

Taints and Tolerations

Lab 4: Taints and Tolerations

Objective:

- Apply taints to nodes and use tolerations to control Pod scheduling behavior.

Steps

1. Taint a Worker Node

```
kubectl taint nodes minikube key=dedicated:NoSchedule  
kubectl taint nodes minikube-m02 key=dedicated:NoSchedule  
kubectl taint nodes minikube-m03 key=dedicated:NoSchedule
```

What this taint tells Kubernetes:

🛑 “Do not schedule pods here unless they explicitly tolerate this taint.”

This allows you to isolate certain nodes — like those for production workloads, GPU jobs, or in this case, anything "dedicated".

2. Create YAML file: `no-toleration.yaml`

```
apiVersion: v1
kind: Pod
metadata:
  name: no-toleration
spec:
  containers:
  - name: busybox
    image: busybox:musl
    command: ["sleep", "3600"] # keep container running for 1 hour
```

Apply it:

```
kubectl apply -f no-toleration.yaml
```

Observe that the Pod is not scheduled.

3. Create YAML file: `tolerate-dedicated.yaml` to deploy a Pod With a Matching Toleration

```
apiVersion: v1
kind: Pod
metadata:
  name: tolerate-dedicated
spec:
  containers:
  - name: busybox
    image: busybox:musl
    command: ["sleep", "3600"] # keep container running for 1 hour
  tolerations:
  - key: "key"
    value: "dedicated"
    operator: "Equal"
    effect: "NoSchedule"
```

Apply it:

```
kubectl apply -f tolerate-dedicated.yaml
```

4. Verify Scheduling

```
kubectl get pod -o wide
```

The pod is running on one of the nodes.

5. Clean Up

```
kubectl delete pod no-toleration tolerate-dedicated
```

```
# Remove taints
```

```
kubectl taint nodes minikube key=dedicated:NoSchedule-
```

```
kubectl taint nodes minikube-m02 key=dedicated:NoSchedule-
```

```
kubectl taint nodes minikube-m03 key=dedicated:NoSchedule-
```