eBook

Ready to run: A FinOps guide to Kubernetes cost optimization

CLOUDBOLT
The cloud ROI company®

Table of contents

Overview

Kubernetes is showing up in your cloud bill whether you're ready for it or not.

The problem is, Kubernetes doesn't behave like traditional infrastructure. It's dynamic, shared, and abstracted in ways that make cost allocation difficult. Tagging is unreliable, forecasting becomes guesswork, and standard FinOps practices—like tracking spend by service or holding teams accountable to budgets—start to break down.

That's because Kubernetes was built for engineering speed, not financial visibility.

Instead of fixed environments and provisioned resources, you're dealing with shared clusters, ephemeral workloads, and loosely coupled teams. Usage fluctuates. Ownership is unclear. And by the time costs show up on the invoice, it's often too late to do anything about them.

Worse, many teams make the mistake of treating the cluster as the cost unit. But clusters are shared infrastructure—the cloud bill may show you what they cost, but not what's driving it. **Unless you can surface cost at the workload level, you're not doing FinOps for Kubernetes. You're just watching spend happen.**

For FinOps teams, that creates two gaps:

A visibility gap—because Kubernetes costs are difficult to allocate, predict, or trace.

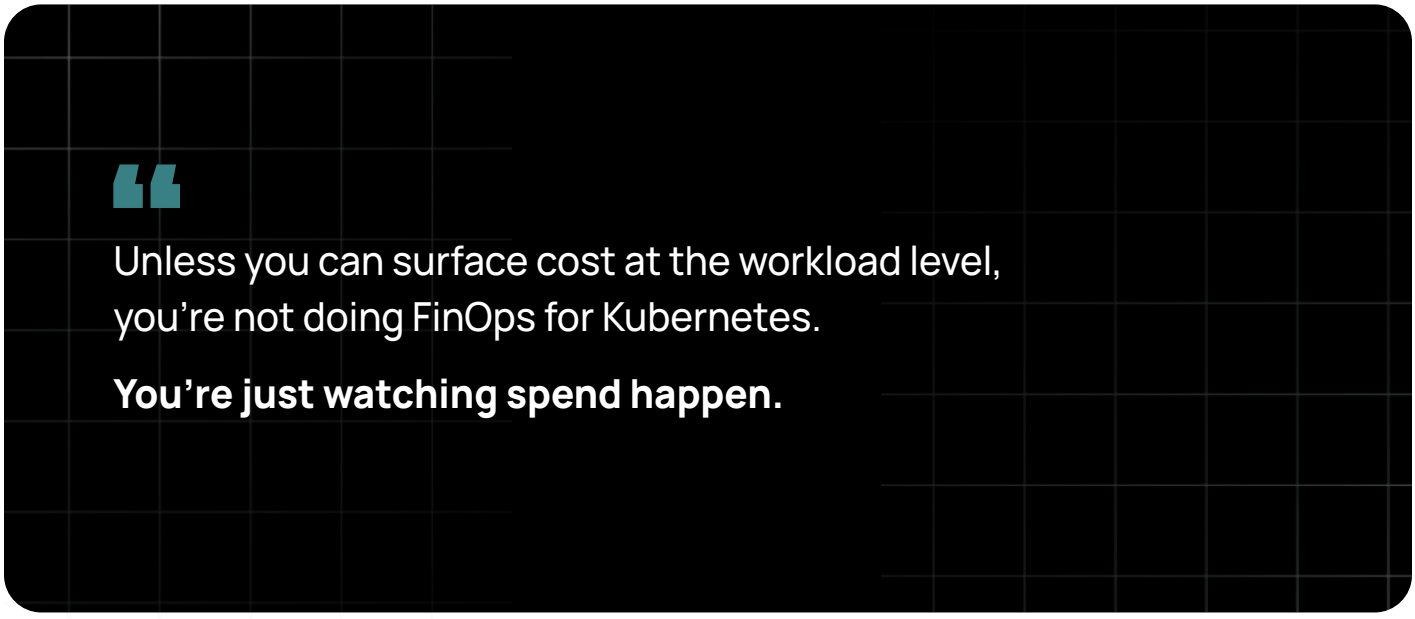
A credibility gap—because without understanding Kubernetes, it's hard to earn influence with the engineers who own it.

