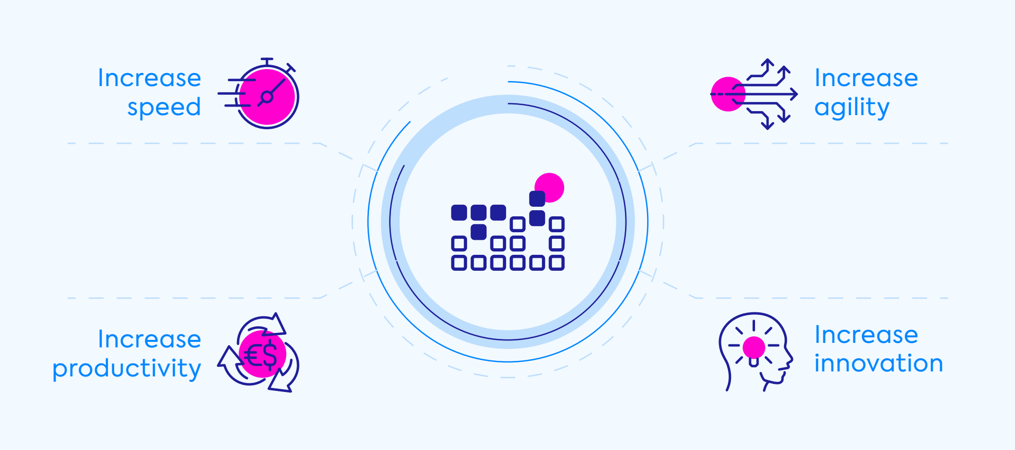
White Paper

Reducing container costs with Kubernetes 2020

Introduction

When you talk about cloud native tools, Kubernetes is one of the technologies that you'll hear the most about. An open source container orchestration tool created by Google, Kubernetes is playing a key role in many organizations' digital transformations.

Companies are increasingly shifting their systems and applications into the cloud with the hope of realizing its benefits — increased speed, agility, productivity and innovation. For many on this cloud journey, containerization presents a path toward achieving those goals, enabling IT teams to move fast, deploy software efficiently and scale to unprecedented levels. The introduction of Kubernetes has pushed the adoption of containers even further in recent years, with all the major cloud providers offering services that support Kubernetes workloads.



Somewhere along the way to achieving that speed and scale however, an increasing number of organizations have found themselves saddled with higher IT costs, despite the cloud's promise of potential cost savings. Driving these costs for many companies is the underlying cloud infrastructure that containerized applications are running on. Kubernetes and containers both add a layer of abstraction that can obstruct visibility, making it harder to use compute wisely and efficiently. As applications grow, they need more resources, but new complexities introduced by containers can make proper infrastructure management challenging.

3 out of 4 companies that implement containers, are using Kubernetes for container orchestration[1]

– Cloud Native Computing Foundation

[1] https://www.cncf.io/blog/2020/03/04/2019-cncf-survey-results-are-here-deployments-are-growing-in-size-and-speed-as-cloud-native-adoption-becomes-mainstream/

Kubernetes is a powerful tool for managing the deployment and life cycle of containers, but doesn't actually manage the cloud infrastructure that containers run on. Spending on infrastructure to support cloud computing now accounts for more than a third of all IT spending worldwide, so organizations can't afford to overlook this key component of their cloud operations.

