SignalFx

MONITORING MICROSERVICES
ON
KUBERNETES

eBook

According to the latest bi-annual survey from CNCF, monitoring is one of the top challenges for organizations adopting Kubernetes.

This e-Book provides insights into how to best monitor both Kubernetes clusters and microservices deployed on Kubernetes in order to pinpoint performance issues and minimize mean time to resolution (MTTR).

Managing performance issues in real-time is a controllable way to enable a flawless end-user experience. Enterprises need to develop best practices for operating Kubernetes clusters as well as for understanding performance across the entire stack, including distributed traces across all microservices.

Kubernetes

The Brain Behind the Orchestration of Containerized Applications

Kubernetes is a leading container orchestration solution for deploying containerized applications. Enterprises across every industry are adopting containerized microservices in order to gain:

- **SPEED** — to accelerate the pace of innovation

- **SCALE** — to deliver optimized application experiences across different geographies and cloud providers

- **EFFICIENCY** — to provide more value at a lower cost by optimizing resource consumption

Kubernetes has several logical layers to support various use cases and manage the lifecycle of containerized microservices.

There are three foundational pillars of the Kubernetes deployment philosophy:

INFRASTRUCTURE ABSTRACTION	DECLARATIVE CONFIGURATION	IMMUTABILITY
Application orchestration from the infrastructure resources such as compute servers, storage volumes, and networks	Kubernetes continuously monitors the state of the applications deployed and Kubernetes objects; It also implements the desired state of the system as expressed by the operators such as how many replicas of a service should be running	Continuing the immutability requirements set forth by Docker where container images are immutable, Kubernetes objects are also immutable e.g. different versions of Kubernetes Services are completely separate and are not interchangeable

Why Use Kubernetes?

In addition to the benefits that containerization provides for application developers, Kubernetes makes the life of operations easier in many different ways:

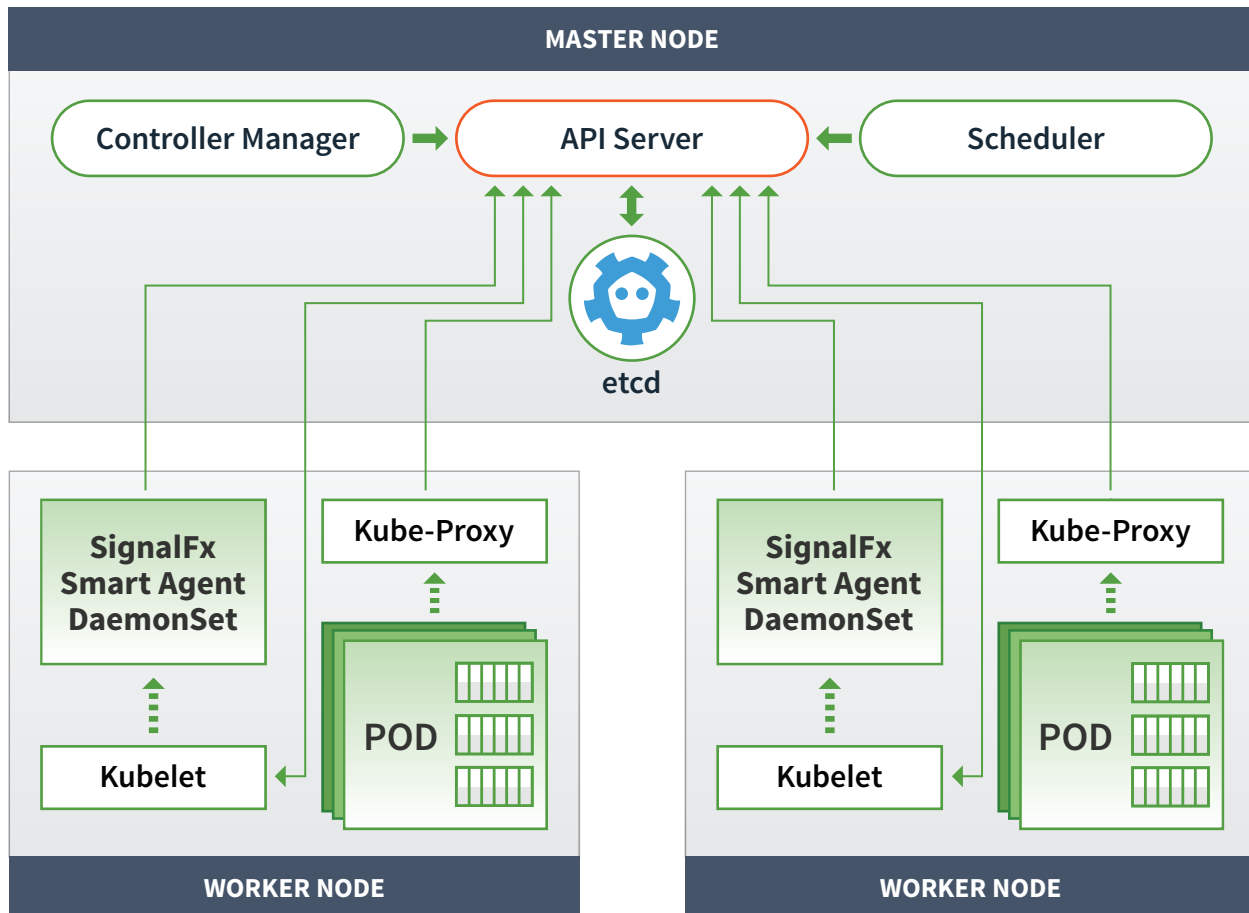| AUTOMATED APPLICATION LIFECYCLE MANAGEMENT	CLOUD INTEROPERABILITY	SELF-HEALING SYSTEM
Kubenetes automates application lifecycle management as an end-to-end solution including application provisioning, autoscaling, and replication management as well as CI/CD	Kubenetes provides abstraction over the infrastructure so that applications become truly portable across different clouds	Kubernetes guards applications against any failures or unreliable behavior that destabilizes the system. It continuously takes actions to ensure the state of the system conforms to the desired state, as expressed in the declarative configuration

Kubernetes Controller Manager continuously compares the desired state with the actual state, evaluates the differences between them, and reconciles the differences by taking appropriate actions.

