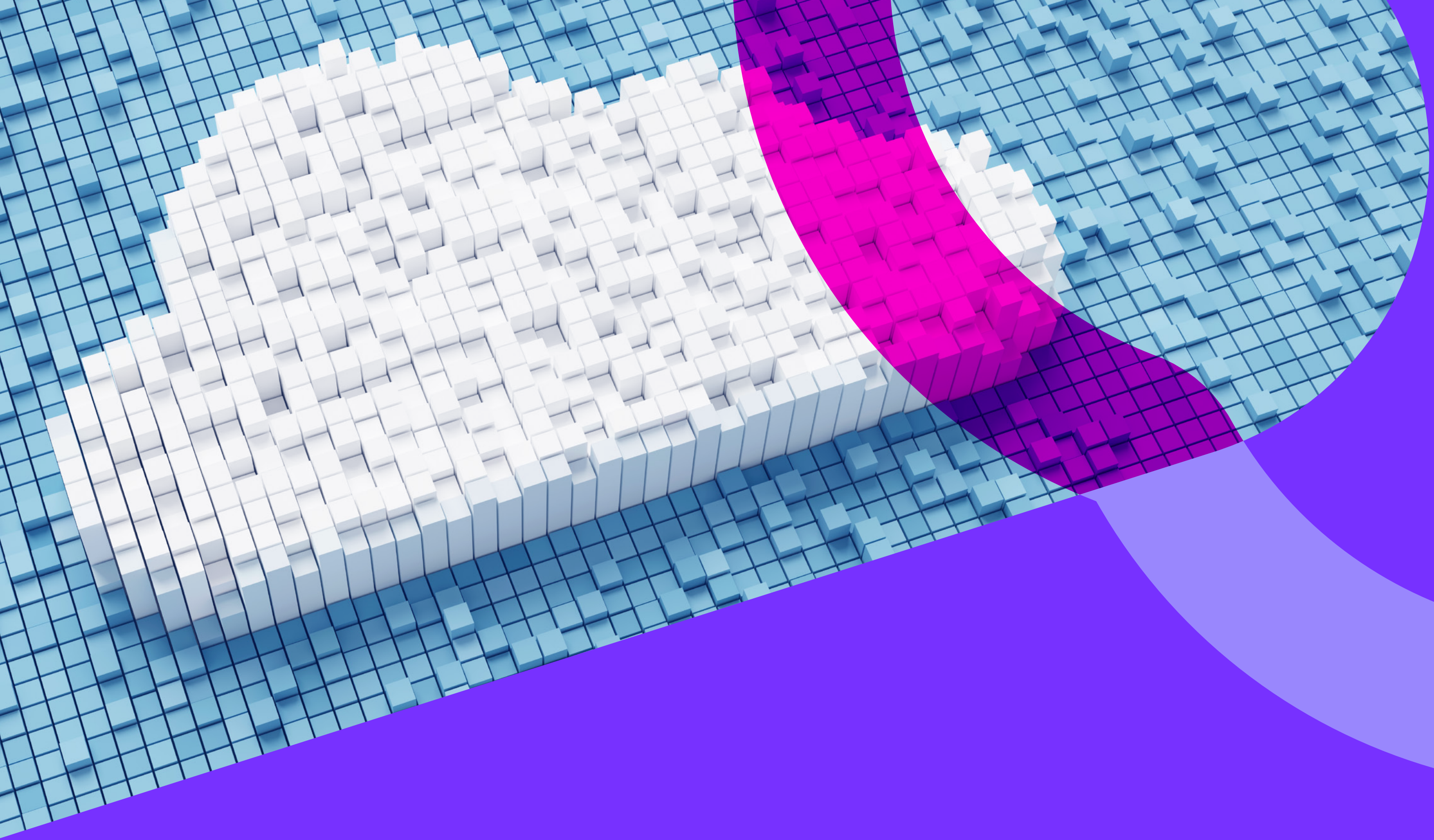


Guide

Rightsizing Kubernetes compute infrastructure

Essential tools & practical guidance for eliminating waste & maximizing resource utilization for cloud-native applications

Spot
by NetApp

Table of contents

Introduction

One of the main drivers of surging container adoption over recent years has been the clear efficiency advantages that containers offer. Yet 66% of respondents in our 2023 State of CloudOps report identified cost management as a key area for improvement.

So why are so many companies failing to see the promised cost benefits of containerization?

