
Overview of the Course

Certified Kubernetes Administrator

What this course is all about?

This is a certification specific course aimed at individuals who intends to gain the “Certified Kubernetes Administrator” certification.

This is a perfect course for certification OR if you plan to start your journey in Kubernetes.



Certification is Beneficial

We will be covering all the domains for the Certified Kubernetes Administrator certification.

- Domain 1 - Cluster Architecture, Installation & Configuration
- Domain 2 - Workloads and Scheduling
- Domain 3 - Services and Networking
- Domain 4 - Storage
- Domain 5 - Troubleshooting

Our Modified Course Structure

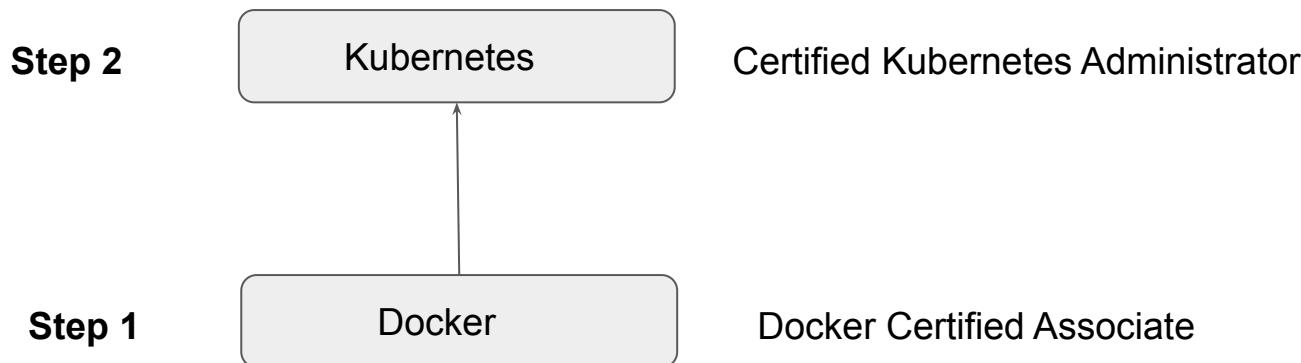
We have a modified course structure that will suite even beginners who are new to Kubernetes.

- Domain 1 - Core Concepts
- Domain 2 - Workloads and Scheduling
- Domain 3 - Services and Networking
- Domain 4 - Storage
- Domain 5 - Security
- Domain 6 - Cluster Architecture, Installation & Configuration
- Domain 7 - Troubleshooting
- Domain 8 - Exam Preparation Section

Prerequisite for this Course

We start our journey on Kubernetes from absolute scratch.

However, a prior Docker knowledge is required.



Our Awesome Docker Course

We have a great course available on Docker.

IT & Software > IT Certification > Docker Certified Associate (DCA)

Docker Certified Associate 2020

Master Course to prepare for Docker Certified Associate certification.

Bestseller | 4.5 ★★★★☆ (968 ratings) 5,744 students

Created by Zeal Vora

Last updated 8/2020 • English

[Wishlist](#) [Share](#) [Gift this course](#)

What you'll learn

- ✓ Strong Fundamentals of Docker
- ✓ Docker Security
- ✓ Orchestration with Docker Swarm
- ✓ Docker Networking, Storage & Image Management

Important Pointer

Certified Kubernetes Administrator is a practical-oriented exam.

You will be given various scenarios which you will have to implement / fix the issue.

One Word: Practice, Practice, Practice.



Small Request - Review the Course

Your time of two minutes writing a text-based review proves to very important.

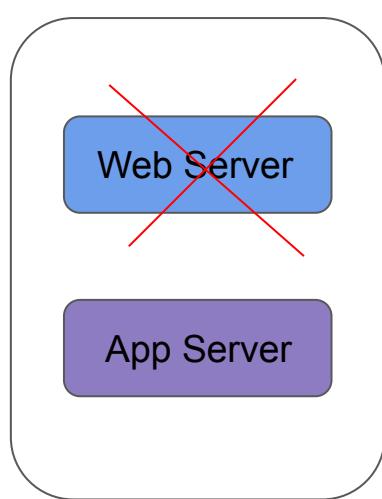


Container Orchestration

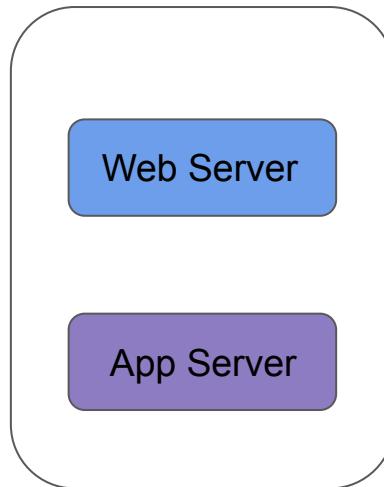
Build once, use anywhere

Getting Started

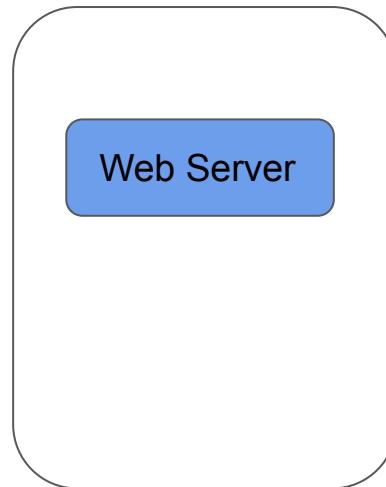
Container orchestration is all about managing the life cycles of containers, especially in large, dynamic environments.



VM 1

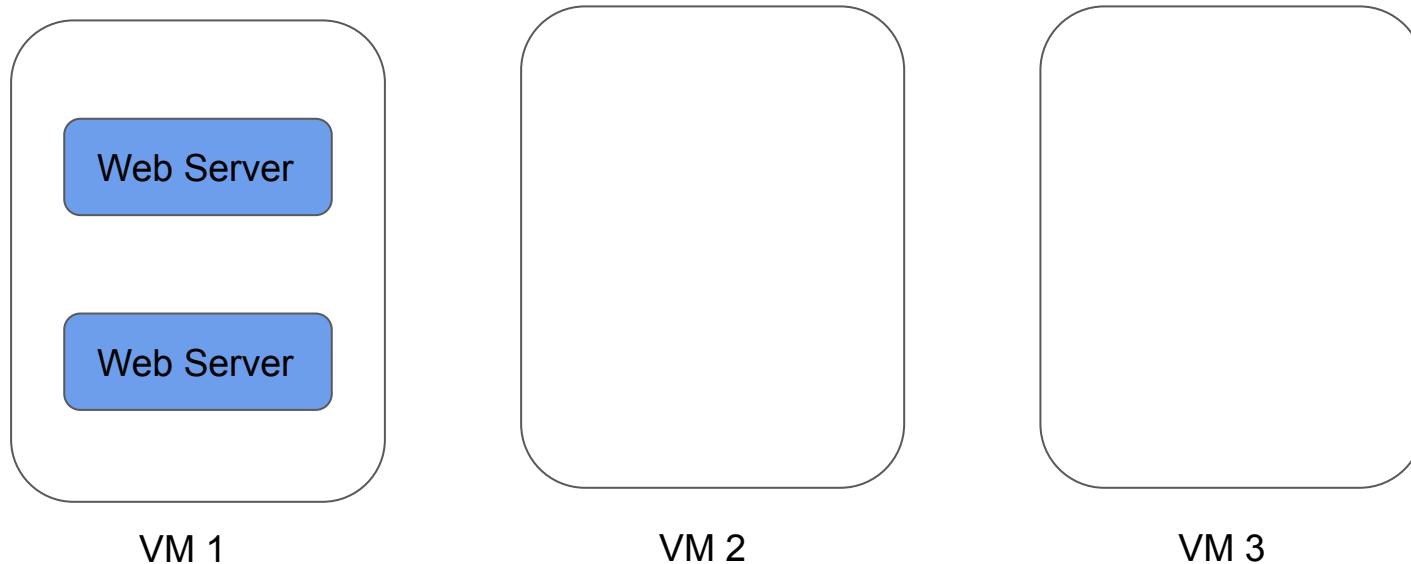


VM 2



VM 3

Requirement: Minimum of 2 web-server should be running all the time.



Importance of Container Orchestration

Container Orchestration can be used to perform lot of tasks, some of them includes:

- Provisioning and deployment of containers
- Scaling up or removing containers to spread application load evenly
- Movement of containers from one host to another if there is a shortage of resources
- Load balancing of service discovery between containers
- Health monitoring of containers and hosts

Container Orchestration Solutions

There are many container orchestration solutions which are available, some of the popular ones include:

- Docker Swarm
- Kubernetes
- Apache Mesos
- Elastic Container Service (AWS ECS)

There are also various container orchestration platforms available like EKS.

Introduction to Kubernetes

Orchestrator Engine

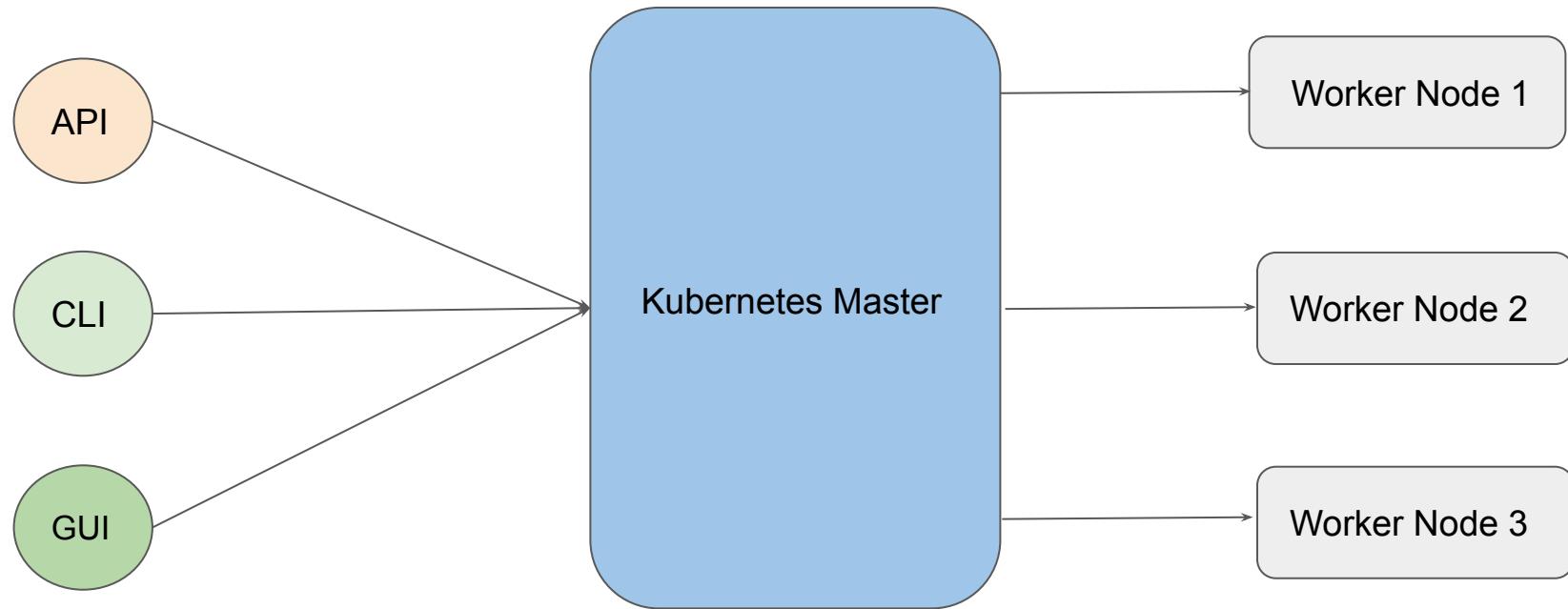
Introduction to Kubernetes

Kubernetes (K8s) is an open-source container orchestration engine developed by Google.

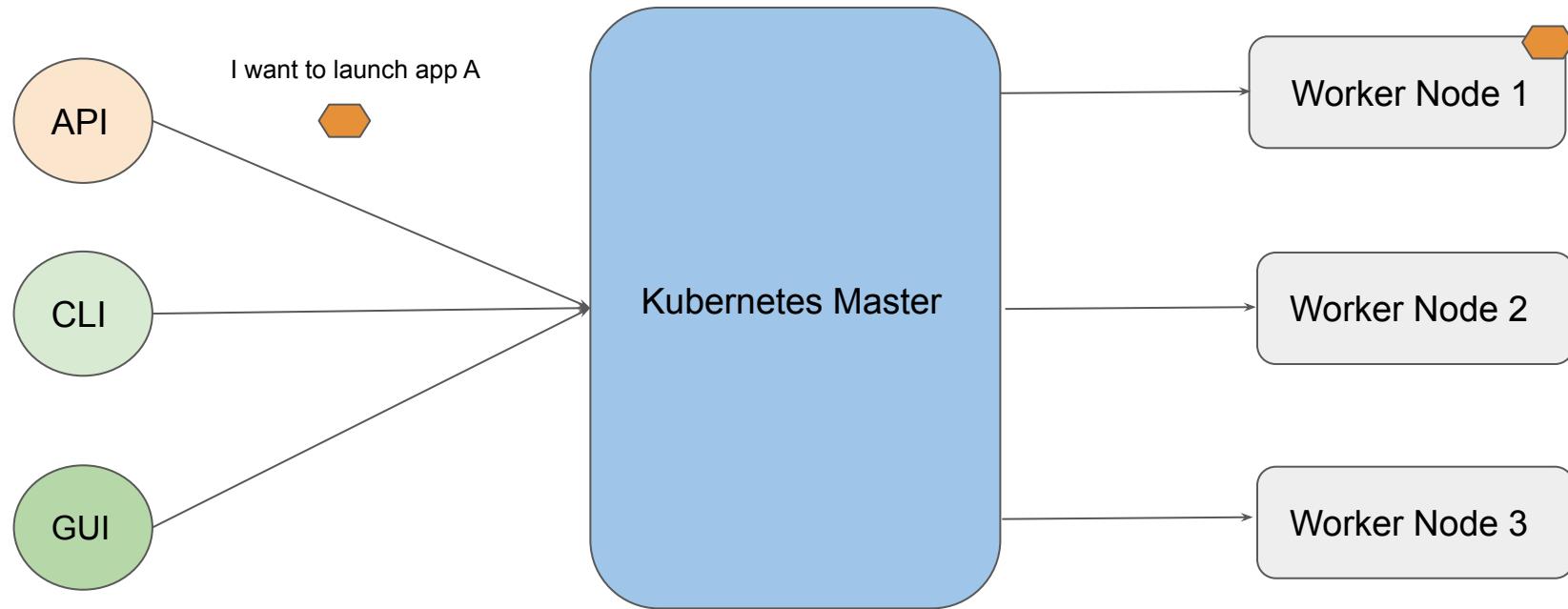
It was originally designed by Google, and is now maintained by the Cloud Native Computing Foundation.