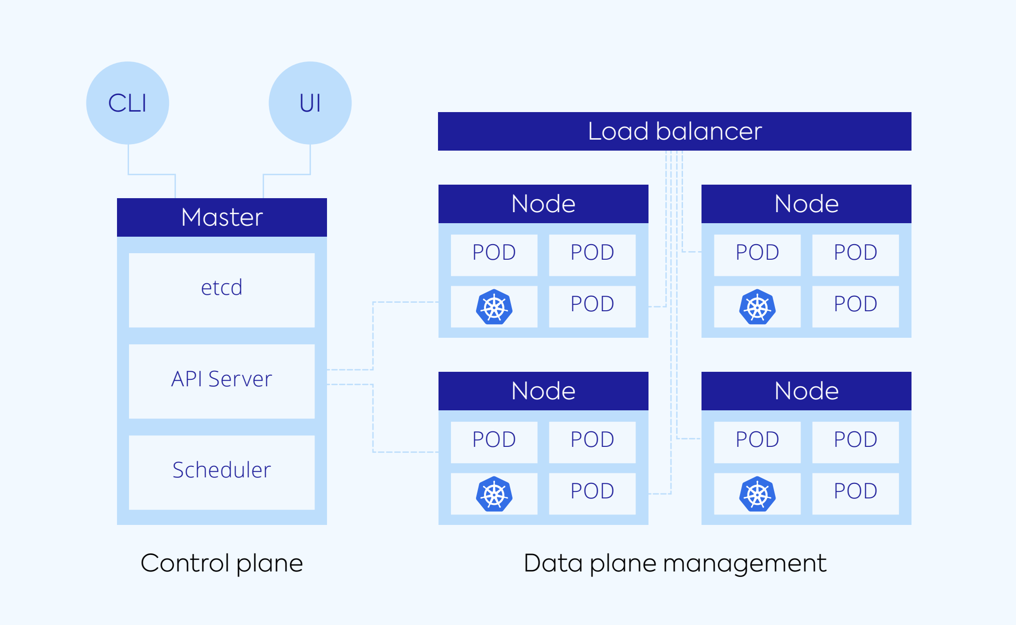
This white paper will present actionable solutions for efficient infrastructure management that will help Kubernetes users reduce costs and optimize resources when operating in the cloud.

The tetris game of auto scaling

In the cloud, you're paying for scalability, and when you work with Kubernetes, everything scales by pods. As a user, you'll interact mostly with the **control plane**, but pods operate on the **data plane**, inside worker nodes (i.e. VMs) and rely on the underlying infrastructure to meet their containers' needs. This is where you'll typically see infrastructure needs grow the most, along with the associated costs.

What's going on inside the master node

The control plane is where Kubernetes carries out communications and sends commands to the worker nodes to execute. With its master node components (etcd, API server, scheduler), Kubernetes knows what needs to be scheduled, configured and accessed.



Kubernetes natively offers pod scaling services (horizontal and vertical pod autoscaling), and while it will schedule a pod to run in any node that meets its requirements, it doesn't automatically scale infrastructure. Kubernetes users can leverage a DIY cluster autoscaler to automatically adjust the size of a cluster and add more resources if there are pods waiting to be run. While this approach ensures that a node is healthy enough for a pod to run on, it can also result in significant inefficiencies and higher costs since Kubernetes doesn't care about the type or size of instance, let alone its cost.

In 2020, computing remains the **largest category of spending** on cloud IT infrastructure at

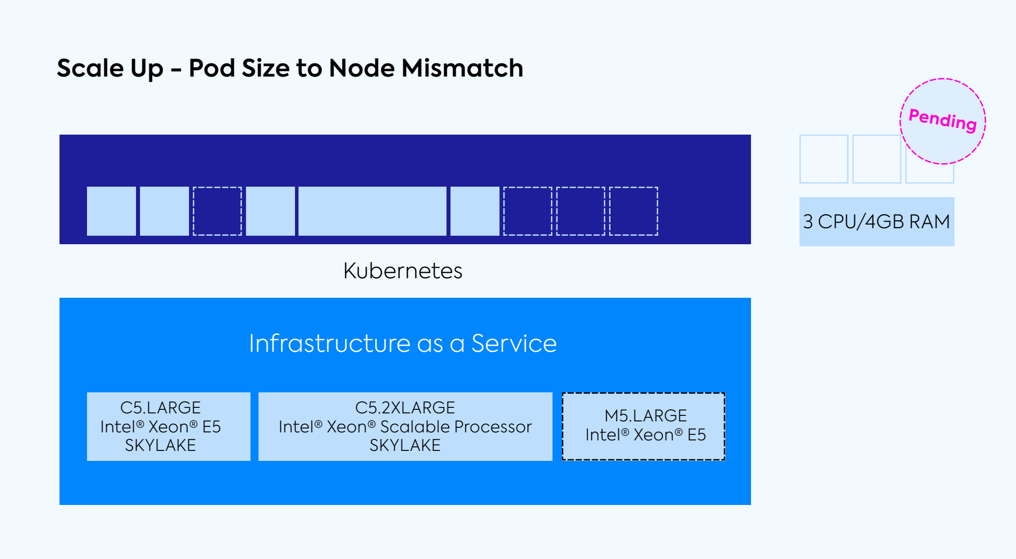
US $34.2B

– IDC[2]

