# KPLABS Course

## Certified Kubernetes Administrator 2022

## Workloads & Scheduling

**ISSUED BY**

Zeal Vora

**REPRESENTATIVE**

instructors@kplabs.in

# Module 1: Labels & Selector

## 1.1 Labels

Labels are key/value pairs that are attached to objects, such as pods

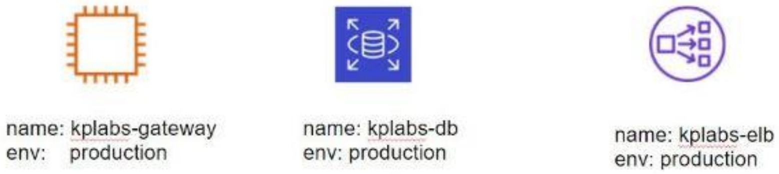
Server
Database
Load Balancer
Load Balancer
Server
Database

## 1.2 Selectors

Selectors allow us to filter objects based on labels.

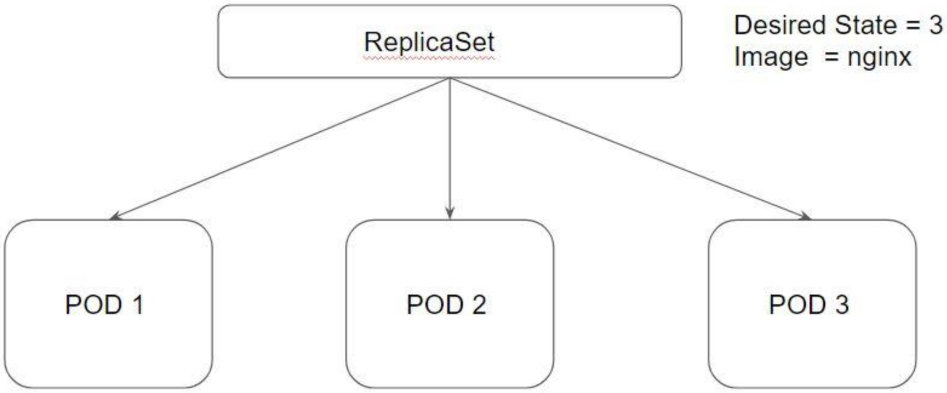
Example:

Show me all the objects which have a label where env: prod



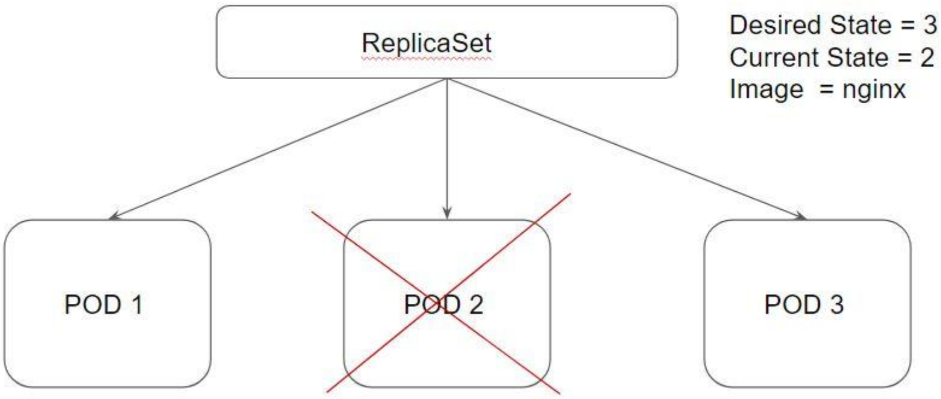
# Module 2:  ReplicaSets

A ReplicaSet purpose is to maintain a stable set of replica Pods running at any given time.



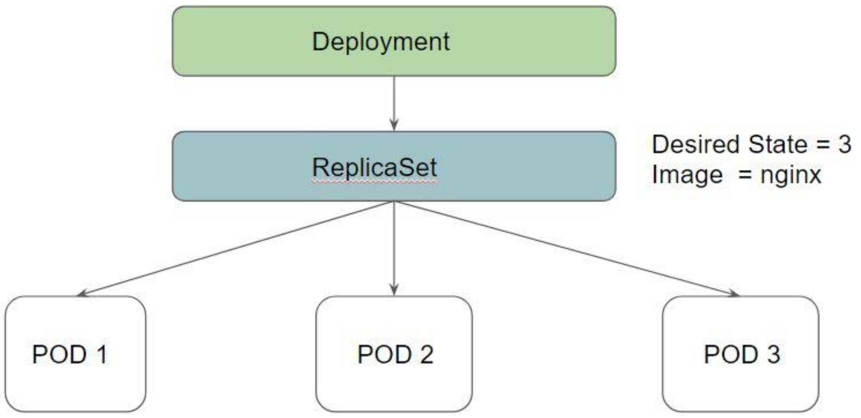
Desired State -  The state of pods which is desired.

Current State - The actual state of pods that are running.