SignalFx
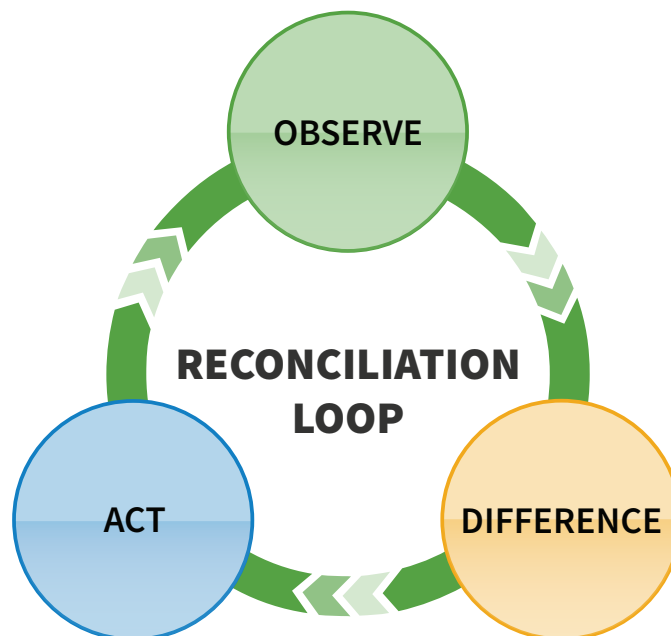
Understanding Kubernetes Components

Kubernetes is a system that assembles a group of machines into a single unit that can be consumed via an API. Kubernetes further segments the compute resources into two groups:

Worker Nodes and Master Nodes

Here are the main components running on master and worker nodes:

MASTER NODES

API SERVER

- Gateway to the Kubernetes cluster

- Mediates all the requests from clients and API objects stored in etcd

- Performs authentication and role-based access control RBAC

- Requests validation and admission control

- More information: http://bit.ly/k8s-apiserver

CONTROL MANAGER

- Daemon process that implements the control loops built into Kubernetes — rolling deployments, replica sets, number of worker nodes, etc.

- More information: http://bit.ly/k8s-ctrl-mgr

SCHEDULER

- Decides where pods should run based on multiple factors — affinity, available resources, labels, QoS, etc.

- More: http://bit.ly/k8s-scheduler

etcd

- Heart of the Kubernetes cluster; persists key-value information on all Kubernetes objects

  More: http://bit.ly/2COkMx9

WORKER NODES

KUBELET — AGENT ON EVERY WORKER

- Initiates pods (a group of one or more containers) using PodSpec and ensures all pods are running and healthy

- Interacts with containers — e.g. Docker

- More: http://bit.ly/k8s-kubelet

KUBE PROXY — AGENT ON EVERY WORKER

- Network proxy and load balancer for Kubernetes Services

- More: http://bit.ly/k8s-proxy

SignalFx

These are the Kubernetes add-ons that are required for the most applications:

KUBE-DNS	KUBECTL
• Provisioned as a pod and a service on Kubernetes • Every service gets a DNS entry in Kubernetes • Kube-DNS resolves DNS of all services in the cluster	• The official command line for Kubernetes • Behind the scenes uses REST-based API calls to Kubernetes API server

Key Kubernetes Constructs and Objects

NAMESPACES	PODS
Virtual segmentation of single clusters	A logical grouping of one or more containers that are managed by Kubernetes
NODES	**REPLICASET**
Infrastructure fabric of Kubernetes (host of worker and master components)	Continuous loop that ensures given number of pods are running
ROLES	**INGRESSES**
Role-based access controls for Kubernetes cluster	Manages external HTTP traffic to hosted service
DEPLOYMENTS	**SERVICES**
Manages a ReplicaSet, pod definitions, updates and other concepts	A logical layer that provides IP, DNS, etc. persistence to dynamic pods

Monitoring Challenges in Kubernetes Environments

According to the latest bi-annual survey from CNCF, monitoring is one of the top challenges for organizations adopting Kubernetes. Monitoring strategies of yore do not work in the cloud-native era primarily because:

• There are many more components to monitor

• Containers are ephemeral and dynamic

• Kubernetes automatically schedules pods based on the best resource utilization. While it does increase efficiency, it also adds unpredictability on where the pods get deployed and run unless specific intent is expressed using affinity or taints. Containerized microservices are more complex to troubleshoot

## This results in the following set of conditions for Kubernetes environments:

• There are many more components to monitor

• Containers are ephemeral and have unpredictable pod placement

• Containerized microservices are more complex to troubleshoot

MORE COMPONENTS TO MONITOR

In the monolithic world, there are only two components to monitor — applications and the hosts on which the applications were deployed.

In the cloud-native world, containerized applications orchestrated by Kubernetes have multiple components that require monitoring:

- Hosts
- The Kubernetes platform itself
- Docker containers
- Containerized Microservices