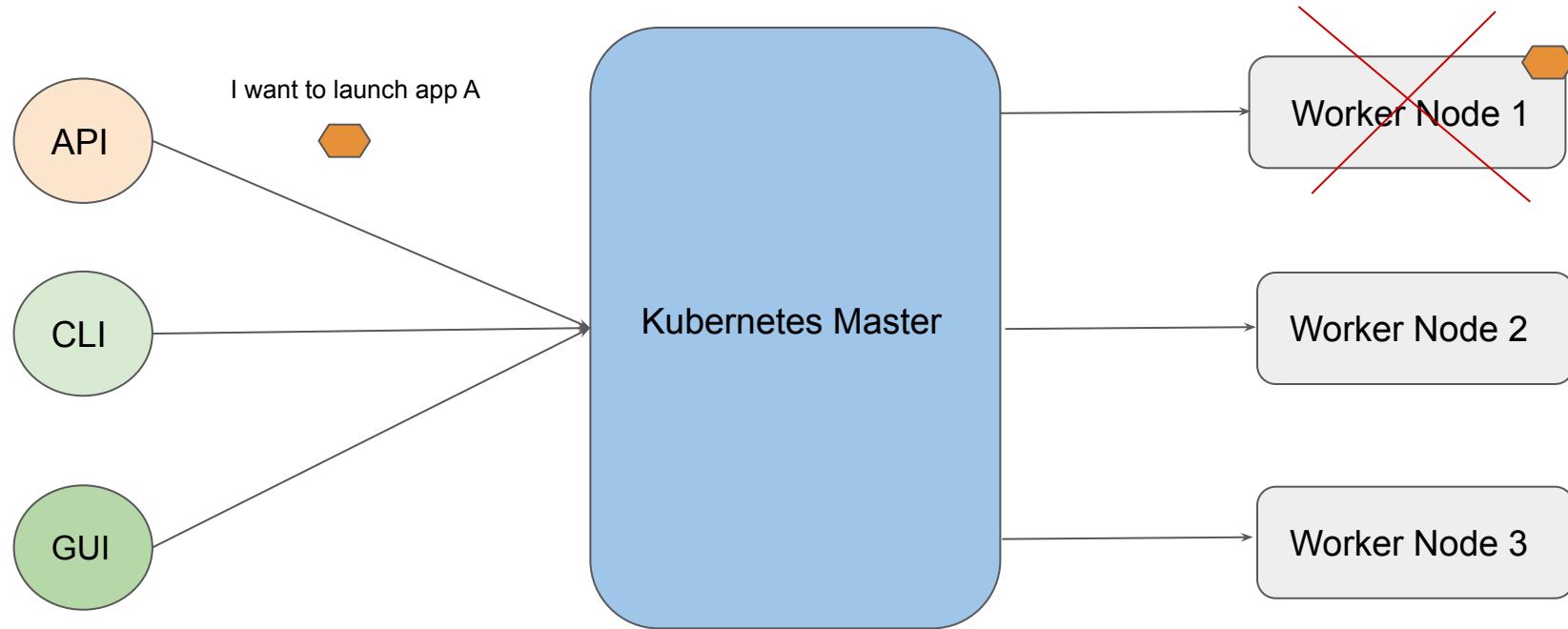
Architecture of Kubernetes



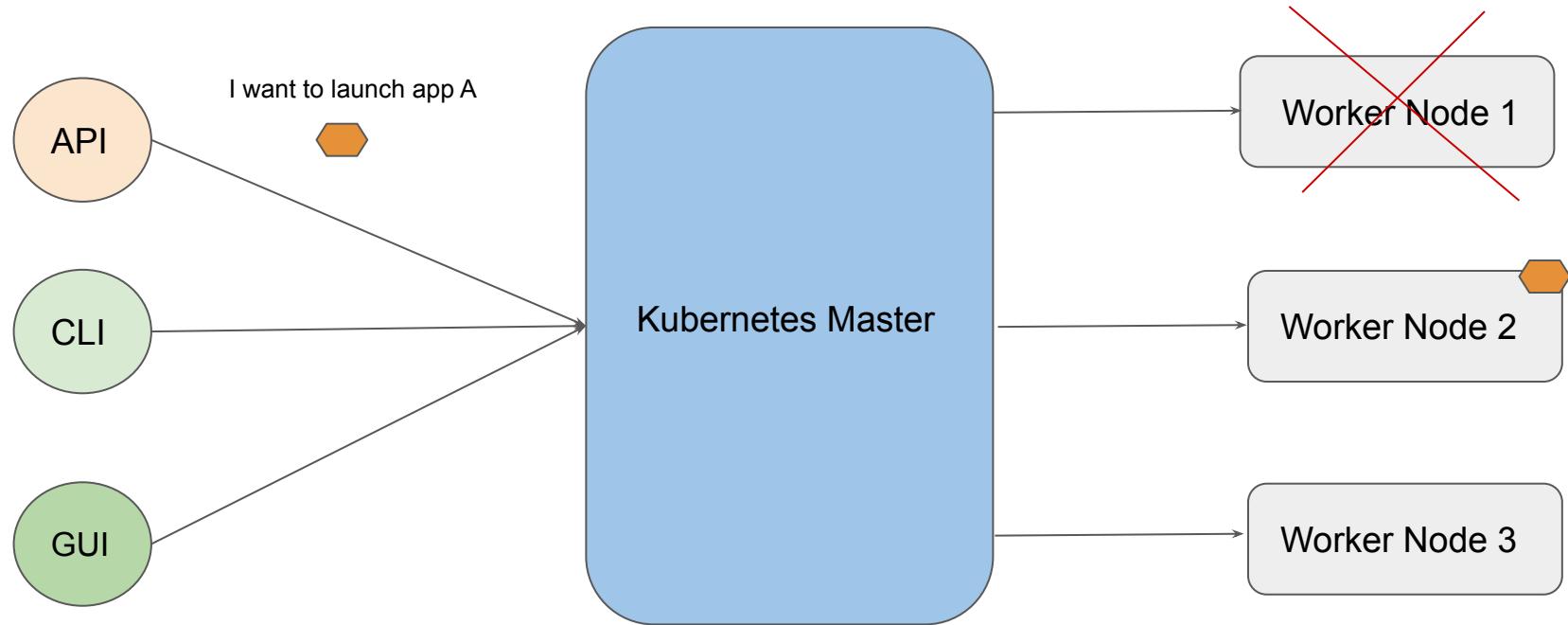
Architecture of Kubernetes



Architecture of Kubernetes



Architecture of Kubernetes



Installation Options

Let's Get Started

Understanding Installation Methods

There are multiple ways to get started with fully functional kubernetes environment

1. Use the Managed Kubernetes Service
2. Use Minikube
3. Install & Configure Kubernetes Manually (Hard Way)

1. Managed Kubernetes Service

Various providers like AWS, IBM, GCP and others provides managed Kubernetes clusters.

Most organizations prefer to make use of this approach.



2. Minikube

Minikube is a tool that makes it easy to run Kubernetes locally.

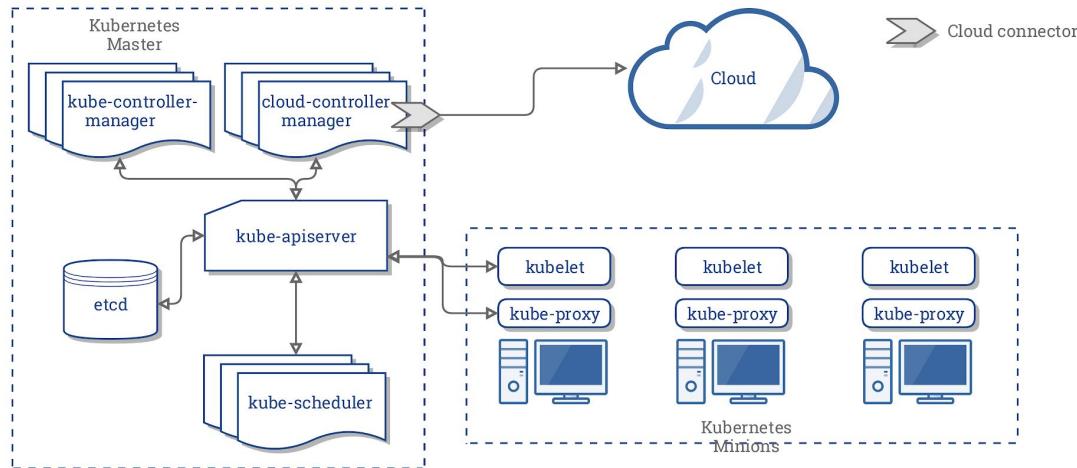
Minikube runs a single-node Kubernetes cluster inside a Virtual Machine (VM) on your laptop for users looking to try out Kubernetes or develop with it day-to-day.



minikube

Install & Configure Kubernetes Manually (Hard Way)

In this approach, you install and configure components of Kubernetes individually.

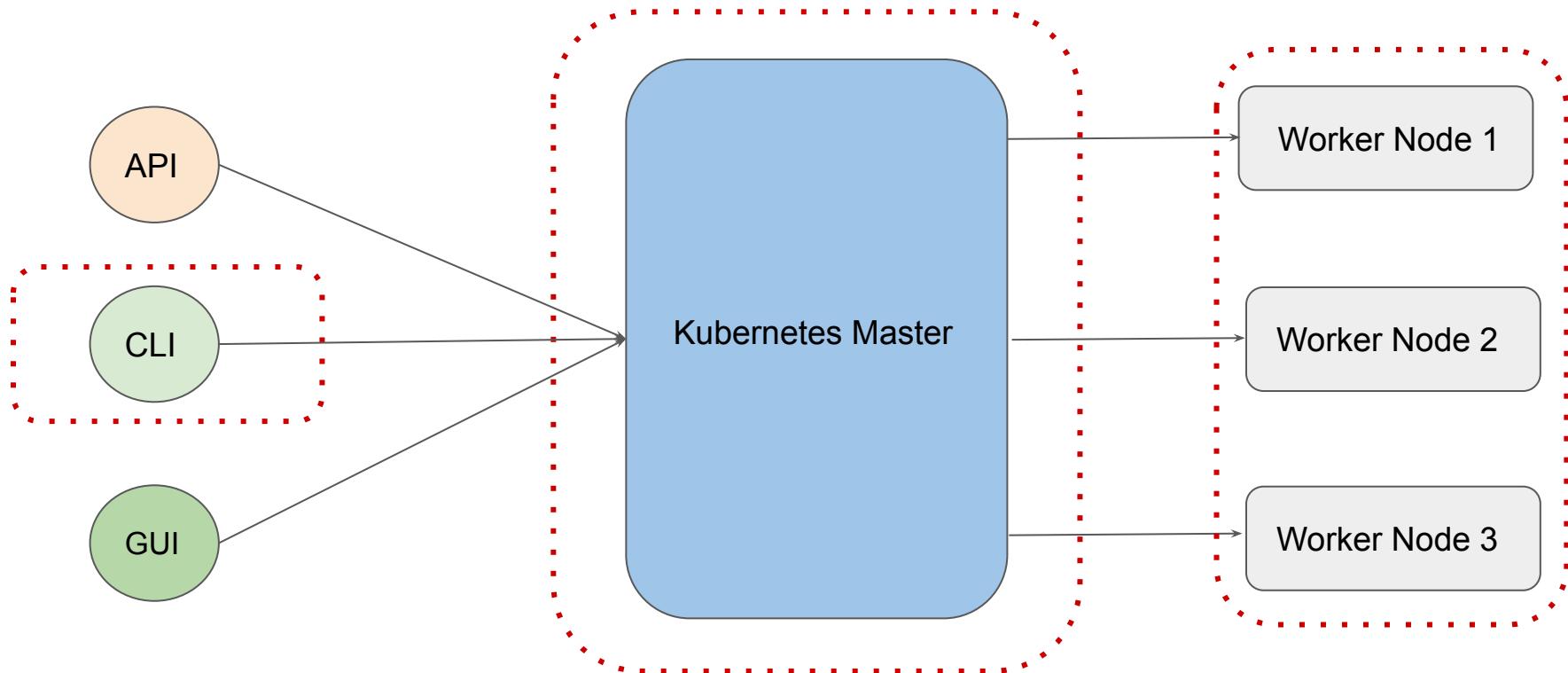


Installation Aspects

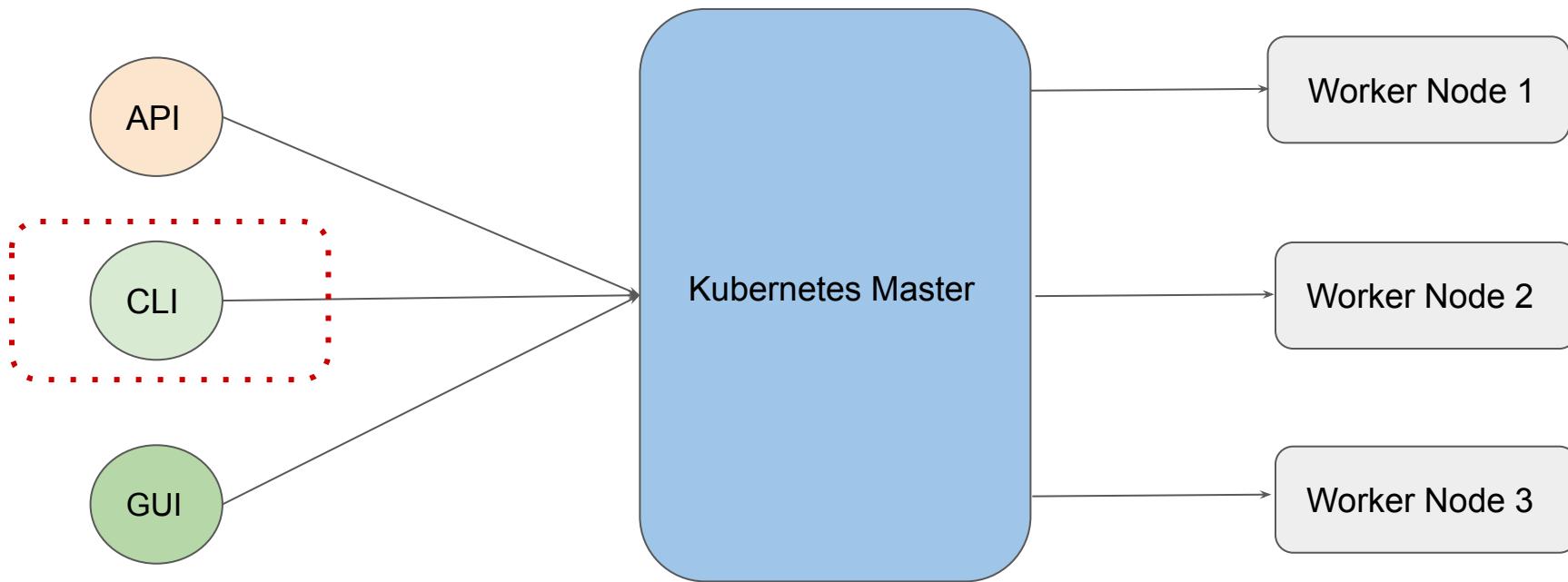
Things to configure while working with Kubernetes.

Sr No	Things to Install	Description
1	kubectl	CLI for running user commands against cluster.
2	Kubernetes Master	Kubernetes Cluster by itself.
3	Worker Node Agents	Kubernetes Node Agent

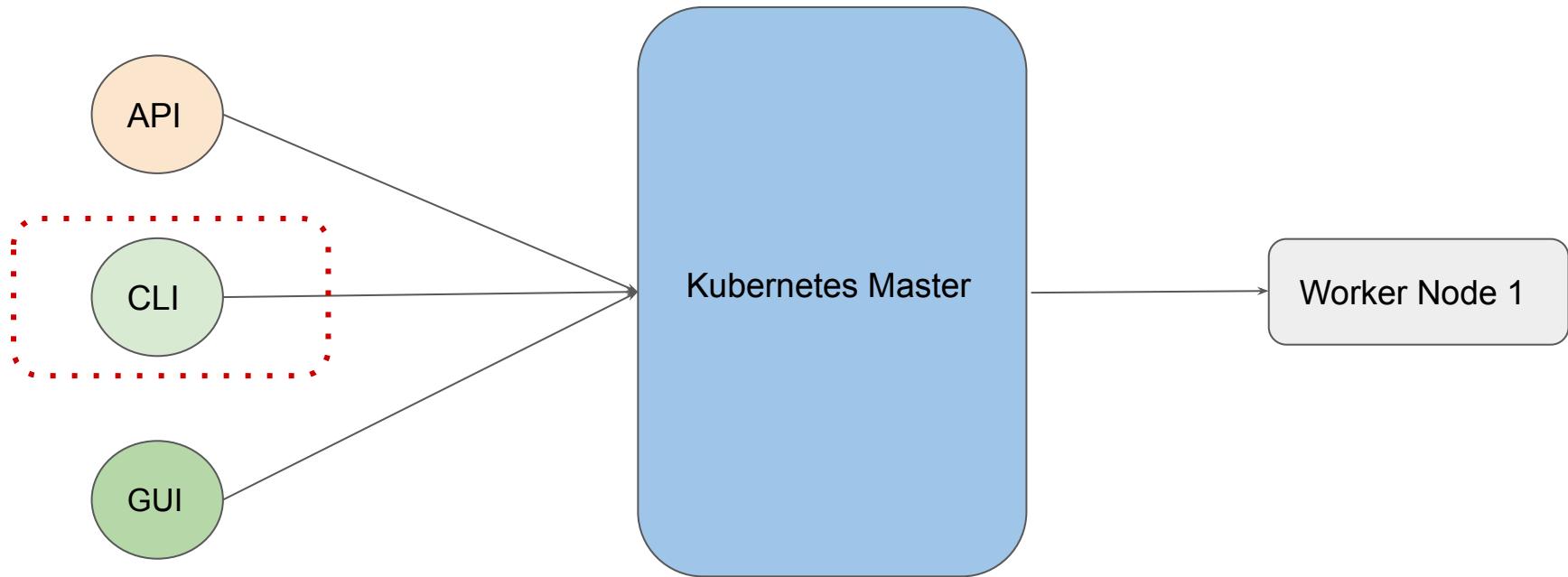
Components to be Configured - Hard Way



Components to be Configured - Managed Service



Components to be Configured - Minikube



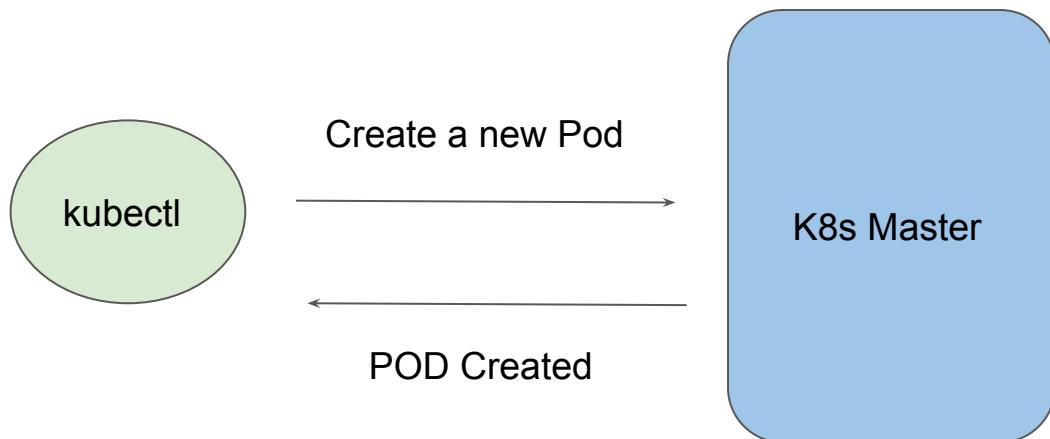
Installing kubectl

Kubernetes Level

Overview of kubectl

The Kubernetes command-line tool, kubectl, allows you to run commands against Kubernetes clusters.