Horizontal Pod Autoscaler (HPA)	Vertical Pod Autoscaler (VPA)
Both the HPA and VPA are fed by the metrics server, which reports CPU and memory utilizations	
HPA measures metrics on deployments and automatically scales the number of pods available in a cluster by replicating them across the environment. As cluster complexity grows, **Ocean** observes scaling decisions made by the HPA and ensures that the cluster has the most optimized infrastructure in real time.	Allocates more or less CPU and memory to a single pod. **Ocean** offers its own vertical container auto scaling solution, which measures in real-time the CPU/memory of pods and provides revsource suggestions based on cluster consumption.

[2] https://www.idc.com/getdoc.jsp?containerId=prUS45293719

For example, in the diagram below, a pod is waiting to be scheduled. The underlying infrastructure has enough space for the pod, which needs 3vCPU and 4GBs of memory, but no single node has enough capacity. Since a pod can only run on a single node, Kubernetes will wait to schedule this pod until one with enough capacity becomes available. This delay could potentially translate into an interruption of service to the customer while you're paying for resources that remain unused.

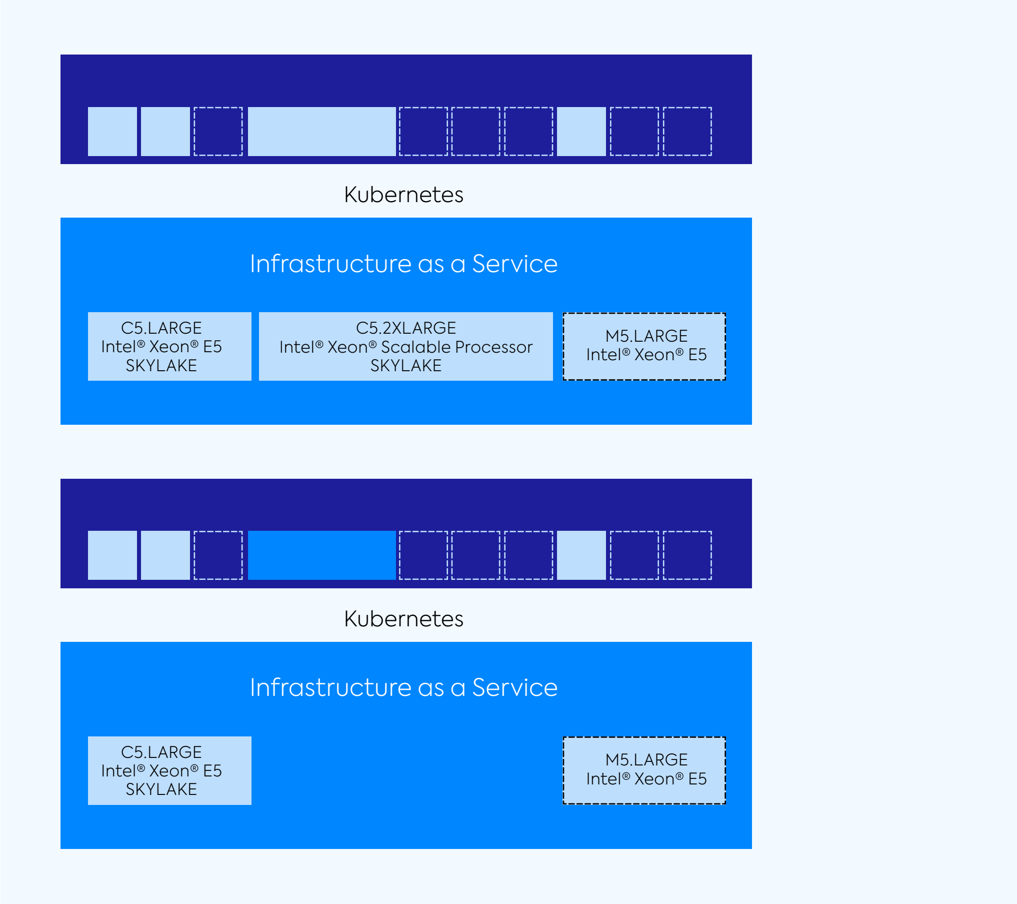

Scale Up – Pod Size to Node Mismatch
Pending
3 CPU/4GB RAM
Kubernetes
Infrastructure as a Service
C5.LARGE Intel® Xeon® E5 SKYLAKE
C5.2XLARGE Intel® Xeon® Scalable Processor SKYLAKE
M5.LARGE Intel® Xeon® E5

Organizations try to get ahead of this lag by adding different types of instances to match different types of pods. But if you bring that same pod from the graphic above to a cluster that is not yet fully provisioned and spin up the wrong instance, your pod will stay unscheduled. To avoid this, container workloads tend to utilize more large instances and a wider variety than non-container workloads, making cost management of this changing infrastructure even more critica.