Microservices

Docker Containers

Kubernetes Platform

Host-Bare-Metal or Virtual Machines
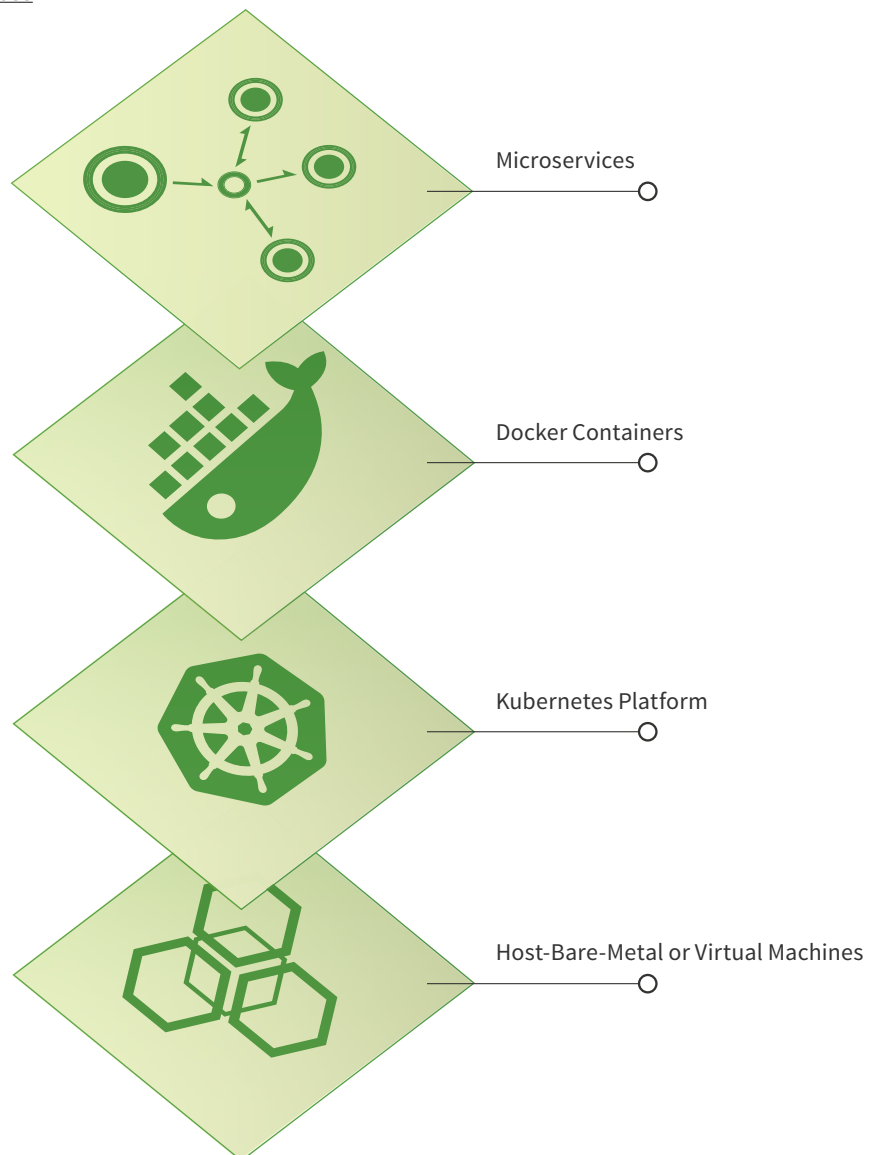
CONTAINER EPHEMERALITY AND UNPREDICTABLE PLACEMENT

Unlike the traditional long-running host model, modern microservices-based applications are typically deployed on containers that are dynamic and ephemeral. Kubernetes makes sure the desired number of application replicas are running. Kubernetes will place the pods on the nodes that it deems fit unless specifically instructed not to do so via node affinity or taints. In fact, letting Kubernetes schedule pods is the key design goal for this self-adjusting system.

Traditional monitoring approaches do not work in these highly-dynamic environments because they tend to follow long-running hosts by using hostnames or IP addresses. For containerized environments, monitoring tools must provide immediate service discovery and automatically detect the lifecycle events of containers, while also adjusting metric collection when containers are spun up or restarted in seconds.

MONITORING THE PERFORMANCE OF MICROSERVICES

Pinpointing issues in a microservices environment is more challenging than with a monolithic one, as requests traverse both between different layers of the stack and across multiple services. Modern monitoring tools must monitor these interrelated layers while also efficiently correlating application and infrastructure behavior to streamline troubleshooting.

Key Performance Metrics to Monitor

POD METRICS

It's essential to monitor all Kubernetes objects in order to ensure each cluster is healthy and resource utilization is optimized. Monitoring Kubernetes pod metrics as well as Deployments and Services will help determine whether Kubernetes is working as intended in your environment.

PODS	
	Number of Desired Pods
	Number of Available Pods
	Pods by Phase (failed, pending, running)
	Desired Pods per Deployment
	Desired Pods per Service
	Available Pods by Deployment
	Available Pods by Service
	Pods Desired by ReplicaSet
	Running Pods Per Node

RESOURCE UTILIZATION METRICS

It's important to also keep track of resource utilization to ensure that your applications and Kubernetes clusters remain healthy.

Docker Socket provides container metrics and node-level resource utilization metrics, such as CPU and memory usage. Kubernetes provides collectd metrics as well as metrics emitted by Kubernetes.

Correlating CPU, memory, Disk IO, and network performance metrics with application performance and Kubernetes events help to get to the root cause of a performance issue quicker.

Monitoring Docker and Kubernetes events, such as container or pod lifecycle events, helps to pinpoint misconfigurations or resource starvation.

Monitoring Kubernetes with SignalFx

Kubernetes provides detailed information about its components and application resource usage at the cluster, pod, and service level that can easily be collected by a vast array of monitoring solutions, but the task of making this data actionable is left to end users.

SignalFx enables you to monitor Kubernetes with flexible, open instrumentation and pre-built dashboards that provide immediate visibility into the various layers of their environment. By combining streaming analytics with distributed tracing, SignalFx users can go beyond basic data collection — leveraging real-time alerting and performance analysis to find and resolve issues in seconds.

COLLECT METRICS FROM A CLOUD PROVIDER

For basic monitoring of a managed Kubernetes service like AWS Elastic Container Service for Kubernetes (EKS), Google Container Engine (GKE), and Azure Kubernetes Service, the simplest way to collect metrics is by integrating SignalFx with services such as AWS CloudWatch, Google Stackdriver, and Azure Monitor.

This approach is the most straightforward, and enables SignalFx to collect Kubernetes metrics without having to install an agent or modify application code. However, these services are configured by default to aggregate and report metrics at relatively infrequent intervals (AWS CloudWatch updates every 5 minutes by default) and do not provide insight into the specific services deployed on your Kubernetes clusters.

