

# Full Stack Visibility with Kubernetes in 15 Minutes



DATADOG

# Trivia

Why do they call it **K8s!**?





# Kubernetes with Datadog

A look from 50,000 feet

## Integration Auto Discovery

### LIVE CONTAINER MONITORING



**Infrastructure Metrics**  
Custom and out-of-the-box

**Logs & Events**  
Structured and unstructured

**Tags/Metadata**  
Custom and out-of-the-box

**Retention**  
15 months by default (extendable on request)

**Granularity**  
No roll ups, full granularity

**Secure Focused**  
Encrypted at Rest, SOC 2 Compliant

**APM**  
Structured and unstructured

**Highly Available**  
Spanning multiple AZ's

**Encrypted Communication**  
Outbound only via HTTPS/443

**System Metrics**  
80+ CPU, Disk, Load, Network, etc.

**250+ Integrations**  
Kubernetes, Docker, AWS (Lambda, S3, etc.), Ansible, Kubernetes, MapR and more

**High resolution**  
15 sec host granularity, 1 sec. custom and business metric granularity



# Deploying Datadog in K8s

Let's start with the basics



# Deployment Options

## Host Based

- Additional Visibility
- Monitor Outside of K8s
- More Granular Configuration
- No Auto-Deployment without Config Management

## Container Based

- Deploy as a *DaemonSet*
- Simplified Management
- Deploy Quickly Everywhere
- Easier to Deploy and Configure

[https://docs.datadoghq.com/agent/kubernetes/host\\_setup/](https://docs.datadoghq.com/agent/kubernetes/host_setup/)



# Tagging and Other Best Practices

Making sense out of all of this data



# Good Tagging Practices

## What Should You Tag?

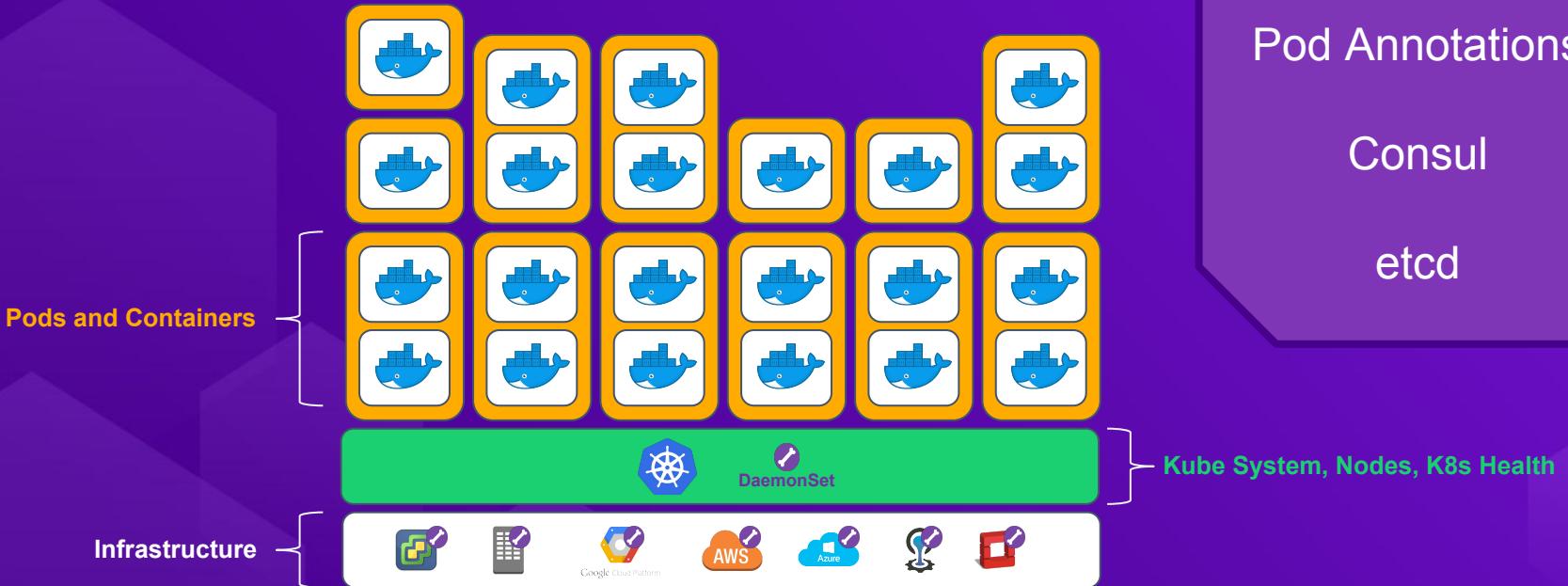
Applications	Roles
Services	Customers
Components	Business Units
Teams	Stores
Departments	Regions
Cost Centers	etc...