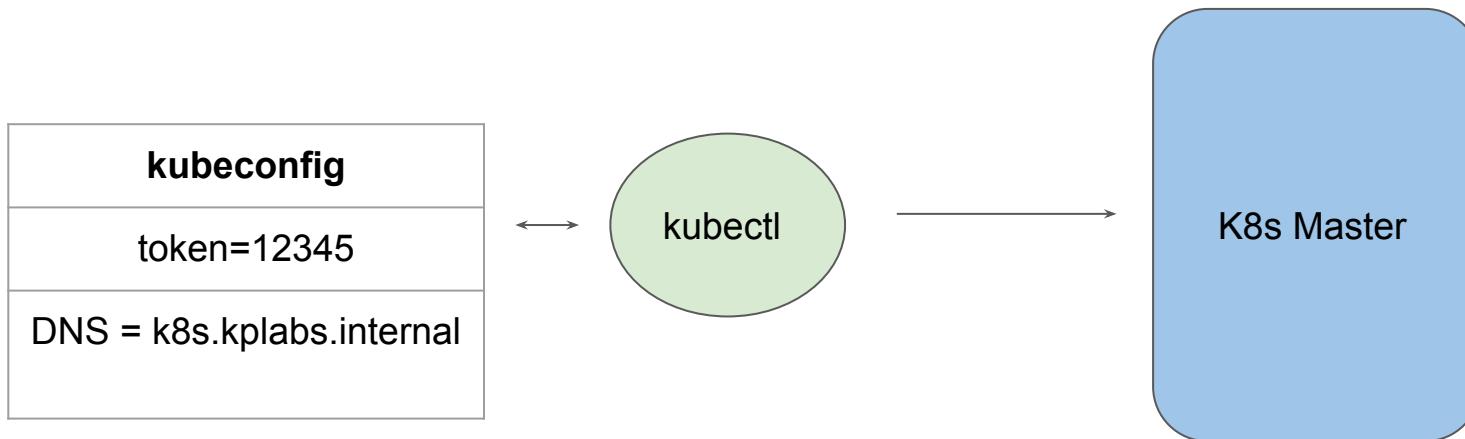
You can use kubectl to deploy applications, inspect and manage cluster resources, and view logs.



High-Level Workflow

To connect to the Kubernetes Master, there are two important data which kubectl needs:

1. DNS / IP of the Cluster
2. Authentication Credentials



Installing kubectl

kubectl is installable on a variety of Linux platforms, macOS and Windows

Install kubectl binary with curl on Windows [🔗](#)

1. Download the [latest release v1.21.0](#).

Or if you have `curl` installed, use this command:

```
curl -LO https://dl.k8s.io/release/v1.21.0/bin/windows/amd64/kubectl.exe
```

Note: To find out the latest stable version (for example, for scripting), take a look at <https://dl.k8s.io/release/stable.txt>.

PODS

Let's get started

The first thing to do!

The initial thing we love to do in any orchestrator is to run our first container.

Docker Command:

```
docker run --name mywebserver nginx
```

Kubectl command:

```
kubectl run mywebserver --image=nginx
```

Exec into Containers

The initial thing we love to do in any orchestrator is to run our first container.

Docker Command:

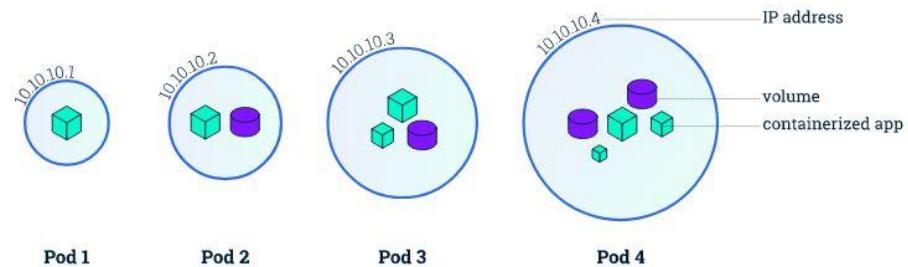
```
docker exec -it mywebserver bash
```

Kubectl command:

```
kubectl exec -it mywebserver -- bash
```

Kubernetes POD

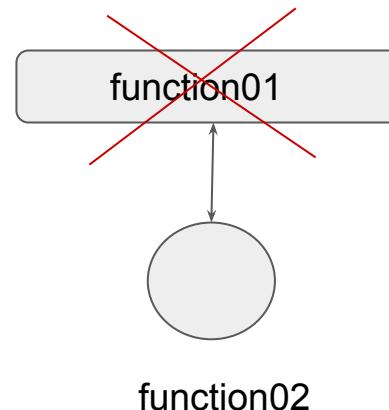
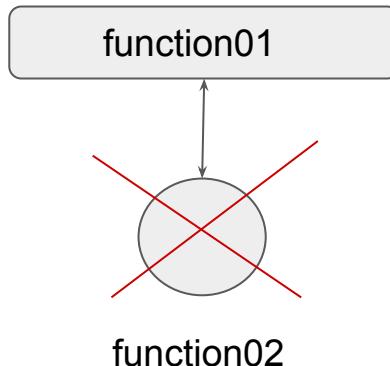
A Pod in Kubernetes represents a group of one or more application containers , and some shared resources for those containers.



Benefits of Pods

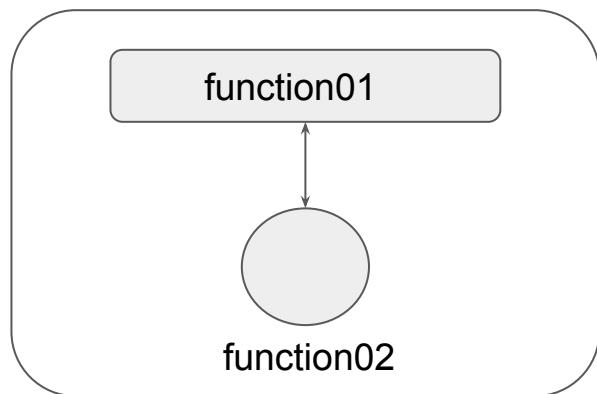
Many application might have more then one container which are tightly coupled and in one-to-one relationship.

- docker run -dt --name myweb01 function01
- docker run -dt --name myapp01 function02

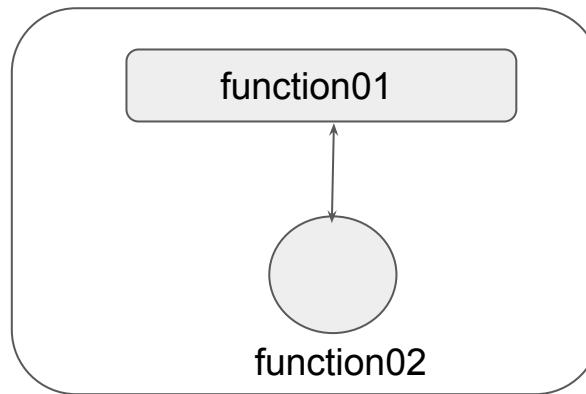


Dealing with Tightly Coupled Application

Containers within a Pod share an IP address and port space, and can find each other via localhost.



POD 1



POD 2

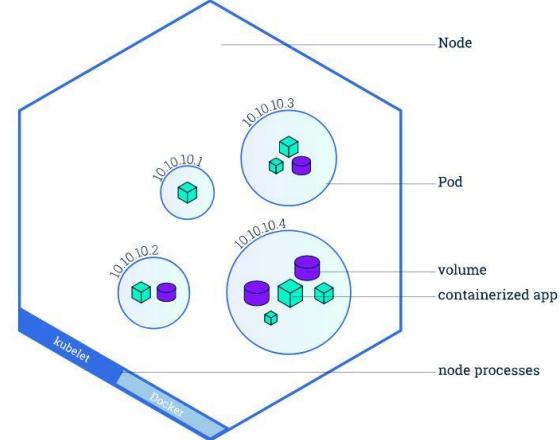
Kubernetes POD

A Pod always runs on a Node.

A Node is a worker machine in Kubernetes.

Each Node is managed by the Master.

A Node can have multiple pods.

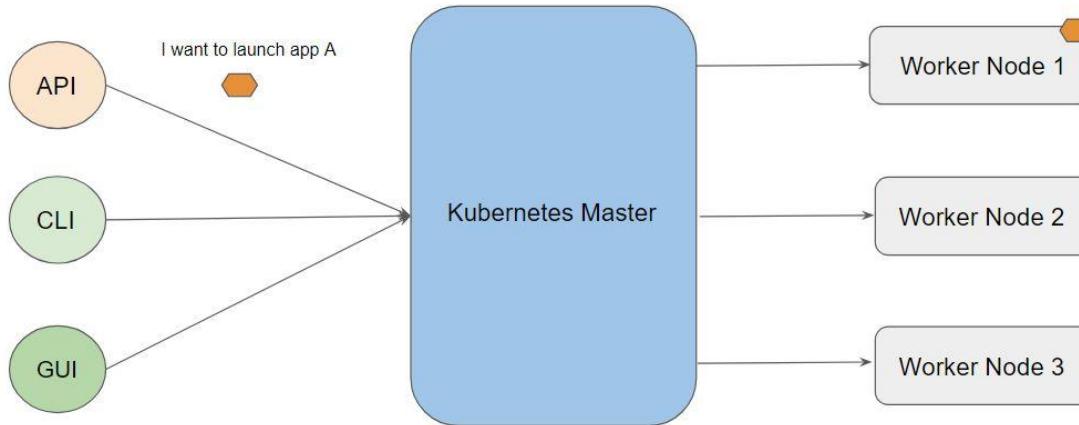


Kubernetes Objects

Let's get started

Overview of Kubernetes Objects

Kubernetes Objects is basically a record of intent that you pass on to the Kubernetes cluster. Once you create the object, the Kubernetes system will constantly work to ensure that object exists.



Approach to Configure an Object

There are various ways in which we can configure an Kubernetes Object.

- First approach is through the kubectl commands.
- Second approach is through configuration file written in YAML.

```
pod.yaml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: mywebserver
5  spec:
6    containers:
7      - name: mywebserver
8        image: nginx
```

Importance of YAML

YAML is a human-readable data-serialization language.

It designed to be human friendly and works perfectly with other programming languages.

XML	JSON	YAML
<pre><Servers> <Server> <name>Server1</name> <owner>John</owner> <created>123456</created> <status>active</status> </Server> </Servers></pre>	<pre>{ Servers: [{ name: Server1, owner: John, created: 123456, status: active }] }</pre>	<pre>Servers: - name: Server1 owner: John created: 123456 status: active</pre>

Benefits of Configuration File

Integrates well with change review processes.

Provides the source of record on what is live within the Kubernetes cluster.

Easier to troubleshoot changes with version control.

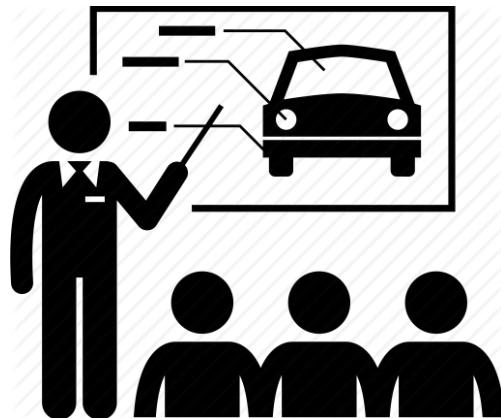
Kubernetes Architecture

Analogy is the Key

Understanding Analogy

Before you learn to drive, the instructor teaches you the basics about the car and it's components

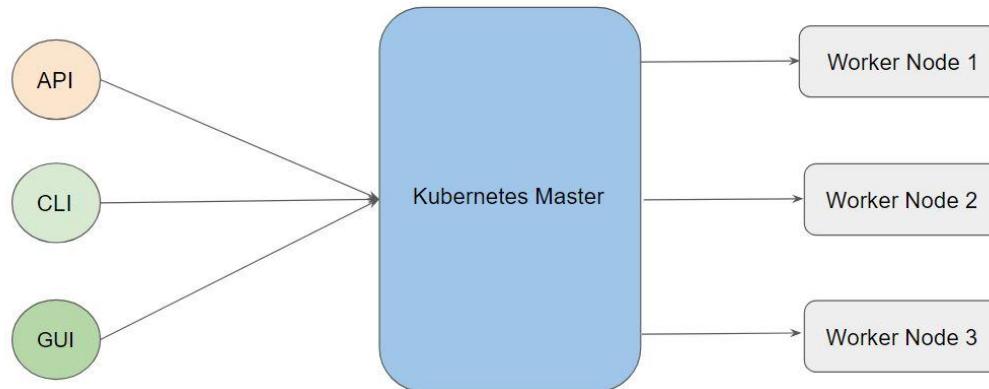
Accelerator, Brakes, Clutch, Steering, Indicators, Side and Rear View mirrors



Kubernetes - Before we Drive

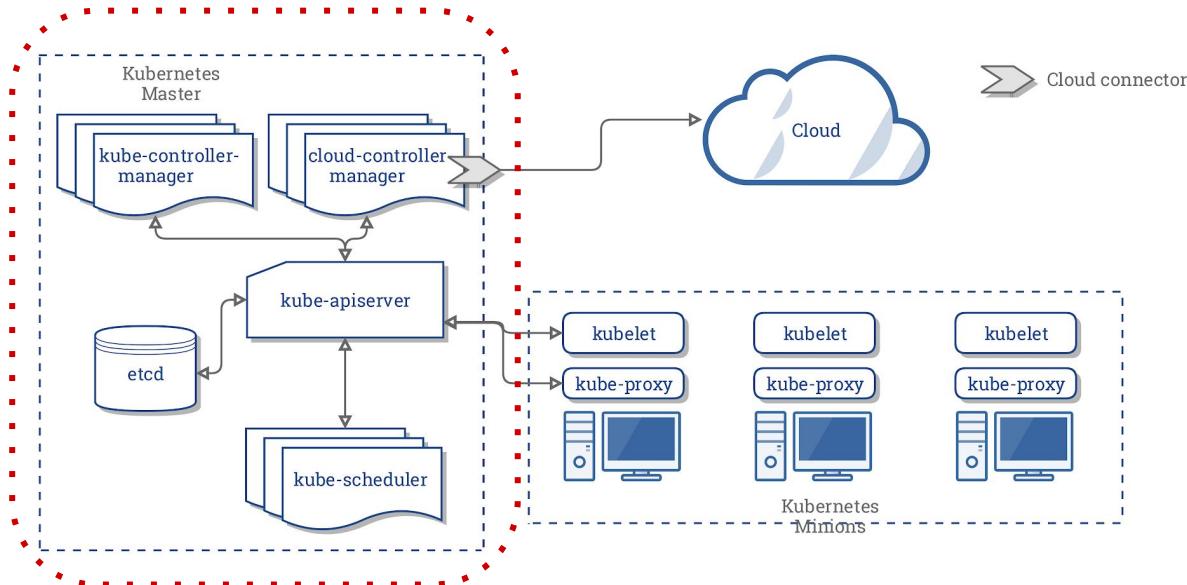
Before we start full-fledged deep dive into Kubernetes, understanding the architecture is beneficial.

ETCD, API Server, Scheduler, Controller Manager, Cloud Controller Manager.



Kubernetes Master Components

Kubernetes Master consists of five major components.

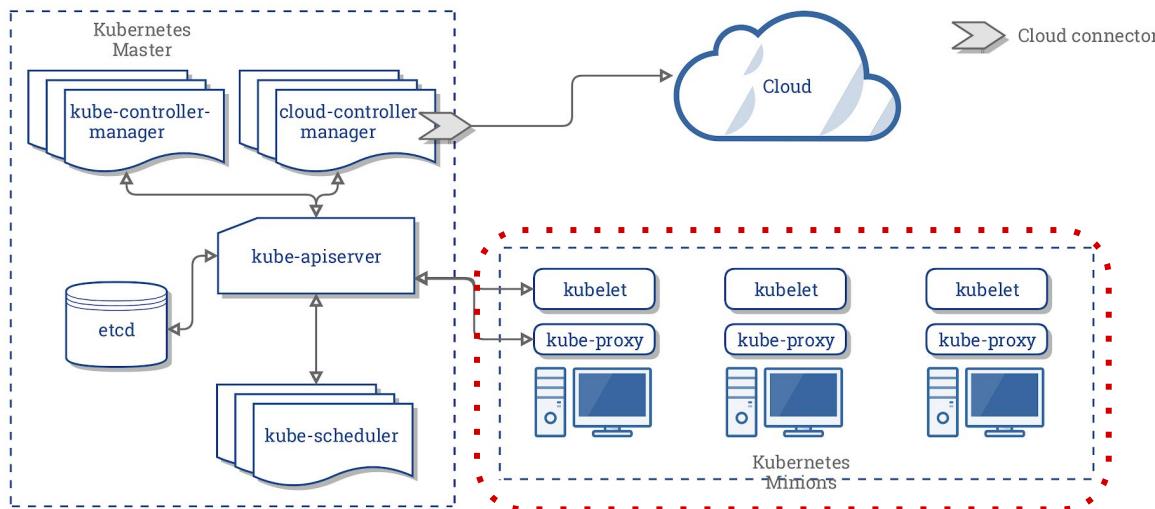


Kubernetes Master Components

Component Name	Description
kube-apiserver	Component on the master that exposes the Kubernetes API.
etcd	Key value store used as Kubernetes' backing store for all cluster data.
kube-scheduler	Component on the master that watches newly created pods that have no node assigned, and selects a node for them to run on.
kube-controller-manager	Responsible for controlling various aspects, including: Node Controllers: Responsible when node goes down. Replication controllers, endpoint controllers, Service account & Token controllers.
cloud-controller-manager	Runs controllers that interact with the underlying cloud providers.