INSTALL THE SIGNALFX SMART AGENT

For greater insight into services and finer-grained monitoring of container metrics, we recommend installing the Signalfx Smart Agent throughout your Kubernetes clusters.

An open-source metrics collection agent built on top of collectd, the Smart Agent provides automatic service discovery and configuration for monitoring content. One advantage with this approach is the Smart Agent's ability to submit metrics to SignalFx at 1-second resolution, making it especially well-suited for the ephemeral nature of Kubernetes pods.

The Smart Agent runs as a DaemonSet in Kubernetes to ensure that it is installed on every node in the cluster. It collects data from Kubernetes and uses cAdvisor to get resource and performance characteristics from running Docker containers.

INSTALLATION FOLLOWS THREE SIMPLE STEPS:

1. Storing a SignalFx access token in a Kubernetes secret

2. Configuring the Kubernetes daemon set and SignalFx agent for your cluster

3. Running two commands on your Kubernetes cluster to install the Smart Agent and begin streaming data into SignalFx

```
$ kubectl create  -f configmap.yaml  \
                  -f daemonset.yaml
```

Not only can the Smart Agent monitor upstream Kubernetes, it is also capable of collecting metrics from managed Kubernetes services like AWS Elastic Container Service for Kubernetes (EKS) and Google Container Engine (GKE), as well as platforms like Openshift and Pivotal Kubernetes Service (PKS).

The Smart Agent also provides monitors that can collect data from other metrics protocols like Prometheus and StatsD, so that teams with existing metrics pipelines can easily take advantage of SignalFx.

| Kubernetes | Google Kubernetes Engine | Amazon EKS | Azure AKS | Red Hat Openshift | Pivotal PKS |

INSTRUMENT APPLICATIONS FOR DISTRIBUTED TRACING

Those using SignalFx Microservices APM™ can submit trace spans for their containerized applications by using Zipkin, OpenTracing, or OpenCensus. For Istio service mesh deployments, we provide a telemetry adapter that sends metrics and traces collected by Istio directly to SignalFx.

Navigating Through Your Kubernetes Environment

## Pre-Built Kubernetes Dashboards

SignalFx Infrastructure Navigator gives you an immediate, at-a-glance view of your Kubernetes architecture, as well as nodes and pods, colored by critical health metrics. The Infrastructure Navigator also provides visibility all the way through the stack as you drill down and across elements of your environment, reflecting the fact that the infrastructure, the orchestrator, the containers and the apps are all related layers, not just individual system components.

