Linux Academy

Kubernetes Essentials

Cheat Sheet

Will Boyd williamb@linuxacademy.com

December 14, 2018

Contents

Kubernetes Essentials Cheat Sheet

This cheat sheet provides a quick reference for the commands and flags that are used in the Kubernetes Essentials course.

Building a Cluster

This section describes how to build a Kubernetes cluster with Kubeadm. This guide is for a three-server cluster with one master and two worker nodes. It is designed for servers running Ubuntu 18.04 LTS, and assumes that network communication between these servers is possible on all ports.

Install Docker

Install Docker on all three servers.

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) \
   stable"
sudo apt-get update
sudo apt-get install -y docker-ce=18.06.1~ce~3-0~ubuntu
sudo apt-mark hold docker-ce
```

Install Kubeadm, Kubelet, and Kubcectl

Install these Kubernetes components on all three servers.

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
cat << EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF
sudo apt-get update
sudo apt-get install -y kubelet=1.12.2-00 kubeadm=1.12.2-00 kubectl=1.12.2-00
sudo apt-mark hold kubelet kubeadm kubectl
```

Initialize the Master

Initialize the Kubernetes master, only on the master server.

```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
```

Set up Kubeconfig to enable Kubectl on the master server.

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Join Worker Nodes to the Cluster

When you run the `kubeadm init` command on the master, you should get a `kubeadm join` command as part of the output. Copy that command to get the token and hash values, then run it on both workers with `sudo`.

```
sudo kubeadm join $controller_private_ip:6443 --token $token \
--discovery-token-ca-cert-hash $hash
```

Set Up Networking with Flannel

First, run these commands on all three servers.

```
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

Apply the flannel config with this command, only on the master server.

```
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/
bc79dd1505b0c8681ece4de4c0d86c5cd2643275/Documentation/kube-flannel.yml
```

Working with Basic Kubernetes Objects

Node

A node is a server that is capable of running containers in a Kubernetes cluster.

List nodes:

```
kubectl get nodes
```

Get more information about a specific node:

```
kubectl describe node $node_name
```

Pod

The pod is the most basic Kubernetes object. A pod consists of one or more containers which share storage resources and an IP address in the cluster.

List pods:

```
kubectl get pods
```

Get more information about a specific pod:

```
kubectl describe pod $pod_name
```

Deployment

A deployment allows you to specify a desired configuration for a group of pods, and the cluster will constantly work to maintain that desired state. For example, you can specify a number of replicas, and if one dies or gets deleted, the cluster will immediately spin up a new replica to replace it and maintain the desired number of replicas.

List deployments:

```
kubectl get deployment
```

Get more information about a specific deployment:

```
kubectl describe deployment $deployment_name
```

Service

A service provides dynamic access to a set of replica pods. Services provide a single access point for other pods and external entities to use in order to access a set of replicas in a high-availability, load-balanced fashion.

List services:

```
kubectl get service
```

Get more information about a specific service:

```
kubectl describe service $service_name
```

Kubectl create

Creating Kubernetes objects can be done with `kubectl create`. You can pass in a YAML file representing the object to be created with the `-f` flag. You can also enter YAML directly from the command line like this:

```
cat << EOF | kubectl apply -f -
$yaml_goes_here
EOF
```

General Kubectl Flags

These are some useful `kubectl` flags that can be useful with almost any `kubectl` command.

- `-n $namespace`: All Kubernetes objects have namspaces. When no namespace is specified, the default namespace is assumed. When working with objects in any namespace other than default, you will always need to use this flag to specify what namespace you are working with. You can create new namespaces with `kubectl create namespace $namespace`.
- `-w`: This flag allows you to watch output for changes. For example, `kubectl get pods -w` will show a list of pods, but the command will continue to run and will display new information if the state of any of the pods changes, or if a new pod is created.

Deploying Stan's Robot Shop

In the Kubernetes Essentials course, we use Stan's Robot Shop as a sample microservice app to help demonstrate the value of Kubernetes. You can deploy that app to a cluster with these commands on the master server:

```
cd ~/
git clone https://github.com/linuxacademy/robot-shop.git
kubectl create namespace robot-shop
kubectl -n robot-shop create -f ~/robot-shop/K8s/descriptors/
```