Kubernetes Node Components

Kubernetes Worker Node consists of two major components:



Kubernetes Node Components

Node components runs on every worker node of the Kubernetes Cluster.

Component Name	Description
kubelet	An agent that runs on each node in the cluster. It makes sure that containers are running in a pod.
kube-proxy	Acts as a network proxy which maintains network rules on the host and performing connection forwarding.
Container Runtime	Software which is responsible for running containers. Supported Runtimes: Docker, containerd, rktlet and others.

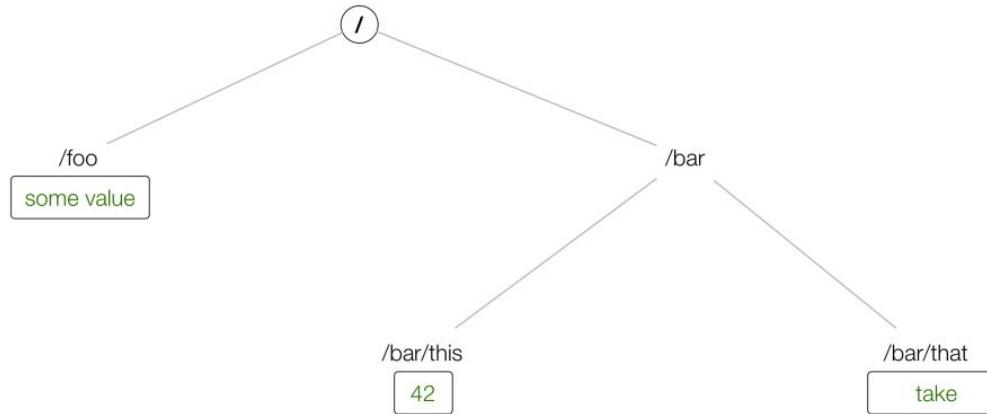
etcd

Distributed Key-Value Store

Central Configuration Storage

In a Linux environment, all the configurations are stored in the /etc directory.

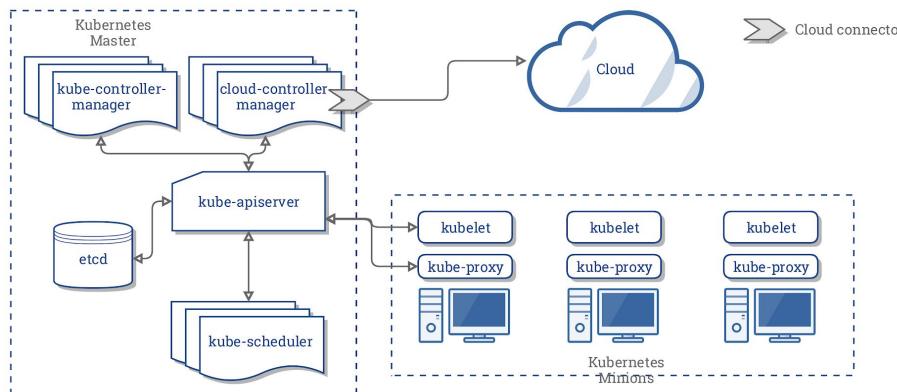
etcd is inspired from that and there is an addition of d which is for distributed.



Understanding the etcd

etcd is a distributed reliable key-value store.

etcd reliably stores the configuration data of the Kubernetes cluster, representing the state of the cluster (what nodes exist in the cluster, what pods should be running, which nodes they are running on, and a whole lot more) at any given point of time.



Important Pointers

A Kubernetes cluster stores all its data in etcd.

Anything that you read while running kubectl get pods is stored in etcd

Any node crashing or process dying causes values in etcd to be changed.

Whenever you create something with kubectl create / kubectl run will create an entry in the etcd.

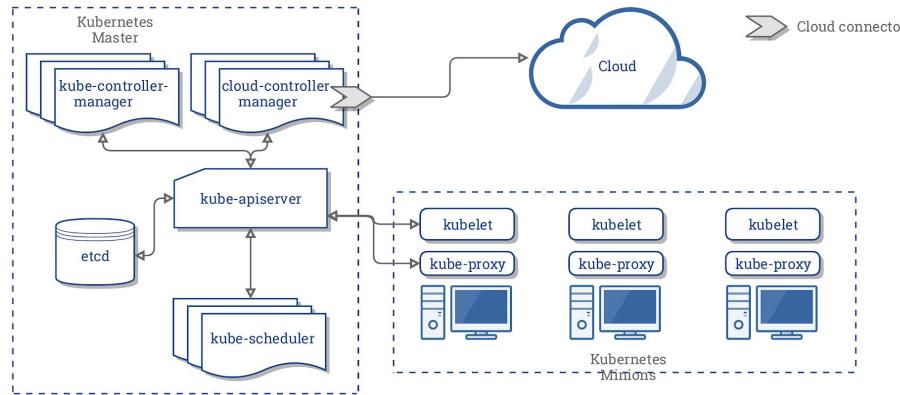
kube-apiserver

The Gateway to Kubernetes Master

Understanding the API Server

API server acts as a gateway to the Kubernetes Cluster.

When you interact with your Kubernetes cluster using the kubectl command-line interface, you are actually communicating with the master API Server component.

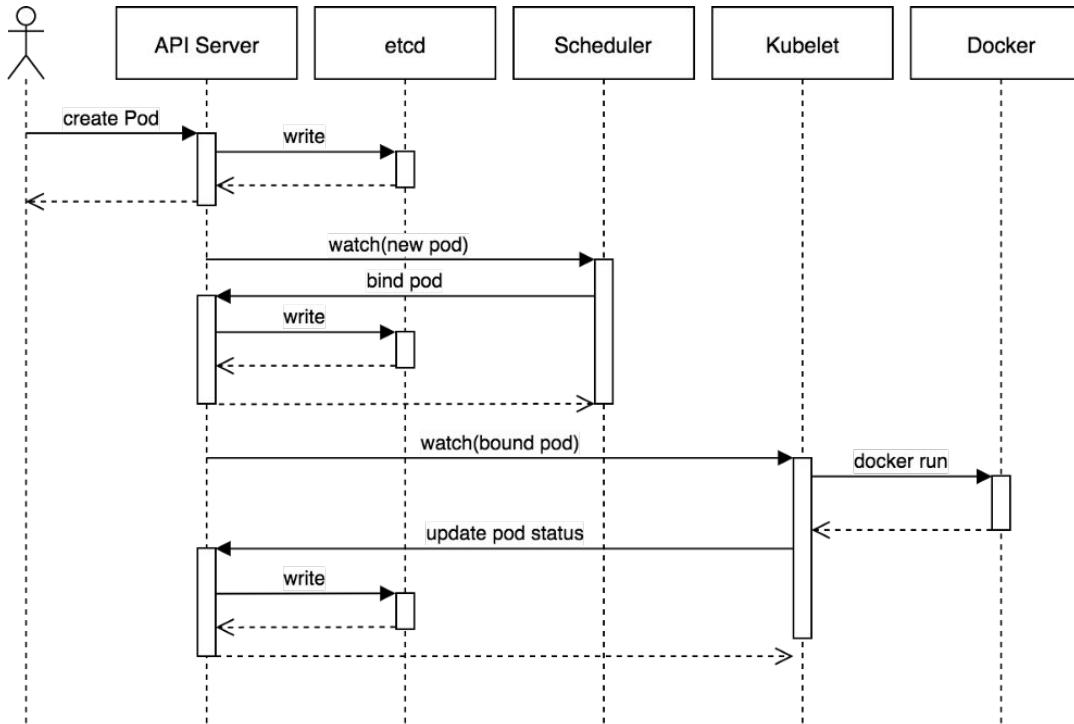


Understanding the API Server

The API Server is the only Kubernetes component that connects to etcd; all the other components must go through the API Server to work with the cluster state.

The API Server is also responsible for the authentication and authorization mechanism. All API clients should be authenticated in order to interact with the API Server.

The flow diagram



Flow Diagram

1. Kubectl writes to the API server (`kubectl run mywebserver --image=nginx`)
2. API server will authenticate and authorize. Upon validation, it will write it to etcd.
3. Upon write to etcd, API Server will invoke the scheduler.
4. Scheduler decides which node the pod should run and return data to API server. API will in-turn write it back to etcd.

Flow Diagram

5. API Server will invoke the kubelet in the node decided by the scheduler.
6. Kubelet communicates to the docker daemon via Docker socket to create the container.
7. Kubelet will update the status of the POD back to the API server.
8. API Server will write the status details back to etcd.

Basics of API



Understanding the Challenge

Book Distributor maintains the list of available books in it's backend systems.

Operator has access to Backend system to check the availability.

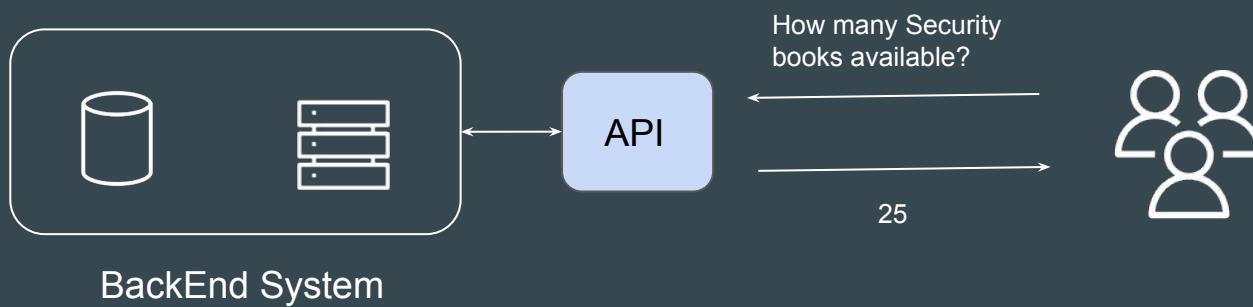
Clients they connect to Operator via Phone call / Chat option



API Based Approach

The book distributor could provide an API to check stock availability.

APIs let you open up access to your resources while maintaining security and control.



Simple Use-Case

James wants to build a weather report application.

OpenWeatherMap is an online service that provides global weather data via API.

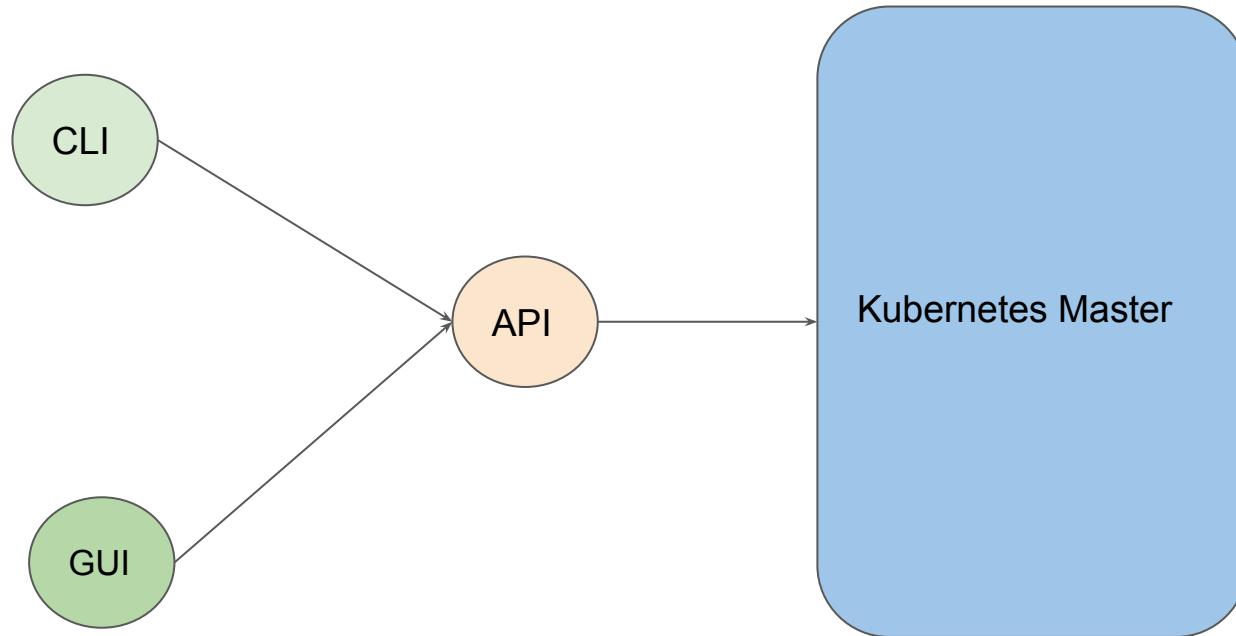
He decided to connect his application to OpenWeatherMap API to fetch the latest reports and populate it in application.



Kubernetes API Primitives

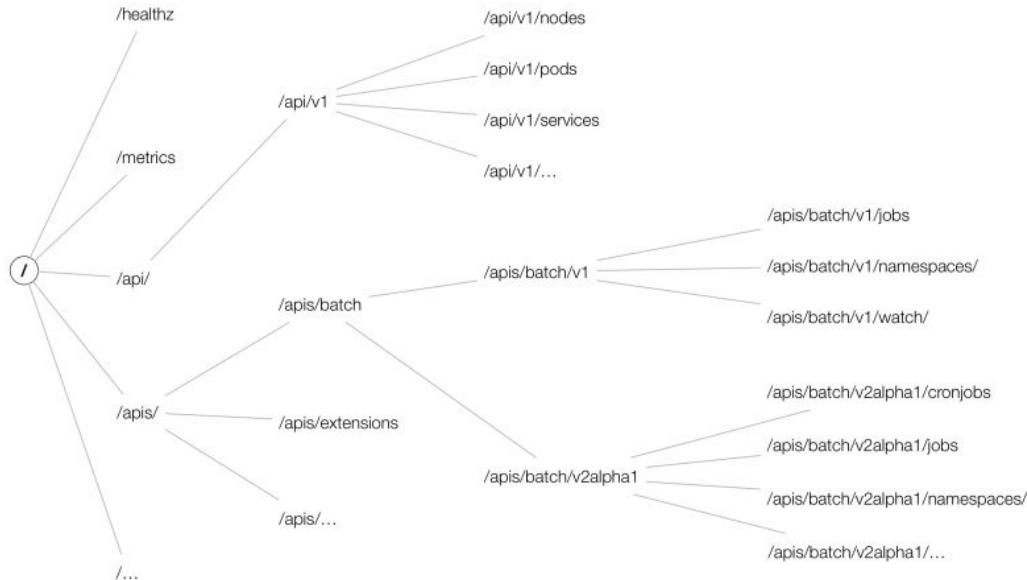
Analogy is the Key

Architecture of Kubernetes



Overview of API's Available

Depending on the operation you intend to do, there are various API's which are available.



Example of POD Object

```
pod.yaml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: mywebserver
5  spec:
6    containers:
7      - name: mywebserver
8        image: nginx
```

Levels	Description
Alpha Level	<ul style="list-style-type: none">• Version name contains alpha.• Example: v1alpha• May be buggy.• Enabling the feature may expose bugs.• Disabled by default.
Beta Level	<ul style="list-style-type: none">• The version names contain beta (e.g. v2beta3).• Code is well tested. Enabling the feature is considered safe. Enabled by default. <p data-bbox="514 616 1743 697">Recommended for only non-business-critical uses because of potential for incompatible changes in subsequent releases.</p>
Stable Level	<ul style="list-style-type: none">• The version name is vX where X is an integer. Example: v1 Recommended to use in production environments.

POD Configuration in YAML

Let's get started

Approach to Configure an Object

There are various ways in which we can configure an Kubernetes Object.

- First approach is through the kubectl commands.
- Second approach is through configuration file written in YAML.

```
pod.yaml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: mywebserver
5  spec:
6    containers:
7      - name: mywebserver
8        image: nginx
```

Understanding Important fields

Key	Description
apiVersion	Version of API
kind	Kind of object you want to create.
metadata name	Name of object that uniquely identifies it.
spec	Desired state of the object.



```
pod.yaml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: mywebserver
5  spec:
6    containers:
7      - name: mywebserver
8        image: nginx
```

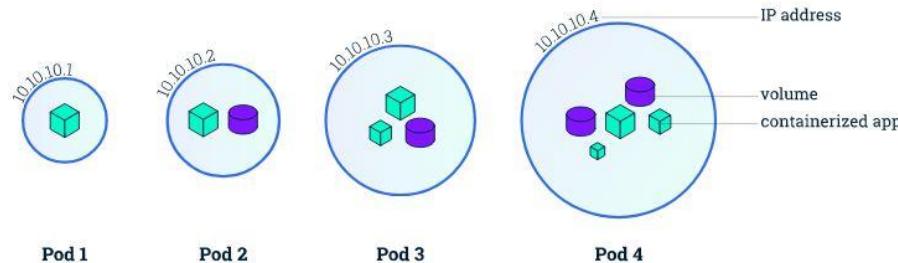
Multi-Container Pods

Let's get started

Kubernetes POD

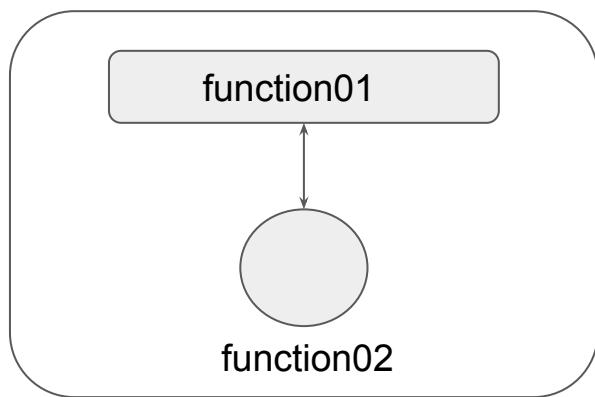
A Pod in Kubernetes represents a group of one or more application containers , and some shared resources for those containers.

