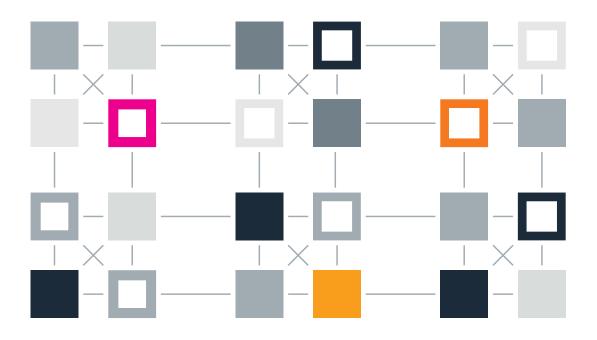
Monitoring Microservices on Kubernetes



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

This White Paper provides insights into how to best monitor both Kubernetes clusters and microservices deployed on Kubernetes in order to pinpoint performance issues, minimizing 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.

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.

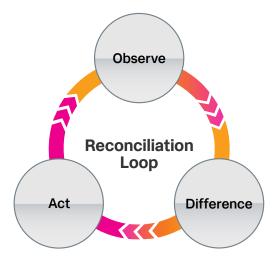
Infrastructure Abstraction	Declarative Configuration	Immutability
Application orchestration from the infrastructure resources such as compute servers, storage volumes, and networks	Kubernetes continuously monitors the state of the applications deployed and Kubernetes objects; It also implements the desired 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

There are three foundational pillars of the Kubernetes deployment philosophy

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:

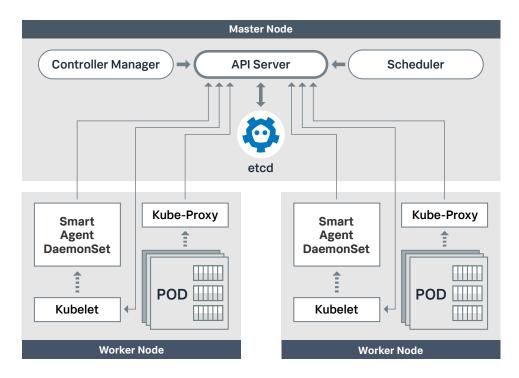
Automated Application Lifecycle Management	Cloud Interoperability	Self-Healing System
Kubernetes automates application lifecycle management as an end-to-end solution including application provisioning, autoscaling, and replication management as well as CI/CD	Kubernetes provides abstraction over the infrastructure 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.

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

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 survey from CNCF, complexity, monitoring, and security are amongst 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

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

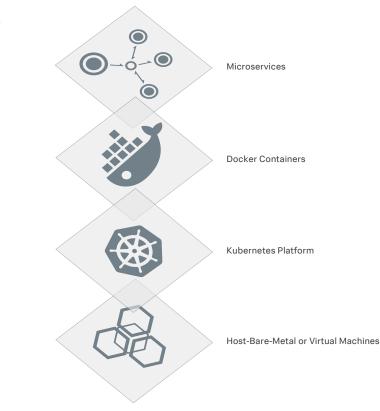
- · There are many more components to monitor
- · Containers are ephemeral and have unpredictable pod placement
- · Containerized microservices are more complex to troubleshoot

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



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 selfadjusting 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.

1				
	Number of Desired Pods			
	Number of Available Pods			
	Pods by Phase (failed, pending, running)			
	Desired Pods per Deployment			
PODS	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 Splunk

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.

Splunk 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, Splunk 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 Splunk with services such as AWS CloudWatch, Google Stackdriver, and Azure Monitor.

This approach is the most straightforward, and enables Splunk 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.

Fully Automated Kubernetes Monitoring

For greater insight into services and finer-grained monitoring of container metrics, we recommend installing the 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 Splunk 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. Zero-touch configuration with automatic discovery of Kubernetes components and containerized services instantly monitors the entire stack. The Smart Agent is installed with one simple step:

helm install splunk --set splunkAccessToken=<
YOUR_ACCESS_TOKEN> --set
clusterName=<YOUR_CLUSTER_NAME> splunk/
splunk-agent

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 Splunk.