# Module 3: Deployments

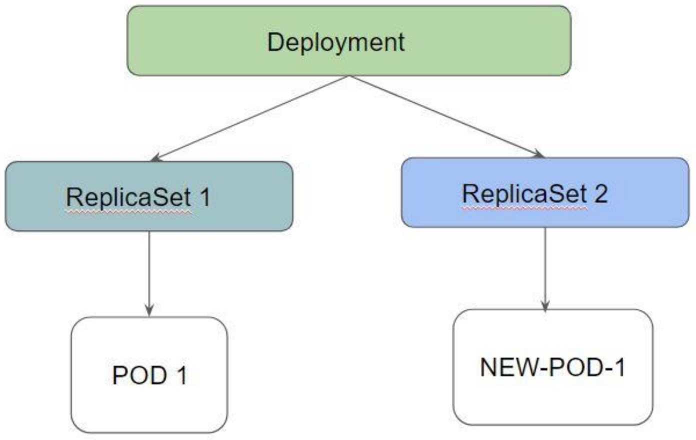
Deployments provide replication functionality with the help of ReplicaSets, along with various additional capabilities like rolling out of changes, rollback changes if required.



## 3.1 Benefits of Deployment - Rollout Changes

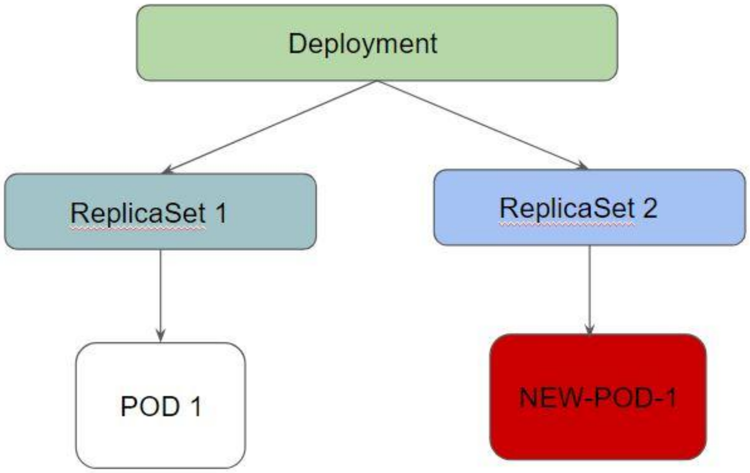
We can easily roll out new updates to our application using deployments.

Deployments will perform an update in a rollout manner to ensure that your app is not down.



### 3.2 Benefits of Deployment - Rollback Changes

Sometimes, you may want to rollback a Deployment; for example, when the Deployment is not stable, such as crash looping



Deployment ensures that only a certain number of Pods are down while they are being updated.

By default, it ensures that at least 25% of the desired number of Pods are up (25% max unavailable)

Deployments keep the history of revision which had been made.

# Module 4:  Important Pointer - Deployments

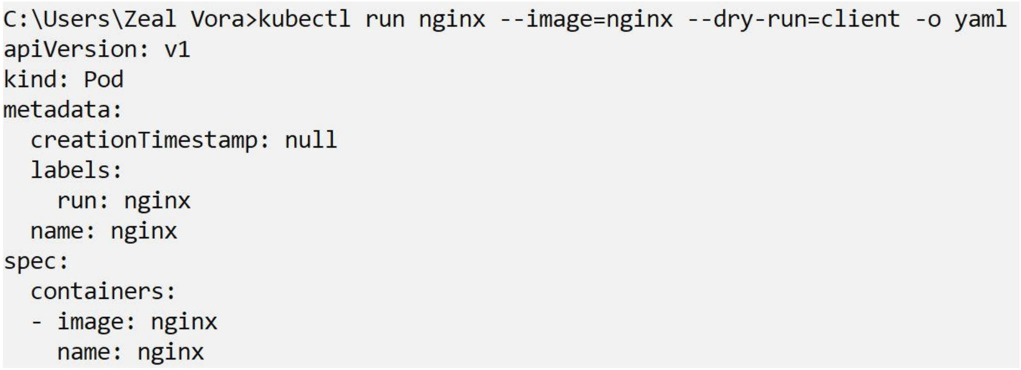
1. You should know how to set a new image to deployment as part of rolling update.
2. You should know the importance of --record instruction.
3. You should know how to rollback a deployment.
4. You should be able to scale the deployment

| Command | Command |
|---|---|
| kubectl set image deployment nginx-deployment nginx=nginx:1.91 --record | Set Image |
| kubectl scale deployment nginx-deployment --replicas 10 | Scale Deployment |
| kubectl rollout undo deployment nginx-deployment | Rollout Undo |

# Module 5: Generating Deployment Manifests via CLI

Till now, we have been referencing the documentation to get the manifests for objects like Pods.

This can be a tedious process and can consume time.

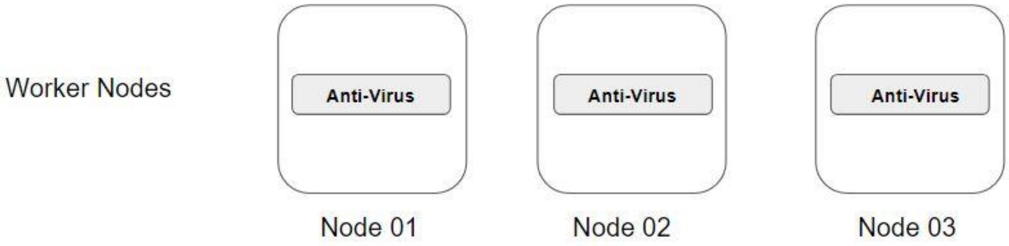


```
C:\Users\Zeal Vora>kubectl run nginx --image=nginx --dry-run=client -o yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx
  name: nginx
spec:
  containers:
  - image: nginx
    name: nginx
```

# Module 6:  Daemonsets

A DaemonSet can ensure that all Nodes run a copy of a Pod.

