Node Selector

```
ubuntu@k8-master:~$ kubectl get nodes
NAME         STATUS   ROLES           AGE   VERSION
k8-master    Ready    control-plane   24h   v1.31.7
k8-worker1   Ready    <none>          24h   v1.31.7
k8-worker2   Ready    <none>          24h   v1.31.7
ubuntu@k8-master:~$ kubectl get nodes --show-labels
NAME         STATUS   ROLES           AGE   VERSION   LABELS
k8-master    Ready    control-plane   24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-master,kubernetes.io/o
nux,node-role.kubernetes.io/control-plane=,node.kubernetes.io/exclude-from-external-load-balancers=
k8-worker1   Ready    <none>          24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-worker1,kubernetes.io/
inux
k8-worker2   Ready    <none>          24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-worker2,kubernetes.io/
inux
```

```
ubuntu@k8-master:~$ kubectl get nodes --show-labels
NAME         STATUS   ROLES           AGE   VERSION   LABELS
k8-master    Ready    control-plane   24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-master,kubernetes.io/os=li
ux,node-role.kubernetes.io/control-plane=,node.kubernetes.io/exclude-from-external-load-balancers=
k8-worker1   Ready    <none>          24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-worker1,kubernetes.io/os=l
inux
k8-worker2   Ready    <none>          24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-worker2,kubernetes.io/os=l
inux
ubuntu@k8-master:~$ kubectl label nodes k8-worker1  cup=two
node/k8-worker1 labeled
ubuntu@k8-master:~$ kubectl get nodes --show-labels
NAME         STATUS   ROLES           AGE   VERSION   LABELS
k8-master    Ready    control-plane   24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-master,kubernetes.io/os=li
ux,node-role.kubernetes.io/control-plane=,node.kubernetes.io/exclude-from-external-load-balancers=
k8-worker1   Ready    <none>          24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,cup=two,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-worker1,kubernetes
.io/os=linux
k8-worker2   Ready    <none>          24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-worker2,kubernetes.io/os=l
inux
ubuntu@k8-master:~$ kubectl label nodes k8-worker1  cup=two-
error: invalid label value: "cup=two-": a valid label must be an empty string or consist of alphanumeric characters, '-', '_' or '.', and must start and end with an alphanumeric character (
e.g. 'MyValue',  or 'my_value',  or '12345', regex used for validation is '(([A-Za-z0-9][-A-Za-z0-9_.]*)?[A-Za-z0-9])?')
ubuntu@k8-master:~$ kubectl label nodes k8-worker1  cup-
node/k8-worker1 unlabeled
ubuntu@k8-master:~$ kubectl label nodes k8-worker1  cpu=two
node/k8-worker1 labeled
ubuntu@k8-master:~$ kubectl get nodes --show-labels
NAME         STATUS   ROLES           AGE   VERSION   LABELS
k8-master    Ready    control-plane   24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-master,kubernetes.io/os=li
ux,node-role.kubernetes.io/control-plane=,node.kubernetes.io/exclude-from-external-load-balancers=
k8-worker1   Ready    <none>          24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,cpu=two,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-worker1,kubernetes
.io/os=linux
k8-worker2   Ready    <none>          24h   v1.31.7   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=k8-worker2,kubernetes.io/os=l
inux
```

I attached a label to the k8-worker1 node using the kubectl label command and verified whether the label was attached using kubectl get nodes --show-labels.

```
ubuntu@k8-master:~$ cat nodeselector.yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    env: test
spec:
  containers:
  - name: nginx
    image: nginx
    imagePullPolicy: IfNotPresent
  nodeSelector:
    cpu: two
```

Created a nodeselector.yaml file using pod and give nodeSelector to it.

```
buntu@k8-master:~$ vi nodeselector.yaml
buntu@k8-master:~$ kubectl apply -f nodeselector.yaml
od/nginx created
buntu@k8-master:~$ kubectl get pods
AME                     READY   STATUS    RESTARTS        AGE
ginx                     1/1     Running   0               6s
ginx-7fcb78bd77-7fkgb    1/1     Running   1 (4m56s ago)   21h
ginx-7fcb78bd77-c9wg8    1/1     Running   1 (4m56s ago)   21h
ginx-7fcb78bd77-m8rcg    1/1     Running   1 (4m53s ago)   21h
ginx1-965647b5c-6sg5b    1/1     Running   1 (4m56s ago)   20h
ginx1-965647b5c-99x2m    1/1     Running   1 (4m53s ago)   20h
ginx1-965647b5c-q6c48    1/1     Running   1 (4m53s ago)   20h
buntu@k8-master:~$ kubectl get deploy
AME     READY   UP-TO-DATE   AVAILABLE   AGE
ginx    3/3     3            3           21h
ginx1   3/3     3            3           20h
buntu@k8-master:~$ kubectl delete deploy nginx nginx1
eployment.apps "nginx" deleted
eployment.apps "nginx1" deleted
buntu@k8-master:~$ kubectl get pods
AME    READY   STATUS    RESTARTS   AGE
ginx   1/1     Running   0          50s
buntu@k8-master:~$ kubectl get pods -o wide
AME    READY   STATUS    RESTARTS   AGE   IP          NODE         NOMINATED NODE   READINESS GATES
ginx   1/1     Running   0          59s   10.44.0.4   k8-worker1   <none>           <none>
buntu@k8-master:~$ cat nodeselector.yaml
```