Navigating Through Your Kubernetes Environment

Pre-Built Kubernetes Dashboards

Starting with the bird's eye view, Kubernetes Navigator enables teams to quickly understand the performance of the entire Kubernetes environment with intuitive and hierarchical navigation. Select, filter, or search for any Kubernetes entity and drill-down for detailed analysis, e.g., node, pod, and container level within seconds. Understand relationships between dynamic Kubernetes components and quickly fix interdependent performance issues arising from noisy neighbors.

← → C ③ Not Str DASHEGMRDS µAPM					"CLUSTER","property":"kuberne	and a state of the	gnito 💮 🌖
Mastructure Type Kubernetes	MAP NODES WORKLOW	DS NODE DETAIL WOR	WLOAD DETAIL FOD D	ETAIL CONTAINER DE	TAIL	ANALYZER INFO	
Cluster optional	Filter optional					Detail on Node: ip-10-4-79-226.ec2.inter	rnal
Cluster Map						Workloads on this Node	
	1-1-0					Search workloads	
	lab0	-				Workload	Wa
						ave-node	Dav
						blacklist-tags-cron-1582606140	Jot
						Wed 11 Mar 2020 01:55:10 P Showing 25 of 35 - Show more	PM
				©	lp-10-4-87-95.ec.2.internal Pods 51 Node Condition Node Ready	CPUM Used by Pod (m) 1	hded.

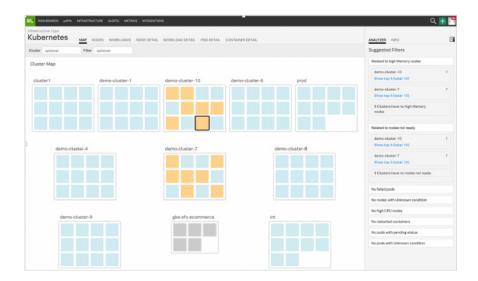
Drilling down to an individual node will display system metrics for that particular node, as well as dashboards for the specific services running on that node.

Cluster gka-cluster × Node Name kuber	metes_node							
Node Properties		Workloads on this Node			Search workloads			Group By Nor
		Workload	Workload Type	Pending Pods	Running Pods	Succeeded Pods	Failed Pods	Unknown Peds
		1.2.0-splunk-kubernetes-logging	DeemonSet	0	0	0	0	0
		1.2.0-splunk-kubernetes-metrics	DeemonSet	0	0	0	0	0
Chart filters result in ambiguous data s Node Properties	stream for	1.2.0-splunk-kubernetes-metrics-agg	Deployment	0	0	0	0	0
Thomas Programma		1.2.0-splunk-kubernetes-objects	Deployment	0 Sun 15 Mar	0 2020 0447:35 PM	0	0	0
200 Node CPU 1 my-release 14.1000	00% resolution		Netwo	rk in by Pod 100		10MB BMB	at by Pod 100	L.
Node CPU 9 10 10 10 10 10 10 10 10 10 10	% used by pod - sysdig-kw19p i 00% resolution e rollups applied Click	igke-standard-c	TCMB 5MB		16-40 16-45	10A48 8A48 6A48 2A48 2A48 0	ut by Pod 190 16.35 16	40 16.45
200 100 100 100 100 100 100 100	% used by pod - sysdig-kw19p i 00% resolution e rollups applied Click	Igke-standard-c 105 Ito source data) In chart for details	TCMB 5MB		16.40 16.48	10A48 8A48 6A48 2A48 2A48 0		40 16.45
200 100 100 100 100 100 100 100	% used by pod - sysdig-kw19p i 00% resolution e rollups applied Click	Igke-standard-c 105 Ito source data) In chart for details	TCMB 5MB			10A48 8A48 6A48 2A48 2A48 0		Search containe
Containers on this Node	% used by pod e-sysdig-kwf9p i 00% resolution of the spinor Click United ta5	Igke-standard-c 105 Ito source data) In chart for details	TCMB 5MB	1635		10A48 8A48 6A48 2A48 2A48 0	16.35 16	Search containe
Add CPU myresec 100 100 100 100 100 100 100 10	% used by pod exysdig-kwt9p 00% recolution Click Click 45	(de mandero e 50 50 50 50 50 50 50 50 50 50	TCMB 5MB	1635	# bernetes-metrics	10A48 8A48 6A48 2A48 2A48 0	16.35 16. Namespa	Search containe
Definition of the second secon	% used by pod exysdig-kwt9p 00% recolution Click Click 45	lgle standard c 199 0 Bosene dia) 10 dia statu fu detala 10 dia titu fu detala 10 dia titu fu detala 10 dia titu fu detala 10 dia titu fu detala	TCMB 5MB	1635 Verkloed Nam 1,20-splank-ko kube dra-7901	# bernetes-metrics	10A48 8A48 6A48 2A48 2A48 0	16.35 16 Rerector gbox	Search containe see CPU% 0% tem 0%
And CPU Survey S	% used by pod exysdig-kwt9p 00% recolution Click Click 45	(de exandre e 10) 20 autor dita) 20 auto	TCMB 5MB	1635 Verkloed Nam 1,20-splank-ko kube dra-7901	e demetes-metrics 18754c5 scaler-bb58c6784	10A48 8M8 6A68 2M8 2M8 0	16.35 16 Ramespa splank kabe-syst	Search containe ce CPU% 0% tem 0%

