

Manual Scheduling

In this lecture, we look at the different ways of manually scheduling a pod on a node. What do you do when you do not have a scheduler in your cluster? You probably do not want to rely on the built-in scheduler and instead want to schedule the pod yourself. So how exactly does a scheduler work in the backend?

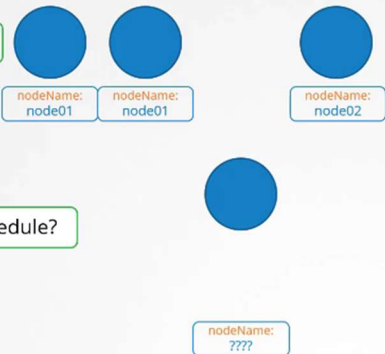
Let's start with a simple pod definition file. Every pod has a field called Node Name that, by default, is not set. You don't typically specify this field when you create the pod manifest file. Kubernetes adds it automatically. The scheduler goes through all the pods and looks for those that do not have this property set. Those are the candidates for scheduling. It then identifies the right node for the pod by running the scheduling algorithm.

How scheduling works

What to Schedule?

Which node to schedule?

```
pod-definition.yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    name: nginx
spec:
  containers:
  - name: nginx
    image: nginx
    ports:
      - containerPort: 8080
  nodeName: 
```



Once identified, it schedules the pod on the node by setting the node name property to the name of the node by **creating a binding object**.

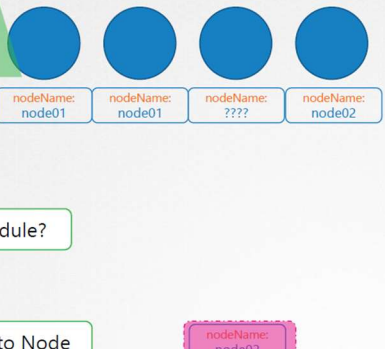
How scheduling works

What to Schedule?

Which node to schedule?

(Schedule)Bind Pod to Node

```
pod-definition.yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    name: nginx
spec:
  containers:
  - name: nginx
    image: nginx
    ports:
      - containerPort: 8080
  nodeName: node02
```



So if there is no scheduler to monitor and schedule nodes, what happens? The pods continue to be in a pending state. So what can you do about it? You can manually assign pods to nodes yourself. Well, without a scheduler, the easiest way to schedule a pod is to simply set the node name field to the name of the node in your pod specification file while creating the pod. The pod then gets assigned to the specified node. You can only specify the node name at creation time.

```
kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
nginx	0/1	Pending	0	3s

```
kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE
nginx	1/1	Running	0	9s	10.40.0.4	node02

```
pod-definition.yaml
```

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    name: nginx
spec:
  containers:
  - name: nginx
    image: nginx
    ports:
    - containerPort: 8080
  nodeName: node02
```

What if the pod is already created and you want to assign the pod to a node?

Kubernetes won't allow you to modify the node name property of a pod, so another way to assign a node to an existing pod is to create a **binding object and send a POST request to the pod's binding API**, thus mimicking what the actual scheduler does. In the binding object, you specify a target node with the name of the node, then send a POST request to the pod's binding API with the data set to the binding object in a JSON format. So you must convert the YAML file into its equivalent JSON form.

I No Scheduler!

```
Pod-bind-definition.yaml
```

```
apiVersion: v1
kind: Binding
metadata:
  name: nginx
target:
  apiVersion: v1
  kind: Node
  name: node02
```

```
pod-definition.yaml
```

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    name: nginx
spec:
  containers:
  - name: nginx
    image: nginx
    ports:
    - containerPort: 8080
```

```
curl --header "Content-Type:application/json" --request POST --data '{"apiVersion":"v1", "kind": "Binding" ... }' http://$SERVER/api/v1/namespaces/default/pods/$PODNAME/binding/
```