You can view all of the Kubernetes clusters in your environment by selecting **Overview**, which opens a built-in dashboard showing key resource utilization metrics for each cluster.
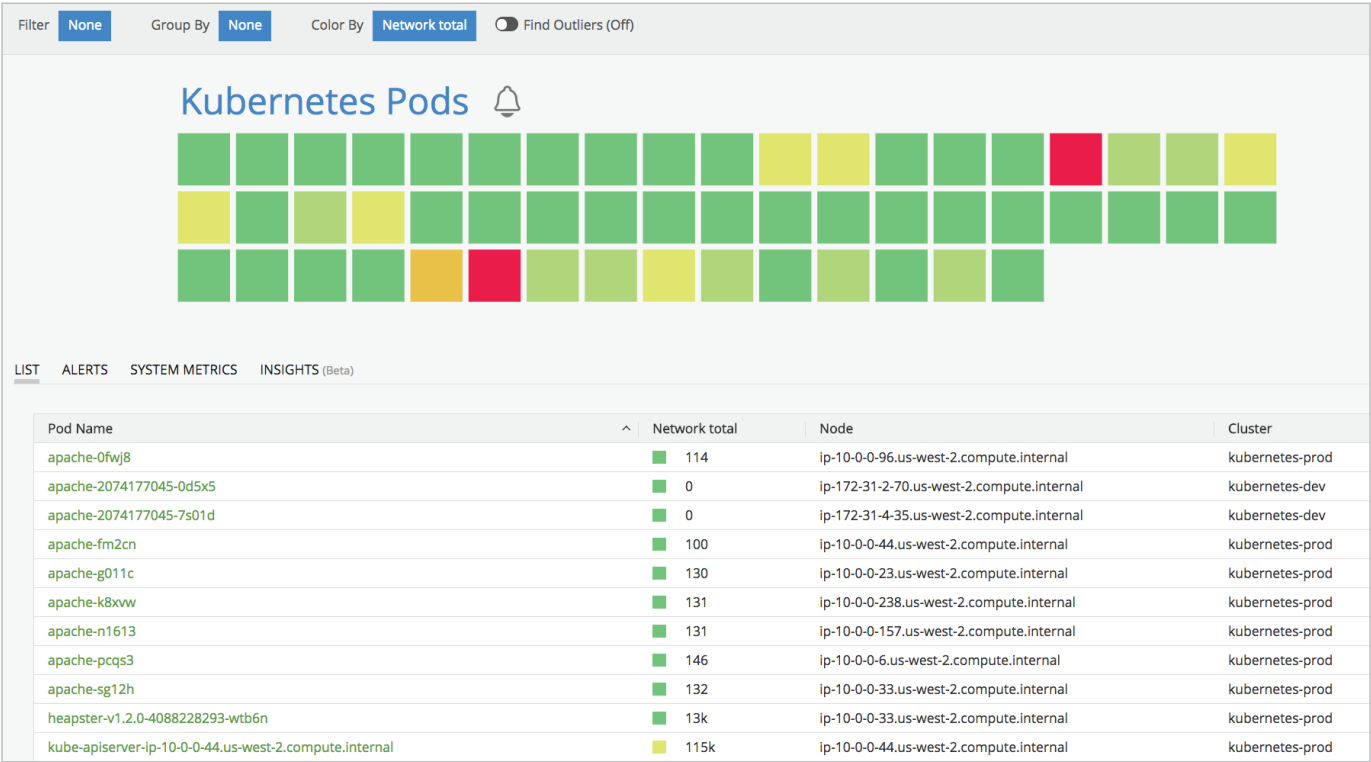
The Nodes view shows all of the nodes backing your Kubernetes clusters.
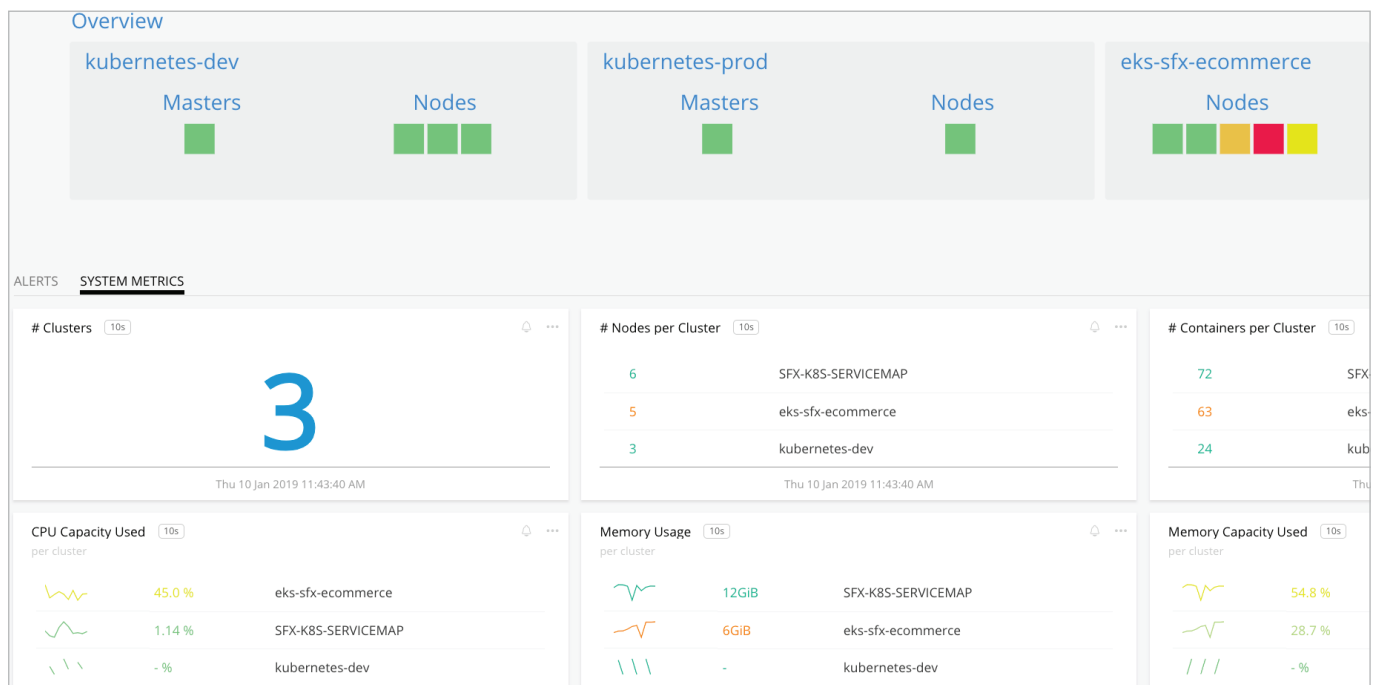
Drilling down to an individual node will display system metrics for that particular node, as well as dashboards for the specific services running on that node.
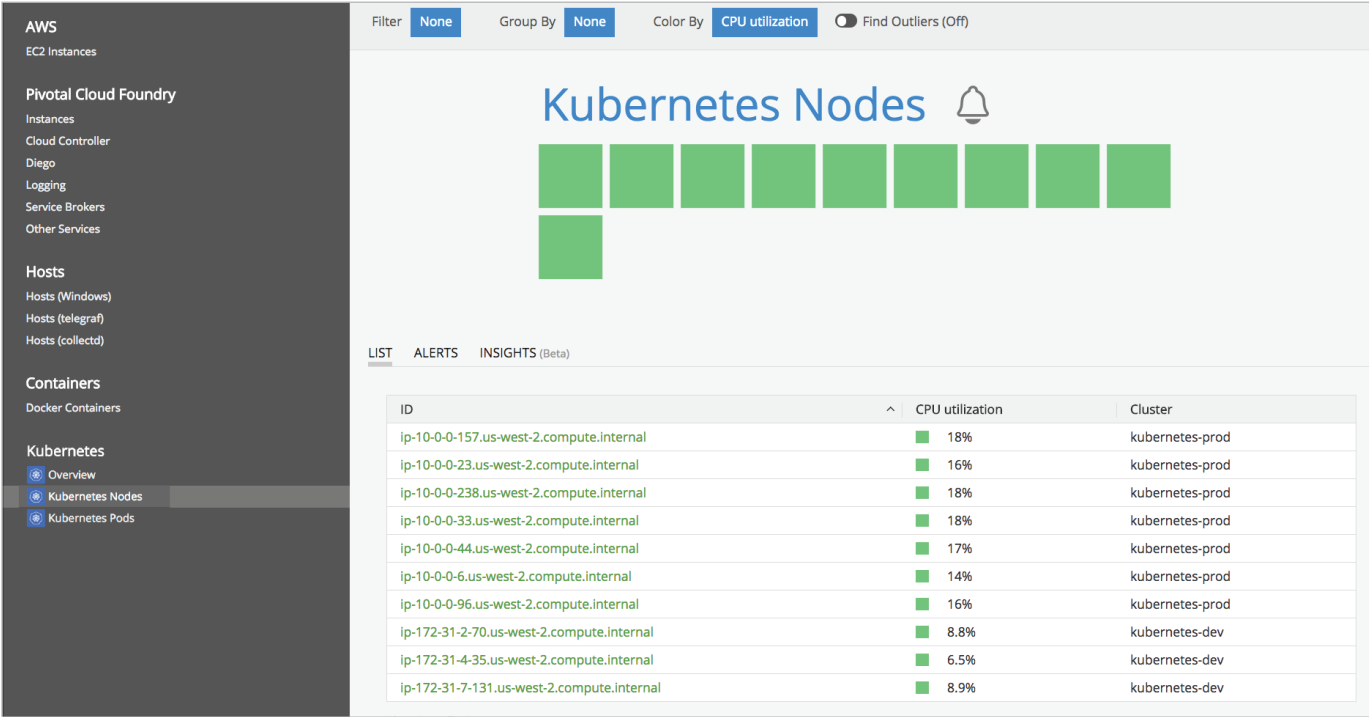
You can also view all of the pods running in your Kubernetes cluster, and track activity across a particular pod or across all of the pods in your cluster. From here, it's possible to drill down to individual Docker containers in each pod.