A single pod can have multiple containers running.

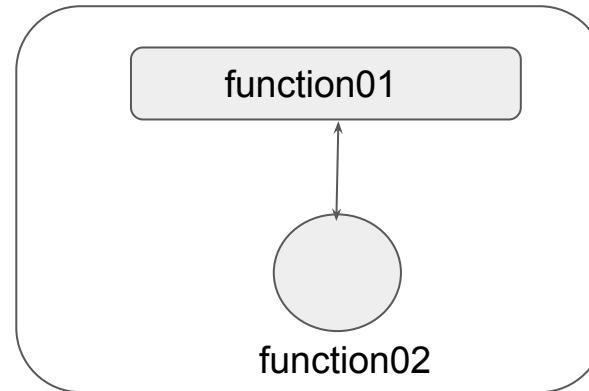


Dealing with Tightly Coupled Application

Containers within a Pod share an IP address and port space, and can find each other via localhost.



POD 1



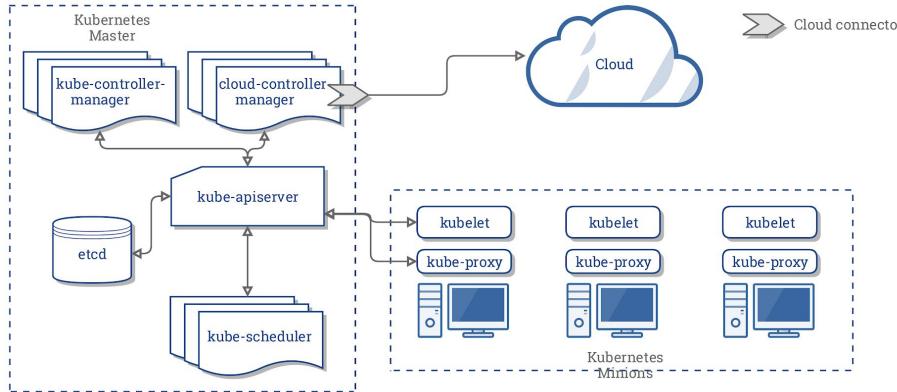
POD 2

kube-scheduler

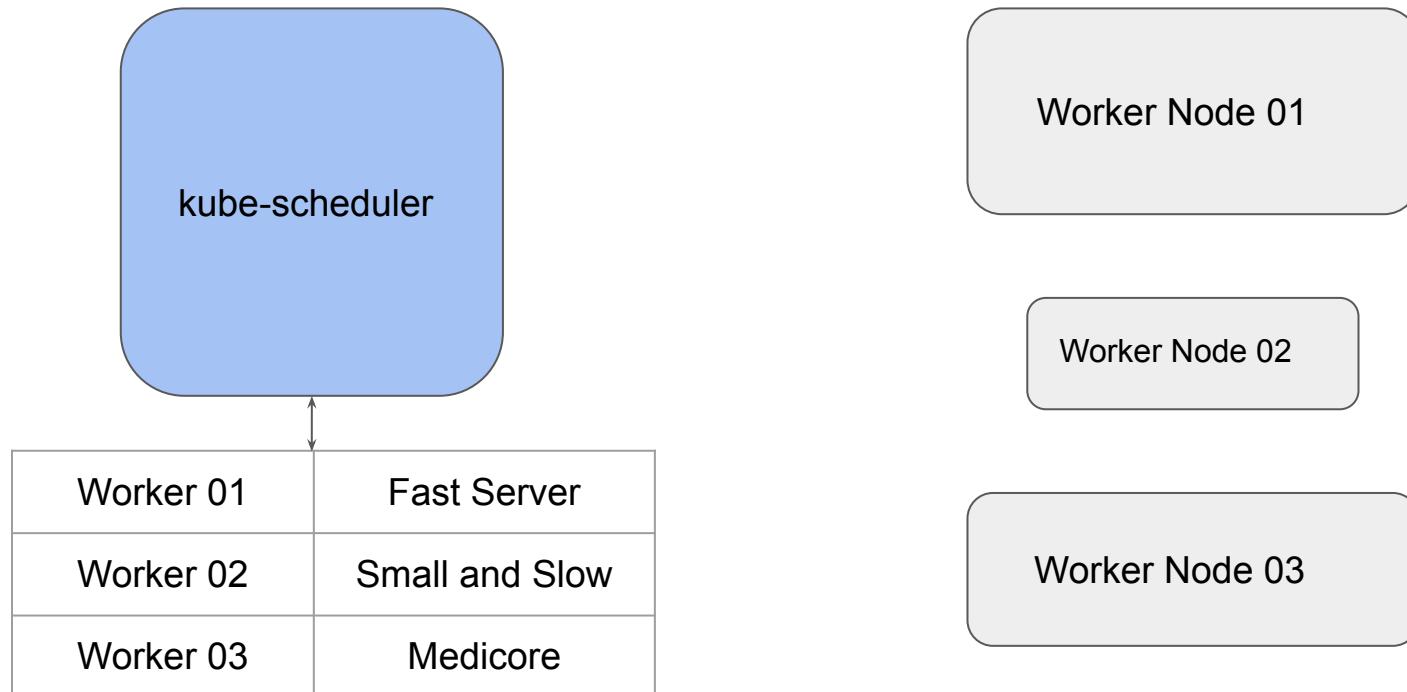
Finding right home for a Pod

Understanding the Kubernetes Scheduler

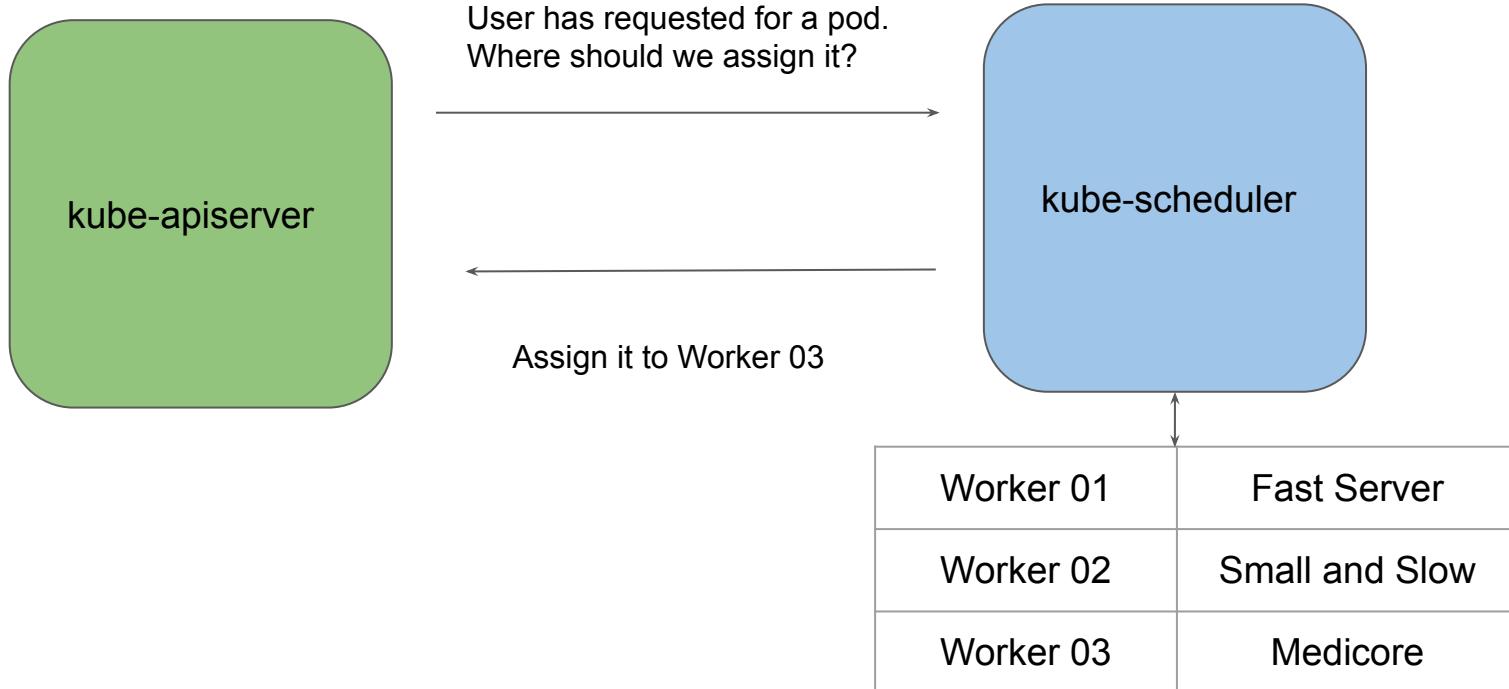
Kube-scheduler watches for newly created pods that have no node assigned, and selects a node for them to run on.

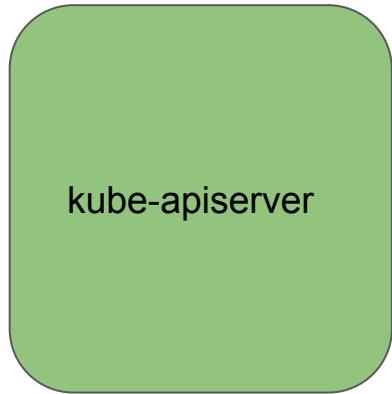


Understanding the kube-scheduler

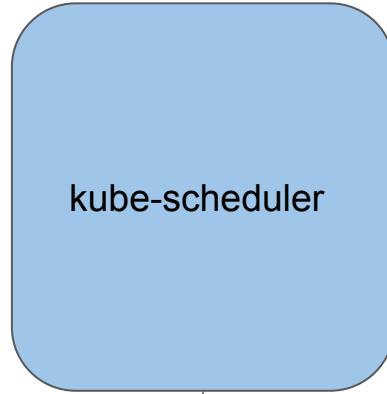


Understanding the kube-scheduler





User has requested for a pod and specified that it needs to run on fast server. Where should we assign it?

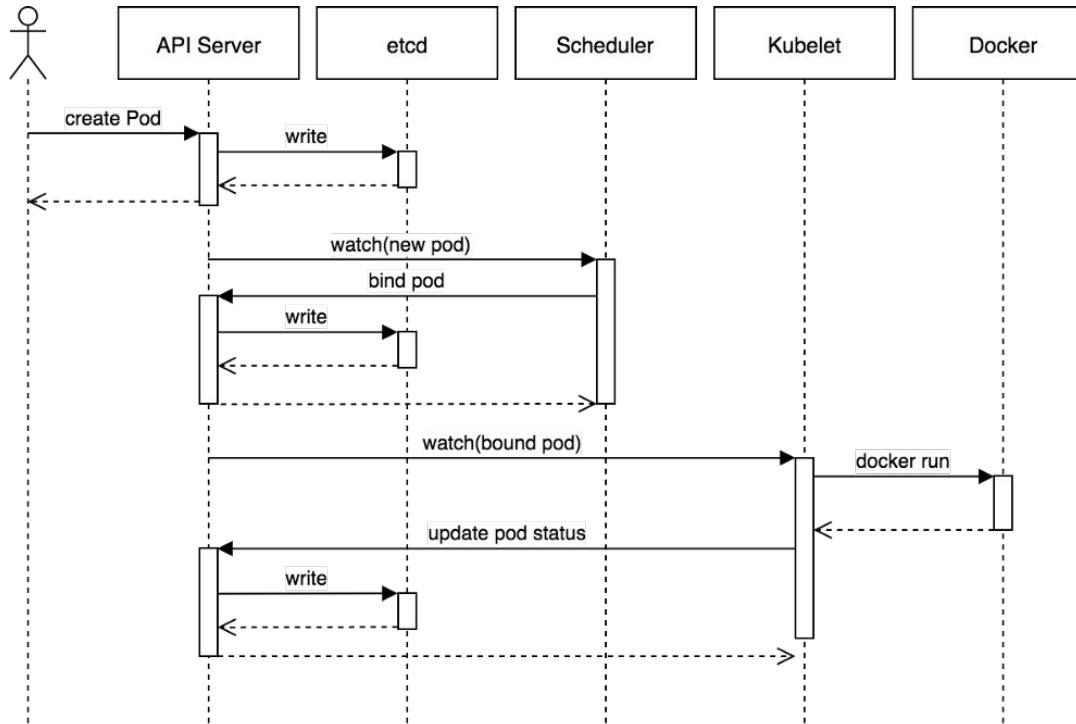


Alright, assign it to Worker 01



Worker 01	Fast Server
Worker 02	Small and Slow
Worker 03	Medicore

The flow diagram



Factors For Scheduling

There are several factors which are taken into consideration before a pod is scheduled to a node.

Some of these includes:

- Resource Requirements
- Hardware/Software policy constraints
- Affinity & Anti-Affinity
- Data Locality

Dockerfile - ENTRYPOINT

Build once, use anywhere

Overview of ENTRYPOINT

The best use for ENTRYPOINT is to set the image's main command

ENTRYPOINT doesn't allow you to override the command.

It is important to understand distinction between CMD and ENTRYPOINT.

Commands and Arguments

Kubernetes Level

Overview of Command and Arguments

During the video of ENTRYPPOINT, we discussed the difference between CMD and ENTRYPPOINT instruction in Dockerfile.

We can also refer them as Image Command and Image Entrypoint.

In Kubernetes, we can override the default entrypoint and CMD with command and args field.

DockerFile vs K8s Manifest Perspective

Docker Field Name	Kubernetes Field name	Description
ENTRYPOINT	command	Command that will be run by container.
CMD	args	Argument passed to the container.

Use-Case 1: Image Entrypoint and CMD

When there is an entrypoint and CMD set for a Docker image and if we are not manually overriding it at k8s manifest level, then final decision is to use the default image specifications.

Image Entrypoint	Image Command	Container Command	Container Argument	Final
sleep	3600	<not set>	<not set>	sleep 3600

Use-Case 2: Setting Container Command

When there is an entrypoint and CMD set for a Docker image and if we are not manually overriding it at k8s manifest level, then final decision is to use the default image specifications.

Image Entrypoint	Image Command	Container Command	Container Argument	Final
sleep	3600	ping -c5 google.com	<not set>	ping -c5 google.com

Use-Case 3: Setting Container Arguments

When container argument is set in k8s manifest, the image command gets overridden.

Image Entrypoint	Image Command	Container Command	Container Argument	Final
sleep	3600	<not set>	5000	sleep 5000

Use-Case 4: Setting Container Command & Arguments

When container command and arguments are specified in k8s manifest, they will override the image command and entrypoint.

Image Entrypoint	Image Command	Container Command	Container Argument	Final
sleep	3600	ping	yahoo.com	ping yahoo.com

CLI Documentation of Resources

Kubernetes Level

Documentation of K8s Resources

Till now we have been going through documentation from browser to understand about fields.

There is better way through which we can achieve similar functionality via CLI

This is through `kubectl explain` command.

EXPOSE

Build once, use anywhere

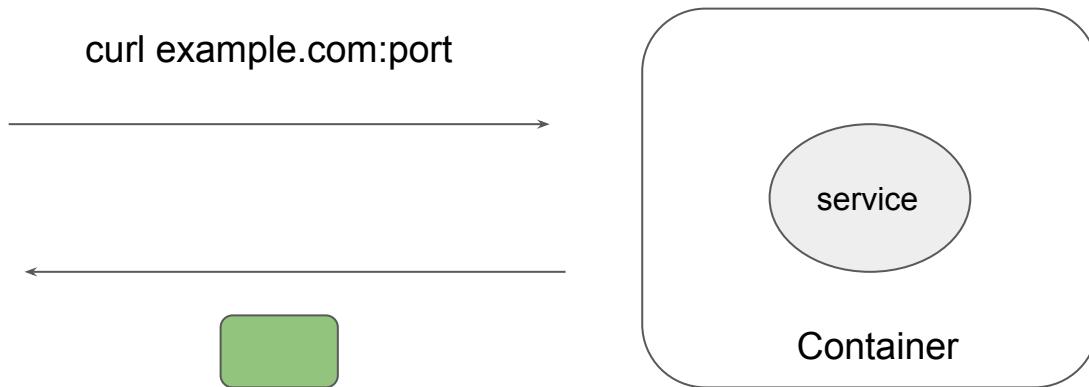
Overview of EXPOSE instruction

The EXPOSE instruction informs Docker that the container listens on the specified network ports at runtime.

