spotinst

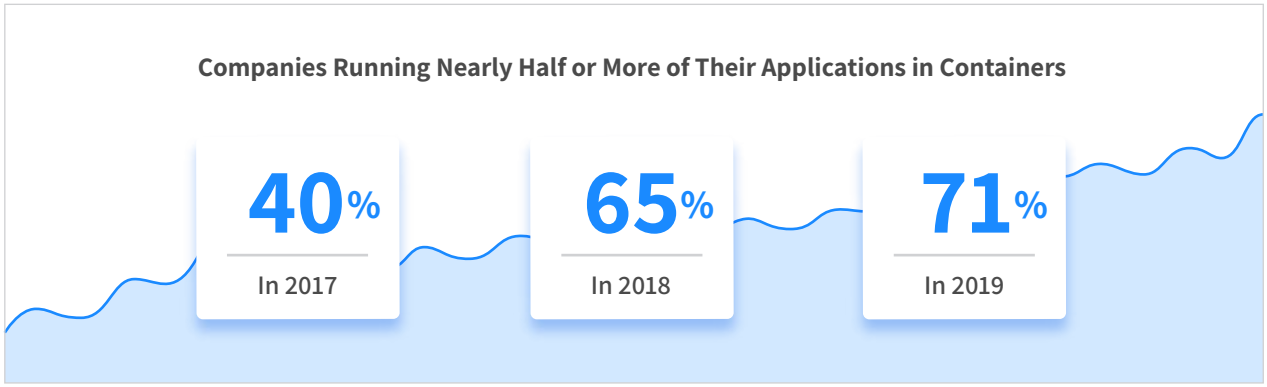
WHITE PAPER

The State of Container Workloads 2019

Introduction

While containers are not a new concept in IT, their large scale adoption to encapsulate applications is a fairly recent trend. This is primarily due to the success of Docker container format and secondly, to Kubernetes as a way to orchestrate containers at scale.
Within the last few years, modern applications are increasingly deployed on containers rather than on virtual machines. The containerized workloads are deployed in cloud with the help of various orchestration engines. The lightweight nature of containers, the declarative approach to container deployment and the seamless scaling offered by cloud providers has changed application deployment significantly.

The adoption of containers is increasing with the success of Kubernetes and serverless container services like AWS Fargate, Azure Container Instances, Google Cloud Run and Spotinst Ocean. As these services get more traction, we expect to see further increase in containerized workloads.



Credit: 2019 Container Adoption Survey by Portworx and Aqua Security

For almost 5 years now, Spotinst has been managing millions of instances, and as of today concurrently runs ~500K servers for our 1,500+ customers at any given moment.

Spotinst's flagship product Elastigroup, has delivered dramatic cost optimization for our customers by leveraging Spot Instances for their production workloads, while ensuring high availability with the highest levels of SLA.

Our more recent cloud-native offering, Ocean, has provided a unique "serveless container" experience by abstracting infrastructure management and cost optimization for Kubernetes and ECS users.

By researching the data of hundreds of Spotinst customers running 5 million containers, we are pleased to present our first report on the state of container workloads in 2019. These organizational trends are indicative of the underlying dynamics of container workloads and we want to share our learnings with the larger community.

State of Container Workloads

A typical container workload traverses from developer environment to production and has the following DevOps flow:

1 | Encapsulation of application code and dependencies on containers

2 | Containers traverse the CI/CD pipeline towards production

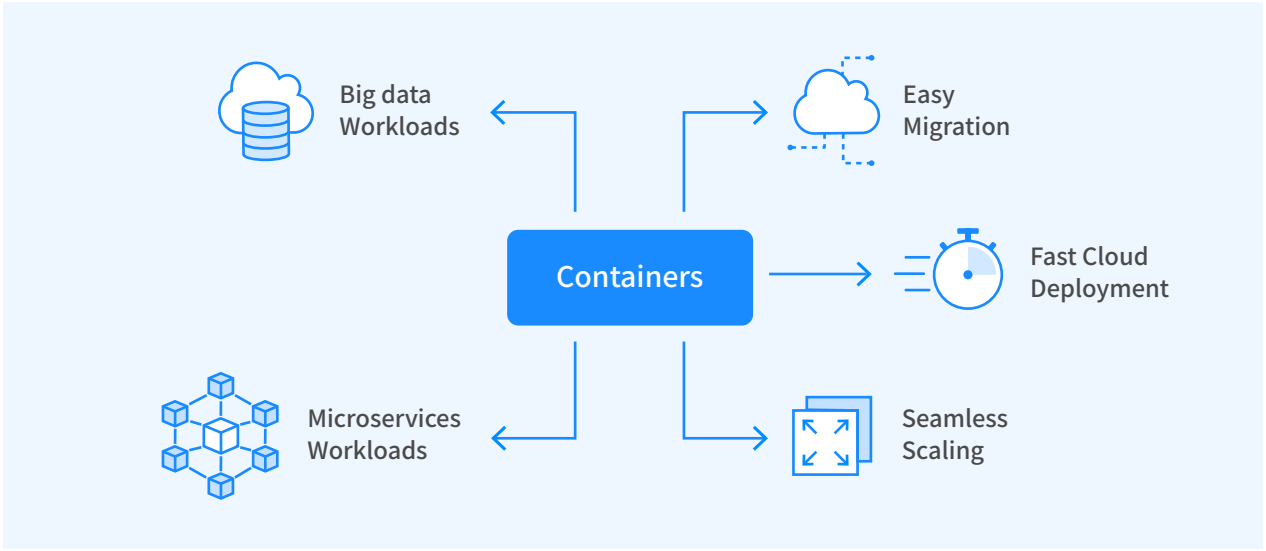
3 | Deployed in production and Day 2 Operations begin

We are presenting our data to mimic this typical flow of container workloads and we hope this gives insight into how organizations are using container workloads in production.

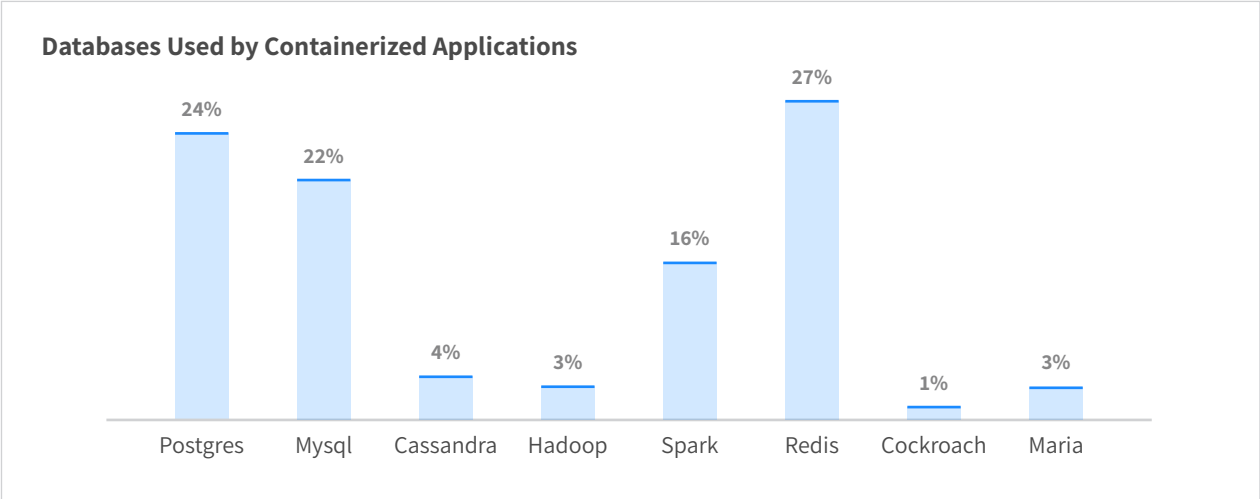
Application Encapsulation

Containers are versatile encapsulation for various types of applications. Even though containers are seen as a pretty good fit for Microservices, they are used for many different types of applications including stateless and stateful applications. Some of the advantages of using containers are:

- Containers can be used to encapsulate legacy workloads and move them to cloud easily
- Containers help modernize legacy workloads by splitting the applications into smaller components for cloud deployment and seamless scaling
- Containers are a perfect fit for Microservices
- Containers help run big data workloads because it provides better isolation and they can be managed with a single unified interface



Even though there is a myth that containerized workloads are better for stateless workloads, we are seeing many stateful applications being deployed on containers.
The following graph shows some of the databases used by applications running on containers.



With the right automation, running stateful containerized workloads is not a problem at all. As the Kubernetes community is working on improving StatefulSets, we expect to see more stateful workloads being containerized.

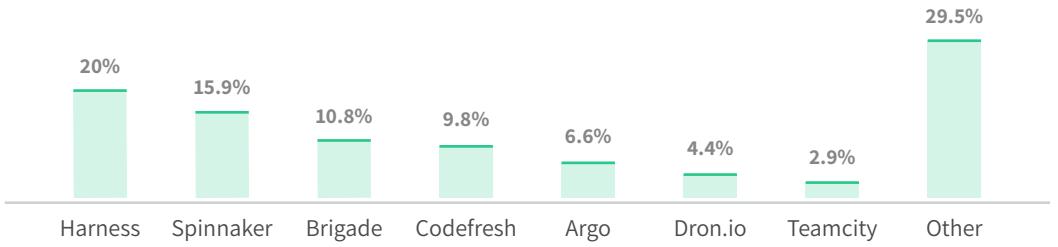
CI/CD Tools

With containers, we have seen a shift in the DevOps workflow as the application bits travel from developer environments to production. Typical CI/CD workflow of containerized workloads can be broadly described as:

1. Developers check in their code in a code repository. Then automation is used to build containers and, after the relevant tests, the container image is pushed to a container registry.
2. Developers build containers on their laptop which are then pushed to the container registry after passing through the testing stage.

This has led to a proliferation of CI/CD tools in the market with some of the modern tools gaining lot of traction in deploying containerized workloads. Some of the tools used by our customers are listed in the following figure.

CI/CD Being Used for Container Deployment



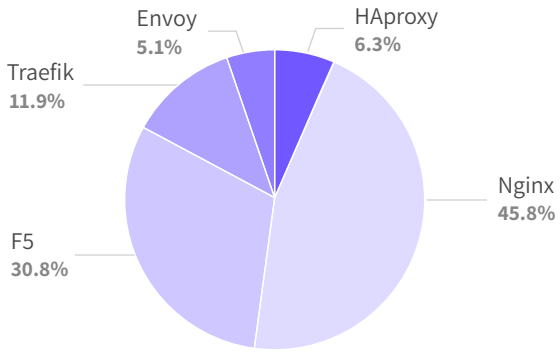
The interesting point in this data is the traction for Harness and Spinnaker rather than traditional continuous integration (CI) tools. This is an indication of how DevOps workflow has changed with containers. Continuous Integration tools are still relevant and widely used but containerization has shortened the flow from developer laptop to deployment, making continuous delivery/deployment (CD) tools like Harness and Spinnaker more relevant.

Container Deployment

Service Proxy

Service proxy frontends all the client side traffic to the application components running on containers. This is a critical component of container workload deployments because service proxy is needed to ensure that the application performance needs, are met. We see various Service Proxies in use in the container workloads managed through the Spotinst platform. NGINX is the most used service proxy followed by F5, Traefik, HAproxy and Envoy.