You can also view all of the pods running in your Kubernetes cluster, and track activity across a particular pod or across all the pods in your cluster. From here, it's possible to drill down to individual Docker containers in each pod.

Kubernetes Analyzer

Al-driven analytics automatically surfaces insights and recommendations to precisely answer, in real-time, what is causing anomalies across the entire Kubernetes cluster — nodes, containers, and workloads. Sophisticated algorithms, including Historical Performance Baselines and Sudden Change, detect system-level issues such as a sudden increase in Goroutines or container restarts and alert within seconds.



Logs in Context

Seamlessly pivot to logs and get granular visibility into application, Kubernetes, and container logs to correlate performance across the entire stack without any context switching. Visibility into lifecycle events of Kubernetes and API Server Audit logs help you understand and maintain your security and compliance postures.

tap : gin-cluster : gite-standard-cluster-1-E-pare	interse-Dev-poor-rizele4b41-p1su				
			ANALYZER INFO	2 D	=
			Detail on F payments t549p •••	Pod: ervice-6599bd64b6-	
			Examine in Splunk 🕞		
			Configure data links		
			Name	paymentservice- 6599bd64b6-t549p	
splunk>enterprise App: Search	& Reporting *		Pod ID	5a687f7f-704f-11ea-	1
Inneh Analytics Patronets	Basada Baste Da			ba4f-42010a800114	e Door
Search Analytics Datasets	Reports Alerts Da	ashboarda	Pod age	ba4f-42010a800114 136m	& Repo
Search Analytics Datasets	Reports Alerts Da	sihboerds	Pod age Pod phase		& Repo
	vLcz-6599bd54b6-t543p*	sinboards		136m	
New Search 1 Index**Balm hostmare**paymentser 2998 of 2,205 events matched No Ev	rvLet-6599bd54b6-t543p* vert Sampling =	antoards	Pod phase	136m Running	* Ci 36 *
New Search Lindex/mails hostnares*"paynestser 2,998 of 2,205 events matched No Events Events (2,198) Patterns Statistic	rvLet-6599bd54b6-t543p* vert Sampling =	Noticed	Pod phase	136m Running gke-standard-cluster-1-	* Ci 36 *
New Search Index*main hostname**paynestser 2098 of 2,205 events matched No Events Nemte (2,398) Patterns Stastster	rvLce-6599bd84b6-t548p* vert Sangling = cs Visualization		Pod phase	136m Running gke-standard-cluster-1-	 Cl se ▼ ≡est Mod
New Search Index*main hostname**paynestser 2098 of 2,205 events matched No Events Nemte (2,398) Patterns Stastster	rvLce-6599bd84b6-t548p* vert Sangling = cs Visualization		Pod phase	136m Running gke-standard-cluster-1-	 Cl se ▼ ≡est Mod
New Search Lindex/mails hostnares*"paynestser 2,998 of 2,205 events matched No Events Events (2,198) Patterns Statistic	vutce-6599041466-1541p* void Sampling = cs Visualization + Zaom to Solection	× Deserved	Pod phase Node	136m Running gke-standard-cluster-1-	▼ Cl sel ▼ Enst Med Enst Med