I applied the nodeselector.yaml file using the kubectl apply command. This will schedule the Pod on the node with matching labels specified in the NodeSelector.

taint and tolerations

```
buntu@k8-master:~$ kubectl get nodes
AME          STATUS   ROLES           AGE   VERSION
8-master     Ready    control-plane   24h   v1.31.7
8-worker1    Ready    <none>          24h   v1.31.7
8-worker2    Ready    <none>          24h   v1.31.7
buntu@k8-master:~$ kubectl taint nodes k8-worker1 key1=value1:NoSchedule
ode/k8-worker1 tainted
buntu@k8-master:~$ kubectl describe nodes k8-worker1 | grep -i taint
aints:             key1=value1:NoSchedule
buntu@k8-master:~$ vi taint_tola.yaml
```

I applied a taint on the node k8-worker1 using the command kubectl taint node k8-worker1 key1=value1:NoSchedule. To verify that the taint was applied, I used kubectl describe node k8-worker1

```
buntu@k8-master:~$ kubectl get pods -o wide
o resources found in default namespace.
buntu@k8-master:~$ kubectl get pods
o resources found in default namespace.
buntu@k8-master:~$ kubectl get pods
o resources found in default namespace.
buntu@k8-master:~$ vi taint_tola.yaml
buntu@k8-master:~$ kubectl apply -f taint_tola.yaml
od/nginx4 created
buntu@k8-master:~$ kubectl get pods -o wide
AME     READY   STATUS    RESTARTS   AGE   IP          NODE         NOMINATED NODE   READINESS GATES
ginx4   1/1     Running   0          5s    10.36.0.1   k8-worker2   <none>           <none>
buntu@k8-master:~$ vi taint_tola.yaml
buntu@k8-master:~$ kubectl apply -f taint_tola.yaml
od/nginx3 created
buntu@k8-master:~$ kubectl get pods -o wide
AME     READY   STATUS    RESTARTS   AGE   IP          NODE         NOMINATED NODE   READINESS GATES
ginx3   1/1     Running   0          5s    10.44.0.1   k8-worker1   <none>           <none>
ginx4   1/1     Running   0          46s   10.36.0.1   k8-worker2   <none>           <none>
buntu@k8-master:~$ cat taint_tola.yaml
```

Created a Pod definition YAML file (taint_tola.yaml) with a toleration matching the taint. Pods without this toleration won't be scheduled on the node with the taint. However, Pods

with the toleration can be scheduled on any node, including those without the taint

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-notoleration
  labels:
    env: test
spec:
  containers:
  - name: nginx
    image: nginx
    imagePullPolicy: IfNotPresent
~
~
~
~
```

```
ubuntu@k8-master:~$ vi no_tola.yaml
ubuntu@k8-master:~$ vi no_tola.yaml
ubuntu@k8-master:~$ kubectl apply -f no_tola.yaml
pod/nginx-notoleration created
ubuntu@k8-master:~$ kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE     IP          NODE          NOMINATED NODE   READINESS GATES
nginx-notoleration  1/1     Running   0          10s     10.36.0.2   k8-worker2    <none>           <none>
nginx3              1/1     Running   0          3m53s   10.44.0.1   k8-worker1    <none>           <none>
nginx4              1/1     Running   0          4m34s   10.36.0.1   k8-worker2    <none>           <none>
ubuntu@k8-master:~$ vi no_rola.yaml
ubuntu@k8-master:~$ vi no_tola.yaml
ubuntu@k8-master:~$ kubectl apply -f no_tola.yaml
pod/nginx-notoleration1 created
ubuntu@k8-master:~$ kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE     IP          NODE          NOMINATED NODE   READINESS GATES
nginx-notoleration  1/1     Running   0          78s     10.36.0.2   k8-worker2    <none>           <none>
nginx-notoleration1 1/1     Running   0          6s      10.36.0.3   k8-worker2    <none>           <none>
nginx3              1/1     Running   0          5m1s    10.44.0.1   k8-worker1    <none>           <none>
nginx4              1/1     Running   0          5m42s   10.36.0.1   k8-worker2    <none>           <none>
ubuntu@k8-master:~$
```

Created a yaml file using pod definition without toleration. Now this pod cannot schedule on the node which have taint.

