Table of Contents

1. Helm Kebernetes

- Helm is a package manager for Kubernetes applications.
- It is a command-line tool that enables you to create and use so-called Helm Charts.
- It uses a YAML file form called Charts. Charts are used to describe, install, and update Kubernetes.
- A Helm Chart is a collection of templates and settings that describe a set of Kubernetes resources.
- Helm is directly communicating with Kubernetes cluster via Rest.
- Helm itself is stateful. When a Helm Chart gets installed, the defined resources are getting deployed and meta-information is stored in Kubernetes secrets.
- Helm gives you a very convenient way of managing a set of applications that enables you to deploy, upgrade, rollback and delete.

➢ Prerequisites for Installing And Using Helm

- A Kubernetes version 1.8 + cluster, enabled with Role-Based Access Control (RBAC).
- The command-line tool **kubectl** installed on your local machine, configured to connect with your cluster.

- The following command can test your connectivity:

```
# kubectl cluster-info
```

- If you access multiple clusters with **kubectl**, be sure to verify that you've selected the correct cluster context:

```
# kubectl config get-contexts
```

1.1. Creating A Kubernetes Deployment Using Helm Charts:

➢ Installing Helm

- Change to a writable directory, and download the GitHub repository script from Helm:

```
# cd /tmp
# curl https://raw.githubusercontent.com/kubernetes/helm/master/scripts/get > install-helm.sh
# chmod u+x install-helm.sh
# ./install-helm.sh
```

➢ Installing A Helm Chart

- Packages of Helm software are called charts.
- Helm comes pre-configured with a collection of curated charts called a stable.

➢ installing the Kubernetes Dashboard.

```
# helm install stable/kubernetes-dashboard --name dashboard-demo
# helm list → To list the installed Dashboard.
```

```
NAME            REVISION   UPDATED              STATUS    CHART
dashboard-demo    1        Aug 8 20:11:11 2018  DEPLOYED  kubernetes-dashboard-0.7.1
```

- To check the deployment of a new service on the cluster:

```
# kubectl get services
```

```
NAME                                  TYPE        CLUSTER-IP
dashboard-demo-kubernetes-dashboard   ClusterIP   10.32.104.73
kubernetes                            ClusterIP   10.32.0.1
```

1.2. Deploy a Simple Helm Application

- Connect to a Kubernetes cluster.
  - To connect to docker-desktop cluster

```
$ kubectl config use-context docker-desktop
  Switched to context "docker-desktop".
```

  - To validate the cluster node

```
$ kubectl get node
  NAME             STATUS   ROLES    AGE   VERSION
  docker-desktop   Ready    master   20d   v1.19.3
```

- Deploy an Apache webserver using Helm.
  - Add the help repository

```
$ helm repo add bitnami https://charts.bitnami.com/bitnami
```

  - install the container:

```
$ helm install my-apache bitnami/apache --version 8.0.2
```

  - Validate:

```
$ kubectl get pods
  NAME                                READY   STATUS    RESTARTS   AGE
  my-apahe-apache-589b8df6bd-q6m2n    1/1     Running   0          2m27s
```

  - open the google Chrome and access apache

```
http://localhost
```

  - List the deployment with Helm list.

```
$ helm list
  NAME         REVISION       STATUS         CHART          VERSION
  my-apache    1              deployed       apache-8.0.2   2.4.46
```

1.3. Upgrade a Helm Application

```
$ helm upgrade my-apache bitnami/apache --version 8.0.3
$ helm list
  NAME         REVISION       STATUS         CHART          VERSION
  my-apache    2              deployed       apache-8.0.3   2.4.46
```

1.4. Rollback a Helm Application

```
$ helm rollback my-apache 1
  Rollback was a success! Happy Helming!

$ helm list
  NAME         REVISION       STATUS         CHART          VERSION
  my-apache    3              deployed       apache-8.0.2   2.4.46
```

  - Helm stores deployment information in secrets, list all secret

```
$ kubectl get secret
  NAME                            TYPE                       DATA   AGE
  default-token-nc4hn                kubernetes.io/sat       3      20d
  sh.helm.release.v1.my-apache.v1    helm.sh/release.v1      1      1m
  sh.helm.release.v1.my-apache.v2    helm.sh/release.v1      1      1m
  sh.helm.release.v1.my-apache.v3    helm.sh/release.v1      1      1m
```

1.5. Remove a Deployed Helm Application

```
$ helm delete my-apache
  release "my-apache" uninstalled
```

1.6. Updating A Release

- The helm upgrade command can be used with a new or updated chart to upgrade a release, or to update the configuration options.

```
# helm upgrade dashboard-demo stable/kubernetes-dashboard --set
fullnameOverride="dashboard"

# kubectl get services
NAME                  TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)
kubernetes            ClusterIP   10.32.0.1          443/TCP    dashboard
```

