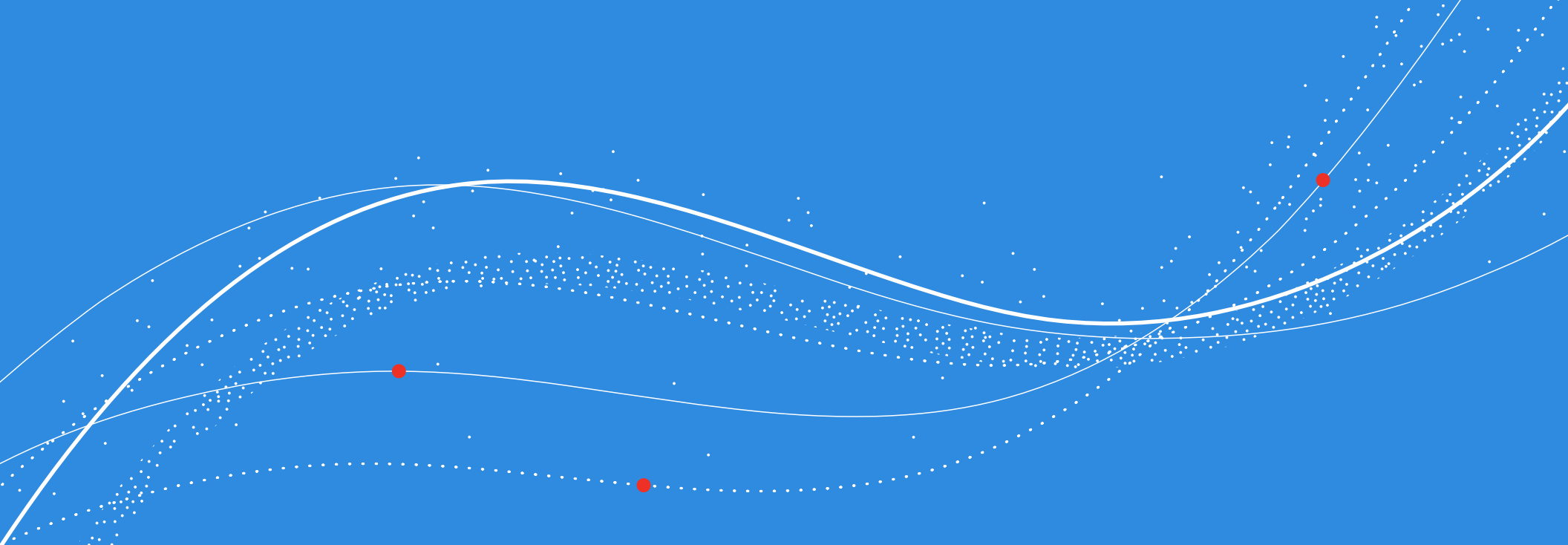
With Kubernetes now powering mission-critical workloads such as financial transactions and patient data processing, legacy approaches to compliance can't keep up. Every misconfigured role or excessive permission is a direct risk to uptime, regulatory standing, and customer trust.

KSPM provides a proactive framework for reducing these risks. By following essential practices like enforcing least privilege with RBAC and continuously monitoring the control and data planes, teams can

secure clusters from control plane to workloads while staying audit-ready every step of the way.

[Qualys' Kubernetes and container security features](#) make it easier for businesses to achieve end-to-end visibility and proactive risk reduction across environments. Users are backed by Qualys' 25+ years of expertise and global customer trust, helping reduce patching costs by up to 30%, improve audit success, and increase security maturity.





**Try Qualys TotalCloud free for 30 days to see
how our platform simplifies KSPM so you can
de-risk your cloud and container environments.**

Schedule a Call



About Qualys

Qualys, Inc. (NASDAQ: QLYS) is a pioneer and leading provider of disruptive cloud-based Security, Compliance and IT solutions with more than 10,000 subscription customers worldwide, including a majority of the Forbes Global 100 and Fortune 100. Qualys helps organizations streamline and automate their security and compliance solutions onto a single platform for greater agility, better business outcomes, and substantial cost savings. Qualys, Qualys VMDR® and the Qualys logo are proprietary trademarks of Qualys, Inc. All other products or names may be trademarks of their respective companies.

For more information, please visit qualys.com