## Why It Matters:

If we can't see or alert on what we want when we need to then monitoring anything has little point!

# Where should we tag?

And what tags do we already get?





# Kubernetes State Metrics

## What are these?

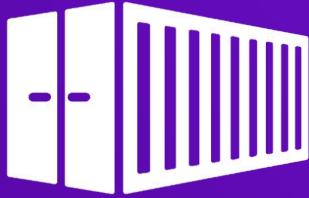
***kube-state-metrics*** is a simple service that listens to the Kubernetes API server and generates metrics about the state of the objects.

It is not focused on the health of the individual Kubernetes components, but rather on the health of the various objects inside, such as deployments, nodes and pods.

## How do I get them?

1. Download the [Kube-State manifests folder](https://github.com/kubernetes/kube-state-metrics/tree/master/kubernetes) from GitHub:  
<https://github.com/kubernetes/kube-state-metrics/tree/master/kubernetes>
2. Apply them to your Kubernetes cluster:  
`kubectl apply -f <KUBE_STATE_FOLDER>`

[https://docs.datadoghq.com/agent/kubernetes/host\\_setup/](https://docs.datadoghq.com/agent/kubernetes/host_setup/)



# Building on the Basics

Going further with Integrations, Logs, and APM

# Integrations in K8s

Getting Into the Middleware

# Two Approaches to Integration

## Mount a Local Directory

- Must deploy configuration to every node
- Have to manage configurations and track changes across nodes
- Config management can ease deployment.

## Use Config Maps

- Manage configurations in Kubernetes configs
- Easy to track configurations
- Change configurations quickly and across-the cluster

<https://docs.datadoghq.com/agent/kubernetes/integrations/>

# Live Container Monitoring

Monitoring Is All About Good Process

# Setting Up Live Containers

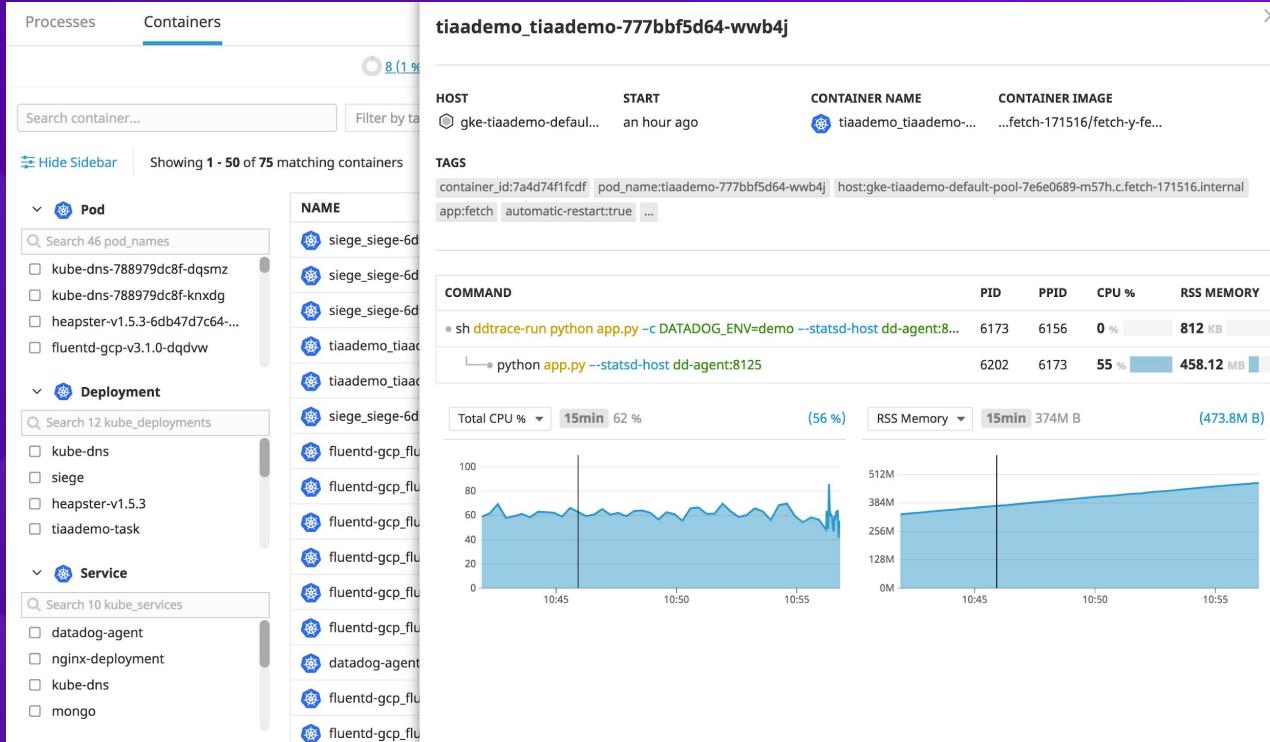
## Live Container Configuration

```
(...)  
  env:  
    (...)  
      - name: DD_PROCESS_AGENT_ENABLED  
        value: "true"  
  volumeMounts:  
    (...)  
      - name: passwd  
        mountPath: /etc/passwd  
        readOnly: true  
  volumes:  
    (...)  
      - hostPath:  
          path: /etc/passwd  
          name: passwd  
(...)
```