Scaling down applications also presents infrastructure inefficiencies, and can result in over provisioned nodes. When traffic is low during off hours or night time, it makes sense to reduce capacity. In the case below, where there are only a few pods running across a lot of infrastructure, the cluster will become unscheduled.

80%

80% of instance types used in Kubernetes deployments are made up from 2XL and 4XL instances sizes

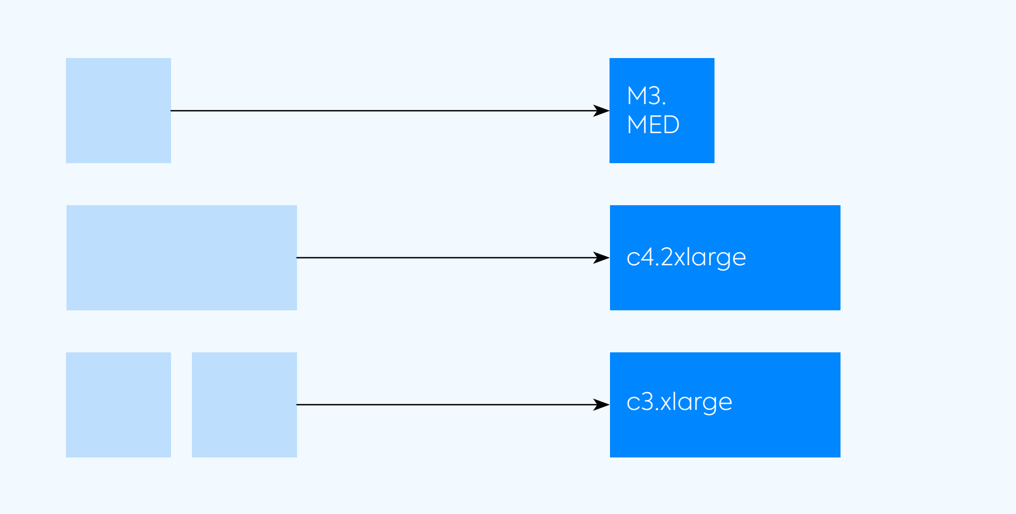


In this scenario, it would have been better to take away the C5.Large or M5.Large, rather than the C5.XLarge. The pods on those smaller instances could have been rescheduled on another node, and the cluster would have remained efficient, running on only what it needs, with all pods scheduled.

These mismatches between pod and node sizes/ types hampers Kubernetes's inherent agility, and leaves resources that you're paying for under-utilized. To be at the maximum of efficiency, infrastructure should benefit from the same scaling flexibilities as containers.

The concept of **container-driven scaling** presents a different approach where pods and containers are treated as the unique entities that they are. Real-time container requirements are used to determine how infrastructure is provisioned, instead of trying to fit containers into predetermined or pre-existing instances. Container and pod characteristics, including labels, taints, and tolerations, define the kind of instance that it gets matched to, whether it needs 5 vCPUs or 1000 GPUs.

Container-driven infrastructure scaling
Infrastructure is provisioned to match what scheduled pods need



Ocean

Simplify infrastructure scaling with Spot's Ocean

As a fully managed data plane for container orchestration, Ocean by Spot is inherently container-driven. With Kubernetes, Ocean connects to your cluster's control plane via a controller pod, and continuously monitors the status of the cluster, its resources and implements changes accordingly. Events are observed at the Kubernetes API Server, giving Ocean visibility that enables accurate provisioning for performance, and lower cloud infrastructure costs.

Right-sizing puzzle pieces

There are more than

1.7 million

possible compute instance combinations to choose from

While the concept of container-driven scaling is a paradigm shift that can ensure infrastructure meets application requests, for Ops teams, scaling infrastructure is just one piece of the puzzle. If pods are asking for more infrastructure than they actually use, it doesn't matter how efficiently your infrastructure scales, you'll still end up paying for more compute than you need.