But that's exactly where the opportunity lies.

FinOps teams that engage earlier—and more strategically—can drive meaningful change. They can help engineering teams make smarter infrastructure decisions, lead cost-performance conversations, and introduce automation that scales governance without slowing down innovation.

This guide is built to help you do just that.

We'll walk you through that path—and how teams like yours are using it to become Run-Ready.



What you need to know to start the journey

You don't need to be a Kubernetes expert to make an impact—but you do need to understand the basics well enough to ask the right questions, interpret what you're seeing in cost reports, and collaborate meaningfully with engineering and platform teams.

What is Kubernetes?

Kubernetes is an open-source system for managing containerized applications. It automatically handles where and how workloads run across a cluster of servers—scaling them up or down based on demand, restarting them if they fail, and distributing them efficiently.

Think of it as an orchestrator. Developers define what they want to run (like a web app or database), and Kubernetes decides when, where, and how those containers get deployed and managed.

It's powerful—but abstract. And that abstraction is where cost visibility often starts to break down.

Core concepts (and why they matter for cost)

Understanding Kubernetes architecture isn't about memorizing every term. It's about knowing which components influence usage, ownership, and—ultimately—spend. Here are the key ones FinOps professionals should know:

Containers

The executable unit of software. Lightweight, fast, and ephemeral.

Pods

The smallest deployable unit in Kubernetes. A pod can run one or more containers and is what Kubernetes schedules onto a node.

Nodes

The virtual machines (or physical servers) where pods actually run. You pay for these directly in your cloud bill.

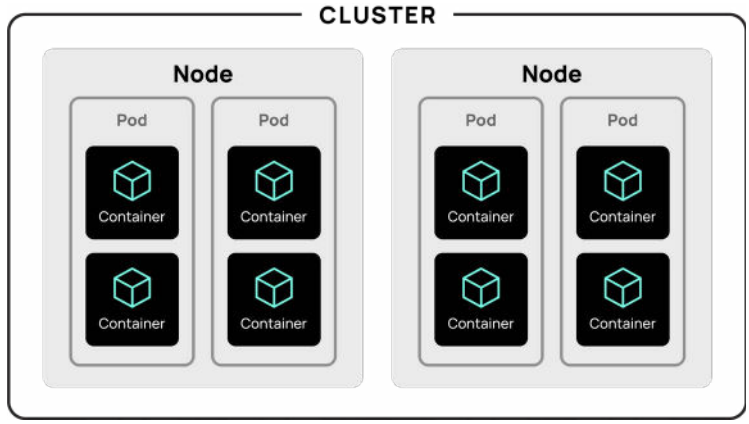


Fig 1. Cost accrue at the node level based on the resources consumed by scheduled pods

Clusters

A collection of nodes managed by Kubernetes. Most organizations run multiple applications in shared clusters—which makes cost attribution tricky.

Namespaces

A way to group workloads by team, project, or function. Often used for cost separation—but only works if workloads are properly labeled and maintained.

Workloads

A general term for the applications and services running on Kubernetes (e.g., deployments, jobs, daemonsets).

Requests & limits

When a pod is created, it defines how much CPU and memory it requests. Kubernetes uses that request to schedule the pod and reserve resources on a node.

Note: You pay for the underlying node, not the pod itself. But if requests are consistently inflated, those nodes end up underutilized—and that's where waste creeps in.

