GRAFANACON
AMS2018

Kubernetes & Grafana

Jacob Lisi

What is Kubernetes?

“Kubernetes is a portable, extensible open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation.“

(https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/, 2018)

Collecting K8s Metrics

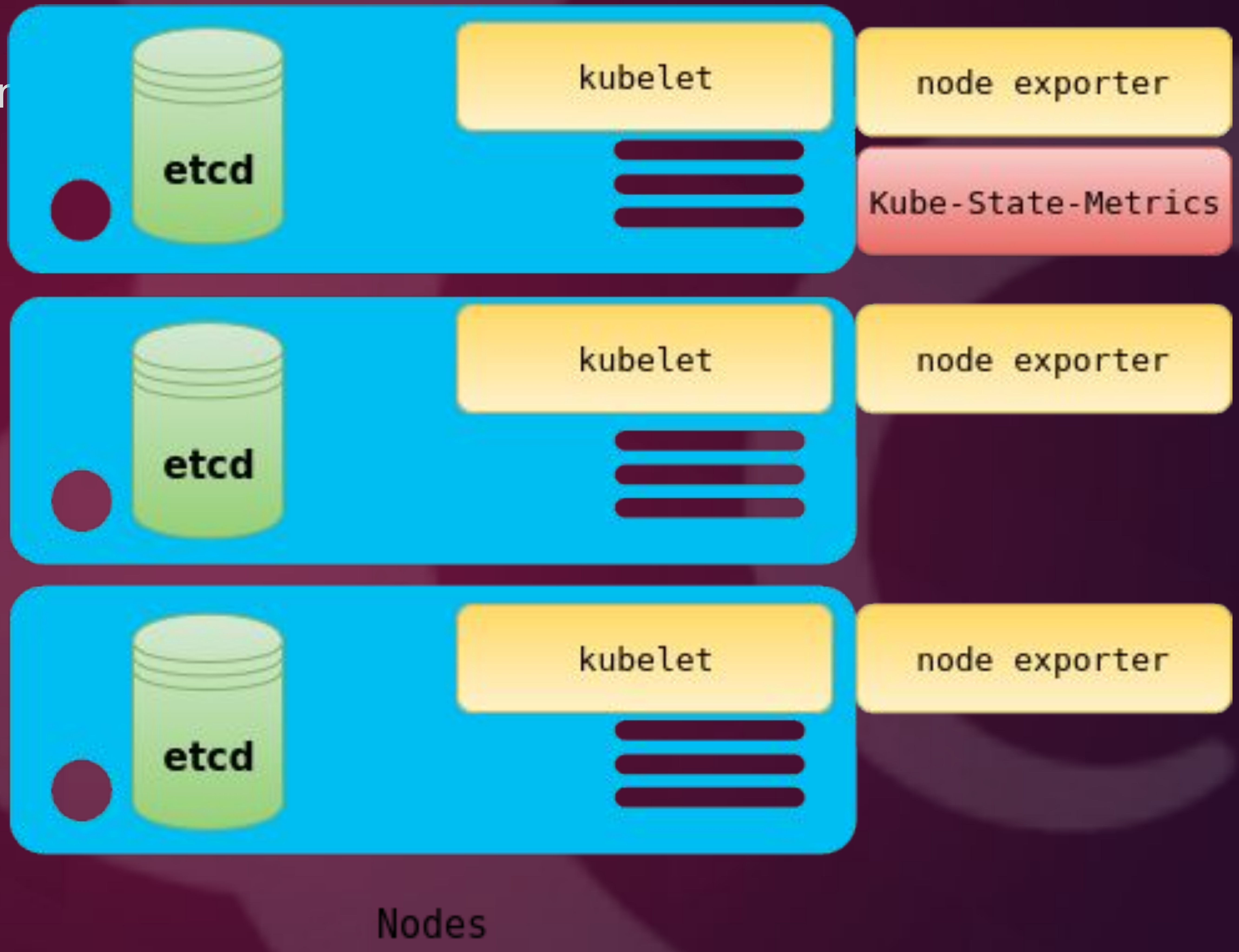
Service Discovery

Prometheus^ is a great kubernetes monitorin tool.

