Managing your Kubernetes Deployments with Helm

Abubakar Siddiq Ango
GitLab BV.
Abubakar Siddiq Ango
Support Engineer @ GitLab.com
Lead Organizer, GDG Bauchi
Executive Director, uplift.ng
abuango.me, @sarki247
Containers have made us more efficient and productive by accelerating innovation and reducing IT costs.
Kubernetes

Efficient, highly scalable and cloud agnostic infrastructure.

Basic Objects are:

- Cluster
- Nodes
- Pods
- Service
- Volume
- Namespace
- Ingress
Deployment.yaml

apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80

Service.yaml

apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  ports:
  - port: 8000
    targetPort: 80
  protocol: TCP
  selector:
    app: nginx

Commands:

kubectl create -f ./deployment.yaml
kubectl get deployment
kubectl get pods -l app=nginx
kubectl delete deployment nginx-deployment

kubectl create -f ./Service.yaml
kubectl get services
kubectl delete service nginx-service
Managing raw manifests can be difficult!

You have to individually create & manage deployments, secrets, config maps, services, volumes, ingress, etc.

- No template parameterization
- No application lifecycle hooks
- No history of releases
Helm comes to the rescue!

Helm helps you manage Kubernetes applications using Helm Charts, which helps you define, install, and upgrade even the most complex Kubernetes application.

Like apt, yum or pip, Helm is a package manager for Kubernetes.
Helm has 2 components:

**The Helm Client:** command line client for end users, which is used for chart development, managing chart repositories and interacting with the Tiller Server.

**The Tiller Server:** an in-cluster server that interacts with the Helm Client and interfaces with the Kubernetes API Server. Manages Charts installations and release in the cluster.
Why Helm?

- Ease of Use
- Reusability
- Grouping of Resources
- Versioning
- Dependency
- Managing releases
- Continuous Integration & Deployment
Helm Charts

A chart is a collection of files in a particular directory tree, that describe a related set of Kubernetes resources.

```yaml
wordpress/
  Chart.yaml  # A YAML file containing information about the chart
  LICENSE     # OPTIONAL: A plain text file containing the license for the chart
  README.md   # OPTIONAL: A human-readable README file
  requirements.yaml # OPTIONAL: A YAML file listing dependencies for the chart
  values.yaml  # The default configuration values for this chart
  charts/     # OPTIONAL: A directory containing any charts upon which this chart depends.
  templates/  # OPTIONAL: A directory of templates that, when combined with values,
               # will generate valid Kubernetes manifest files.
  templates/NOTES.txt # OPTIONAL: A plain text file containing short usage notes
```
Chart Repositories

A chart repository consists of packaged charts for reuse. You can roll your own private repo or use the official repository at: https://github.com/kubernetes/charts.

It's the default repo or Helm and hosted at https://kubernetes-charts.storage.googleapis.com/.

The Charts in the repo are organized in 2 folders, stable and incubator, you will have to add the incubator repo separately to use the charts in it.
Installing Helm

**MacOS:** brew install kubernetes-helm

**Others:** Binary Releases at https://github.com/kubernetes/helm/releases

**Source Installation (Linux/MacOS):**

```
$ cd $GOPATH
$ mkdir -p src/k8s.io
$ cd src/k8s.io
$ git clone https://github.com/kubernetes/helm.git
$ cd helm
$ make bootstrap build
```
Installing Tiller

- Install gcloud
- gcloud auth login
- gcloud components install kubectl
- gcloud projects create helm-demo
- gcloud config set compute/zone us-central1-b
- gcloud config set project helm-demo
- gcloud container clusters create demo-cluster
- gcloud container clusters get-credentials demo-cluster

- helm init #installs tiller in cluster
Installing Tiller Locally - Using minikube

- Minikube start

- helm init #installs tiller in cluster
Helm Commands

- **Search for a chart:**
  helm search wordpress

- **Check details of a chart:**
  helm inspect wordpress

- **Install a chart:**
  helm install -n [release-name] wordpress

- **Install with customized values:**
  helm install -f values.yaml -n [release-name] stable/wordpress

- **Check history of a release:**
  helm history [release-name]

- **Check chart for errors:**
  helm lint

- **List releases:**
  helm list

- **List Repos:**
  helm repo list

- **Delete/Uninstall a release:**
  helm delete [release-name]

- **Rollback a Release:**
  helm rollback [release-name] [revision]
Learn more: https://helm.sh

Thank you!
Everyone can contribute

Fancy Remote work, join us at:
https://about.gitlab.com/jobs/