Autoscaling (HPA, VPA, Cluster Autoscaler)

Kubernetes can scale workloads up or down based on resource usage:

- **HPA (Horizontal Pod Autoscaler):** Adds/removes replicas
- **VPA (Vertical Pod Autoscaler):** Adjusts resource requests
- **Cluster Autoscaler:** Adds/removes underlying nodes

Note: Not all teams configure this properly—and "we use autoscaling" doesn't always mean cost-efficient.

Shared vs. dedicated clusters: Why it matters

Most organizations don't run one app per cluster—they run dozens (sometimes hundreds) of workloads in the same environment. This is efficient from an engineering perspective, but a challenge for FinOps.

Without clear boundaries—like namespaces, team labels, or workload ownership—cost attribution becomes nearly impossible. You may know the total cost of the cluster, but not which teams or products are driving it.

Because workloads are the true cost drivers, workload-level visibility is the only reliable way to operationalize FinOps in Kubernetes.This is why even mature organizations struggle with showback or chargeback in Kubernetes.

Why tagging and ownership break down

Tagging in Kubernetes is fundamentally different from tagging in IaaS environments like EC2 or Azure VMs.

- Tags are applied at the container or pod level, not the billing entity
- They're often inconsistent, overwritten, or missing entirely
- There's no native financial construct—Kubernetes was designed for availability and scalability, not financial accountability

As a result, cost visibility requires more than clean tags—it requires collaboration between FinOps and engineering, custom data pipelines, and often external tools.

Your journey to run-ready FinOps

Kubernetes cost management isn't something you solve with a tool or a one-time cleanup—it's a capability you build over time. Most FinOps teams start in Crawl, stuck in dashboards and disconnected reporting. But the ones that move forward aren't necessarily more advanced—they just have a clearer path.

This maturity model helps FinOps leaders understand where they are today, what's holding them back, and what it takes to evolve from visibility to continuous, intelligent optimization.

Crawl: Get visibility & build accountability

Most FinOps teams enter Kubernetes without the visibility or influence they need to make an impact. Costs are rising, but the environment feels like a black box. The goal in Crawl isn't to optimize—it's to orient. You're building the foundation: who's running what, what it costs, and who needs to be looped in.

Common signs you're here:

- You can't tie Kubernetes spend to teams, products, or services
- Tagging and labeling are inconsistent or nonexistent
- Engineering teams may be aware of overprovisioning—but don't see it as their responsibility to fix
- Costs are growing, but no one feels responsible

Steps to take in *Crawl*

Tag workloads by team, environment, and application

Partner with your platform team to define a labeling schema (e.g., team:, env:, app:) and enforce it at deployment. If you reply on manual tagging, coverage will fall short. Many teams use CloudBolt early on to apply tagging policies programmatically and normalize metadata across clusters, helping FinOps build reliable cost views while cutting down inconsistencies.

Build shared cost dashboards—even if they're rough

Start visualizing costs by namespace, cluster, or pod using your observability stack (e.g., Prometheus, Grafana, Datadog). You don't need perfect accuracy—your goal is directional insight. Focus on trends, not totals.

Identify and document workload ownership

Create a running inventory of namespaces and who's responsible for them. If no one owns it, it's a risk—and likely a cost sink. A shared Google Sheet is better than nothing while you're early in the journey.

Collaborate early with engineering on cost-performance decisions

Visibility isn't enough—you need buy-in. Start by partnering with engineers to understand how CPU and memory requests are set, and where overprovisioning may be happening. Use this as a launch point for performance-efficiency conversations, not just cost-cutting.

To build momentum:

- Align around shared metrics. Use KPIs that speak to both sides—like cost-per-transaction or infra cost as a % of revenue.
- Frame decisions in terms of performance. Engineers engage more when optimization improves system behavior, not just spend.
- Connect cost to technical choices. Show how things like autoscaler configs or node types shape the cloud bill.
- Empower FinOps champions. Identify engineers interested in optimization and equip them with data and tools—they'll help spread best practices internally.

SPOTLIGHT

Bridging the Engineering-FinOps Divide

One of the biggest challenges in Kubernetes cost optimization isn't technical—it's cultural. FinOps teams often speak a different language than platform engineers, which creates friction when trying to drive financial accountability.

