

EXECUTIVE PAPER · INFRASTRUCTURE

Kubernetes Is a Commitment, Not a Default

Kubernetes runs most of the industry's containers and asks for a platform team, three upgrades a year, and a steady stream of component migrations in return. CTOs and CFOs should adopt it when the workload justifies that bill, and budget for it as a standing cost when they do.

AUTHOR

Chris Watkins

SERVICE

Infrastructure

FOR

CTO, CFO, CEO · 11 min read

AT A GLANCE

The problem. Kubernetes has become the assumed answer for running software: 80% of organizations in the 2024 CNCF survey run it in production. Each minor version gets about a year of community patches, so a cluster needs a planned upgrade roughly every four months. Missing the window costs money on managed services and, from 2027, can mean the provider upgrades the cluster on its own schedule.

Why it persists. Kubernetes is chosen as a technical standard and funded as a one-time project. The platform team, the upgrade cadence and the add-on components are recurring work that rarely appears in the business case. Smaller teams adopt it because larger companies did.

The recommendation. Use Kubernetes when you run enough services, with enough engineers and enough releases, that a shared platform pays for itself. Below that line, use managed container and serverless runtimes. If you adopt it, fund a platform team, keep every cluster on a supported version, and track each add-on's lifecycle as carefully as the cluster's.

The Decision in Front of You

Kubernetes is open-source software that schedules containers across a fleet of machines, restarts them when they fail, and gives teams a common interface for deploying software. It is the most widely used way to run containers. The CNCF's 2024 survey of 750 members of its community found 80% of organizations running it in production, up from 66% in 2023 [1][13]. The same survey found cultural and operational change had overtaken security as the top challenge [1].

The decision is whether your organization should be one of them. Kubernetes gives a large engineering organization one consistent platform for many services. It also carries a fixed operating cost: people who understand it, upgrades on the project's schedule, and a set of add-on components that each have their own maintainers, vulnerabilities and retirement dates.

Those costs do not shrink with the size of the application. A company running six services pays most of the same platform overhead as one running sixty. The question is whether you have enough work on the platform to spread that cost.

What's Happening

The release clock does not stop

The Kubernetes project ships a new minor version about every four months, and each one receives roughly a year of patch support [2][4]. Only the three most recent minor versions are maintained [2]. A cluster that is not upgraded at least once a year falls out of security support.

Upgrades are more than a version number. Kubernetes removes beta APIs on a published schedule: a beta API is deprecated within nine months or three releases of its introduction and stops being served nine months or three releases after that [3]. Any deployment file, chart or tool that still uses a removed API fails on the new version. Each upgrade is a small migration project.

Managed services charge for falling behind. Amazon EKS charges $0.10 per cluster per hour during the first 14 months of a version and $0.60 per cluster per hour during the following 12 months of extended support [5]. Azure AKS offers a second year of support only on its Premium tier [4]. From June 1, 2027, AKS will upgrade clusters left on unsupported versions after a 60-day grace period, and that upgrade will not follow the customer's maintenance window, will not stop for deprecated APIs, and can override safeguards that keep applications running during node drains [4].

Teams reserve far more than they use

Datadog's telemetry from thousands of customers shows most workloads across Kubernetes, AWS Fargate, Google Cloud Run, Azure Container Apps and AWS Lambda use less than half of their requested memory and less than a quarter of their requested CPU [7]. Its 2023 report found over 65% of Kubernetes workloads using less than half their requested CPU and memory, and fewer than 1% of Kubernetes organizations using the Vertical Pod Autoscaler that adjusts those requests [6]. Datadog notes its customers skew toward heavier cloud users, so the sample is not the whole market [6][7].

The same 2023 data showed version lag. The most common Kubernetes release among Datadog customers was about 16 months old [6].