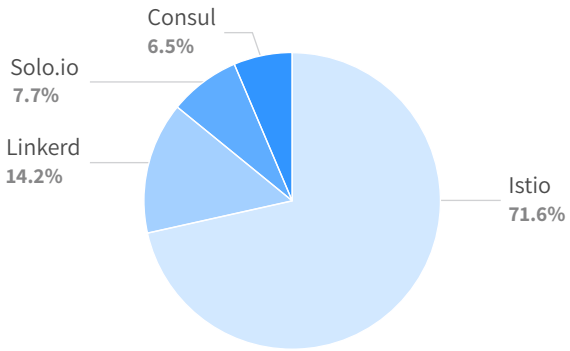
Service Proxy



Service Discovery and Service Mesh

As containers are used for deploying the applications, leading to a more distributed architecture, Service Discovery and Service to Service Communications comes into focus. There are many tools that offer both service discovery and service mesh (for inter-service communication). With microservices architecture, the role of service discovery and service mesh has become even more critical. Open Source tools rule the service discovery and service mesh market segment and we are seeing Istio, the open source service mesh project by Google, has gained significant traction in the past year. Kubernetes offers necessary primitive to implement rudimentary service discovery and service mesh solution. Istio and other tools leverage these Kubernetes primitives to provide more comprehensive tools needed for microservices and other containerized workloads.

Mesh & Service Discovery Being Used For Container Workloads



Cluster Size

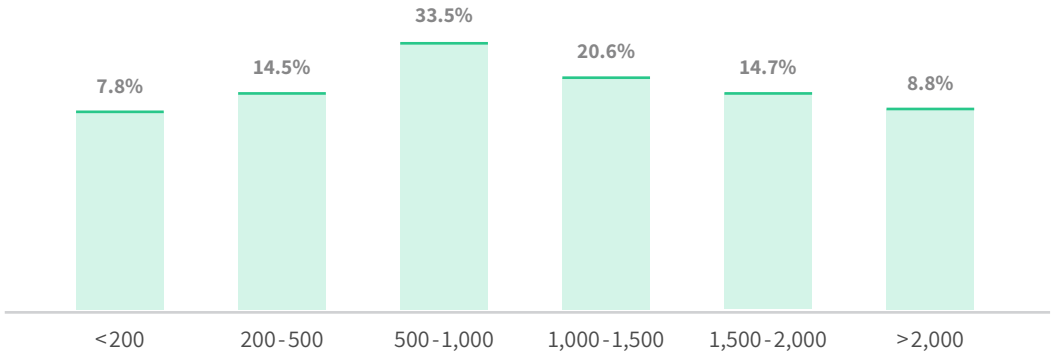
The advantage of containers in modern application deployment lies in the resource efficiency and scale. We have seen deployments with a handful of containers in a cluster up to several thousands of them. The customer data indicates that scale is one of the biggest considerations with >500 containers per cluster being the sweet spot for containerized workloads running on SpotInst platform.

Different orchestration planes may impose limitations on the maximum number of containers one can use in a cluster but there are clear trends indicative of specific ways in which these clusters are used. They are:

- Multiple services and multiple environments (Dev, Test & Production) in the same cluster. Customers also use different clusters for different environments
- Some prefer a single large cluster for their application whereas others prefer small clusters to package microservices
- Some prefer to use large clusters to set up multi-tenant environments for better resource usage

Containerized workloads provides better efficiency and ease of scale to meet the diverse needs of modern enterprises.

