Multiple Scheduler

- **Default scheduler**: -

It has an algorithm that distribute pods across nodes evenly as well as takes into consideration the various condition specifies through taints, toleration and node affinity.

- If you have a specific application that requires its components to be placed on nodes after performing some additional checks. So, you decide to have your own scheduling algorithm to place pod on nodes and you decide that you can add your own custom conditions and checks in it.

  Kubernetes is highly extensible. Kubernetes cluster can have multiple schedulers. You can write your own kubernetes scheduler program, package it and deploy it as kubernetes cluster. You can use this custom scheduler for specific application. When creating a pod or a deployment you can instruct kubernetes have the pod schedule by a specific scheduler.

- **leader-elect=true**

The leader-elect option is used when you have multiple copies of the scheduler running on different master nodes. In a High Availability (HA) setup where you have multiple master nodes with the kube-scheduler process running on both of them.

If multiple copies of the same scheduler are running on different nodes only one can be active at a time. That's where the leader-elect helps in choosing a leader who will lead scheduling activities.

- **leader-elect=false**

To get multiple schedulers working you must either set the leader-elect option to false. In case where you don't have multiple master. (we use this in example)

- If you have multiple master, you can pass in an additional parameter to set a lock object name.
  - --lock-object-name=my-custom-scheduler

Multiple scheduler example

Deploy an additional scheduler to the cluster following the given specification.

Namespace = kube-system

Name = my-scheduler

my-scheduler.yml

```
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    component: kube-scheduler
    tier: control-plane
  name: my-scheduler
  namespace: kube-system
spec:
  containers:
  - command:
    - kube-scheduler
    - --authentication-kubeconfig=/etc/kubernetes/scheduler.conf
    - --authorization-kubeconfig=/etc/kubernetes/scheduler.conf
    - --bind-address=127.0.0.1
    - --kubeconfig=/etc/kubernetes/scheduler.conf
    - --leader-elect=false
    - --scheduler-name=my-scheduler
    - --port=0
    image: k8s.gcr.io/kube-scheduler:v1.19.0
    imagePullPolicy: IfNotPresent
    livenessProbe:
      failureThreshold: 8
      httpGet:
        host: 127.0.0.1
        path: /healthz
        port: 10259
        scheme: HTTPS
      initialDelaySeconds: 10
      periodSeconds: 10
      timeoutSeconds: 15
    name: my-scheduler
    resources:
-- INSERT (paste) --
```

```
controlplane $ kubectl create -f my-scheduler.yml
pod/my-scheduler created
controlplane $ kubectl -n kube-system get pods
NAME                                   READY   STATUS    RESTARTS   AGE
coredns-f9fd979d6-g7k6v                1/1     Running   0          20m
coredns-f9fd979d6-pkxwf                1/1     Running   0          20m
etcd-controlplane                      1/1     Running   0          20m
kube-apiserver-controlplane            1/1     Running   0          20m
kube-controller-manager-controlplane   1/1     Running   0          20m
kube-flannel-ds-amd64-gqxhq            1/1     Running   0          20m
kube-flannel-ds-amd64-ksvcc            1/1     Running   0          20m
kube-proxy-475kj                       1/1     Running   0          20m
kube-proxy-pvprv                       1/1     Running   0          20m
kube-scheduler-controlplane            1/1     Running   0          20m
my-scheduler                           0/1     Running   0          5s
controlplane $
```

Create a nginx POD with this (my-scheduler) new custom scheduler.

```
controlplane $ kubectl apply -f nginx-pod.yaml
pod/nginx created
controlplane $ kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          15s
controlplane $ cat nginx-pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  schedulerName: my-scheduler
  containers:
  -   image: nginx
      name: nginx
controlplane $
```