Simpler runtimes are already mainstream

Managed runtimes that hide the cluster are now common. Datadog's 2025 data shows Google Cloud Run used by 70% of its Google Cloud customers, AWS Lambda by 65% of AWS customers, and Azure App Service by 56% of Azure customers [7]. A third of organizations using serverless functions run no container orchestration at all in that cloud [7]. Its 2023 report found 46% of container organizations already running serverless containers, up from 31% two years earlier [6].

The ecosystem's components retire too

Ingress NGINX, 2025 to 2026. Ingress NGINX is the add-on that routed outside web traffic into a large share of Kubernetes clusters. The Kubernetes steering and security committees said about half of cloud native environments relied on it, citing Datadog research, and that one or two people had maintained it in their spare time for years [9]. On November 11, 2025 the project announced its retirement: best-effort maintenance until March 2026, then no releases, bug fixes or security patches [8]. The committees warned that none of the alternatives are drop-in replacements and that existing installations would keep working, so a later compromise might be the first sign of trouble [9]. Users had about four months between the announcement and the end of maintenance.

IngressNightmare, 2025. The same component carried the year's most serious Kubernetes vulnerability. In March 2025 Wiz disclosed a chain of flaws, including CVE-2025-1974, rated 9.8 out of 10, that let an unauthenticated attacker run code inside the controller and read every secret in every namespace, which can lead to full cluster takeover [10]. Wiz estimated about 43% of cloud environments were vulnerable and found more than 6,500 clusters exposing the vulnerable component to the internet [10].

Gitpod, 2024. In October 2024 Gitpod, a cloud development environment company with about 1.5 million users, published why it was leaving Kubernetes after six years [11]. Its workload, interactive developer machines that need root access, bursty CPU and fast startup, fought the platform at every layer, and the team built extensive custom machinery for CPU, storage, networking and security before replacing it with its own system [11]. The authors were explicit that the lesson is about fit, writing that for application workloads Kubernetes "continues to be a fine choice" [11].

Platforms help large organizations most

Google's 2024 DORA research found internal developer platforms raise developer productivity, are more common in larger organizations, and often cause a temporary drop in performance before the gains appear [12]. The best results came from platforms run as products, designed around their users [12]. A platform needs enough users to be worth building and a team to run it as a product.

Why the Usual Approach Fails

Kubernetes is often chosen as a résumé standard. Engineers want experience with it, hiring managers ask for it, and architecture reviews treat it as the safe choice. The business case covers the migration and stops there.

The recurring work is invisible at approval. Three upgrades a year per cluster, API removals, add-on lifecycles, node image updates, certificate rotation, and on-call for the control plane all land on the same few people. When those people leave, the knowledge goes with them.

Managed Kubernetes reduces the work without removing it. EKS, AKS and GKE run the control plane. The customer still owns the version, the add-ons, the workload configuration and the upgrade timing, and pays more for falling behind [4][5].

Add-ons are treated as part of the product. Ingress controllers, certificate managers, service meshes and monitoring agents come from separate projects with separate maintainers. Ingress NGINX showed that a component used by half the ecosystem can be retired with months of notice [8][9].

The Framework: Three Thresholds

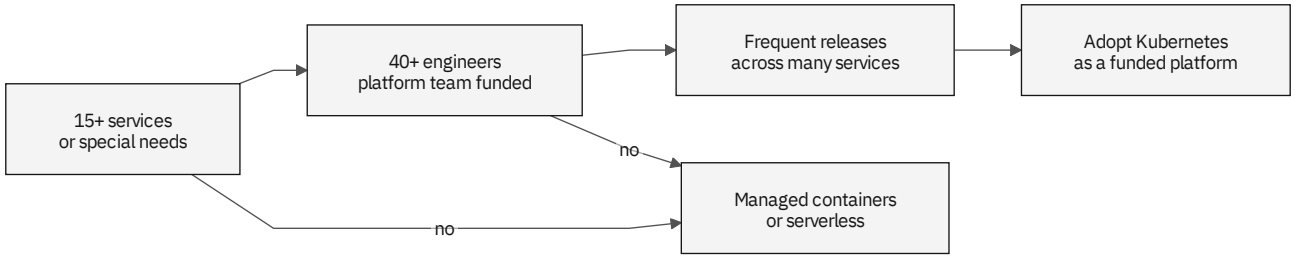
Adopt Kubernetes when all three of these hold. If one fails, a managed container or serverless runtime is the better default.