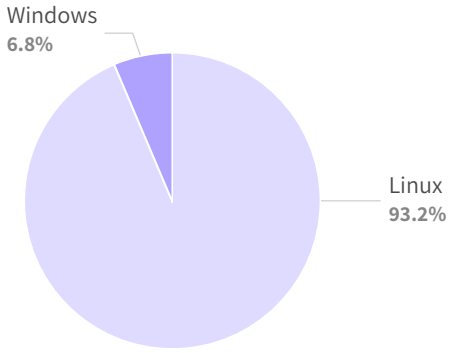
Containers Per Cluster



Operating Systems

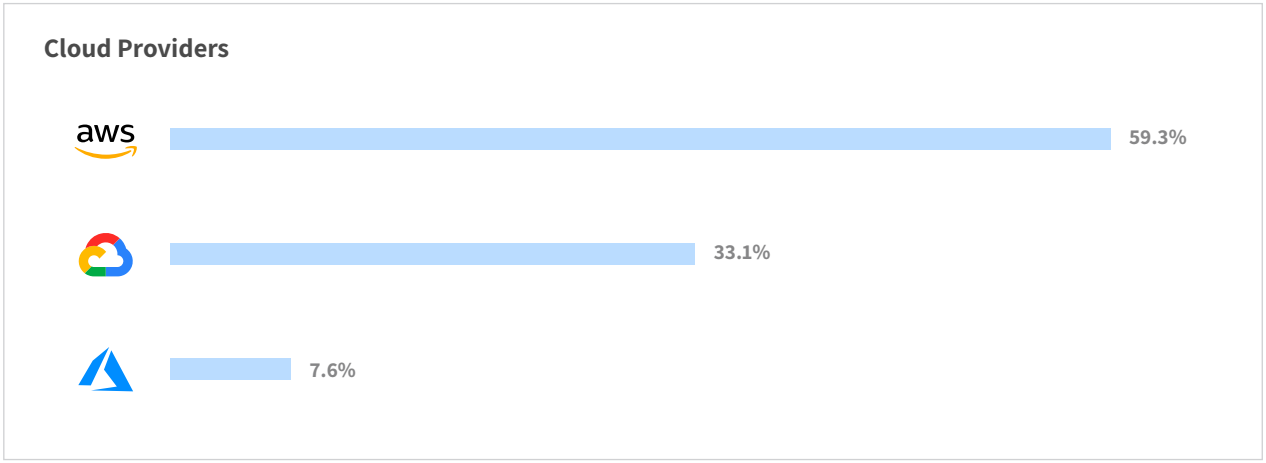
Containers began with chroot system call introduced in the 1970s on Unix systems and BSD. In the 2000s, a similar idea was implemented as jails in FreeBSD, as a way to partition directory folders. Eventually LinuX Containers (LXC) was introduced in 2008 which took advantage of cgroups and namespaces to isolate resources. Until recently, Windows didn't have a counterpart equivalent to containers. So it is no surprise that most of the containerized workloads run on Linux containers.