One of the main culprits is the misallocation — and resulting waste — of compute resources. It has been estimated that **nearly half of all containers are using less than 30% of requested CPU and memory*.**

Estimating exactly how much CPU, memory, disk and network your pods are going to require is one of the trickiest areas to get right during deployment of cloud-native applications. At worst, this uncertainty can lead to underprovisioning of resources resulting in serious application performance issues, but most commonly developers play it safe by overprovisioning, which results in wasted resources and spiraling costs.

Fortunately, in the cloud you are not stuck with physical servers: you can always rightsize your compute infrastructure, even mid-flight. **But one of the challenges is in ensuring you have the right metrics to know when it's appropriate to rightsize.** Once you have the relevant data, you should make investing time and resources into rightsizing efforts a priority.

But while moment-in-time rightsizing efforts can be effective, the ultimate goal should be to build rightsizing automation into your deployment process. There are several established open source and commercial tools that can help, but we found only 45% of DevOps and platform teams are using automation to optimize resource utilization (State of CloudOps 2023).

1.5%

of organizations are maximizing cluster utilization

Only a handful of engineering teams optimize their Kubernetes clusters to maximize resource utilization.

*Annual container report, Datadog, 2020

In this guide we will delve deeper into the challenges of estimating resource needs, learn how to identify which workloads require attention, and access the metrics needed to make accurate rightsizing decisions. Finally, we'll explore the options available for automating rightsizing and ensuring resource utilization is continuously optimized.

Pod resource requests: fine-tuning efficiency in Kubernetes

Challenges in estimating pod resource requests

Developers configuring pod resource requests can provide an estimate by trial and error or run simulations with a test deployment. However, the time and effort that development teams need to invest to provide accurate metrics of their application's CPU and Memory consumption can be quite extensive and often ineffective.

Even after developers dedicate time and resources to establish an accurate measurement, test simulation metrics will almost always differ from actual production usage. Moreover, production resource consumption invariably changes over time, exacerbating deviations from the initial estimate.

These challenges lead many engineering teams to adopt a t-shirt sizing approach to selecting instances - small, medium, large and extra-large - with engineers often rounding up and greatly overprovisioning to be on the safe side.

Spot Ocean

Accurate pod resource requests are critical to driving significant cost savings. Spot Ocean monitors workload utilization in real time, providing accurate recommendations for rightsizing based on actual cluster CPU and Memory usage.

Supersizing really does save you money!

Example: Pod requires 6,500mb of memory and 1.5vCPUs

	Pod count	Pod M	Instance M	Memory %	Pod vCPU	Instance vCPU	CPU %	Cost per pod (OD hourly)
m5.large	1	6,500	8,000	81.25%	1.5	2	75%	$0.096
m5.xlarge	2	13,000	16,000	81.25%	3	4	75%	$0.096
m5.2xlarge	5	31,500	32,000	98.44%	7.5	8	93.75%	$0.077

In this example, knowing the precise memory and CPU requirements of the pod enables you to optimize node resource allocation and **reduce costs by 20%.**

Where to start your rightsizing efforts

It's tempting to start your rightsizing project off with your most expensive applications and environments where your upfront investment may yield higher savings, especially as monitoring costs are typically linked to the number of assets and metrics being monitored.

However, often the quickest way to show tangible results is to start off with less resource intensive workloads that you can easily identify as overprovisioned. Most likely, this low hanging fruit is already on your engineers' radar and can prove to be a good testing ground for your first rightsizing project. From that initial success your organization will be more willing to allocate time and resources to further rightsizing efforts.

Spot Ocean

To obtain the most accurate assessment of your environment, regardless of which one you choose to focus on, you should monitor all instances running in it, not just a subset. The Spot Ocean controller queries Metrics Server every five minutes, aggregating data using maximum and mean pod utilization values to shape its recommendations.

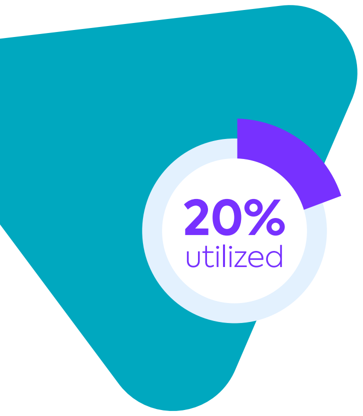
Insights into compute utilization allows proper rightsizing of container requirements.

Gaining visibility to monitor performance

To ascertain whether your workloads can be moved to machines with fewer resources requires, at a minimum, visibility into their CPU and memory utilization. Metrics such as disk and network consumption can also play an important role in determining whether to increase or decrease resource requirements. Let's take a look at the various options.