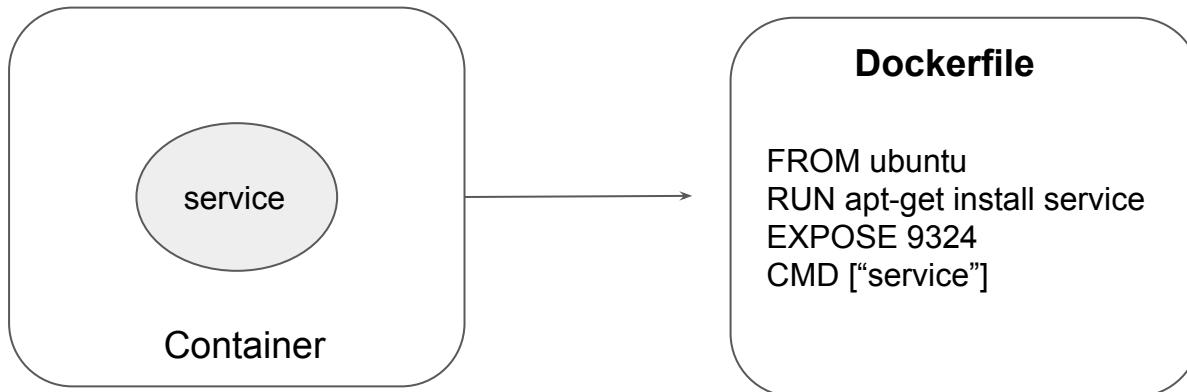
The EXPOSE instruction does not actually publish the port.

It functions as a type of documentation between the person who builds the image and the person who runs the container, about which ports are intended to be published.

Understanding the Use-Case



Understanding the Use-Case



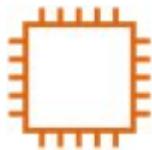
Labels and Selectors

Let's get started



Overview of Labels

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



Server



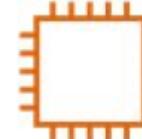
Database



Load Balancer



Load Balancer

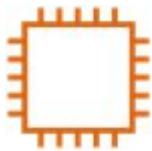


Server



Database

Adding Labels to Resource



name: kplabs-gateway
env: production



name: kplabs-db
env: production



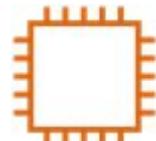
name: kplabs-elb
env: production



name: kp-elb
env: dev



name: kp-db
env: dev



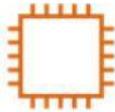
name: kp-gateway
env: dev

Overview of Selectors

Selectors allows us to filter objects based on labels.

Example:

1. Show me all the objects which has label where **env: prod**



name: kplabs-gateway
env: production



name: kplabs-db
env: production



name: kplabs-elb
env: production

Overview of Selectors

Selectors allows us to filter objects based on labels.

Example:

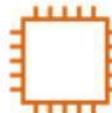
2. Show me all the objects which has label where **env: dev**



name: kp-elb
env: dev



name: kp-db
env: dev



name: kp-gateway
env: dev

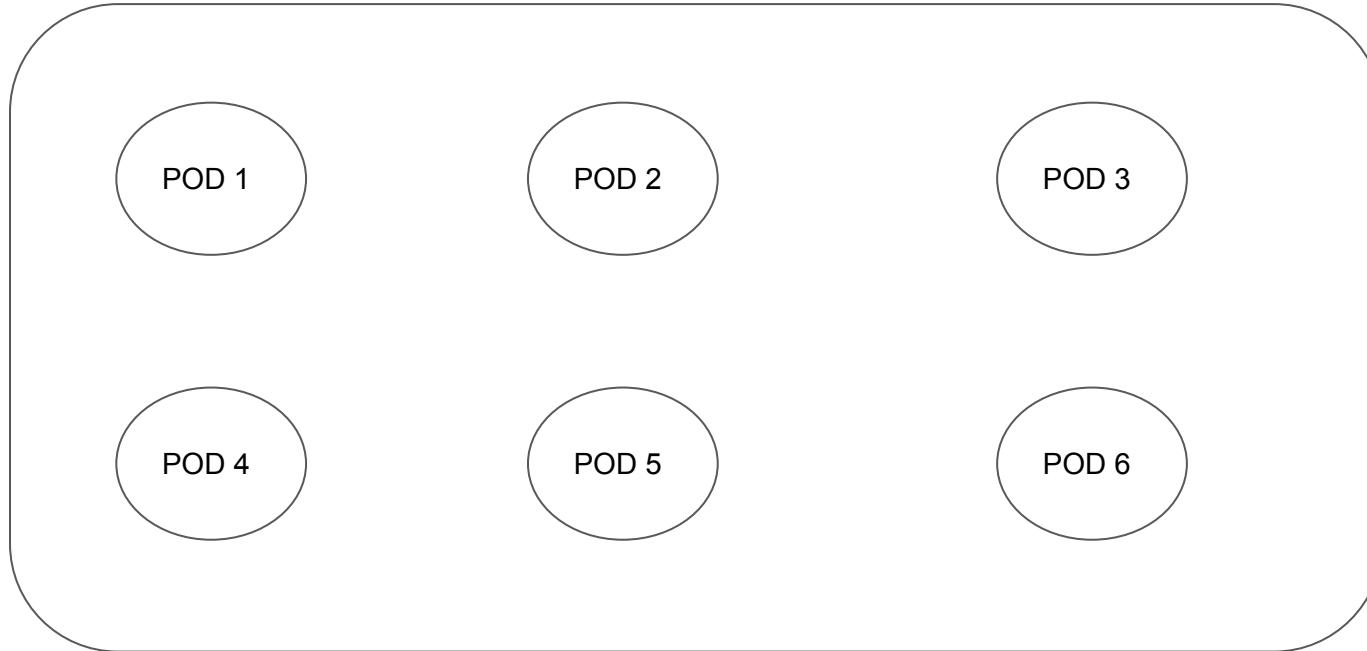
Kubernetes Perspective

There can be multiple objects within a Kubernetes cluster.

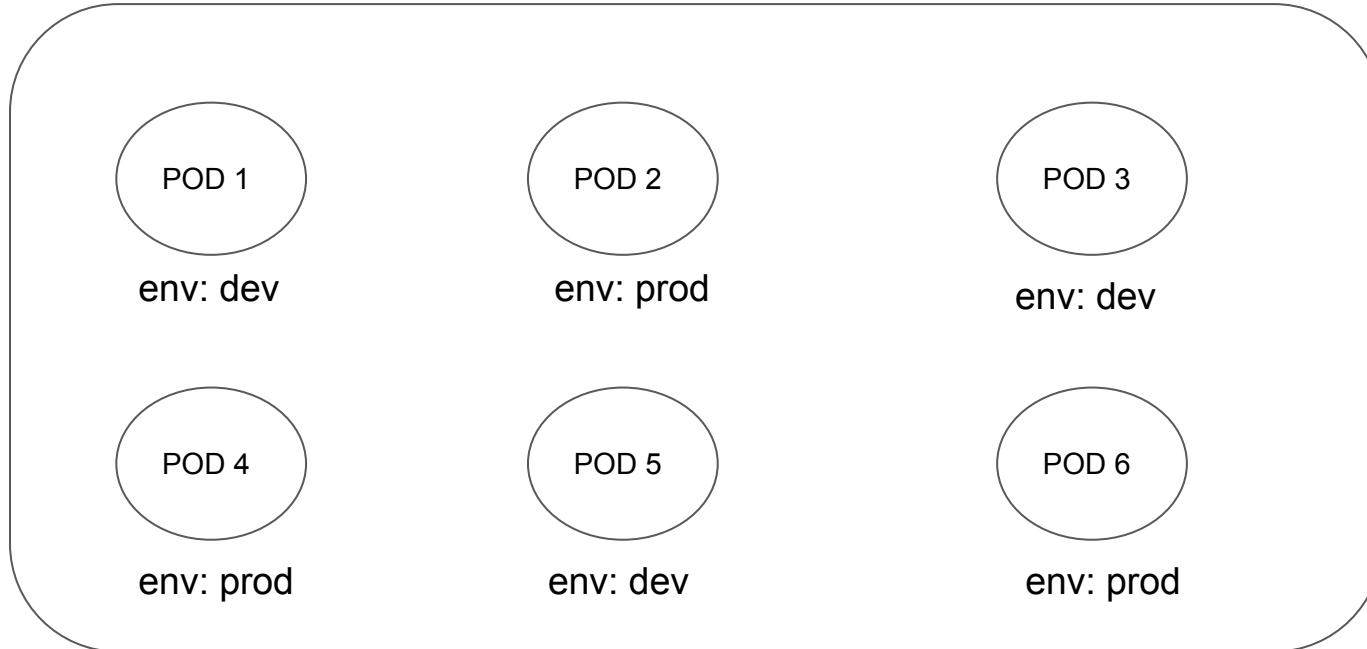
Some of the objects are as follows:

1. Pods
2. Services
3. Secrets
4. Namespaces
5. Deployments
6. Daemonsets

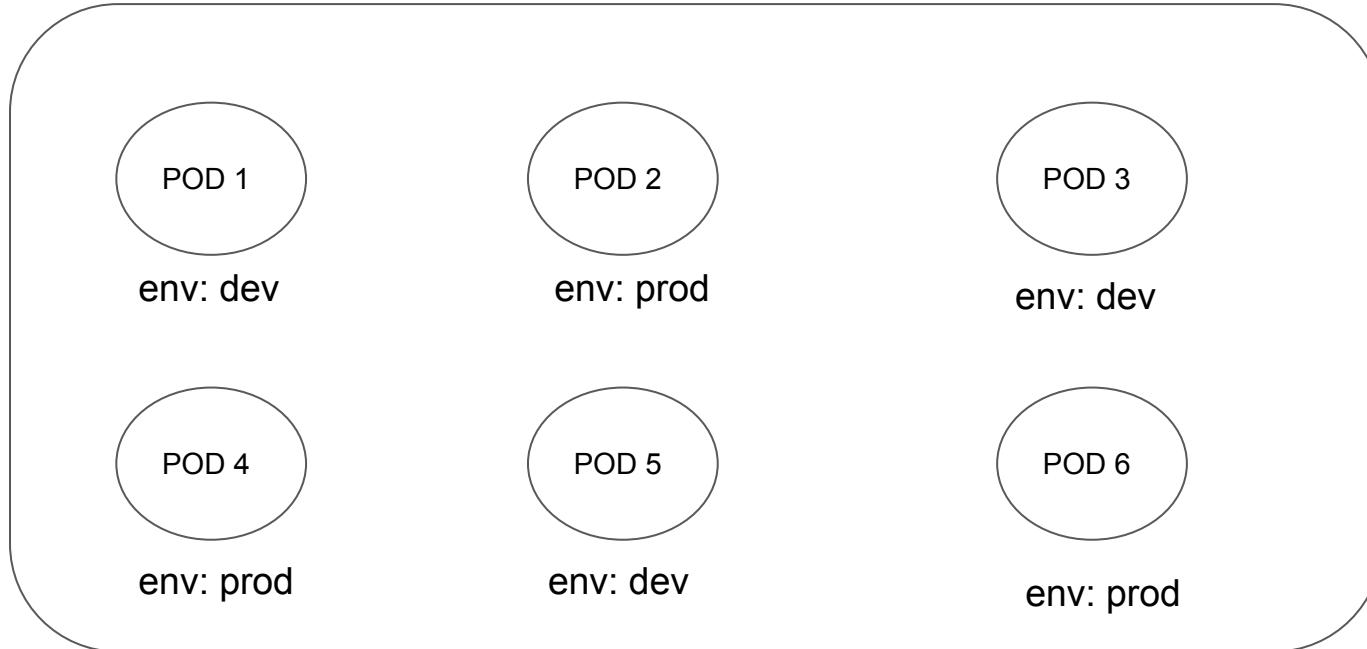
Kubernetes Perspective



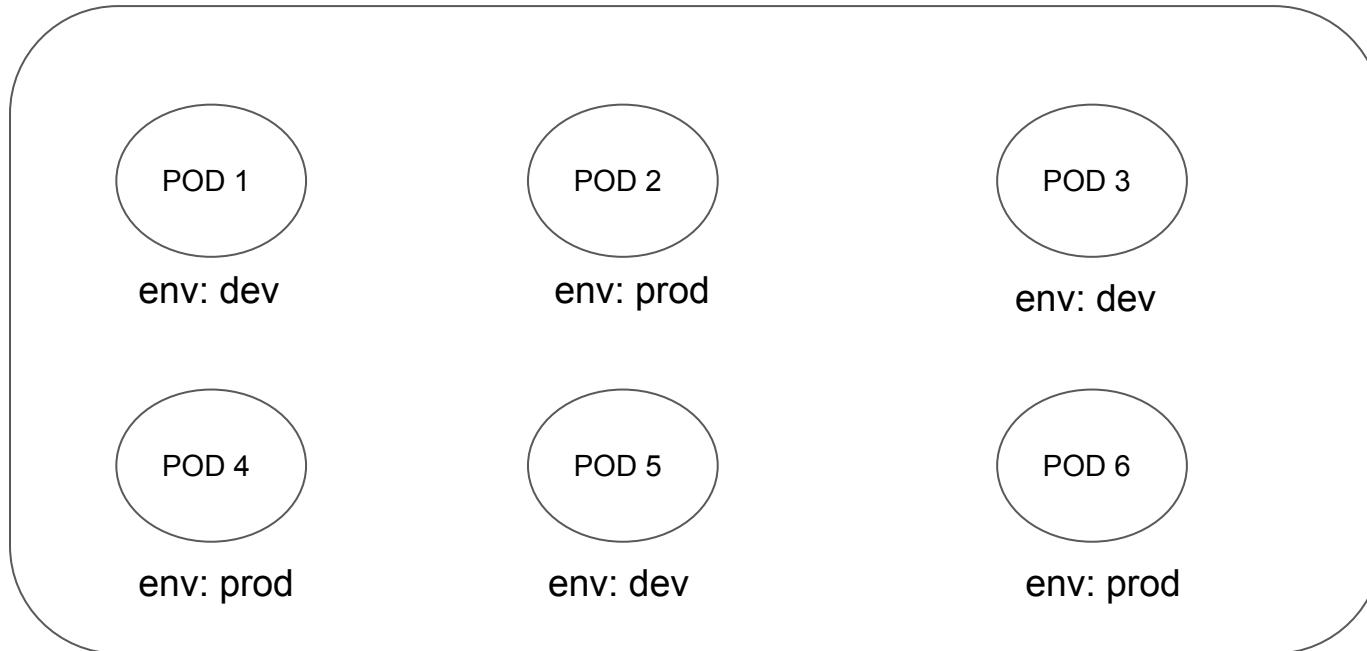
Assigning Labels to Kubernetes Objects



Selector: List All Pods from Dev Environment



Selector: List All Pods from Production Environment

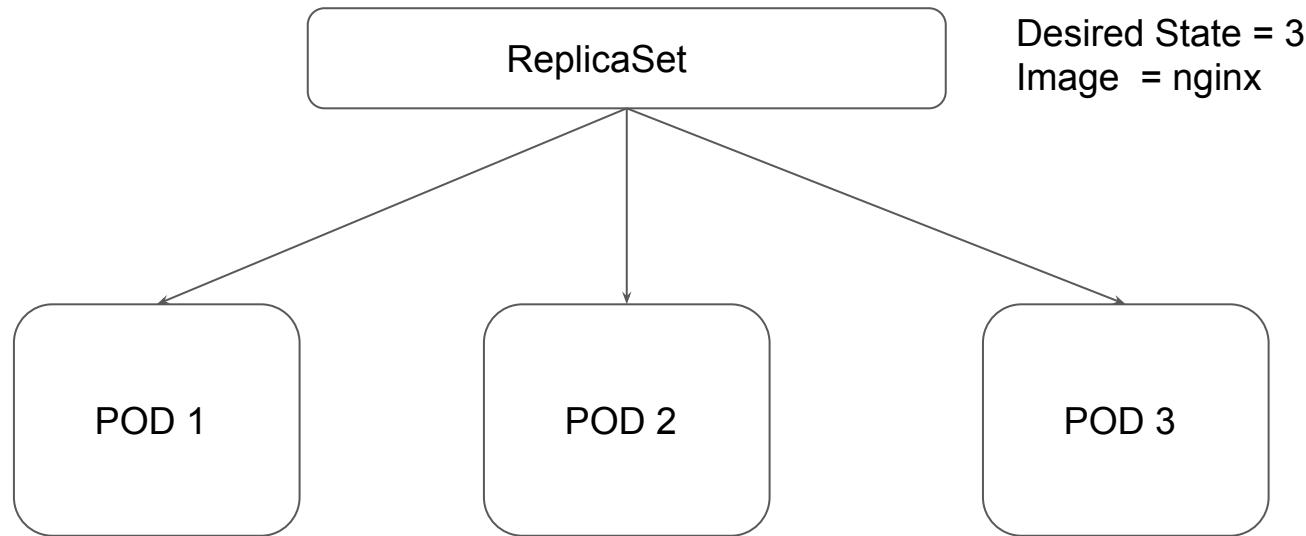


ReplicaSets

Let's get started

Overview of Kubernetes ReplicaSets

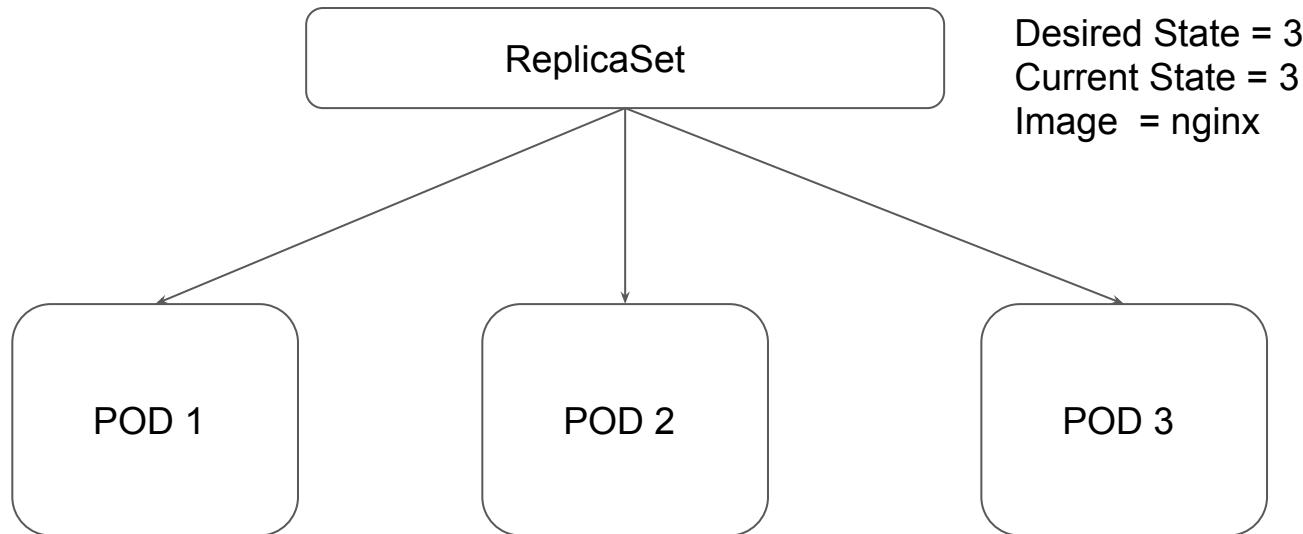
A ReplicaSet purpose is to maintain a stable set of replica Pods running at any given time.