1.7. Rolling Back a Release

- Helm retains all previous release details in case you need to roll back to a previous configuration or chart

```
# helm list
NAME            REVISION  UPDATED                  STATUS     CHART
dashboard-demo   2        Aug 8 20:13:15 2018  DEPLOYED   kubernetes-dashboard-0.7.1

# helm rollback dashboard-demo 1
Rollback was a success! Happy Helming!
```

1.8. Deleting A Release

- Helm releases can be deleted with the Helm delete command:

```
# helm delete dashboard-demo
release "dashboard-demo" deleted
# helm list --deleted
NAME          REVISION   UPDATED                  STATUS    CHART
dashboard-demo  3        Aug  8 20:15:21 2018  DELETED  kubernetes-dashboard-0.7.1
# helm delete dashboard-demo –purge → purge all old revisions
```

1.9. Postgres deployment using helm

- Create a **deployment.yaml** file:

```
apiVersion: apps/v1
     kind: Deployment
     metadata:
       name: {{ .Values.postgres.name }}
       labels:
         app: {{ .Values.postgres.name }}
         group: {{ .Values.postgres.group }}
     spec:
       replicas: {{ .Values.replicaCount }}
       selector:
         matchLabels:
           app: {{ .Values.postgres.name }}
       template:
         metadata:
           labels:
             app: {{ .Values.postgres.name }}
             group: {{ .Values.postgres.group }}
```

```
          spec:
            volumes:
              - name: {{ .Values.postgres.volume.name }}
                persistentVolumeClaim:
                  claimName: {{ .Values.postgres.volume.pvc.name }}
            containers:
              - name: {{ .Values.postgres.name }}
                image: {{ .Values.postgres.container.image }}
                ports:
                  - containerPort: {{ .Values.postgres.container.port }}
                envFrom:
                  - configMapRef:
                      name: {{ .Values.postgres.config.name }}
                volumeMounts:
                  - name: {{ .Values.postgres.volume.name }}
                    mountPath: {{ .Values.postgres.volume.mountPath }}
```

```
# vi values.yaml
  replicaCount: 1postgres:
    name: postgres
    group: db
    container:
      image: postgres:9.6-alpine
      port: 5432
    service:
      type: ClusterIP
      port: 5432
    volume:
      name: postgres-storage
      kind: PersistentVolumeClaim
      mountPath: /var/lib/postgresql/data
      pvc:
        name: postgres-persistent-volume-claim
        accessMode: ReadWriteOnce
        storage: 4Gi
    config:
      name: postgres-config
      data:
         - key: key
           value: value
```

```
# vi Chart.yaml
  apiVersion: v2
  name: postgres
  description: A Helm chart for PostgreSQL database
  type: application
  version: 0.1.0
  appVersion: 1.16.0
  keywords:
    - database
    - postgres
  home: https://github.com/wkrzywiec/k8s-helm-helmfile/tree/master/helm
```

```
maintainers:
  - name: Wojtek Krzywiec
    url: https://github.com/wkrzywiec
```

```
$ helm install -f kanban-postgres.yaml postgres ./postgres
NAME: postgres
LAST DEPLOYED: Mon Apr 13 16:13:16 2020
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
$ helm list
NAME     NAMESPACE REVISION  STATUS    CHART          APP VERSION
postgres default   1         deployed  postgres-0.1.0 1.16.0
$ kubectl get deployments
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
postgres   1/1     1            1           2m14s
```

1.10. Prometheus installation using

- Add the Prometheus charts repository to your helm configuration.
  ```
  # helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
  # helm repo add stable https://kubernetes-charts.storage.googleapis.com/
  # helm repo update
  ```
- install Prometheus
  ```
  # Helm 3
  helm install [RELEASE_NAME] prometheus-community/prometheus
  # Helm 2
  helm install --name [RELEASE_NAME] prometheus-community/prometheus
  ```
- **Bonus point**: Helm chart deploys node-exporter, kube-state-metrics, and alertmanager along with Prometheus.

1.11. Install Traefik

- Add the Repo and install traefik.
  ```
  # helm repo add stable https://kubernetes-charts.storage.googleapis.com/
  # helm install traefik stable/traefik --set metrics.prometheus.enabled=true
  ```

- Verify
  ```
  $ kubectl get svc
  ```
  ```
  k get svc
  NAME         TYPE       CLUSTER-IP    EXTERNAL-IP  PORT(S)  AGE
  kubernetes  ClusterIP   100.64.0.1    <none>       443/TCP  99d
  traefik    LoadBalancer      100.65.9.227 xxx.eu-west-1.elb.amazonaws.com
  443:32164/TCP,80:31829/TCP  72m
  traefik-prometheus ClusterIP 100.66.30.208    <none>    .
  ```