Collecting metrics is as simple as ‘`promtheus.io/scrape`’

Recommended Metrics:

- kubelet cAdvisor
- kube-api-server
- kube-state-metrics
- etcd
- node exporter

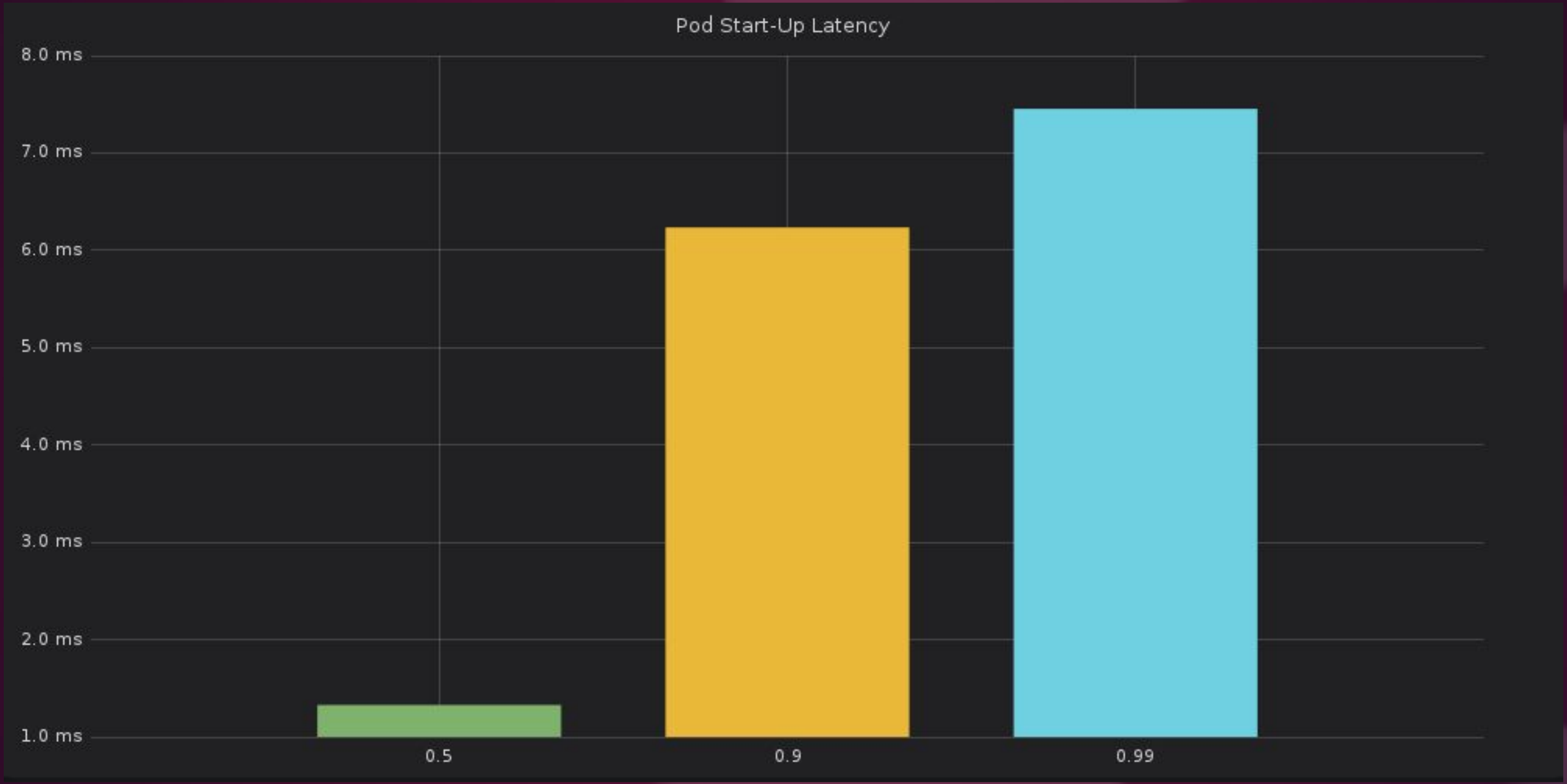


Make sure it works as promised...