Desired State vs Current State

Desired State - The state of pods which is desired.

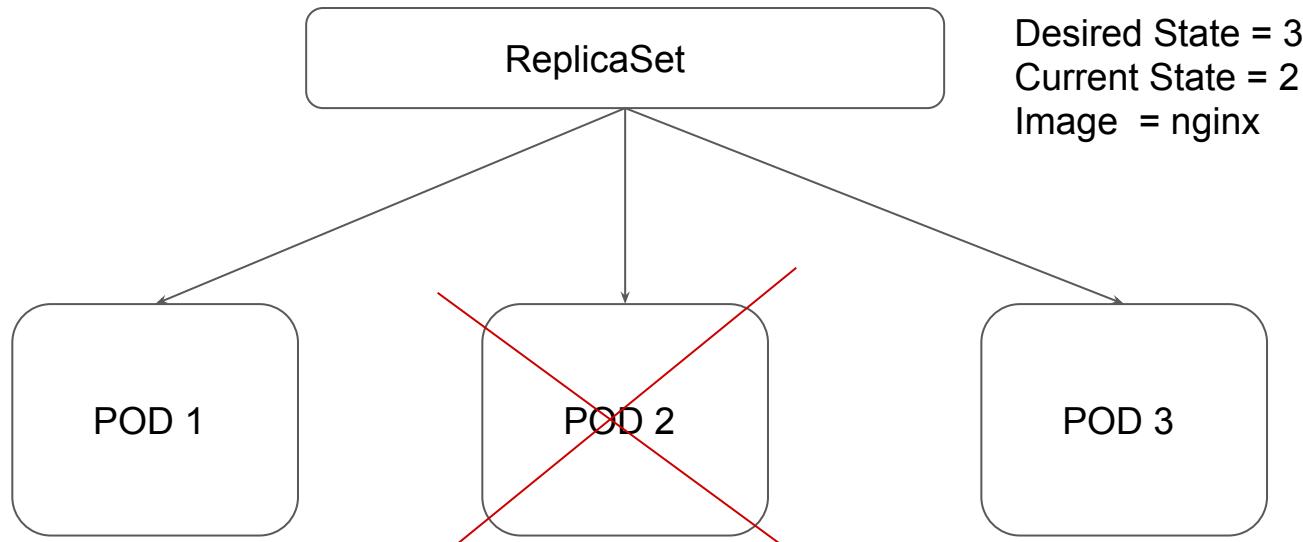
Current State - The actual state of pods which are running.



Desired State vs Current State

Desired State - The state of pods which is desired.

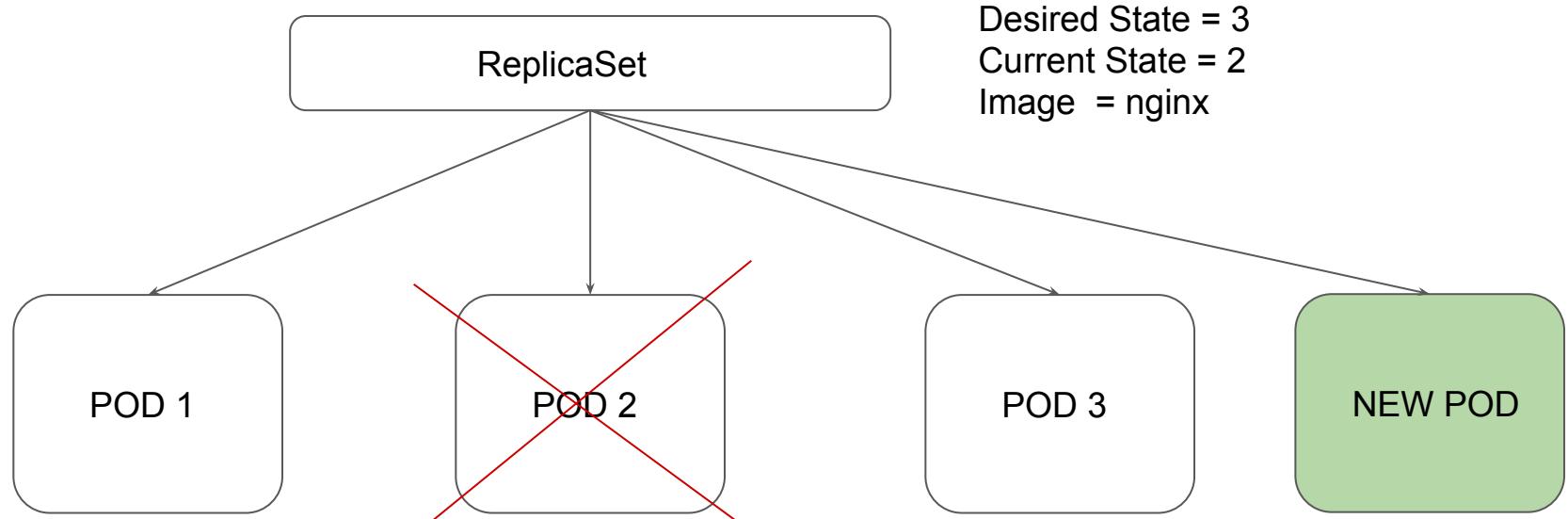
Current State - The actual state of pods which are running.



Desired State vs Current State

Desired State - The state of pods which is desired.

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

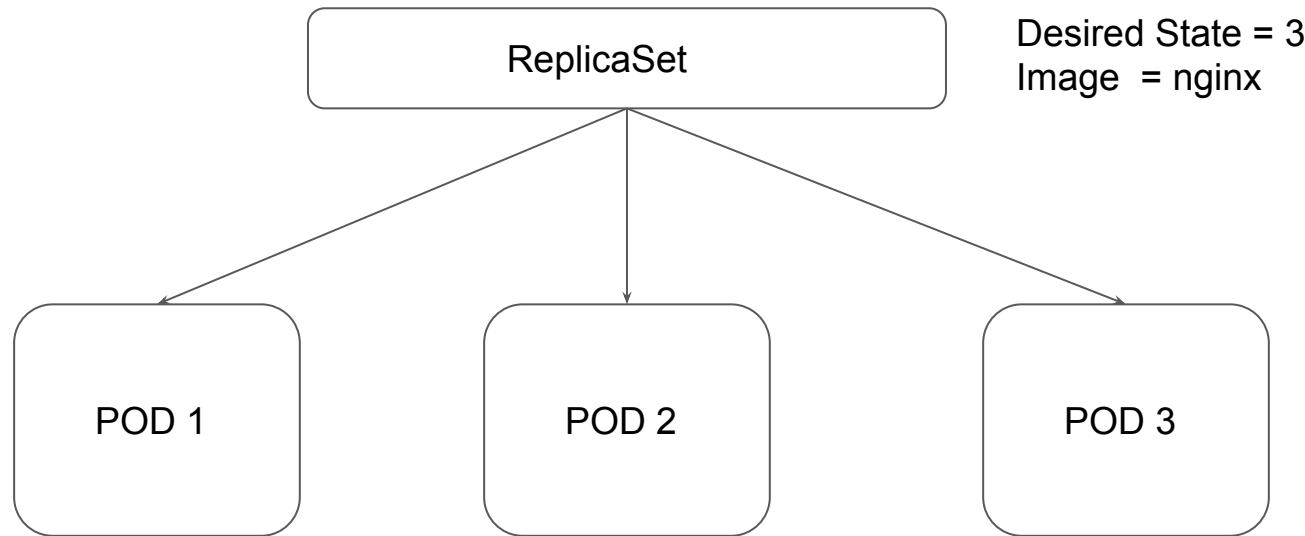


Deployments

Let's get started

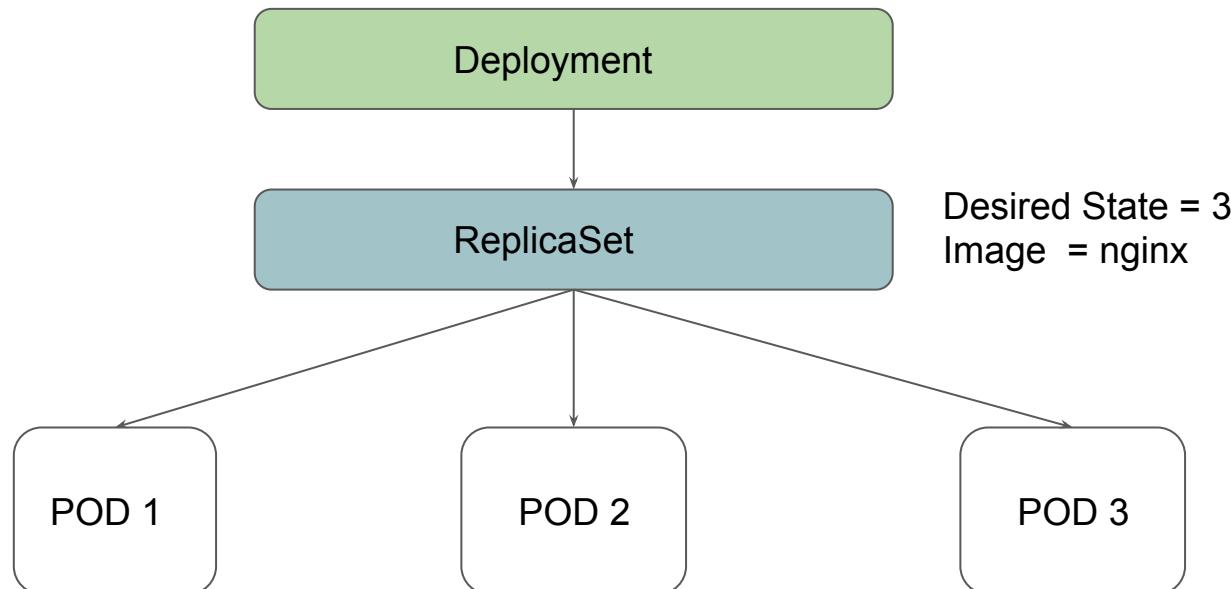
Challenges with ReplicaSets

ReplicaSets works well in basic functionality like managing pods, scaling pods and similar.



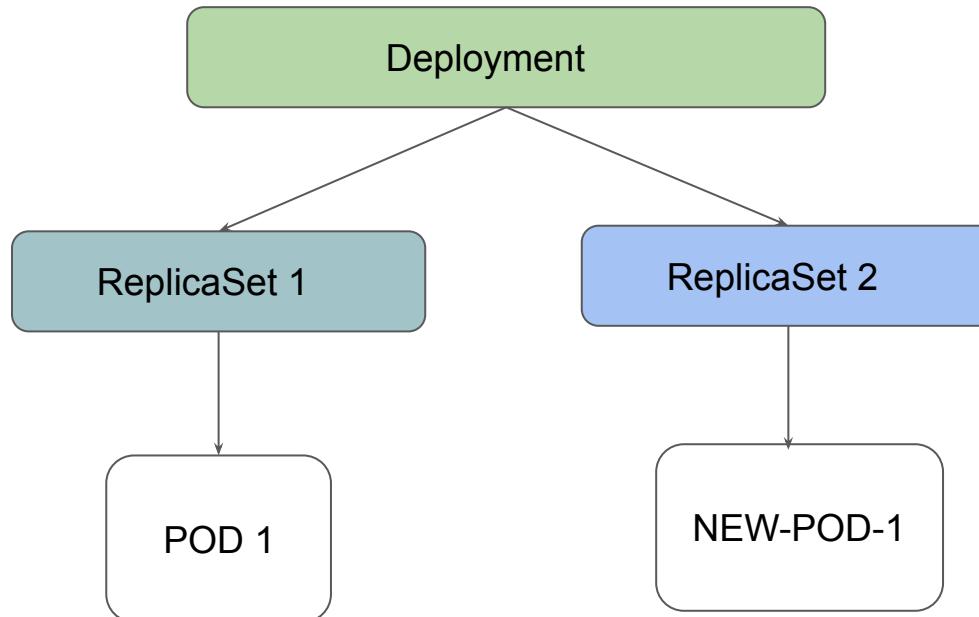
Understanding Deployments

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



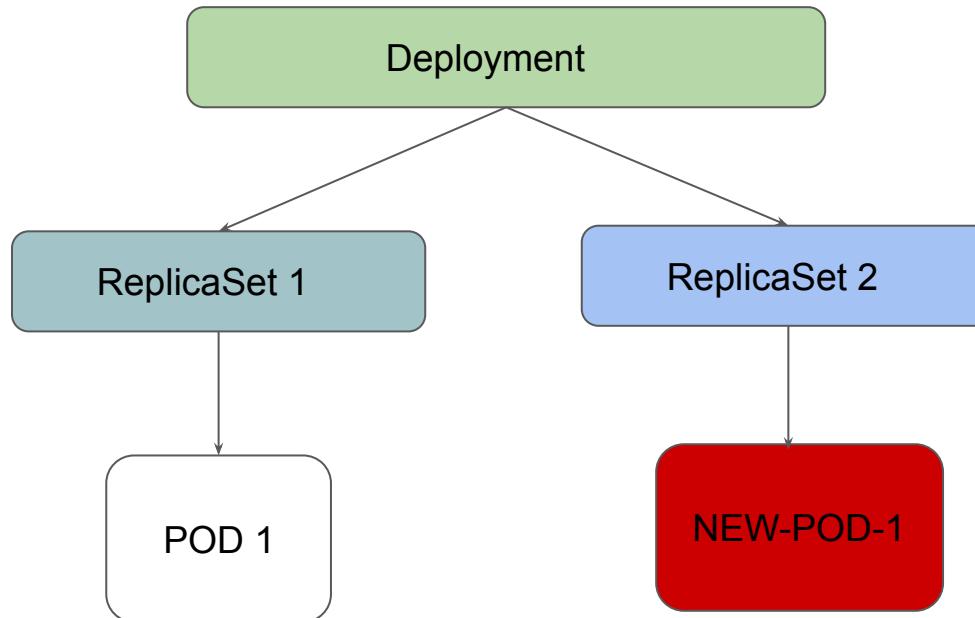
Benefits of Deployments - Rollout Changes

- We can easily rollout new updates to our application using deployments.
- Deployments will perform update in rollout manner to ensure that your app is not down.