| Pod Name		Network total	Node	Cluster
apache-0fwj8	■	114	ip-10-0-0-96.us-west-2.compute.internal	kubernetes-prod
apache-2074177045-0d5x5	■	0	ip-172-31-2-70.us-west-2.compute.internal	kubernetes-dev
apache-2074177045-7s01d	■	0	ip-172-31-4-35.us-west-2.compute.internal	kubernetes-dev
apache-fm2cn	■	100	ip-10-0-0-44.us-west-2.compute.internal	kubernetes-prod
apache-g011c	■	130	ip-10-0-0-23.us-west-2.compute.internal	kubernetes-prod
apache-k8xvw	■	131	ip-10-0-0-238.us-west-2.compute.internal	kubernetes-prod
apache-n1613	■	131	ip-10-0-0-157.us-west-2.compute.internal	kubernetes-prod
apache-pcqs3	■	146	ip-10-0-0-6.us-west-2.compute.internal	kubernetes-prod
apache-sg12h	■	132	ip-10-0-0-33.us-west-2.compute.internal	kubernetes-prod
heapster-v1.2.0-4088228293-wtb6n	■	13k	ip-10-0-0-33.us-west-2.compute.internal	kubernetes-prod
kube-apiserver-ip-10-0-0-44.us-west-2.compute.internal	■	115k	ip-10-0-0-44.us-west-2.compute.internal	kubernetes-prod

Creating Custom Service Dashboards

SignalFx lets you view container metrics alongside other performance indicators to provide visibility across every layer of your environment. For example, a service owner who wants to monitor canary deployments might create the following dashboard, with container metrics displayed next to charts measuring request latency and an event feed that tracks code pushes, alerts, and any remediation actions are taken.
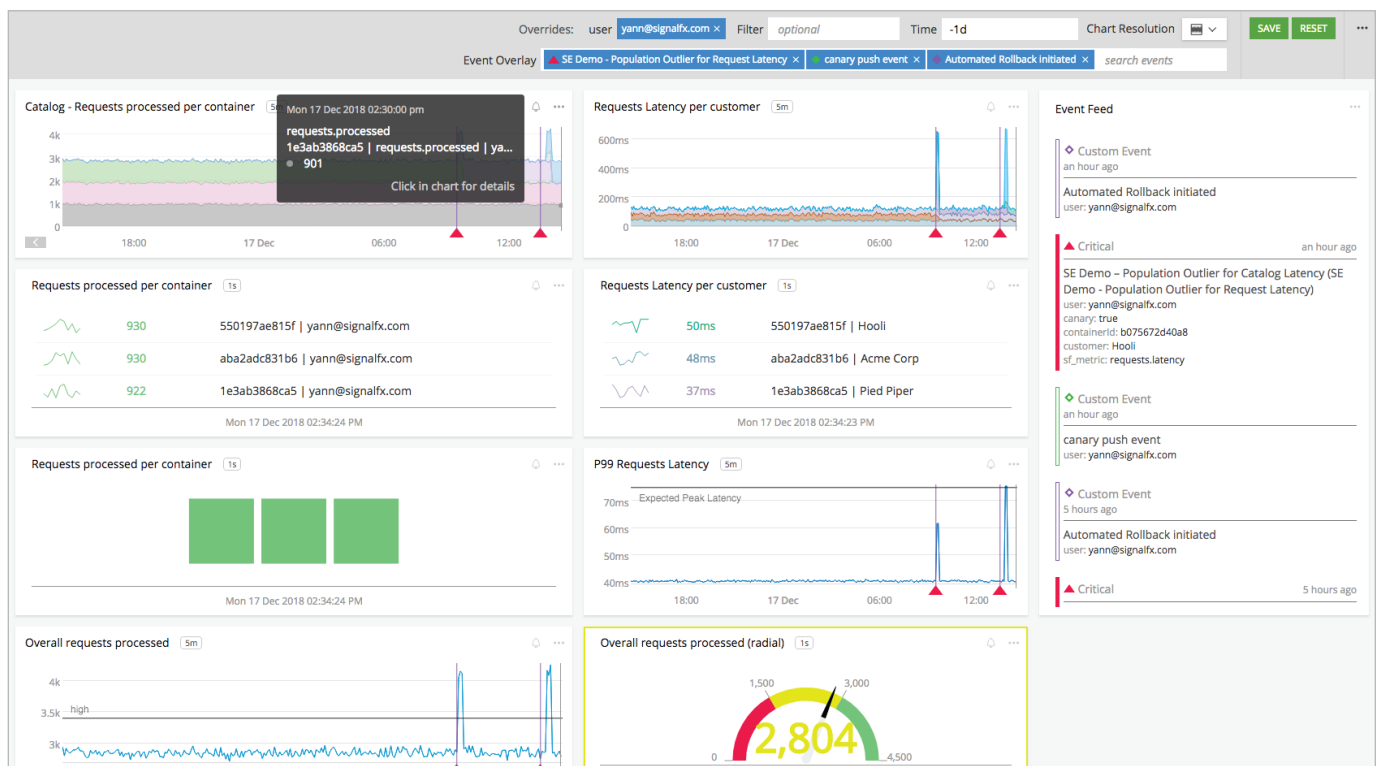
Detect Problems in Real-Time