Operating Systems Containers Run On

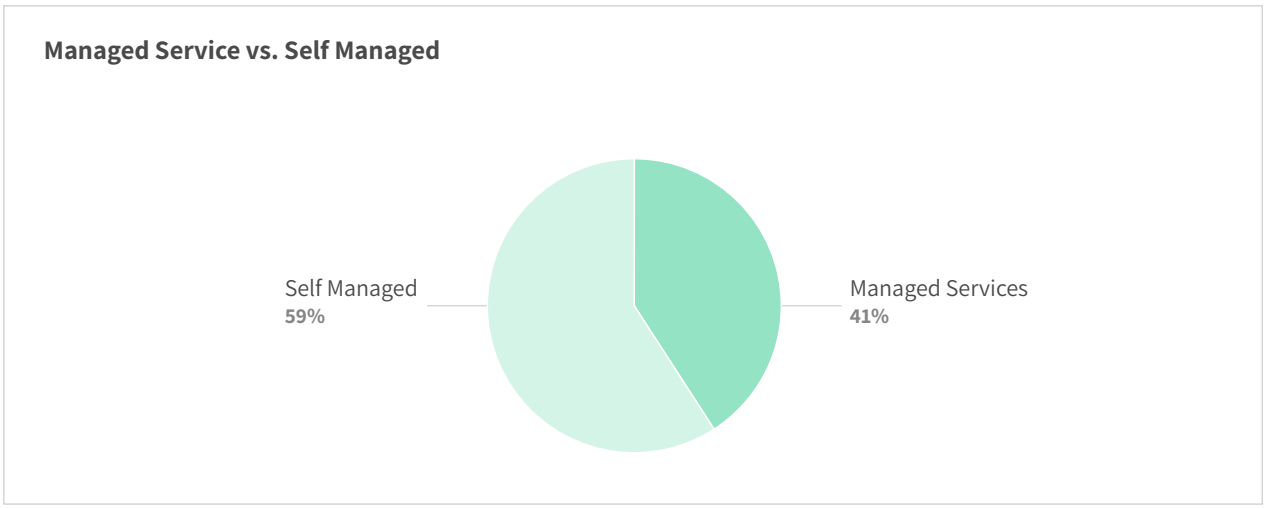


Infrastructure

Spotinst customers use public cloud providers to run their containerized workloads. We support all the three major cloud providers and the container workloads are distributed among these cloud providers with AWS hosting significant number of our customers' container workloads. With the increase in adoption of serverless offerings like AWS Fargate, Microsoft Azure Container Instances and Google Cloud Run, we expect this trend to increase further in the coming years.



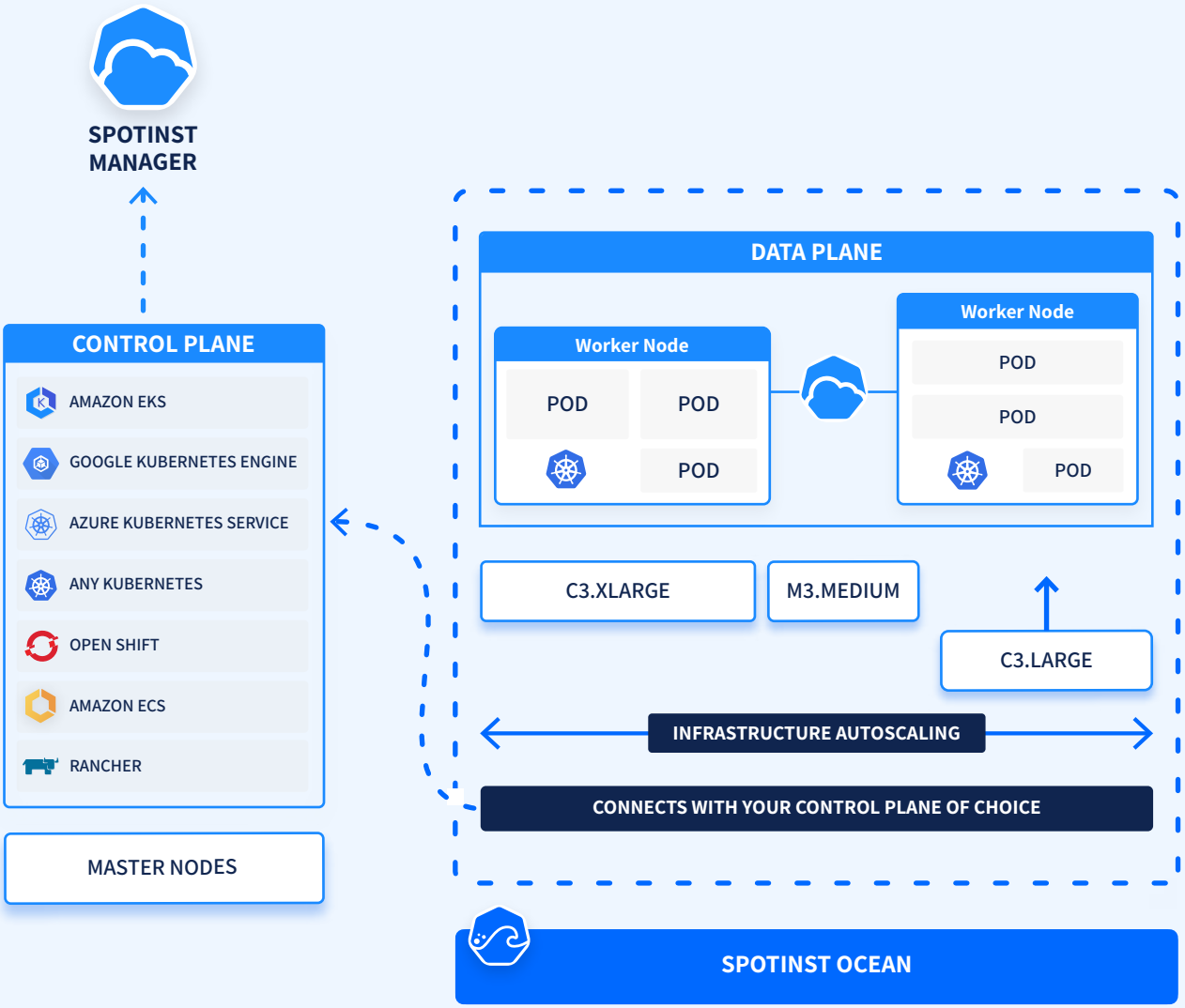
We also see a significant percentage of our customers using self managed container service over managed offerings.