Workload count. Enough independently deployed services that a shared scheduling and deployment layer saves real effort. In the author's experience that starts around 15 to 20 services, or fewer when several have unusual needs such as GPUs, stateful workloads, or strict placement across regions and providers.

Team size. Enough engineers that two or three of them can own the platform full time without starving product work. As a rough guide, an engineering organization of 40 or more can carry that team. Below that, the platform competes with features for the same people.

Release frequency. Enough deployments that consistent, automated rollout matters. Teams shipping several times a day across many services gain the most. Teams releasing monthly gain little.

Exhibit 1. The adoption decision: one failed threshold points to a managed runtime



Source: author's framework. Thresholds are the author's judgment from production experience and should be adjusted to your workload; platform effects drawn from DORA 2024 [12].

Portability alone does not pass the test. Kubernetes runs on every major cloud and on-premises, but applications still depend on each provider's databases, identity and storage. If multi-cloud is a requirement, price it separately.

What the standing cost looks like

The following illustrative model shows a mid-size company running six managed clusters (production, staging and development in two regions).

Exhibit 2. Illustrative annual cost of operating Kubernetes, separate from compute

Item	Assumption	Annual cost
Platform team	3 engineers at $220,000 fully loaded	$660,000
Managed control plane, standard support	6 clusters at $0.10 per hour [5]	$5,256
Subtotal, kept current		$665,256
Extended support if every cluster falls behind	6 clusters at an extra $0.50 per hour [5]	$26,280
Total if versions lag		$691,536

Source: author's illustrative estimate. EKS list prices [5]; 8,760 hours per year. Salary is an assumption; substitute local figures. Excludes worker compute, add-on licenses, monitoring and upgrade project time.

The control plane fee is small. People are the cost, and they do not scale down when the workload is small. A company with six services pays nearly the same $660,000 as one with sixty.

Costs, Risks, and Tradeoffs

Exhibit 3. Kubernetes against managed runtimes

Area	Kubernetes	Managed containers or serverless
Operating staff	Standing platform team	Little or none beyond application teams
Upgrades	About three minor versions a year, with API removals [2][3]	Provider-managed; runtime versions still retire
Flexibility	Broad control of scheduling, networking and placement	Limited to the provider's options
Unit cost at scale	Lower when clusters are well packed	Higher per unit for steady, heavy load
Lock-in	Portable control layer; data services still provider-specific	Tied to one provider's runtime
Component risk	Add-ons retire on their own schedules [8]	Fewer moving parts to track
When this is wrong	Small teams pay full platform cost for few services	Large estates with GPUs, stateful systems or strict placement outgrow the provider's limits

Source: author's analysis, with cited provider and project documentation.

Managed runtimes have their own lifecycle. Language runtimes on Lambda or App Service retire on the provider's schedule, and request limits, cold starts and per-request pricing can bite at high steady load. The difference is the size of the team needed to keep up.

How to Act

Exhibit 4. A one-year roadmap

Window	Actions	Owner	Milestone
First 30 days	List every cluster, its Kubernetes version, the date that version leaves support, and every add-on with its version and maintainer. Score current and planned workloads against the three thresholds.	CTO with platform lead	Inventory complete; every cluster's support end date known
First 30 days	Check for Ingress NGINX and any other retired add-on; schedule replacements.	Platform lead	Zero retired add-ons without a dated migration plan
First 90 days	For teams below the thresholds, move new services to a managed runtime by default. For teams above them, fund the platform team as a permanent budget line.	CTO with CFO	Written runtime standard; platform team in the operating budget
First 90 days	Put the upgrade calendar on a fixed cadence with a test cluster that runs each version first. Scan manifests for removed APIs before each upgrade.	Platform lead	No cluster in paid extended support; upgrade runbook rehearsed
First 365 days	Report quarterly on cluster versions, add-on status, usage against reserved capacity, and platform cost per service. Retire clusters with too few workloads.	Platform lead reporting to CTO and CFO	Every cluster on a supported version all year; platform cost per service tracked

Source: author's recommendation.