You'll get live processes  
on your nodes too!

# What you should see:



# Logs in K8s

All Your Logs Are Belong To Us

# Setting Up Log Collection

## Basic Logs Configuration

```
(...)
env:
  (...)

  - name: DD_LOGS_ENABLED
    value: "true"
  - name: DD_LOGS_CONFIG_CONTAINER_COLLECT_ALL
    value: "true"
(...)
```

## Tracking Our Read Location

```
(...)
volumeMounts:
  (...)

  - name: pointerdir
    mountPath: /opt/datadog-agent/run
(...)

volumes:
  (...)

  - hostPath:
      path: /opt/datadog-agent/run
      name: pointerdir
(...)
```

# What you should see:

Log Explorer Save As

2

0 09:20 09:25 09:30 09:35 09:40 05

Filters Saved Views < Hide sidebar | 24 results found

Manage Facets 27 of 27

Core

Source docker 24

Host

Service docker 24

Status Error 0 Warn 0 Info 24

Others

Availability zone

Name

DATE ↓ HOST

DATE	HOST
Oct 05 10:19:09.069	gke-tiaademo
Oct 05 10:17:58.735	gke-tiaademo
Oct 05 10:17:36.981	gke-tiaademo
Oct 05 10:16:27.898	gke-tiaademo
Oct 05 10:16:02.934	gke-tiaademo
Oct 05 10:14:51.566	gke-tiaademo
Oct 05 10:14:24.640	gke-tiaademo
Oct 05 10:13:15.836	gke-tiaademo
Oct 05 10:12:52.684	gke-tiaademo
Oct 05 10:11:40.435	gke-tiaademo
Oct 05 10:11:09.686	gke-tiaademo
Oct 05 10:09:59.322	gke-tiaademo
Oct 05 10:09:26.638	gke-tiaademo
Oct 05 10:08:04.880	gke-tiaademo
Oct 05 10:07:41.635	gke-tiaademo
Oct 05 10:06:30.226	gke-tiaademo
Oct 05 10:06:07.301	gke-tiaademo

INFO Oct 05, 2018 at 10:17:58.735 (2 minutes ago) View in Context ×

HOST gke-tiaademo-default-pool-7e6e docker SOURCE docker

CONTAINER NAME k8s\_fluentd-gcp-scaler\_fluentd- DOCKER IMAGE gcr.io/google-containers/fluentd-g POD NAME fluentd-gcp-scaler-7c5db745fc-t4hr

TAGS

display\_container\_name:fluentd-gcp-scaler\_fluentd-gcp-scaler-7c5db745fc-t4hmj  
docker\_image:gcr.io/google-containers/fluentd-gcp-scaler:0.3  
kube\_replica\_set:fluentd-gcp-scaler-7c5db745fc kube\_namespace:kube-system  
pod\_name:fluentd-gcp-scaler-7c5db745fc-t4hmj short\_image:fluentd-gcp-scaler  
image\_name:gcr.io/google-containers/fluentd-gcp-scaler image\_tag:0.3  
container\_name:k8s\_fluentd-gcp-scaler\_fluentd-gcp-scaler-7c5db745fc-t4hmj\_kube-system\_564378e2-c683-11e8-9ed9-42010a9600f9\_0  
service:docker ...