The reality is that engineers care about costs too—but they prioritize different metrics:

Engineering Teams

Engineering teams value stability, performance, and time-to-market.

FinOps Teams

FinOps teams focus on utilization, efficiency, and budget alignment.

The most successful organizations don't force one perspective on the other—**they create a common language and shared incentives.**

Walk: Start rightsizing & introduce automation

Once visibility and ownership are in place, the next step is efficiency. FinOps teams in the *Walk* phase begin to identify waste—like over provisioned resources and idle workloads—and partner with engineering to reduce it. At this stage, optimization is still mostly manual, but the insights are real, and momentum is building.

Common signs you're here:

- You've surfaced inefficiencies, but fixes require manual effort
- Engineering is receptive to optimization conversations
- You're tracking cost trends but not automating changes
- Conversations around cost-performance tradeoffs are just beginning

Steps to take in *Walk*

Compare requested vs. actual usage

Use tools like Datadog, Prometheus, or your cloud-native observability stack to see how much CPU/memory is used vs. requested. Some teams begin using CloudBolt here to automate detection of over-requested workloads and surface actionable sizing opportunities without relying on manual reviews.

Identify idle or underutilized workloads

Look for pods or services that are consistently below 10% CPU/memory utilization or haven't been touched in weeks. Flag them for deprecation, scaling down, or scheduling to shut off during non-peak hours.

Automate waste detection and generate optimization recommendations

Use scripts, policies, or tools that automatically flag anomalies—like pods consistently using <25% of requested CPU—or alert you when resources exceed thresholds. Even if the action isn't automated yet, automation in detection saves hours.

Review and refine cloud commitments

Compare your Reserved Instances or Savings Plans to your real-time Kubernetes node consumption. If workloads are shifting or scaling in ways that make long-term commitments risky, adjust accordingly or shift strategy toward more flexible pricing. CloudBolt helps FinOps teams align usage patterns with cloud commitments by correlating resource consumption across Kubernetes and IaaS layers—avoiding costly mismatch and underutilization.

Introduce showback reporting

Start sending regular usage summaries to teams—by namespace, product, or environment. This doesn't have to be perfect or tied to actual budgets yet. The goal is awareness, not enforcement. Use visuals and clear summaries to encourage ownership.

SPOTLIGHT

Turning quick wins into long-term momentum

Once FinOps teams begin rightsizing and surfacing savings opportunities, the temptation is to stop there. But quick wins alone won't sustain optimization.

To turn Walk into a true stepping stone toward maturity:

- Build regular syncs with engineering into your process
- Introduce FinOps metrics into sprint retros or platform reviews
- Track time-to-action from recommendation to implementation

This shift transforms optimization from a series of one-off tasks into a repeatable motion. And that's what separates *Walk* from *Run*.

Run: Optimize continuously & align with business goals

In the Run phase, FinOps and engineering aligned—and cost optimization becomes continuous. Decisions are driven by real-time data, backed by machine learning, and enforced through policy. Manual cleanups are replaced with proactive governance. Optimization isn't just about saving money—it's about scaling with confidence.

Common signs you're here:

- You can forecast Kubernetes costs with confidence
- Engineering trusts and acts on optimization insights
- Guardrails and cleanup actions are automated
- Optimization decisions are tied to product, finance, and SLO priorities

Steps to take in *Run*

Enable continuous workload optimization with ML

Manual tuning simply can't keep up with the pace and scale of Kubernetes. FinOps teams in Run rely on machine learning to continuously adjust resource requests based on actual usage patterns. CloudBolt, with StormForge's ML-powered optimization built in, continuously tunes resources at the workload level—not just the node level—ensuring right-sizing stays aligned with live demand.

Tie optimization decisions to business goals

Move beyond "cost cutting." Align optimization with SLOs, product timelines, and revenue targets. For example, rightsize staging environments aggressively while maintaining headroom for revenue-critical services in prod.