Questions to Ask Your Team

1. How many services run on our clusters, and how many engineers own the platform full time?
2. When does each cluster's Kubernetes version leave support, and who has the upgrade on their calendar?
3. Are we paying extended support fees on any cluster today?
4. Which add-ons do we run, who maintains each one, and which have announced retirement?
5. What share of the CPU and memory we reserve do our workloads use?
6. Which of our services would run more cheaply on a managed container or serverless runtime?
7. If our two most experienced platform engineers left, who could perform the next upgrade?

Closing

Kubernetes is a sound platform for organizations with many services, many engineers and frequent releases. For everyone else, managed runtimes deliver most of the benefit without a standing team. Score your workload against the three thresholds before adopting. If you adopt, carry the platform team and the upgrade cycle in the operating budget every year, the way you would any other system the business depends on.

Sources

[1] Cloud Native Computing Foundation. "CNCF Research Reveals How Cloud Native Technology is Reshaping Global Business and Innovation." Press release on the 2024 Cloud Native Survey. 2025. https://www.cncf.io/announcements/2025/04/01/cncf-research-reveals-how-cloud-native-technology-is-reshaping-global-business-and-innovation/.

[2] The Kubernetes Authors. "Releases." kubernetes.io. 2026. https://kubernetes.io/releases/.

[3] The Kubernetes Authors. "Kubernetes Deprecation Policy." kubernetes.io. https://kubernetes.io/docs/reference/using-api/deprecation-policy/.

[4] Microsoft. "Long-Term Support for Azure Kubernetes Service (AKS) Versions." Microsoft Learn. 2026. https://learn.microsoft.com/en-us/azure/aks/long-term-support.

[5] Amazon Web Services. "Amazon EKS Pricing." 2026. https://aws.amazon.com/eks/pricing/.

[6] Datadog. "2023 Container Report." 2023. https://www.datadoghq.com/container-report/.

[7] Datadog. "2025 State of Containers and Serverless." 2025. https://www.datadoghq.com/state-of-containers-and-serverless/.

[8] Sable, T. "Ingress NGINX Retirement: What You Need to Know." Kubernetes Blog. 2025. https://kubernetes.io/blog/2025/11/11/ingress-nginx-retirement/.

[9] Kubernetes Steering Committee and Security Response Committee. "Ingress NGINX: Statement from the Kubernetes Steering and Security Response Committees." Kubernetes Blog. 2026. https://kubernetes.io/blog/2026/01/29/ingress-nginx-statement/.

[10] Wiz Research. "IngressNightmare: CVE-2025-1974 - 9.8 Critical Unauthenticated Remote Code Execution Vulnerabilities in Ingress NGINX." 2025. https://www.wiz.io/blog/ingress-nginx-kubernetes-vulnerabilities.

[11] Weichel, C., and de Brito Fontes, A. "We're leaving Kubernetes." Gitpod (now Ona). 2024. https://ona.com/stories/we-are-leaving-kubernetes.

[12] Google Cloud. "Announcing the 2024 DORA report." 2024. https://cloud.google.com/blog/products/devops-sre/announcing-the-2024-dora-report.

[13] Cloud Native Computing Foundation. "CNCF Annual Survey 2024." 2025. https://www.cncf.io/reports/cncf-annual-survey-2024/.

The views in this paper are the author's own and do not represent any employer. Scenarios are hypothetical. This paper is not legal, financial, or compliance advice.