Although installing and configuring a Kubernetes cluster is not one of the objectives for the CKAD exam, it is important to get some hands-on experience with the concepts covered in this course. Therefore, it is useful to have a Kubernetes cluster with which you can experiment and try out the things that will be covered throughout the course. This lesson will guide you through the process of building a basic cluster that you can experiment with as you proceed.

# Lesson Reference

If you want to follow along, there is a reference for the commands used in this lesson below.

## On All 3 Servers

First, set up the Docker and Kubernetes repositories:

Make the initial configuration for Docker before installing it.

```
sudo mkdir /etc/docker
```

Create the daemon.json config file that Docker will use.

```
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
 "max-size": "100m"
},
  "storage-driver": "overlay2"
}
EOF
```

Update your packages, and install dependencies.

```
sudo apt-get update
sudo apt-get install apt-transport-https ca-certificates curl gnupg lsb-release
```

Update repositories and gpg keys for Docker.

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg

echo   "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list >
/dev/null
```

Update your repositories.

```
sudo apt-get update
```

Add the packages and gpg key for Kubernetes.

```
sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu \ $(lsb_release -cs) stable"

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
```

Update your repositories one last time.

```
sudo apt-get update
```

Now, you can install Docker and Kubernetes packages.

```
sudo apt-get install -y docker-ce=18.06.1~ce~3-0~ubuntu kubelet=1.14.5-00
kubeadm=1.14.5-00 kubectl=1.14.5-00

sudo apt-mark hold docker-ce kubelet kubeadm kubectl
```

Note that if you want to use a newer version of Kubernetes, change the version installed for kubelet, kubeadm, and kubectl. Make sure all three use the same version.

**Note:** There is currently a bug in Kubernetes 1.13.4 (and earlier) that can cause problems installing the packages. Use 1.14.5-00 to avoid this issue.

Enable iptables bridge call.

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

sudo modprobe br_netfilter

sudo sysctl -p
```

## On the Kube Master Server

Initialize the cluster.

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

Set up local kubeconfig.

```
mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Install Flannel networking.

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

**Note:** If you are using Kubernetes 1.16 or later, you will need to use a newer Flannel installation yaml instead:

```
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/3f7d3e6c24f641e7ff557ebcea1136fdf4b1b6a1/Documentation/kube-flannel.yml
```

## On Each Kube Node Server

Join the node to the cluster. Do this by copying the provided line from the output when initializing the master node. Keep in mind that, when copying the command, the system will add a newline character if it stretches over multiple lines in the web terminal. To get around this, copy the command to a text editor and make sure it fits on one entire line. It should look something like the following:

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

## On the Kube Master Server

Finally, verify all nodes are joined and ready:

```
kubectl get nodes
```

You should see all three servers with a status of Ready.

```
NAME                      STATUS   ROLES    AGE     VERSION
wboyd1c.mylabserver.com   Ready    master   54m     v1.14.5
wboyd2c.mylabserver.com   Ready    <none>   49m     v1.14.5
wboyd3c.mylabserver.com   Ready    <none>   49m     v1.14.5
```