Enforce cost policies with automation

As your Kubernetes footprint scales, governance needs to be proactive—not reactive. Define and apply policies across environments to auto-delete idle workloads, cap oversized requests, enforce labeling, and block noncompliant configurations. CloudBolt supports this by embedding guardrails directly into the platform layer, so cost control happens without constant manual oversight.

Model cost scenarios and forecast usage

Forecasting becomes strategic at this stage. Whether you're planning a rollout, scaling an environment, or shifting architecture, FinOps needs to model cost impact in advance. CloudBolt enables real-time scenario modeling tied to actual usage data, helping FinOps teams predict how infrastructure decisions will affect budgets—before costs land on the invoice.

Operationalize cost-performance reviews

Make optimization a recurring part of sprint cycles, platform syncs, or QBRs. Review ML recommendations, policy enforcement logs, and forecast variances with engineering—so cost becomes part of how teams measure performance.

SPOTLIGHT

How to keep *Run* from slipping backward

Getting to Run is an achievement. But staying there requires discipline. Over time, teams change. Priorities shift. Environments drift. The practices that got you to *Run* don't maintain themselves.

To hold the line:

- Build optimization into dev workflows
- Set alerting thresholds when automation is bypassed or policies aren't applied
- Regularly audit savings and reinvest learnings into platform improvements

Appendix: Your 90-day run-ready roadmap

Whether you're just starting in *Crawl* or building momentum in Walk, operationalizing Kubernetes cost optimization takes more than insight—it takes execution. That's where this 90-day roadmap comes in.

Use it to build your foundation, find quick wins, and start scaling the practices and policies that will carry you into *Run*.

Day 1-30: Build your baseline

- Inventory clusters and map each one to its business purpose
- Implement basic labeling standards—even if they're manual at first
- Stand up a simple dashboard to track cost trends by namespace
- Identify stakeholders across platform and application teams

Days 31–60: Find quick wins

- Right-size your top 5 over-provisioned workloads
- Start implementing automated recommendations for CPU and memory requests
- Create your first showback report and share it with engineering
- Establish a regular sync cadence with platform teams

Days 61–90: Scale and automate

- Enforce automated policies for cost governance (e.g., tagging, idle cleanup)
- Forecast costs based on historical usage
- Integrate cost optimization reviews into sprint planning or QBRs
- Document cost savings and performance gains to show momentum

Want help putting this into practice?

CloudBolt helps FinOps and engineering teams enforce policies, optimize workloads with ML, and align cost with business outcomes—without the manual grind.

Request a CloudBolt demo today to accelerate your journey to Run-Ready Kubernetes › › ›

Ready to run: Where continuous optimization begins

No matter where your team is today—just starting to build visibility, actively rightsizing workloads, or scaling automation across clusters—the goal is the same: to move faster, with more control and less waste.

And that's what continuous optimization is all about.

Kubernetes won't wait for manual reviews and spreadsheets. Workloads change daily. Teams evolve. Costs fluctuate. What was right-sized yesterday might already be misaligned today.

The most successful FinOps teams don't just optimize once—they operationalize it.

They embed automation into their workflows. They enforce policies through code. They tie optimization to product goals and business metrics. And they rely on intelligent, real-time tools that turn cloud cost management from a reactive process into a proactive advantage.

That's where CloudBolt comes in.

CloudBolt combines policy-based governance, dynamic cost visibility, and ML-driven workload optimization—powered by StormForge—to give FinOps and engineering teams a shared platform for smarter decisions. From early visibility to full-scale enforcement, it's designed to meet you where you are and grow with you.

Whether you're still in *Crawl* or pushing into *Run*, CloudBolt helps make continuous optimization your new normal.

Run-Ready isn't a destination. It's a mindset. And with the right foundation, it becomes your default operating model.

Learn more

CLOUDBOLT
The cloud ROI company®