Kubernetes provides users with the option to define resource guidelines for containers, based on specific CPU and memory needs. Developers will often attempt to configure resource requests based on a guess (can be inaccurate), trial and error (can be extensive) or simulations with a test deployment (can be ineffective as test metrics often differ from production usage).

Incorrect provisioning can lead to idle resources and higher operational costs, or result in performance issues within your application because a cluster doesn't have enough capacity to run. Most organizations are unwilling to risk performance, and in order to be prepared for a scaling burst, will typically do one of two things, neither of which are perfect solutions:

Overprovision clusters with more resources than needed, resulting in spare capacity that you're paying for, but that often sits idle.

Pod Priority indicates the importance of a pod relative to other pods, and schedules them as such. Lower priority pods may remain unscheduled.

70%
cloud cost wasted

As much as 70% of cloud costs are wasted[3]
– Gartner

Real time right-sizing with Ocean

With Ocean, users are presented with better right-sizing and container utilization approaches that help to reduce container costs. In order to more accurately estimate and apply the right resource requirements to containers, Ocean's vertical container auto scaling feature measures, in real time, the CPU and memory of pods and provides actionable recommendations to improve resource configurations.

Users also have the option to configure adjustable headroom, a unique solution introduced by Spot, to better provision spare capacity for fast scaling. Adjustable headroom is a spare capacity unit that acts as a placeholder for workloads, configured based on the characteristics of applications running in the cluster. This kind of intelligent overprovisioning is a cost effective way to limit idle resources and ensure pods always have a place to land.

[3] https://www.infosys.com/about/knowledge-institute/insights/rationalizing-cloud-costs.html#:~:text=Gartner%20estimates%20that%20up%20to%2070%25%20of%20cloud%20costs%20are%20wasted.&text=Financial%20CapEx%20models%20or%20manual,CapEx%20rather%20than%20OpEx%20model

Playing monopoly with your instances

Along with the advantages of scalability, the cloud also offers more flexible pricing that can hold the key to significant cost saving when applied the right way. Pay-as-you-go (e.g. on-demand) is the benchmark pricing model for compute, but cloud providers also sell capacity through various discounted pricing models, either via a long-term commitment (e.g. reserved instances), or they sell spare capacity (e.g. spot instances).

Spot instances offer users up to 90% cost reduction compared to on-demand pricing, but their low cost comes with the caveat that cloud providers can take capacity back whenever they need it. Interruption of your EC2 instances can impact and degrade services, or result in data loss, making developers wary to work with them on mission-critical, production workloads. With intelligent orchestration, however, spot instances can be a good fit for container workloads since these are typically fault tolerant and highly available.

On-demand is the **easiest to set up** and implement, but its flexibility comes at a higher cost than other options.

Reserved instances are more **cost effective**, but the dynamic nature of containers make it difficult to make long term commitments.

Spot instances are the **cheapest way to buy** compute capacity, but require a significant amount of configuration, maintenance and expertise to ensure availability.

Ocean

Leverage spot instances for lower cost computing with Ocean

For companies that want to take advantage of the discounted pricing of spot instances, Ocean uses predictive analytics to identify and anticipate spare capacity interruptions, and proactively replaces at-risk instances with new ones. Machine learning and analytics also enable Ocean to implement the optimal mix of pricing options while sustaining highly available applications.

Conclusion

It’s clear that there is a path to scalability and significant cost savings when users leverage Kubernetes and containers in the cloud — if they can overcome the operational complexities of managing cloud infrastructure. Even as Kubernetes adoption continues to gain momentum, and the ecosystem of services for it grows, cost management is still a significant pain point.

At Spot, our customers are facing the same challenges and we’ve worked to create a solution that addresses infrastructure management with containers, following the approaches laid out above. Ocean is a container-driven data plane management platform that automates capacity provisioning and scaling so developers can focus on applications, not on infrastructure. Using these intelligent features for auto scaling, right sizing and capacity purchasing, Ocean is able to help customers realize up to 90% savings on cloud infrastructure, and give their teams a simpler, efficient way to scale out their containers and applications.

More information on containers and Kubernetes

Ocean on AWS EKS Workshop

Challenges and solutions for container cost optimization

7-14jul20