Start with the SLOs:

- **Pod startup time:** 99% of pods and their containers (with pre-pulled images) start within 5s.
- **API-responsiveness:** 99% of all API calls return in less than 1s

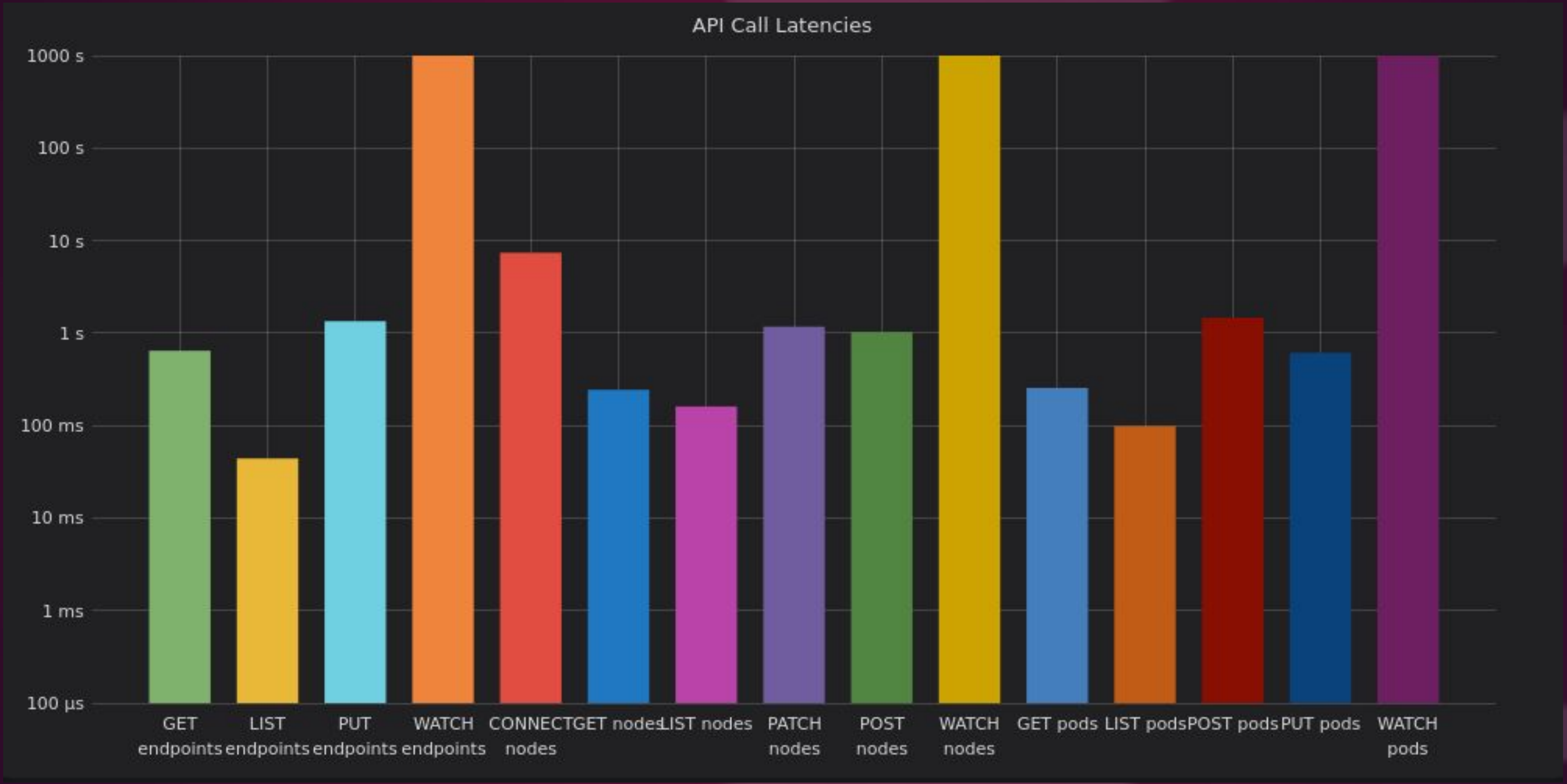
Is your cluster working?



