Monitoring Kubernetes Clusters

Craig Golightly
SENIOR SOFTWARE CONSULTANT

@seethatgo www.seethatgo.com

Overview

Using Prometheus to monitor Kubernetes

- Multiple options
- Least effort with most coverage

Prometheus Operator

- Simplify deployment and configuration
- Use Helm to add to Kubernetes cluster
- Metrics available

Prometheus Operator

Extends Kubernetes API to simplify deployment, config, and management

Automates common tasks

Creates Kubernetes Custom Resource Definitions (CRDs)

Fast and easy to add Prometheus to cluster

https://github.com/prometheus-operator/prometheus-operator

Helm

Chart

Helm package

All definitions to run app, tool or service

Repository

Place where charts can be collected and shared

Release

Instance of a chart running in Kubernetes

"Helm installs *charts* into Kubernetes, creating a new *release* for each installation. And to find new charts, you can search Helm chart *repositories*."

Helm documentation

```
curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3
| bash

helm repo add stable https://kubernetes-charts.storage.googleapis.com

helm install my-prometheus-operator stable/prometheus-operator
```

Install Helm and Prometheus Operator

Running Kubernetes cluster

kubectl connected to cluster

Managing Kubernetes

Cost management vs. having enough resources (CPU, memory)

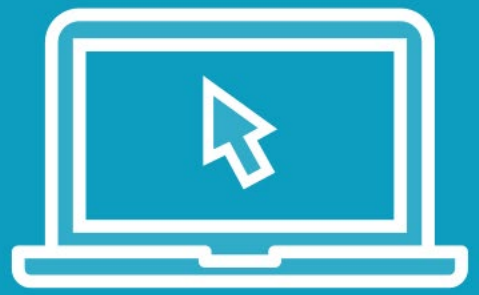
Grouped by cluster, namespace, node, pod, and container

Resource requests (CPU/RAM requested by deployed pods)

Resource utilization (CPU/RAM actually being used)

Resource limits (max CPU/RAM to allocate)

Demo



Starting point:

- Running Kubernetes cluster
- Simple application deployed
- kubectl connected to cluster

Install Helm

- prometheus-operator chart

View Prometheus server

Summary

Install

- Helm
- Prometheus Operator

Variety of metrics

- CPU / memory info
- Grouped by pod, node, cluster, etc.

Easily add to monitor Kubernetes clusters

Up Next:
Monitoring Message Queues