2018-10-05T14:17:58,704932780+00:00 Running: kubectl set resources -n kube system fluentd-gcp-v3.0.0 -c fluentd\_gcp --requests=cpu=100m,memory=200Mi --limits=memory=300Mi

ATTRIBUTES

# APM in K8s

Getting Down To The Code



# Setting Up APM and Trace Search

## APM Configuration

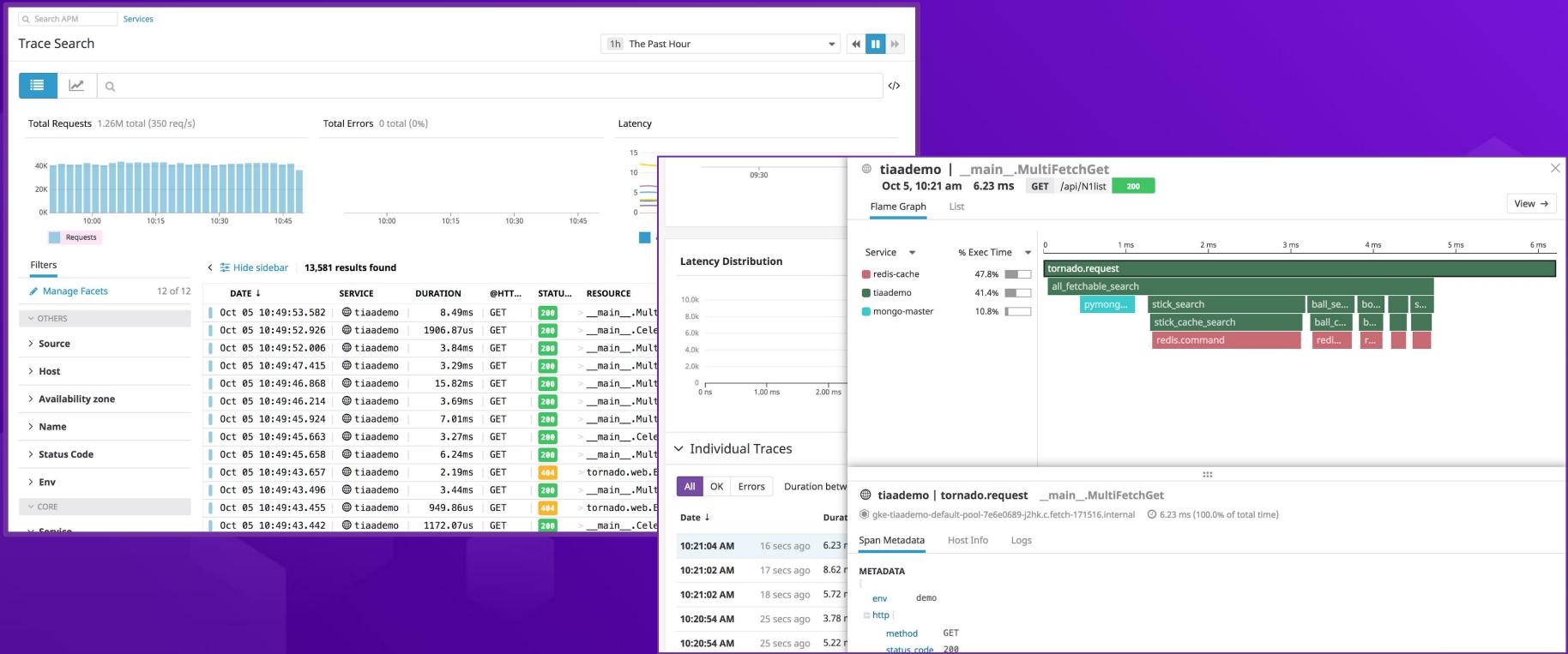
```
(...)  
env:  
(...)  
- name: DD_APM_ENABLED  
  value: "true"  
(...)
```

## Trace Search Configuration

```
(...)  
env:  
(...)  
- name: DD_APM_ANALYZED_SPANS  
  value: "YOURAPP|span.name=1"  
(...)
```

[https://docs.datadoghq.com/agent/kubernetes/daemonset\\_setup/#trace-collection](https://docs.datadoghq.com/agent/kubernetes/daemonset_setup/#trace-collection)

# What you should see:



# Auto Discovery

Monitor Anything and Everything



# Setting Up AutoDiscovery

## Docker Labels

LABEL

```
"com.datadoghq.ad.check.names"="[<CHECK NAME>]"
```

LABEL

```
"com.datadoghq.ad.init.configs"="[<INIT CONFIG>]"
```

LABEL

```
"com.datadoghq.ad.instances"="[<INSTANCE CONFIG>]"
```

```
LABEL "com.datadoghq.ad.logs"="[<LOGS CONFIG>]"
```

## Pod Annotations

annotations:

```
ad.datadoghq.com/<container identifier>.check.names: '[<CHECK NAME>]'  
ad.datadoghq.com/<container identifier>.init.configs: '[<INIT CONFIG>]'  
ad.datadoghq.com/<container identifier>.instances: '[<INSTANCE CONFIG>]'  
ad.datadoghq.com/<container identifier>.logs: '[<LOG CONFIG>]'
```