As nodes are added to the cluster, Pods are added to them.

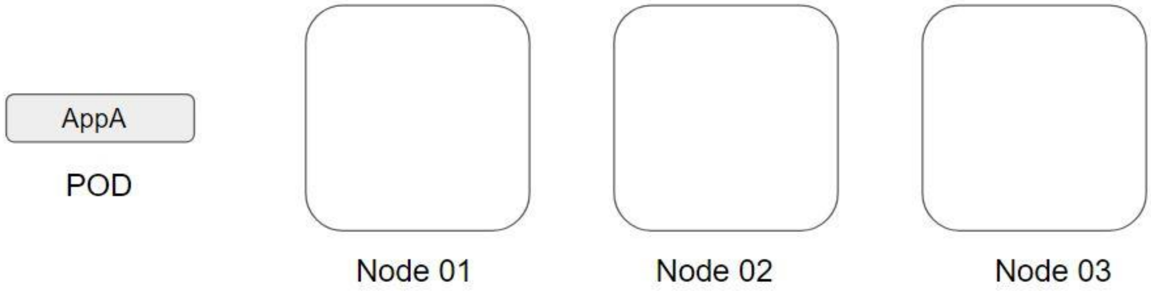


# Module 7: NodeSelector

nodeSelector allows us to add a constraint about running a pod in a specific worker node.

Use-Case:

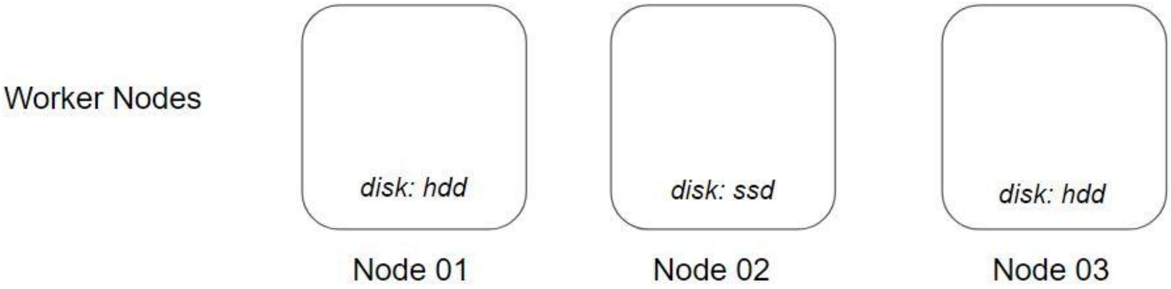
- AppA requires a faster disk in order to be able to run effectively.
- Run AppA in nodes which has SSD.



## Step 1: Adding a Label to the Node

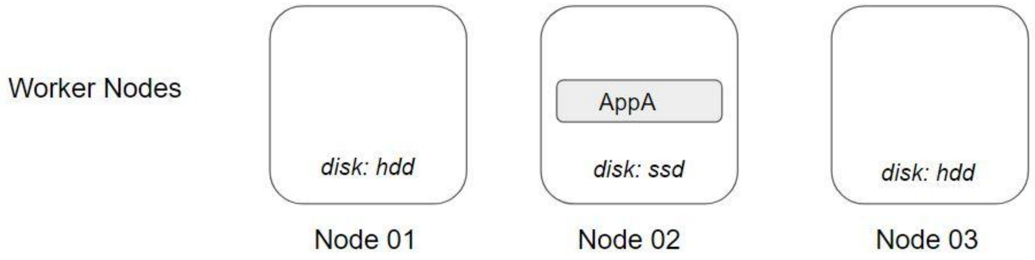
Step 1: Add a Label to your nodes depending on their disk types.

- disk: hdd
- disk:ssd



## Step 2: Using nodeSelector Configuration

Create a nodeSelector configuration to run pods only on nodes which has a label of disk=ssd



## Module 8:  Node Affinity

For multiple reasons, there can be a need to run a pod on a specific worker node.

There can be multiple reasons, node hardware being the common one.

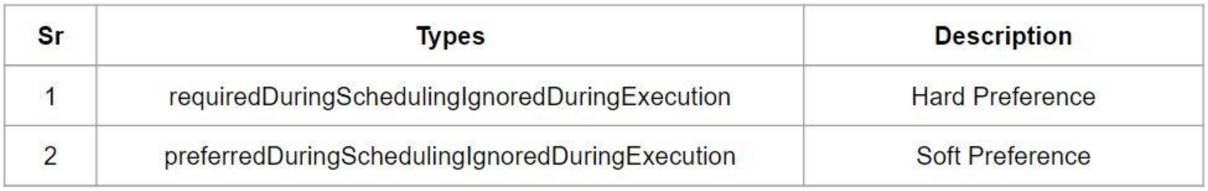
Node affinity is a set of rules used by the scheduler to determine where a pod can be placed

In Kubernetes terms, it is referred to as nodeSelector, and nodeAffinity/podAffinity fields under PodSpec.