Even though it is counter intuitive to see more people managing the underlying infrastructure than using managed container services, there could be various reasons, including historic usage patterns.

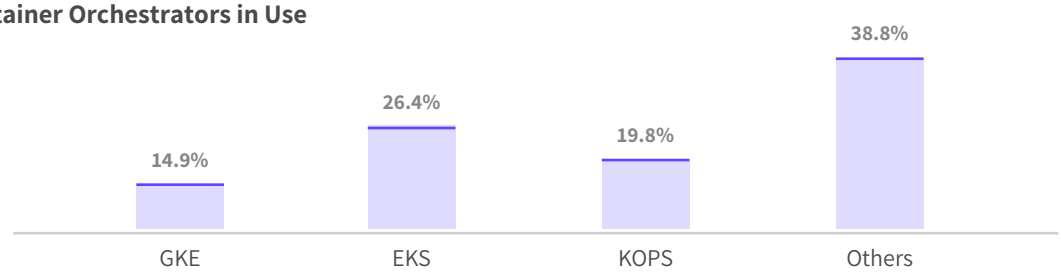
As managed container services are new and most of the enterprise customers already have an ops team that can handle the underlying cloud infrastructure, inertia could be the cause. Other reasons could be related to cost and control. Specifically, the current managed container services are quite expensive and also, completely abstract the infrastructure, so end-users cannot control the types of machines being used.

Spotisnt Ocean on the other hand, while simplifying the infrastructure management with cloud-native autoscaling, rightsizing and similar features, still allows the user to select which instances they wish to use. This allows for fine-tuned performance with minimal effort.



Among the Kubernetes customers, we see a range of deployment options. Almost 43% of our users use Managed Kubernetes Service like GKE or EKS. Among the remaining who use DIY approach on cloud, KOPS stand out as the default mechanism for deploying Kubernetes with 19.8% of users using the deployment tool. The other tools including Kubeadm are used by the remaining DIY Kubernetes customers. Even though it is early, there is clearly a trend that shows the users wanting to use managed services or better automation to deploy Kubernetes and avoid the operational overhead that comes with the platform.

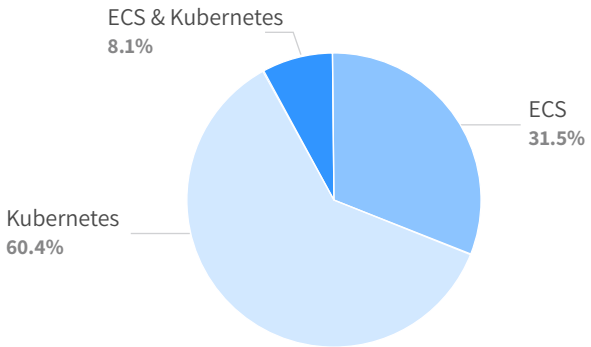
Container Orchestrators in Use



With more customers looking to simplify their container workload management, we expect “serveless container” adoption to greatly increase in the upcoming year.