```
ubuntu@k8-master:~$ kubectl taint nodes k8-worker1 key1=value1:NoSchedule-
node/k8-worker1 untainted
ubuntu@k8-master:~$ kubectl taint nodes k8-worker1 key1=value1:PreferNoSchedule
node/k8-worker1 tainted
ubuntu@k8-master:~$ kubectl describe nodes k8-worker1 | grep -i taint
Taints:             key1=value1:PreferNoSchedule
ubuntu@k8-master:~$ vi no_tola.yaml
ubuntu@k8-master:~$ kubectl get pods -o wide
NAME                  READY   STATUS    RESTARTS   AGE     IP          NODE          NOMINATED NODE   READINESS GATES
nginx-notoleration    1/1     Running   0          6m8s    10.36.0.2   k8-worker2    <none>           <none>
nginx-notoleration1   1/1     Running   0          4m56s   10.36.0.3   k8-worker2    <none>           <none>
nginx3                1/1     Running   0          9m51s   10.44.0.1   k8-worker1    <none>           <none>
nginx4                1/1     Running   0          10m     10.36.0.1   k8-worker2    <none>           <none>
ubuntu@k8-master:~$ kubectl apply -f no_tola.yaml
The Pod "nginx-PreferNoSchedule1" is invalid: metadata.name: Invalid value: "nginx-PreferNoSchedule1": a lowercase RFC 1123 subdoma
 or '.', and must start and end with an alphanumeric character (e.g. 'example.com', regex used for validation is '[a-z0-9]([-a-z0-9
ubuntu@k8-master:~$ vi no_tola.yaml
ubuntu@k8-master:~$ kubectl apply -f no_tola.yaml
pod/nginx-prefernoschedule1 created
ubuntu@k8-master:~$ kubectl get pods -o wide
NAME                       READY   STATUS    RESTARTS   AGE     IP          NODE          NOMINATED NODE   READINESS GATES
nginx-notoleration         1/1     Running   0          8m47s   10.36.0.2   k8-worker2    <none>           <none>
nginx-notoleration1        1/1     Running   0          7m35s   10.36.0.3   k8-worker2    <none>           <none>
nginx-prefernoschedule1    1/1     Running   0          5s      10.36.0.4   k8-worker2    <none>           <none>
nginx3                     1/1     Running   0          12m     10.44.0.1   k8-worker1    <none>           <none>
nginx4                     1/1     Running   0          13m     10.36.0.1   k8-worker2    <none>           <none>
ubuntu@k8-master:~$ vi no_tola.yaml
ubuntu@k8-master:~$ kubectl apply -f no_tola.yaml
pod/nginx-prefernoschedule2 created
ubuntu@k8-master:~$ kubectl get pods -o wide
NAME                       READY   STATUS    RESTARTS   AGE     IP          NODE          NOMINATED NODE   READINESS GATES
nginx-notoleration         1/1     Running   0          9m31s   10.36.0.2   k8-worker2    <none>           <none>
nginx-notoleration1        1/1     Running   0          8m19s   10.36.0.3   k8-worker2    <none>           <none>
nginx-prefernoschedule1    1/1     Running   0          49s     10.36.0.4   k8-worker2    <none>           <none>
nginx-prefernoschedule2    1/1     Running   0          10s     10.36.0.5   k8-worker2    <none>           <none>
nginx3                     1/1     Running   0          13m     10.44.0.1   k8-worker1    <none>           <none>
nginx4                     1/1     Running   0          13m     10.36.0.1   k8-worker2    <none>           <none>
```

Applied a taint with PreferNoSchedule to the node. This ensures that the scheduler will prefer not to schedule pods without a toleration, but if there are no other suitable nodes, the pod may still be scheduled on the tainted node

```
ubuntu@k8-master:~$ kubectl taint nodes k8-worker1 key1=value1:PreferNoSchedule-
node/k8-worker1 untainted
ubuntu@k8-master:~$ kubectl taint nodes k8-worker2 key1=value1:NoExecute
node/k8-worker2 tainted
ubuntu@k8-master:~$ kubectl get pods -o wide
NAME     READY   STATUS    RESTARTS   AGE   IP          NODE          NOMINATED NODE   READINESS GATES
nginx3   1/1     Running   0          15m   10.44.0.1   k8-worker1    <none>           <none>
ubuntu@k8-master:~$ kubectl get pods -o wide
NAME     READY   STATUS    RESTARTS   AGE   IP          NODE          NOMINATED NODE   READINESS GATES
nginx3   1/1     Running   0          15m   10.44.0.1   k8-worker1    <none>           <none>
ubuntu@k8-master:~$
```

Applied a taint with NoExecute that ensures no new pods will be scheduled on the node, and existing pods will be evicted (terminated) if they don't tolerate the taint