In Kubernetes, we can achieve nodeAffinity with the help of:

- nodeSelector
- nodeAffinity (more flexibility)

Node affinity is conceptually similar to nodeSelector – it allows you to constrain which nodes your pod is eligible to be scheduled on, based on labels on the node.



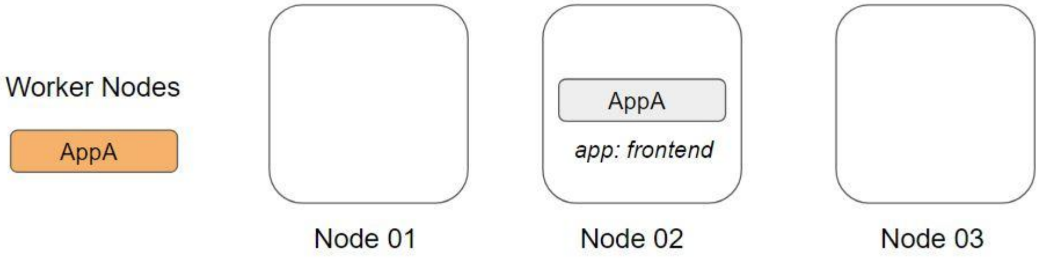
| Sr | Types | Description |
|---|---|---|
| 1 | requiredDuringSchedulingIgnoredDuringExecution | Hard Preference |
| 2 | preferredDuringSchedulingIgnoredDuringExecution | Soft Preference |

# Module 9: Pod Affinity and Pod Anti-Affinity

First Question: Where should I be running this pod?

With Node Affinity, the question became: Should I be running my pod in this node?

The considerations are still about the node. No outside information is considered apart from node.



## Step 1: Pod Selector

The first thing to figure out: "What other POD are we referring to"

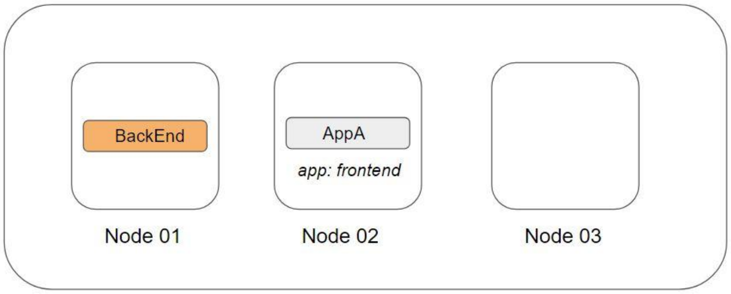
In this case, we are referring to other POD as AppA which has a label of app:frontend

Answer: I want BackEnd Pod to be running in the same place as AppA Pod.

## Step 2: Topology

Topology refers to "what does the same place mean"?

It can mean the same place if we look at the zone or region level (same AZ / same region)
It can mean a different place if we look at the host level.



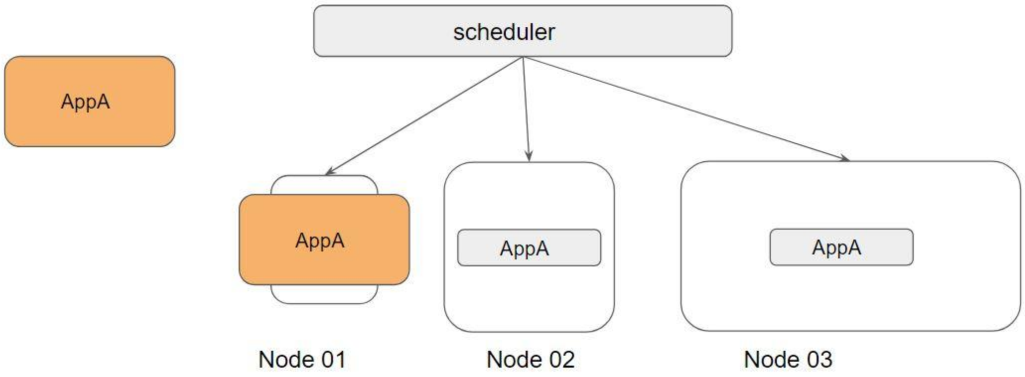
Step 3: Yes/No

Should I run my pod in the same place as the other POD?  (Yes or No)

Yes:  Pod Affinity
No:   Pod Anti-Affinity

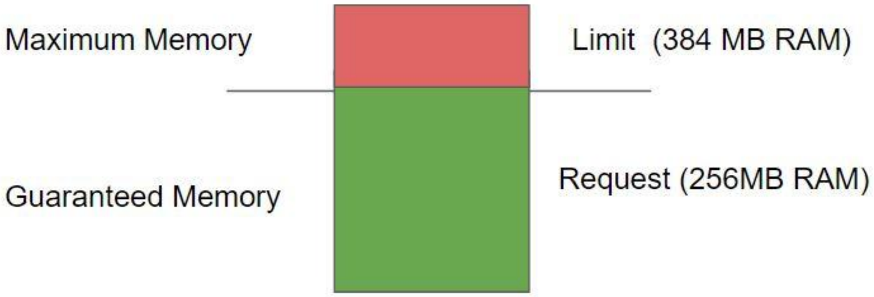
# Module 10:  Resource Requests and Limits

If you schedule a large application in a node which has limited resources, then it will soon lead to OOM or others and will lead to downtime.