PromQL Query:

```
avg (rate(kubelet_pod_start_latency_microseconds[2m]) > 0) by (quantile)
```

Is your cluster working?



PromQL Query:

```
avg (apiserver_request_latencies_summary{quantile="0.99", resource=~"nodes|pods|endpoints"}) by (verb,resource)
```

Metadata

Kubernetes Metadata

Annotations

```
"annotations": {
  "kubernetes.io/key/1" : "value1",
  "kubernetes.io/key/2" : "value2"
}
```

Machine readable metadata consumed by tooling and system extensions

Labels

```
"labels": {
  "key1" : "value1",
  "key2" : "value2"
}
```

Human readable metadata to facilitate the organization and management of API resources

Lots of Metadata!

Any large organization will end up with inordinate amounts of metadata from their kubernetes cluster…

Problems?

Implicit Tags

$host.cpu.system

Implicit Tags get messy

```
$region.$zone.$network.$app.$host.cpu.system
```

Implicit Tags get messy and differs across orgs

$region.$zone.$app.$host.cpu.system

$region.$zone.$app.$host.cpu-seconds.system

$region.$zone.$app.$host.cpu.system.seconds

$region.$zone.$network.$env.$app.$host.cpu.system

$regionID.$region.$zone.$network.$app.$host.cpu.system

snap the open telemetry framework

collectd

Ganglia

DATADOG

Kubernetes Tags Explicitly, So Should You

```
Container_cpu_system_seconds_total{$region.$zone.$app.$host}

Container_cpu_system_seconds_total{$region.$zone.$app.$host}

Container_cpu_system_seconds_total{$region.$zone.$app.$host}

Container_cpu_system_seconds_total{$region.$zone.$network.$env.$app.$host}

Container_cpu_system_seconds_total{$regionID.$region.$zone.$network.$app.$host}
```

The Curse of Dimensionality

O(2^d)

Desire to maintain consistent metric tags

Containers are ephemeral and that’s ok

API Overview

Workloads

- → Container v1 core
- → CronJob v1beta1 batch
- → DaemonSet v1 apps
- → Deployment v1 apps
- → Job v1 batch
- → Pod v1 core
- → ReplicaSet v1 apps
- → ReplicationController v1 core
- → StatefulSet v1 apps

DISCOVERY & LOAD BALANCING

- → Endpoints v1 core
- → Ingress v1beta1 extensions
- → Service v1 core

Cluster

- → Namemespace v1 core
- → Node v1 core
- → etc...

Custom Resource Definitions

- → etc...

Live Demo

API Overview

Kubernetes has a well defined API with very specific conventions

- ➔ Follows a traditional REST pattern
- ➔ All kubernetes REST objects contain identically structured metadata fields
- ➔ This allows us to leverage the api as a datasource across different any number of standard or user defined kubernetes resources

API Overview

Workloads

- → Container v1 core
- → CronJob v1beta1 batch
- → DaemonSet v1 apps
- → Deployment v1 apps
- → Job v1 batch
- → Pod v1 core
- → ReplicaSet v1 apps
- → ReplicationController v1 core
- → StatefulSet v1 apps

DISCOVERY & LOAD BALANCING

- → Endpoints v1 core
- → Ingress v1beta1 extensions
- → Service v1 core

Cluster

- → Namemespace v1 core
- → Node v1 core
- → etc...

Custom Resource Definitions

- → etc...

So What?

Being able to query on a few extra dimensions is not that special

Monitoring Kubernetes should be Turn-Key and Free

A standard set of defined metrics that are tool and database agnostic ✓

Tools to auto-generate visualizations and alerts for kubernetes based on best practices ✓

A Fractured landscape of tools and practices that differ across companies and teams within companies ✗

Shout Out To Daniel Lee

Thank You / QA