For Annotations, Autodiscovery identifies containers by *name*, NOT image (as it does for auto-conf files and key-value stores). That is, it looks to match <container identifier> to .spec.containers[0].name **not** .spec.containers[0].image

<https://docs.datadoghq.com/agent/autodiscovery/?tab=docker#template-source-kubernetes-pod-annotations>

# AutoDiscovery with a Key/Value Store

## Environment Variables

```
/datadog/  
  check_configs/  
    docker_image_1/ # container identifier,  
      - check_names: [<CHECK_NAME>]  
      - init_configs: [<INIT_CONFIG>]  
      - instances: [<INSTANCE_CONFIG>]
```

## etcd, Consul, Zookeeper

```
etcdctl mkdir /datadog/check_configs/httpd  
  
etcdctl set /datadog/check_configs/httpd/check_names '["apache"]'  
  
etcdctl set /datadog/check_configs/httpd/init_configs '[{}]'  
  
etcdctl set /datadog/check_configs/httpd/instances  
'[{"apache_status_url":  
"http://%%host%%/server-status?auto"}]'
```

<https://docs.datadoghq.com/agent/autodiscovery/?tab=docker#template-source-key-value-store>



# Demo Time

Let's see what this looks like in the wild...

# Going further...

Advanced scaling and K8s cluster monitoring with Datadog

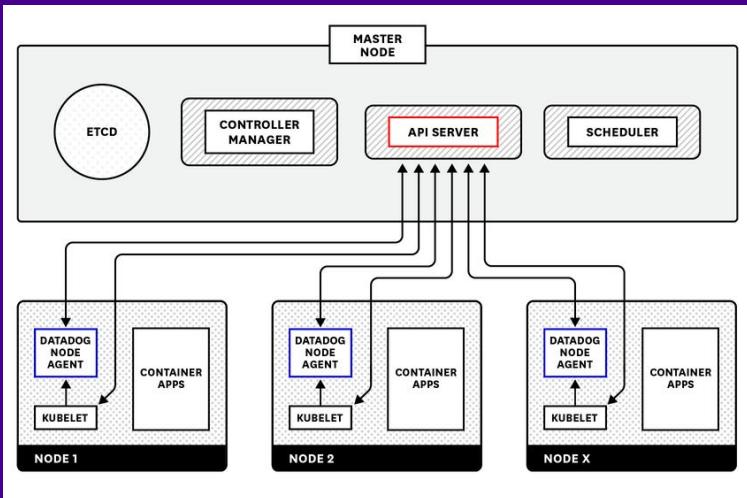


# Datadog Cluster Agent

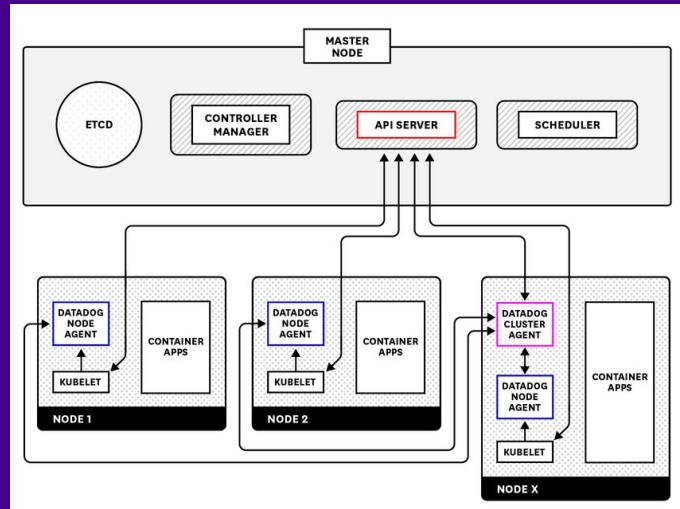
Scaling up to thousands of nodes gracefully...

# Cluster Agent Concepts

## Simple Deployment



## With the Cluster Agent



<https://docs.datadoghq.com/agent/kubernetes/cluster/>



# Autoscale With Any Metric

It's like Burger King... You can have it your way!

# Set Up Horizontal Pod Scaling

*Use the Datadog Cluster Agent  
as an External Metrics Provider!*

Scale your services using any Datadog metric...

<https://www.datadoghq.com/blog/autoscale-kubernetes-datadog/>



# Questions?

# Thank you!



**Mike Moore**

michael.moore@datadoghq.com

Find this presentation and related code at:

<https://dtdg.co/ddk8s>



[webofmike.com](http://webofmike.com)



[themsquared](#)



@themsquared