We also looked at the Kubernetes traction in AWS as they also offer a competing offering called Amazon ECS, the elastic container service using their own proprietary technology. We see an interesting mix in our customer base with more than 60% users using Kubernetes as the orchestration plane. Around 31% of our users use ECS for their container workloads. A small chunk of users use both ECS as well as Kubernetes for their workloads. It could either be due to users evaluating both the options as they decide to standardize on a single orchestration plane. Spotinst Ocean makes it easy to use Kubernetes and/or ECS, giving users the flexibility of choice for container orchestration. **Based on all the data we see among our customers as well as external surveys, Kubernetes has emerged as the standard for container orchestration.**

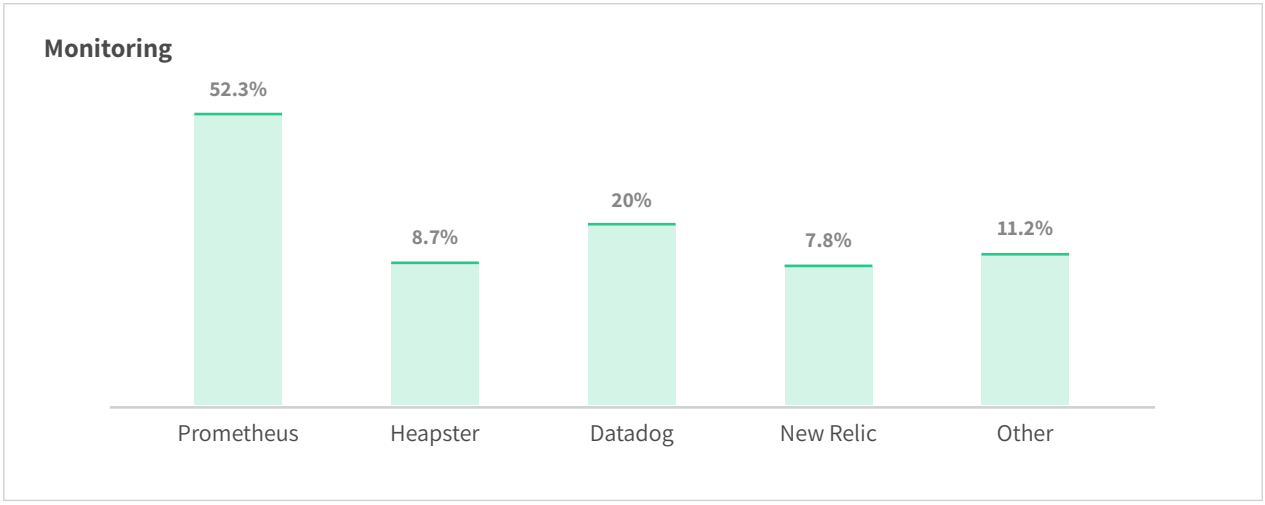
Kubernetes vs. ECS in AWS



Day 2 Operations

Monitoring

Monitoring is critical to keep the lights on in production after the deployment of containerized workloads. Monitoring is critical to ensure that engineers have the necessary information to troubleshoot any issues that may impact their container clusters. Monitoring containers is different from traditional monitoring tools and you need to monitor both at the host level as well as cluster and container levels. Even though there are many monitoring tools available in the market, we see our customers using tools like Prometheus, DataDog, New Relic, Heapster and a few others. As expected, Prometheus comes out as a popular tool followed by DataDog and New Relic.



In Summary

The adoption of containers is increasing with the success of Kubernetes and serverless container services like AWS Fargate, Azure Container Instances, Google Cloud Run and Spotinst Ocean. As these services get more traction, we expect to see increased adoption and more diverse workloads on containers. In our study of customer usage, we find some interesting trends we notice from our data are:

- Stateful container workloads are real and more stateful applications are getting containerized
- Continuous Delivery/Deployment tools are gaining more traction as containers reshape the DevOps workflow
- AWS leads the cloud providers in hosting the containerized workloads and managed container services are getting increasing traction
- Large clusters dominate the containerized workloads and is an interesting trend to watch as multi-tenancy matures in Kubernetes