Requests and Limits are two ways in which we can control the amount of resource that can be assigned to a pod (resource like CPU and Memory)

Requests:  Guaranteed to get.
Limits:      Makes sure that the container does not take node resources above a specific value.

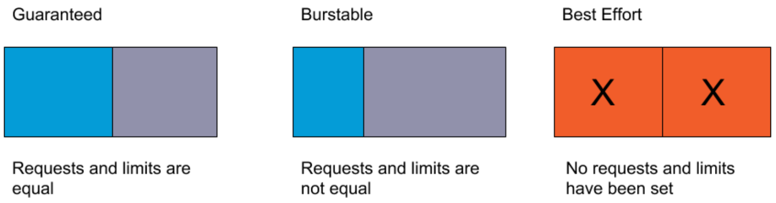


Kubernetes Scheduler decides the ideal node to run the pod depending on the requests and limits.

If your POD requires 8GB of RAM, however, there are no nodes within your cluster which has 8GB RAM, then your pod will never get scheduled.

Kubernetes Scheduler decides the ideal node to run the pod depending on the requests and limits.

If your POD requires 8GB of RAM, however, there are no nodes within your cluster which has 8GB RAM, then your pod will never get scheduled.


Guaranteed
Burstable
Best Effort
X
X
Requests and limits are equal
Requests and limits are not equal
No requests and limits have been set

## Module 11: Static Pods

You can directly inform the kubelet that it needs to run a specific pod.

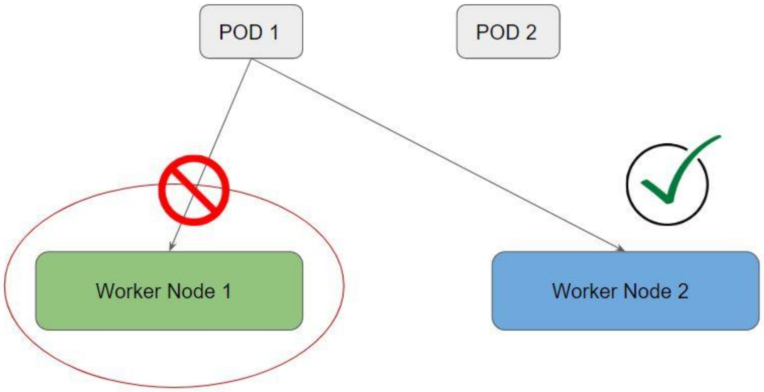
There are multiple ways in which you can tell kubelet to run a pod.

Pod created directly without schedulers are also referred to as Static Pods.

# Module 12: Taints and Toleration

## 12.1 Understanding Taints:

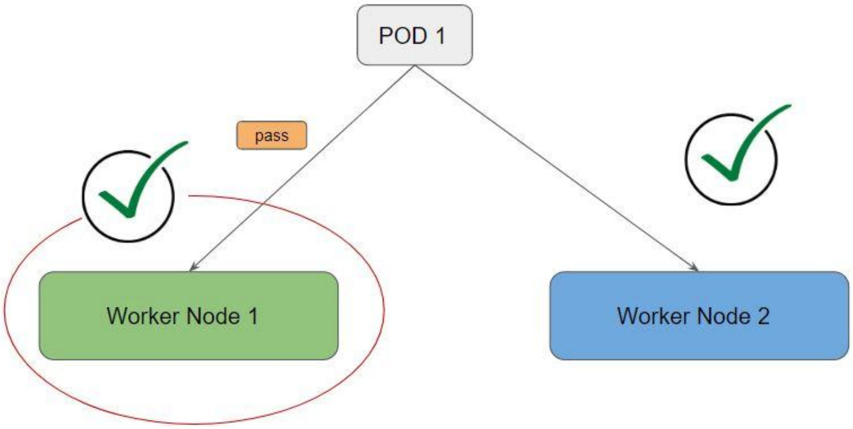
Taints are used to repel the pods from a specific node



## 12.2 Understanding Toleration:

In order to enter the taint worker node, you need a special pass.

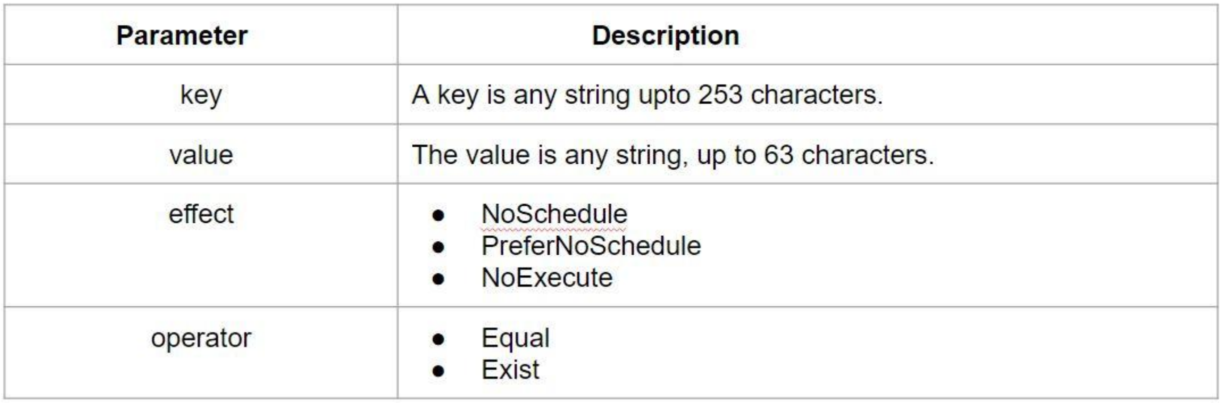
This pass is called toleration.