With SignalFx, setting up alerts doesn't have to be a compromise between accuracy and timeliness. Proactively alert on issues by combining high-resolution metrics from your Kubernetes clusters with a library of statistical functions and algorithms in SignalFx. Preview alerts against historical data to tune them before deployment, and apply data science to alerting to avoid the false positives and alert storms common in highly ephemeral environments.
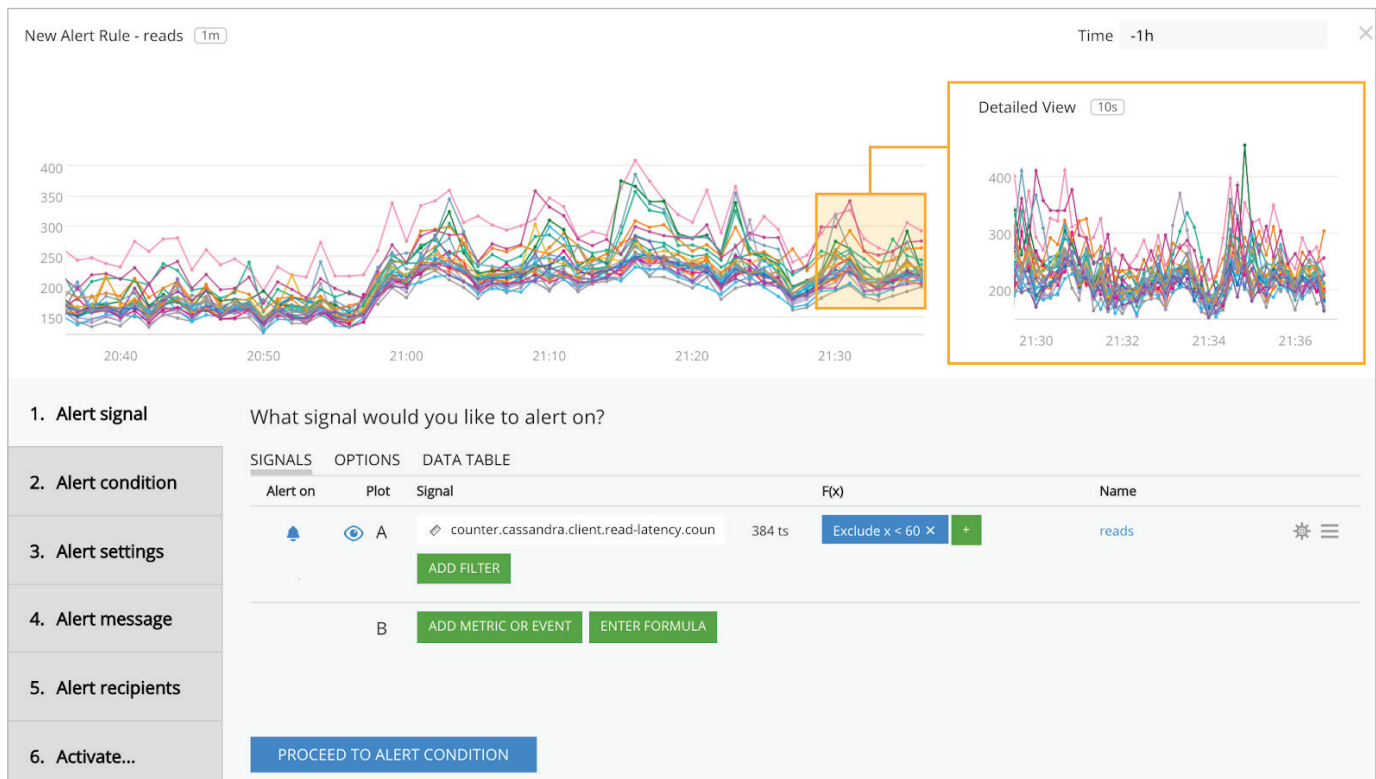
Leverage Distributed Tracing for Directed Troubleshooting

SignalFx **Microservices APM™** provides users with distributed tracing capabilities to dig deep when a performance-impacting event occurs. No matter where an issue arises in your Kubernetes environment, you can navigate from real-time alerts directly to application traces and correlate performance trends between infrastructure, Kubernetes, and your microservices. The SignalFx solution is built on a **NoSample™** Architecture that leverages tail-based tracing methodology to capture errors and anomalous request traces more consistently than traditional APM tools.

SignalFx Outlier Analyzer™ provides directed troubleshooting by surfacing most commonly represented patterns in outlier traces enabling you to quickly narrow down to anomalous Kubernetes nodes, cluster, cloud region or application specific labels or tags.
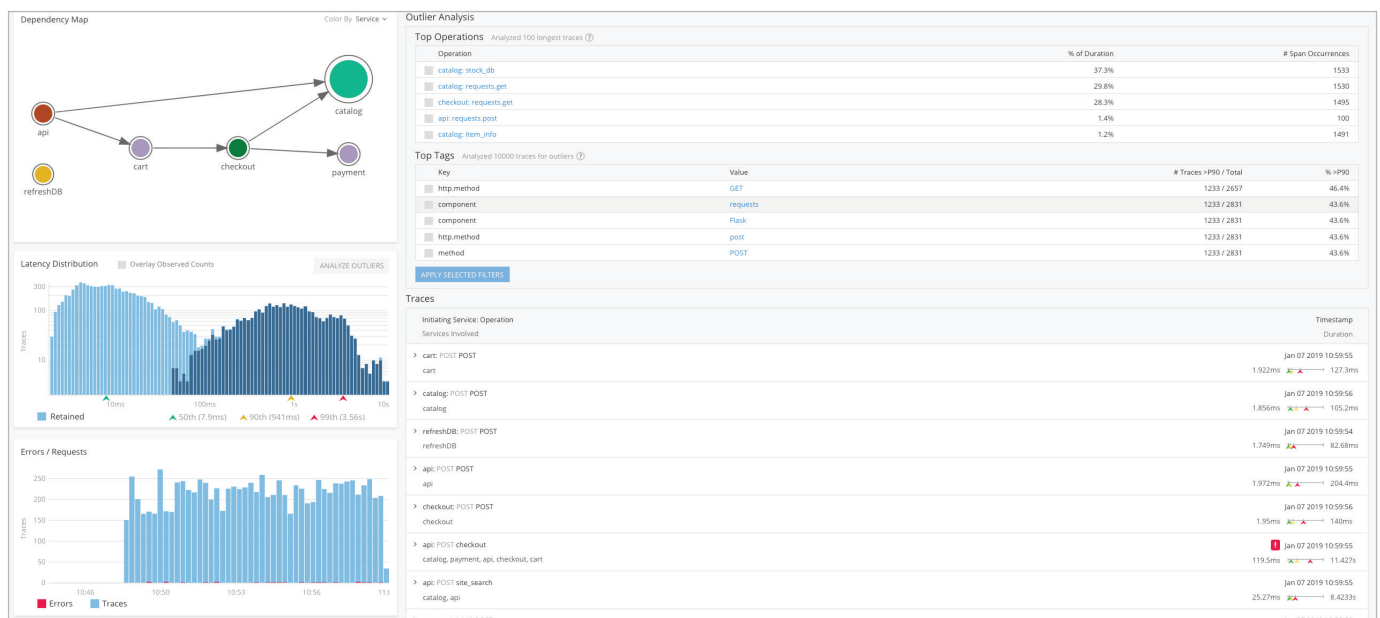
Increase DevOps Velocity

