Building your own Kubernetes cluster is a necessary skill for the CKA certification, according to the exam objectives. In addition to this, you will need your own Kubernetes cluster so you can experiment hands-on with the topics and skills that will be covered in this course. In this lesson, we walk through the process of building a Kubernetes cluster using kubeadm. Follow along to build your own cluster!

#### Relevant Documentation

* [Installing kubeadm](https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/)
* [Creating a cluster with kubeadm](https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/create-cluster-kubeadm/)

#### Lesson Reference

If you are using cloud playground, create three servers with the following settings:

* Distribution: Ubuntu 18.04 Bionic Beaver LTS
* Size: medium

On all nodes, set up containerd. You will need to load some kernel modules and modify some system settings as part of this process.

cat <<EOF | sudo tee /etc/modules-load.d/containerd.conf

overlay

br\_netfilter

EOF

sudo modprobe overlay

sudo modprobe br\_netfilter

cat <<EOF | sudo tee /etc/sysctl.d/99-kubernetes-cri.conf

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip\_forward = 1

net.bridge.bridge-nf-call-ip6tables = 1

EOF

sudo sysctl --system

Install and configure containerd.

sudo apt-get update && sudo apt-get install -y containerd

sudo mkdir -p /etc/containerd

sudo containerd config default | sudo tee /etc/containerd/config.toml

sudo systemctl restart containerd

On all nodes, disable swap.

sudo swapoff -a

sudo sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab

On all nodes, install kubeadm, kubelet, and kubectl.

sudo apt-get update && sudo apt-get install -y apt-transport-https curl

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=1.20.1-00 kubeadm=1.20.1-00 kubectl=1.20.1-00

sudo apt-mark hold kubelet kubeadm kubectl

On the control plane node only, initialize the cluster and set up kubectl access.

sudo kubeadm init --pod-network-cidr 192.168.0.0/16

mkdir -p $HOME/.kube

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

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

Verify the cluster is working.

kubectl version

Install the Calico network add-on.

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

Check the calico-related kube-system Pods to verify that everything is working so far (they may take a few moments to fully start up).

kubectl get pods -n kube-system

Get the join command (this command is also printed during kubeadm init. Feel free to simply copy it from there).

kubeadm token create --print-join-command

Copy the join command from the control plane node. Run it on each worker node as root (i.e. with sudo).

sudo kubeadm join ...

On the control plane node, verify all nodes in your cluster are ready. Note that it may take a few moments for all of the nodes to enter the READY state.

kubectl get nodes