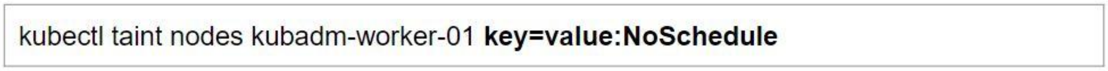
# Module 13: Components of Taints and Tolerations

A taint allows a node to refuse pod to be scheduled unless that pod has matching toleration.

We can apply toleration to a pod within the PodSpec

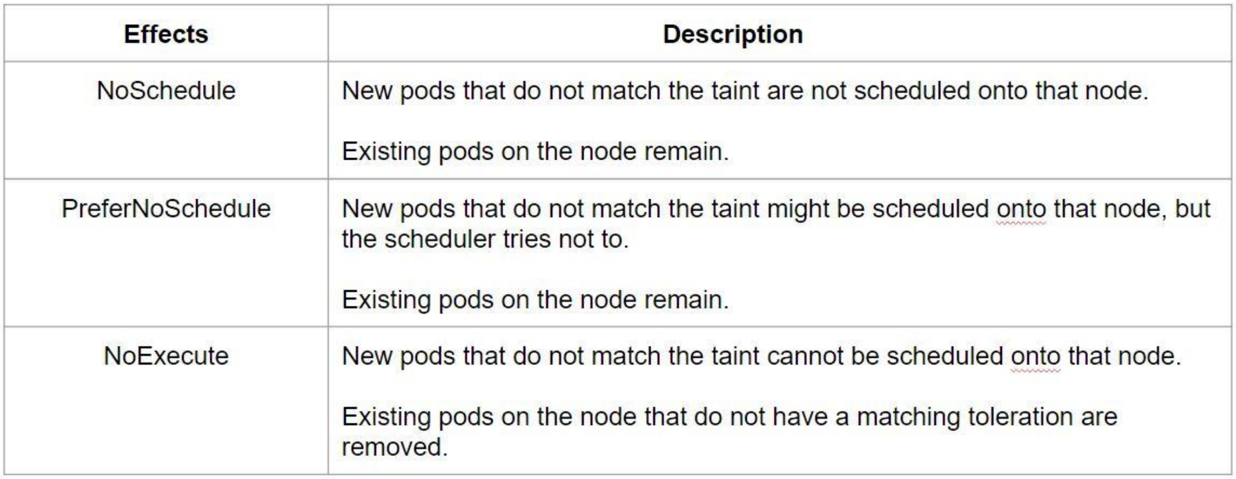


| Parameter | Description |
|---|---|
| key | A key is any string upto 253 characters. |
| value | The value is any string, up to 63 characters. |
| effect | ● NoSchedule<br>● PreferNoSchedule<br>● NoExecute |
| operator | ● Equal<br>● Exist |



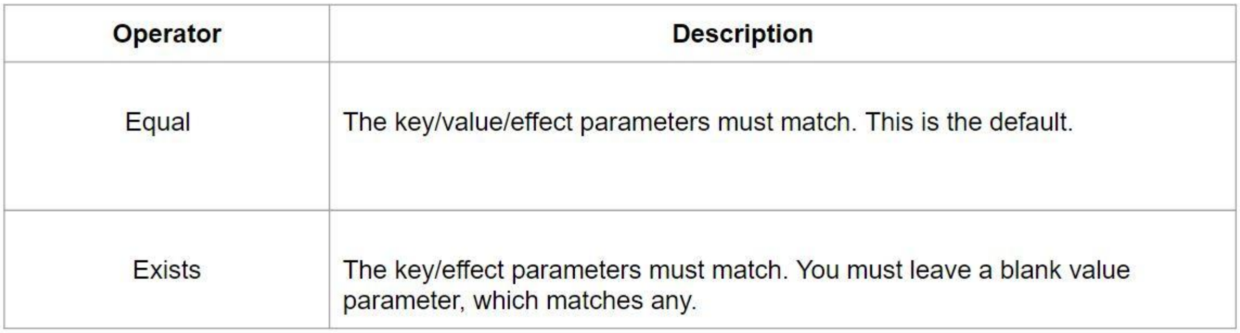
```
kubectl taint nodes kubadm-worker-01 key=value:NoSchedule
```

The following table shows the effects:



| Effects | Description |
|---|---|
| NoSchedule | New pods that do not match the taint are not scheduled onto that node.<br>Existing pods on the node remain. |
| PreferNoSchedule | New pods that do not match the taint might be scheduled onto that node, but the scheduler tries not to.<br>Existing pods on the node remain. |
| NoExecute | New pods that do not match the taint cannot be scheduled onto that node.<br>Existing pods on the node that do not have a matching toleration are removed. |

The following are the two primary operators.



