Lesson 2 Demo 8: Configure a Pod with Custom Scheduler

This section will guide you to:

* Configure a pod with custom scheduler

This lab has one sub-section, namely:

1. Configuring a pod with custom scheduler

**Note:** If you don’t have an existing Kubernetes cluster, refer to the Demo 1.1 of Lesson 1.

**Step 1:** Configuring a pod with custom scheduler

* Start the kubernetes cluster in the lab
* Create the deployment specified in the config in the Kubernetes cluster. For that, create a yaml file as shown below:

*vi my-scheduler.yaml*

* Let’s create the deployment config that manages a ReplicaSet which in turn manages the pods, thereby making the scheduler resilient to failures as shown in screen below.   
    
  **Note:** You might already have the my-scheduler.yaml file created from the earlier demos.

*apiVersion: v1*

*kind: ServiceAccount*

*metadata:*

*name: my-scheduler*

*namespace: kube-system*

*---*

*apiVersion: rbac.authorization.k8s.io/v1*

*kind: ClusterRoleBinding*

*metadata:*

*name: my-scheduler-as-kube-scheduler*

*subjects:*

*- kind: ServiceAccount*

*name: my-scheduler*

*namespace: kube-system*

*roleRef:*

*kind: ClusterRole*

*name: system:kube-scheduler*

*apiGroup: rbac.authorization.k8s.io*

*---*

*apiVersion: apps/v1*

*kind: Deployment*

*metadata:*

*labels:*

*component: scheduler*

*tier: control-plane*

*name: my-scheduler*

*namespace: kube-system*

*spec:*

*selector:*

*matchLabels:*

*component: scheduler*

*tier: control-plane*

*replicas: 1*

*template:*

*metadata:*

*labels:*

*component: scheduler*

*tier: control-plane*

*version: second*

*spec:*

*serviceAccountName: my-scheduler*

*containers:*

*- command:*

*- /usr/local/bin/kube-scheduler*

*- --address=0.0.0.0*

*- --leader-elect=false*

*- --scheduler-name=my-scheduler*

*image: gcr.io/my-gcp-project/my-kube-scheduler:1.0*

*livenessProbe:*

*httpGet:*

*path: /healthz*

*port: 10251*

*initialDelaySeconds: 15*

*name: kube-second-scheduler*

*readinessProbe:*

*httpGet:*

*path: /healthz*

*port: 10251*

*resources:*

*requests:*

*cpu: '0.1'*

*securityContext:*

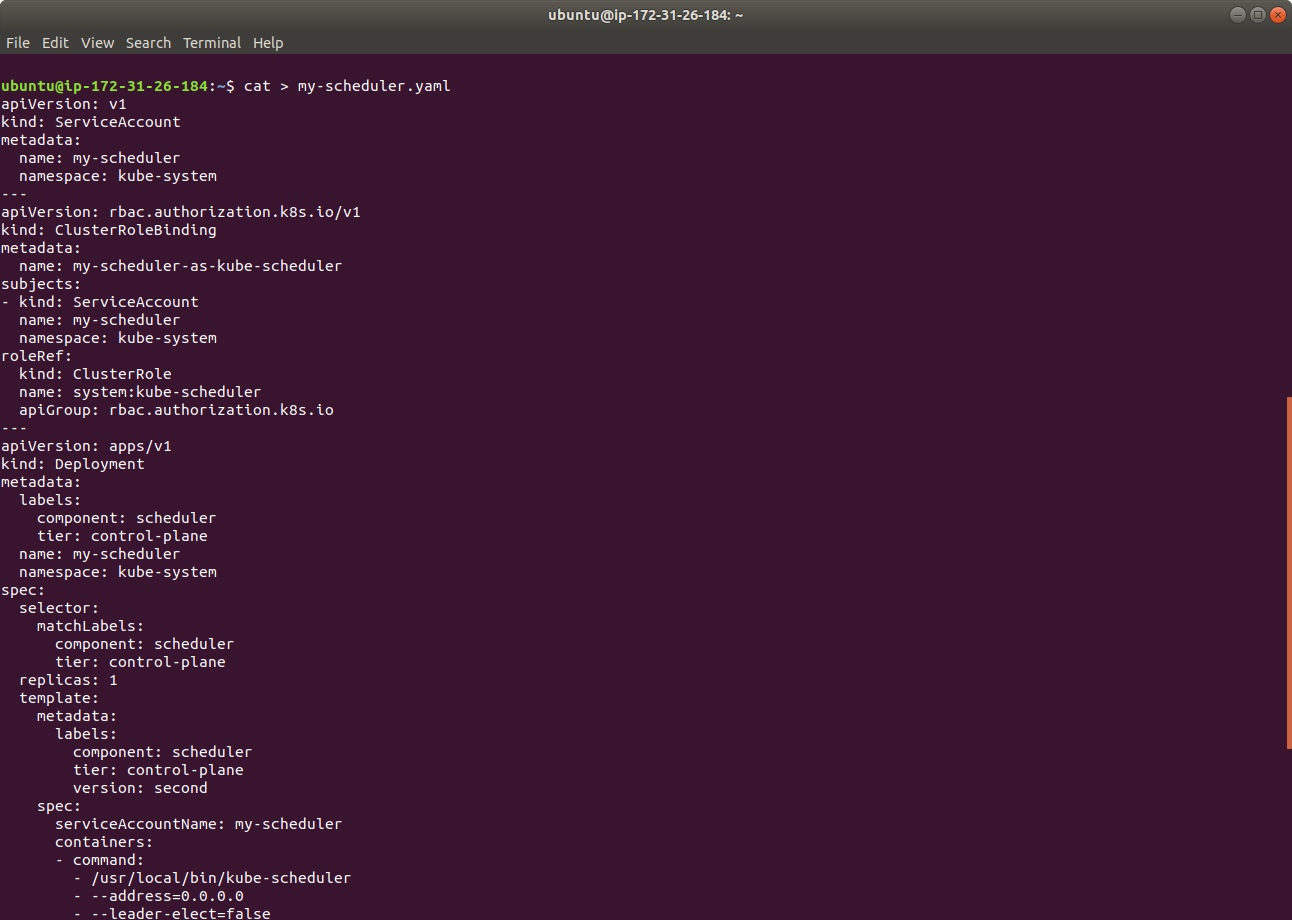
*privileged: false*

*volumeMounts: []*

*hostNetwork: false*

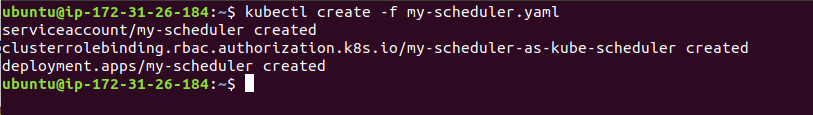
*hostPID: false*

*volumes: []*



* Now that we have a config file that creates the deployment as a custom scheduler, use the kubectl command to run your scheduler in a Kubernetes cluster as shown below:

*kubectl create -f my-scheduler.yaml*



* As the next step, let’s verify that the scheduler pod is running as shown below:

*kubectl get pods --namespace=kube-system*