For AWS users, metrics collection can be done via Amazon CloudWatch Container Insights, enabling you to collect all relevant metrics for your ECS workloads as well as for EKS and Kubernetes. Other options such as Metrics Server, cAdvisor and Prometheus can be used to understand how your pods and containers utilize CPU and memory. There are a number of considerations to take into account when deciding which monitoring tool is right for your organization, including cost, ease-of-use, and multi-cloud capabilities.

With your cluster utilization insights in hand, you can now rightsize your tasks and pods while your autoscaling solution handles the underlying infrastructure.



Downsizing candidate!
Seriously consider moving down a size when utilization is under 20%.

Criteria for downsizing and testing

After observing workload resource utilization, anything less than 20% utilized is typically an excellent candidate for downsizing. Even if utilization is higher, but still nowhere near max capacity, it's recommended that you downsize.

It is essential to test workload performance on rightsized instances to make sure there is no performance degradation. If you have both production and testing/ QA environments for the same application workload, first rightsize on the lower environments and then use load testing tools such as Jmeter, Gatling or others to evaluate performance. Once that's done, ongoing monitoring of your application's behavior in your production environment should be done with APM tools such as New Relic and AppDynamics to ensure that the rightsizing does not impact performance at any point.

Rightsizing tip

Sometimes rightsizing can yield **dramatic results** as seen here with the c5n.large.

It provides all the networking capacity needed for communication-intensive workloads without the excess CPU and Memory baggage that the c4.8xlarge has and for **94% cost savings!**

Instance	Price	vCPU	Memory (GiB)	Instance storage	Network performance
c4.8xlarge	$1.591 per hour	36	60	• EBS only • Dedicated EBS • Bandwidth (Mbps): 4,000	10 Gigabit
c5n.large	$0.108 per hour	2	5.25	• EBS only • EBS Bandwidth (Mbps): Up to 4,750	Up to 25 Gigabit

Embracing automation for continuous rightsizing

Even with properly defined resource requirements there is still a risk of misconfiguration, unless you have implemented automated rightsizing. Someone or something (a CI/CD system for example) can revert any resource update, resulting in misconfigured resource requirements for your services or deployments. Additionally, production resource consumption usually changes over time causing a mismatch between requirements and actual resources.

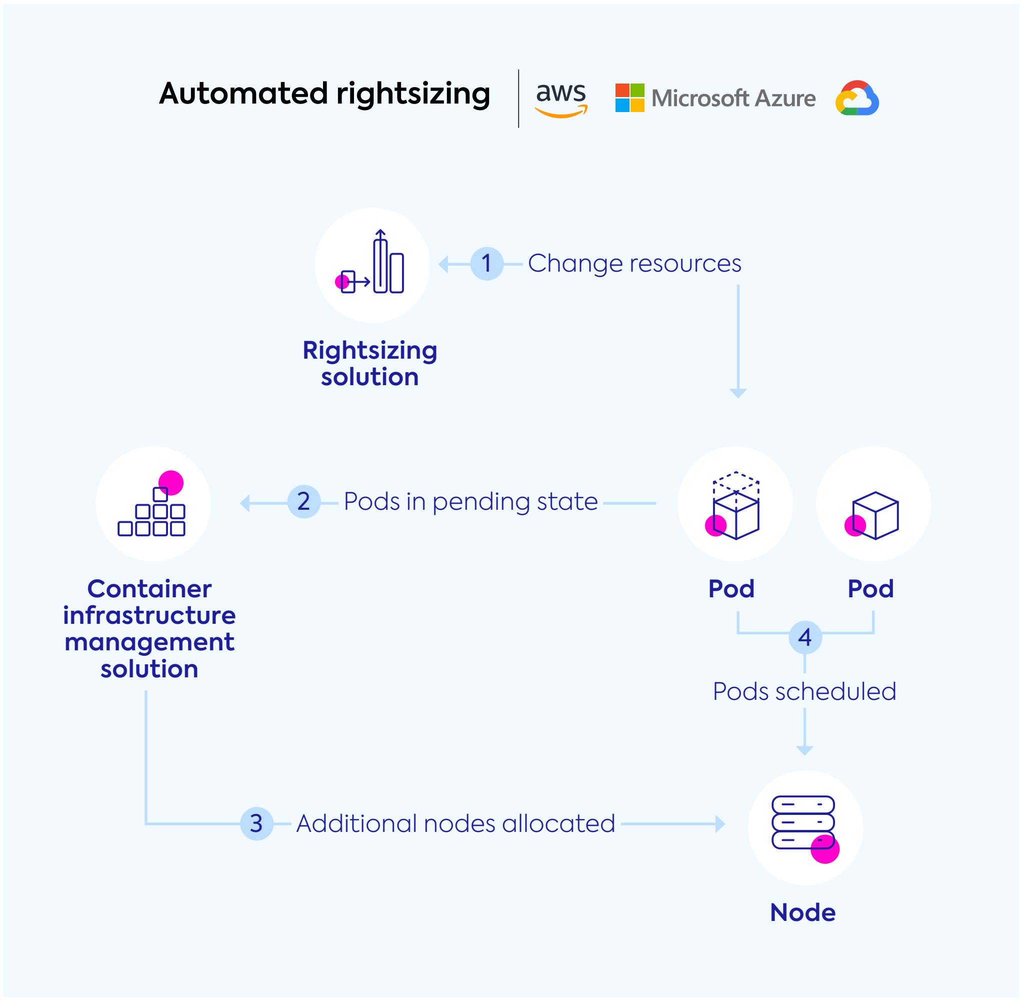
To ensure this doesn't happen there are several options for automating rightsizing. For a DIY approach, you can either implement automated rightsizing at the beginning of the CI process or insert it into your CD process.