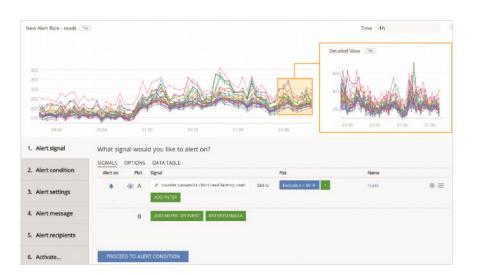
Creating Custom Service Dashboards

Splunk 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.

	c user yumillagaalt.com × Filter opsisoo' Time -1d	Chart Resolution 📄 🗸 56xt 82587 ***
Event Overlay	8 Demo - Population Guiller for Request Lanercy × 👔 caracy push event × 🔹 Automated Ralback	initiated × search events
Catalog - Requests processed per container () won 10 two 2010 022000 pm 0	Requests Latency per customer (5m) 0	Deerst Feed
601 Click in chart for details	400ml	an hour ago Automated Rol back Initiated user: pam@sgnatk.com
1850 17 Dec 0650 12.60	18:00 17 Dec 06:00 12:00	▲ Critical an hour ago
Requests processed per contailer: (3) ↓	Requests Latency per outsomer (%) ↓ ~~√ Storms SS0195/self351 Hool8 ~_√ 48ms abs23dd53196 Aome Cepp	SE Damo - Population Outlier for Catalog Latency (SE Demo - Population Outlier for Request Latency) user paint@againf.com catagory trite containent b0%P23ddbg catagorier Heal d_metric megastrubmeg
V/v 922 fe3ab366ca5 yann@signaltu.com	VV 37ms fe3ab3868ca5 Pied Piper	
Won 17 Dec 2018 02:04:34 PM	Mon 17 Dec 2018 02:34:23 FM	Custom Event an hour ago
Requests processed per container (3)	P93 Requests Latency (Sn) 0	canary push event une: parnifeignatik.com
	Gitrus	S hoars ago Automated Rollback initiated user: pam@tigmatk.com
Mon 17 Dec 2016 02:54/24 PM	40mm 18:00 17 Dec 06:00 12:00	Critical Shours ago
Overall requests processed (an)	Overall requests processed (radia) 📧 🔅 —	
a and a second a	2,804	
a second a second se	0	

Detect Problems in Real-Time

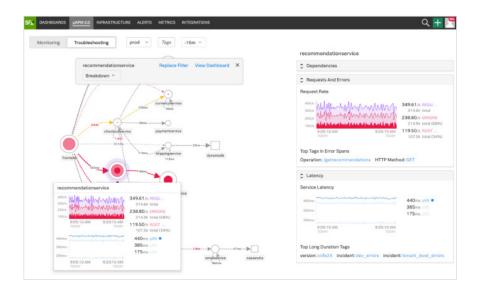
With Splunk, setting up alerts doesn't have to be a compromise between accuracy and timeliness. Proactively alert on issues by combining high-resolution metrics from your Kubernetes clusters with a library of statistical functions and algorithms in Splunk. Preview alerts against historical data to tune 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

Splunk 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. Splunk APM is built on a NoSample™ Full-Fidelity architecture that ingests and analyzes all traces so outliers and anomalies never go undetected.

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 Splunk, DevOps teams can more easily leverage monitoring best practices and solutions to common problems across the entire organization. Splunk 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 Splunk API to programmatically create monitoring content and define analytics, and leverage 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.splunk.com/v2/chart
```

Splunk is Trusted by Leading Organizations

Enterprises from every industry and across the globe are leveraging Splunk to accelerate their journey to cloud-native and adopt Kubernetes in their organizations with confidence. Following is a list of just a few customers.

More information: https://www.splunk.com/en_us/customers.html



Get Started Today with Splunk Infrastructure Monitoring

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

Our customer success managers and solutions engineers will work with your teams to accelerate your Kubernetes adoption, and to monitor Kubernetes in real-time. Get started with a 14-day free trial: https://www.observability.splunk.com



To learn more about Kubernetes Navigator for real-time monitoring and troubleshooting Kubernetes environments, at any scale, download the free **Kubernetes Navigator Data Sheet**.

splunk>

Learn more: www.splunk.com/asksales

www.splunk.com

Splunk, Splunk >, Data-to-Everything, D2E and Turn Data Into Doing are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names or trademarks belong to their respective owners. © 2020 Splunk Inc. All rights reserved.