Once data from Kubernetes is flowing into SignalFx, DevOps teams can more easily leverage monitoring best practices and solutions to common problems across the entire organization. SignalFx provides out of the box, instant visibility into hosts, containers, pods, and Kubernetes objects with zero-touch, built-in dashboards, while also allowing teams to customize dashboards, alerts, and notifications for their specific needs.

Central teams can deliver monitoring as code by using the SignalFx API to programmatically create monitoring content and define analytics, and leverage the SignalFx Service Bureau to control usage, access, and editing permissions for users and teams.

```
curl \
   --request post \
 --header "x-sf-token: your_access_token" \
  --header "content-type: application/json" \
  --data \
  '{
        "name": "cpu load",
        "programtext": "data('cpu.load').publish()"
        }' \
 https://api.signalfx.com/v2/chart
```

SignalFx is Trusted
by Leading Organizations

Enterprises from every industry and across the globe are leveraging SignalFx to accelerate their journey to cloud-native and adopt Kubernetes in their organizations with confidence. Following is a list of just a few customers. For detailed case studies please visit here.
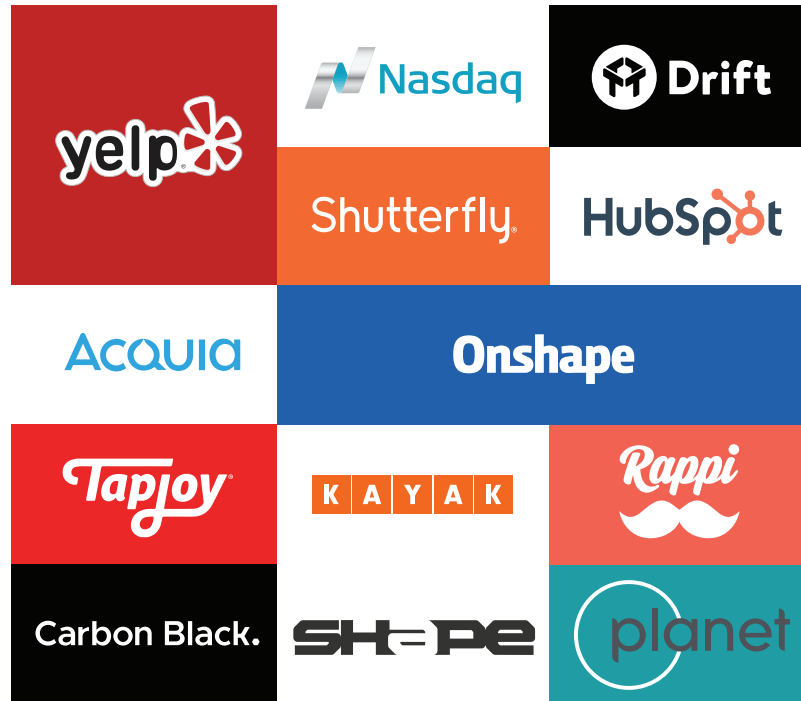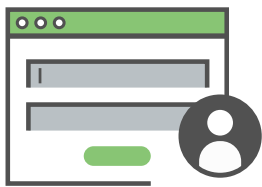
Get Started Today With SignalFx

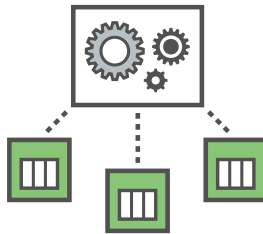Ready to learn how SignalFx can accelerate Kubernetes adoption in your organization?

Contact sales@signalfx.com, or **get started with a 14-day trial of SignalFx**.

Our customer success managers and solutions engineers will work with your teams to accelerate your Kubernetes adoption, and to monitor Kubernetes in real-time.

Sign up for a free trial

GET STARTED

Deploy SignalFx Smart Agent as a DaemonSet

LEARN MORE

Start monitoring Kubernetes environments

LEARN MORE

SignalFx

SignalFx

SignalFx is the only real-time cloud monitoring platform for infrastructure, microservices, and applications. The platform collects metrics and traces across every component in your cloud environment, replacing traditional point tools with a single integrated solution that works across the stack. SignalFx is built on a massively scalable streaming architecture that applies advanced predictive analytics for real-time problem detection. With its NoSample™ distributed tracing capabilities, SignalFx reliably monitors all transactions across microservices, accurately identifying all anomalies. Through data-science-powered directed troubleshooting, SignalFx guides the operator to find the root cause of issues in seconds. SignalFx is used by leading enterprises across high tech, financial services, consumer products, retail, communications, media, entertainment, and web-scale players like Yelp, HubSpot, Acquia, and Kayak. SignalFx is venture-funded by Andreessen Horowitz, Charles River Ventures, and General Catalyst.