For the beginning of the CI process, adding a resource update step to the existing automation of your Deployment Yaml generator will do the trick. This will result in a ready-to-apply Kubernetes Deployment object with the rightsized resource requirements configuration.

For the end of the CD process, an in-cluster mechanism to intercept any updates to the Kubernetes Deployment object will override any misconfigured resource requests and rightsize them on the fly.

Automation is key to CloudOps success!

Intelligent automation ensures your cloud infrastructure is always scalable, responsive and continuously optimized, while freeing DevOps teams from time-consuming management tasks.



Using Vertical Pod Autoscaler (VPA)

If you are looking for a more out-of-box solution that only requires minor configuration and will also dynamically assess whether your deployments are rightsized, there is Kubernetes' native VPA or Vertical Pod Autoscaler.

The VPA uses live data to set limits on container resources.

Most containers adhere more closely to their initial requests rather than to upper limit requests. As a result, the Kubernetes default scheduler overcommits a node's memory and CPU reservations. To deal with this, the VPA increases and decreases the requests made by pod containers to ensure actual usage is in line with available memory and CPU resources.

Some workloads can require short periods of high utilization. Increasing request limits by default entails wasting unutilized resources and limits the nodes that can run those workloads. Horizontal Pod Autoscaler (HPA) may help with this in some cases, but in other cases, the application may not easily support distribution of load across multiple instances.

A VPA deployment calculates target values by monitoring resource utilization, using its recommender component. Its updater component evicts pods that must be updated with new resource limits. Finally, the VPA admission controller overwrites the pod resource requests when they are created, using a mutating admission webhook.

Limitations of the VPA

Updating running pods is still experimental in VPA, and performance in large clusters remains untested. VPA reacts to most out-of-memory events, but not all, and the behavior of multiple VPA resources that match the same pod remains undefined. Finally, VPA recreates pods when updating pod resources, possibly on a different node. As a result, all running containers restart.

Best Practices for using the VPA

Two best practices for making efficient use of Vertical Pod Autoscaler:

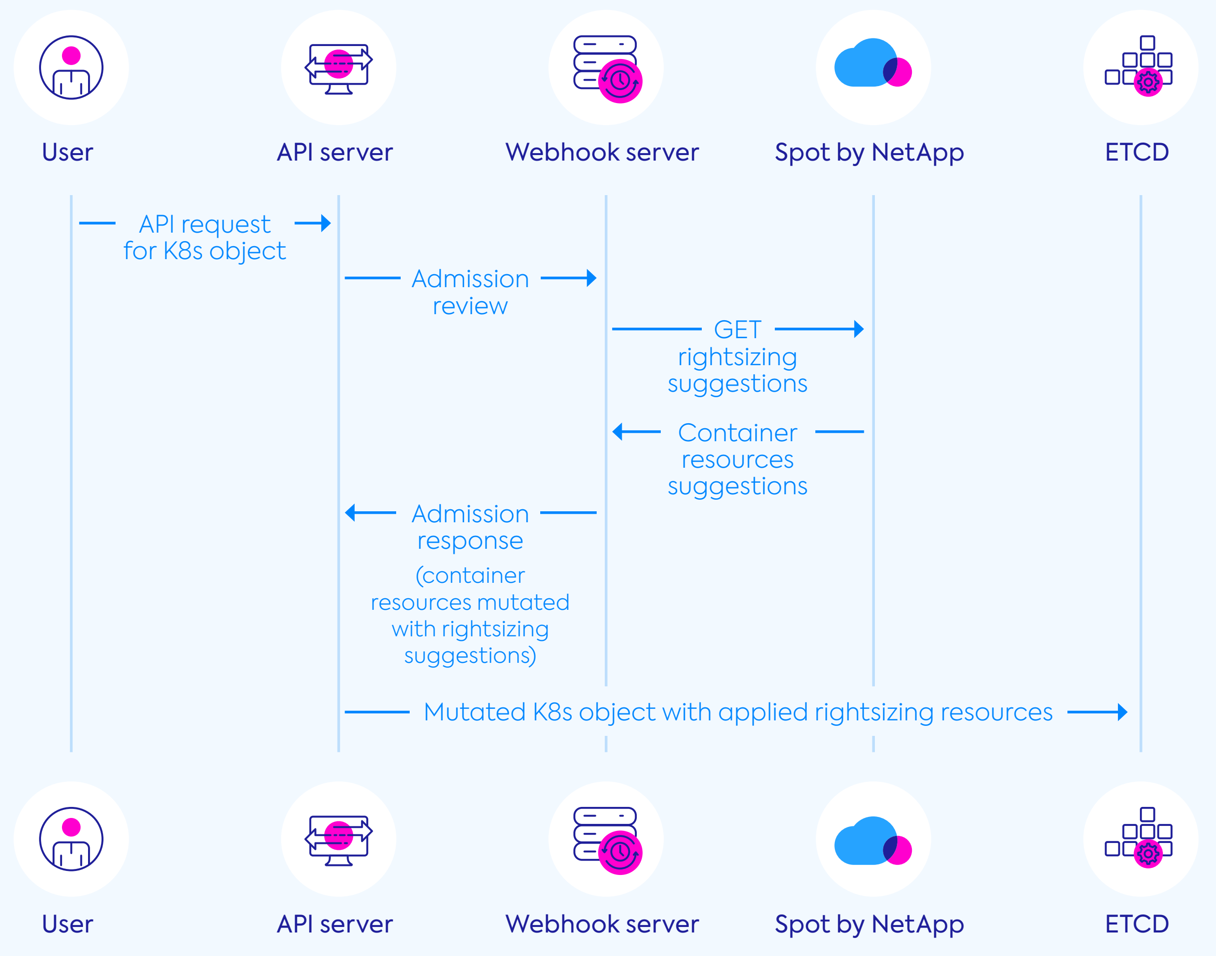
- **Avoid using HPA and VPA in tandem**—HPA and VPA are incompatible. Do not use them together for the same set of pods, unless you configure the HPA to use either custom or external metrics.

- **Use VPA together with Cluster Autoscaler**—VPA might occasionally recommend resource request values which exceed available resources. This can result in resource pressure and cause pods to go into a pending status. The Cluster Autoscaler can mitigate this behavior by spinning up new nodes in response to pending pods.

Automate and optimize CloudOps for Kubernetes with Spot Ocean

Spot Ocean is NetApp's market-leading multi-cloud container management solution. It offers comprehensive automation and optimization that reduces the complexity and overhead of running K8s at scale, freeing operations teams from manual cluster management and delivering highly scalable, available and cost-efficient infrastructure that always meets application demands.

Rightsizing automation during deployment


User
API server
Webhook server
Spot by NetApp
ETCD
API request for K8s object
Admission review
GET rightsizing suggestions
Container resources suggestions
Admission response (container resources mutated with rightsizing suggestions)
Mutated K8s object with applied rightsizing resources
User
API server
Webhook server
Spot by NetApp
ETCD

Automated rightsizing with Spot Ocean

For a completely turnkey solution Spot Ocean provides resource utilization analysis and rightsizing recommendations for all containerized workloads which can be implemented as part of the CI process or when a Deployment is being created on the cluster.

Based on the rightsized requirements, Ocean continuously manages the underlying nodes ensuring that you always have the optimal compute power needed by your cluster.

Learn more about Spot Ocean's automated rightsizing capabilities

- **Blog:** Kubernetes automatic rightsizing with Dynamic Admission Controller
- **Documentation:** Spot Ocean rightsizing
- **Video:** Watch an 8-minute demonstration of Spot Ocean

Spot Ocean

Spot Ocean simplifies infrastructure management for container orchestration tools. With robust, container-driven infrastructure auto-scaling and intelligent rightsizing for container resource requirements, engineers can code more, while operations can literally "set and forget" the underlying spot instance cluster.

>> Try Ocean for free!