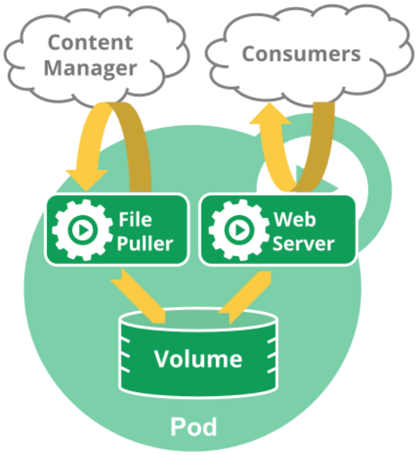
| Operator | Description |
|---|---|
| Equal | The key/value/effect parameters must match. This is the default. |
| Exists | The key/effect parameters must match. You must leave a blank value parameter, which matches any. |

# Module 14: Multi-Container POD Patterns

## 14.1 Overview of Sidecar Pattern

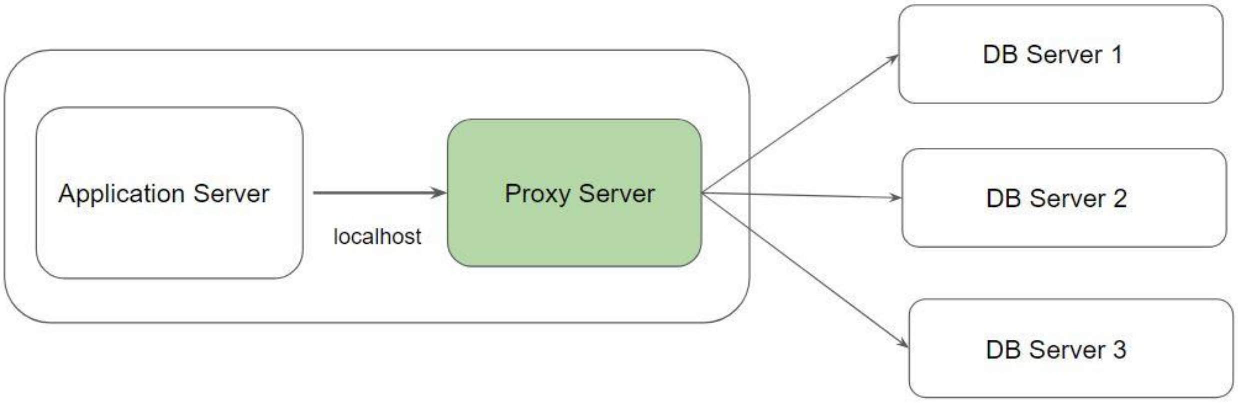
Sidecar pattern is nothing but running multiple containers as part of a pod in a single node.

In the diagram below we see that there is a web-server container which servers files from volumes and a separate sidecar container “file puller” which fetches and updates those files.



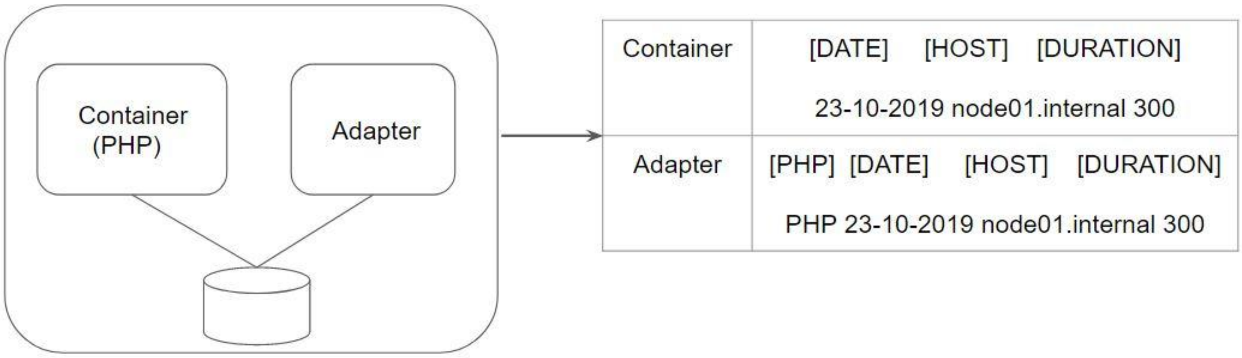
## 14.2 Ambassador Pattern

Ambassador Pattern is a type of sidecar pattern where the second container is primarily used to proxy the requests.



### 14.3 Sidecar Container - Adapter Pattern

Adapter Pattern is generally used to transform the application output to standardize / normalize it for aggregation.

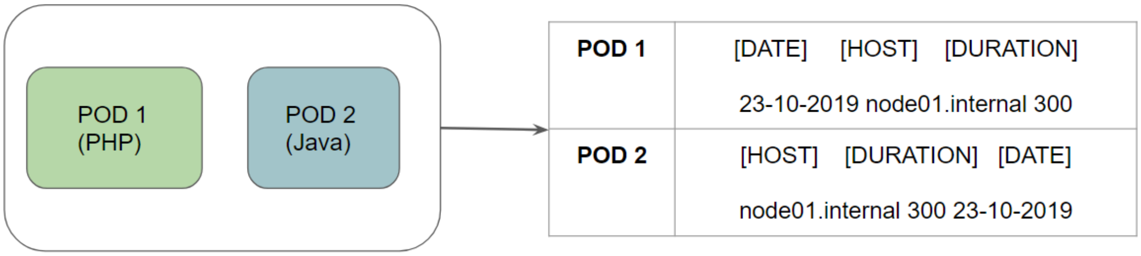


| | |
|---|---|
| Container | [DATE] [HOST] [DURATION]<br>23-10-2019 node01.internal 300 |
| Adapter | [PHP] [DATE] [HOST] [DURATION]<br>PHP 23-10-2019 node01.internal 300 |

# Module 15: Adapter Pattern

### 15.1 Overview of Adapter Pattern

Adapter Pattern is generally used to transform the application output to standardize / normalize it for aggregation.



| | |
|---|---|
| **POD 1** | [DATE] [HOST] [DURATION]<br>23-10-2019 node01.internal 300 |
| **POD 2** | [HOST] [DURATION] [DATE]<br>node01.internal 300 23-10-2019 |

### 15.2 Standardizing Log Format

Within your logging application, you want to have a common standard format.

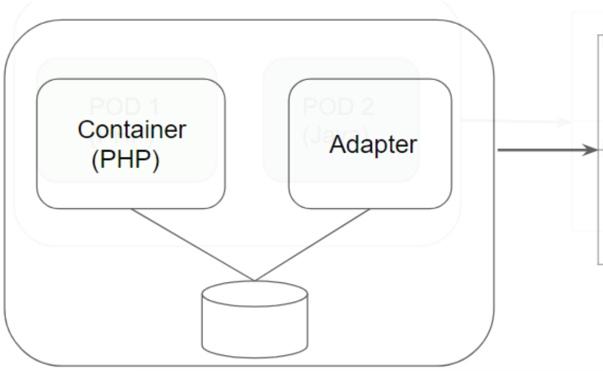
[PHP|JAVA] [DATE] [HOST] [DURATION]

PHP 23-10-2019 node01.internal 300

## 15.3 Transforming Logs

With Adapter Container, you can transform your logs to standardize it.

Since containers in PODS can share volumes, adapter containers can easily access App logs.



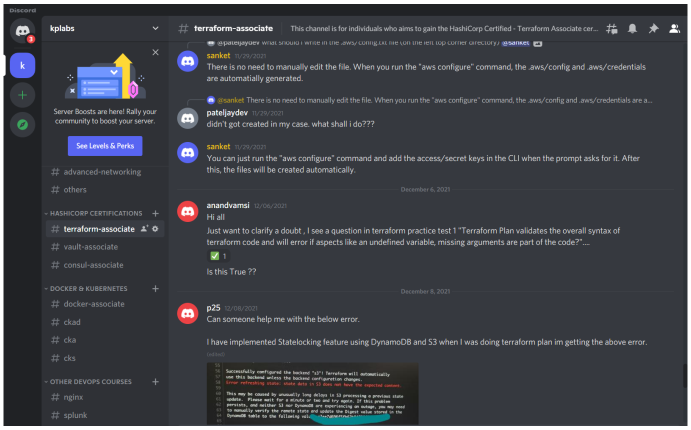
| | |
|---|---|
| Container | [DATE] [HOST] [DURATION]<br>23-10-2019 node01.internal 300 |
| Adapter | [PHP] [DATE] [HOST] [DURATION]<br>PHP 23-10-2019 node01.internal 300 |

## Join Our Discord Community

We invite you to join our Discord community, where you can interact with our support team for any course-based technical queries and connect with other students who are doing the same course.

Joining URL:

http://kplabs.in/chat



# terraform-associate — This channel is for individuals who aims to gain the HashiCorp Certified - Terraform Associate cer...

@pateljaydev what should i write in the .aws/config.txt file (on the left top corner directory) @sanket

sanket 11/29/2021
There is no need to manually edit the file. When you run the "aws configure" command, the .aws/config and .aws/credentials are automatially generated.

@sanket There is no need to manually edit the file. When you run the "aws configure" command, the .aws/config and .aws/credentials are a...

pateljaydev 11/29/2021
didn't got created in my case. what shall i do???

sanket 11/29/2021
You can just run the "aws configure" command and add the access/secret keys in the CLI when the prompt asks for it. After this, the files will be created automatically.

December 6, 2021

anandvamsi 12/06/2021
Hi all
Just want to clarify a doubt , I see a question in terraform practice test 1 "Terraform Plan validates the overall syntax of terraform code and will error if aspects like an undefined variable, missing arguments are part of the code?"....
✅ 1
Is this True ??

December 8, 2021

p25 12/08/2021
Can someone help me with the below error.

I have implemented Statelocking feature using DynamoDB and S3 when I was doing terraform plan im getting the above error.
(edited)

```
Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.
Error refreshing state: state data in S3 does not have the expected content.

This may be caused by unusually long delays in S3 processing a previous state
update.  Please wait for a minute or two and try again. If this problem
persists, and neither S3 nor DynamoDB are experiencing an outage, you may need
to manually verify the remote state and update the Digest value stored in the
DynamoDB table to the following valu[redacted]
```