Benefits of Deployments - RollBack Changes

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

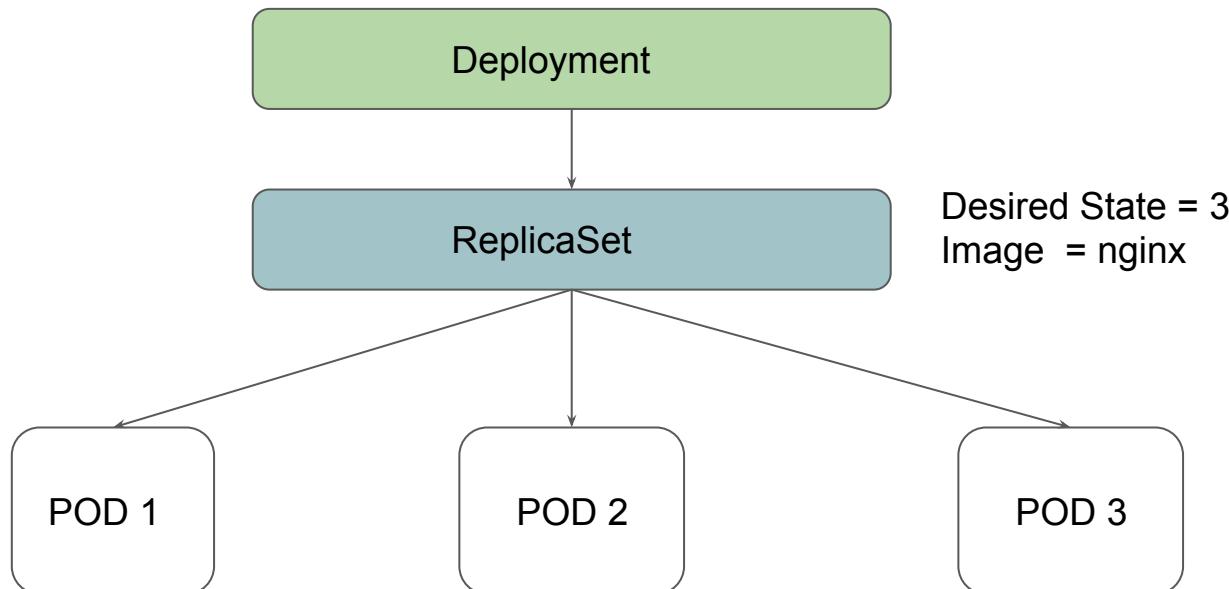


Creating First Deployment

Let's get started

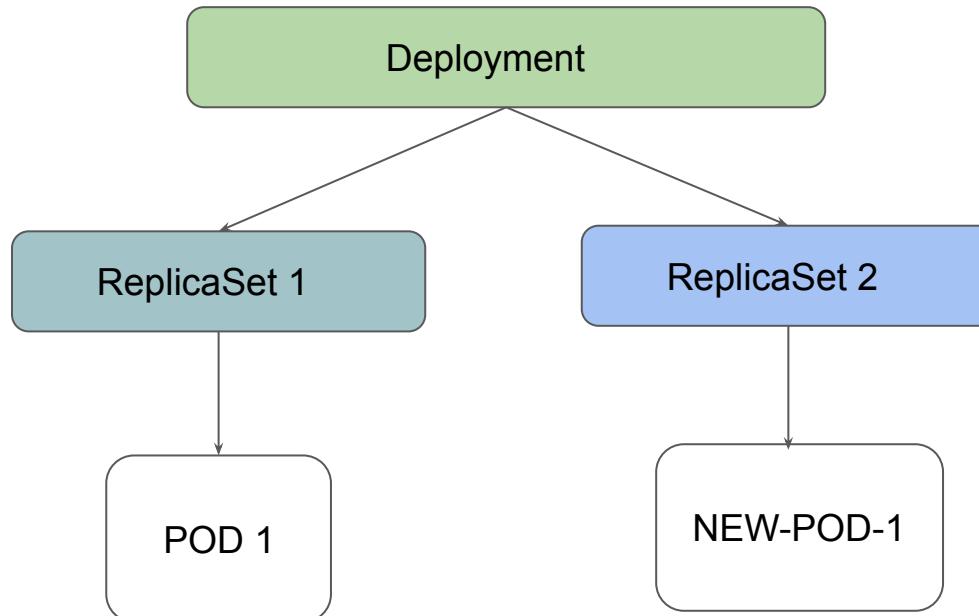
Understanding Deployments

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



Benefits of Deployments - Rollout Changes

- We can easily rollout new updates to our application using deployments.
- Deployments will perform update in rollout manner to ensure that your app is not down.



Important Pointers

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).

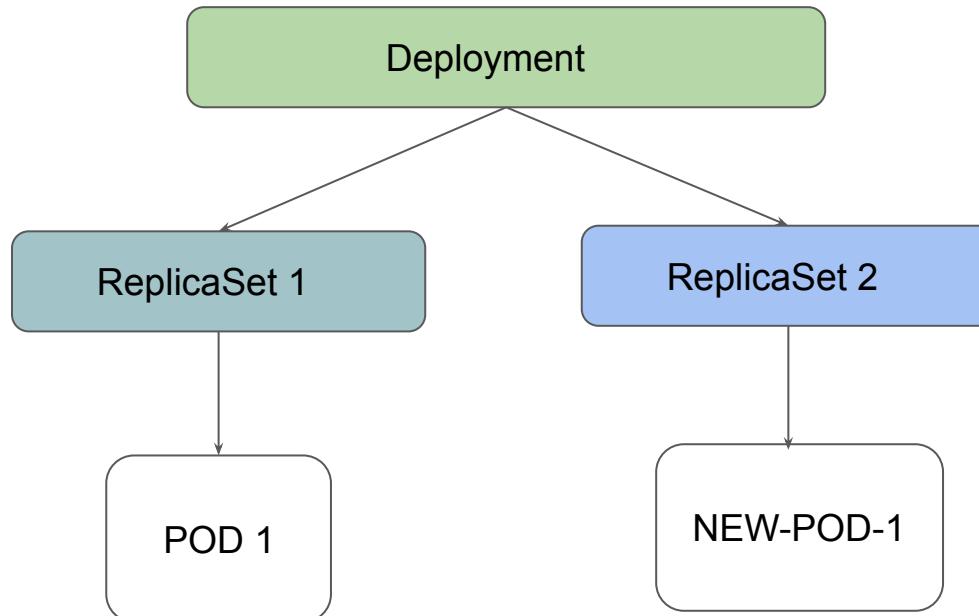
Deployments keep the history of revision which had been made.

Rolling Back Deployments

Let's get started

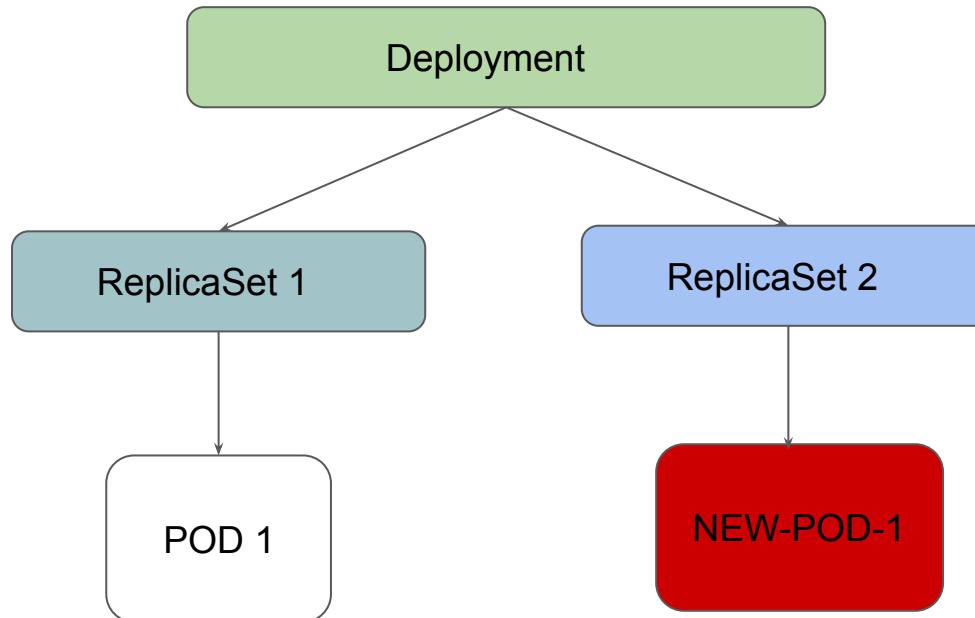
Benefits of Deployments - Rollout Changes

- We can easily rollout new updates to our application using deployments.
- Deployments will perform update in rollout manner to ensure that your app is not down.



Benefits of Deployments - RollBack Changes

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



Creating Deployments via CLI

Let's get started

Challenge with Deployment Manifest

To create a simple deployment, we have to search for the entire deployment manifest file from the documentation and edit it based on our requirements.



```
label-pod.yaml •  
1 apiVersion: apps/v1  
2 kind: Deployment  
3 metadata:  
4   name: kplabs-deployment  
5 spec:  
6   replicas: 5  
7   selector:  
8     matchLabels:  
9       tier: frontend  
10  template:  
11    metadata:  
12      labels:  
13        tier: frontend  
14    spec:  
15      containers:  
16        - name: php-redis  
17          image: nginx
```

Creating Deployment from CLI

We can run the [kubectl create deployment](#) set of commands to create a deployment from CLI.

Command	Description
<code>kubectl create deployment my-dep --image=nginx</code>	Creates new deployment.
<code>kubectl create deployment my-dep --image=nginx --replicas 3</code>	Deployment with 3 replicas

Important Pointer - Deployments

Deployment Related Options

4 Important Pointers for Deployments

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

Important Commands to Remember

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

DaemonSet

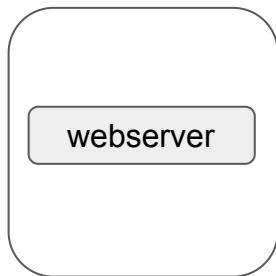
Global Service

Understanding the Use-Case

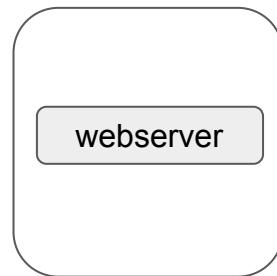
Sample: Use-Case:

Let's say that we have three nodes and we want to run a single copy of pod in every node.

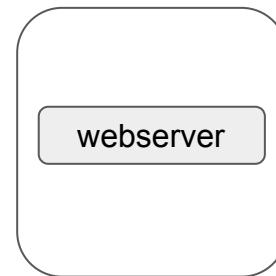
Worker Nodes



Node 01



Node 02



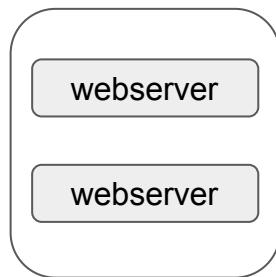
Node 03

Understanding the Use-Case

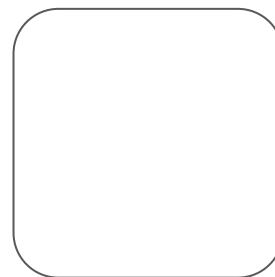
Sample: Use-Case:

Let's say that we have three nodes and we want to run a copy of pod in every node.

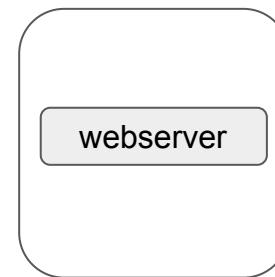
Worker Nodes



Node 01



Node 02



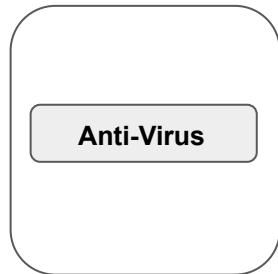
Node 03

Understanding DaemonSet

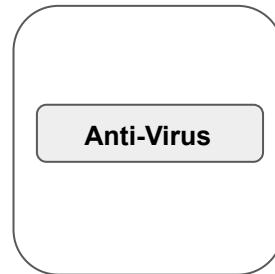
A DaemonSet can ensure that all Nodes run a copy of a Pod.

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

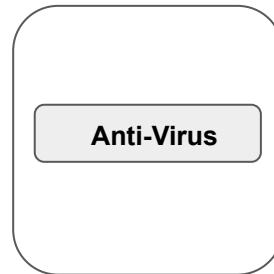
Worker Nodes



Node 01



Node 02



Node 03

nodeSelector

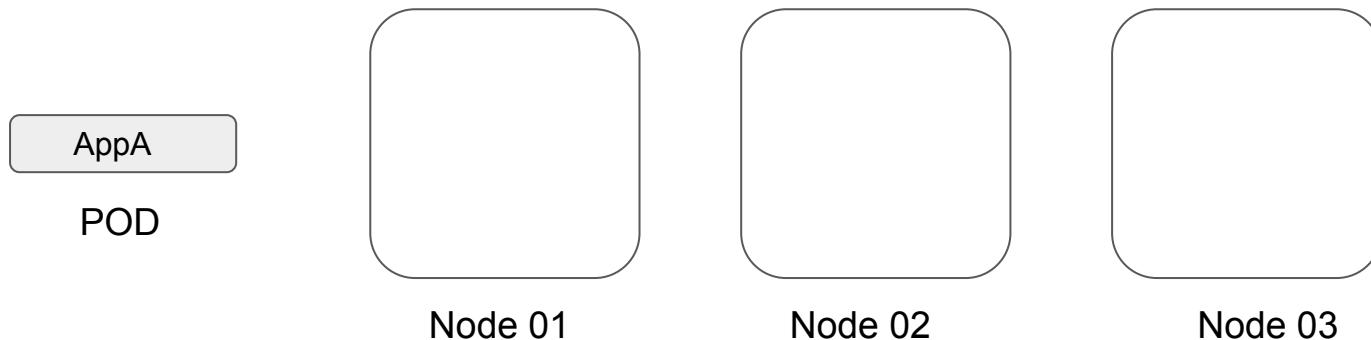
Global Service

Understanding the Use-Case

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

Use-Case:

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



Adding a Label to the Node

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

- disk: hdd
- disk:ssd

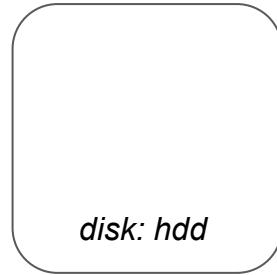
Worker Nodes



Node 01



Node 02



Node 03

Using nodeSelector Configuration

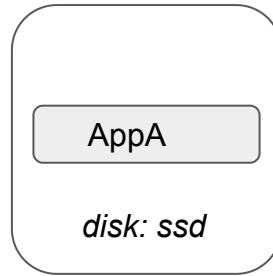
Step 2:

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

Worker Nodes



Node 01



Node 02



Node 03

Built-In Node Labels

Nodes come pre-populated with the standard set of labels. These include:

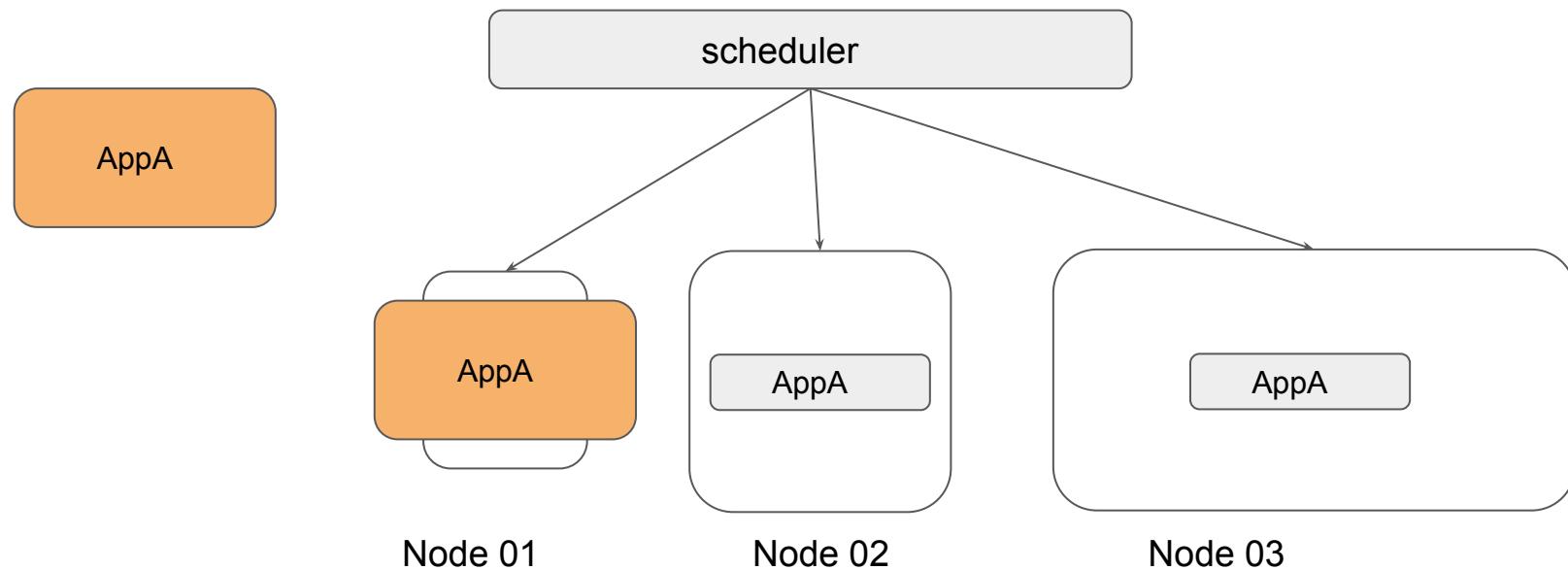
- kubernetes.io/hostname
- kubernetes.io/os
- kubernetes.io/arch
- failure-domain.beta.kubernetes.io/zone
- failure-domain.beta.kubernetes.io/region
- beta.kubernetes.io/instance-type

Resource Requests and Limits

Scheduling Objects

Understanding the Basics

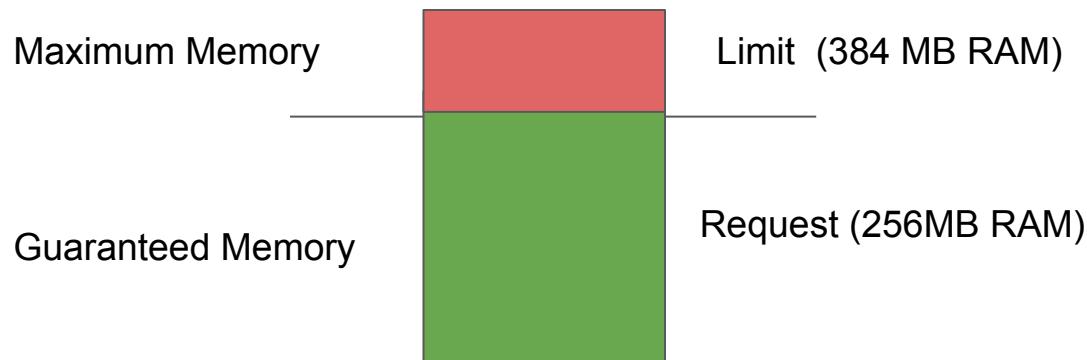
If you schedule a large application in a node which has limited resource, then it will soon lead to OOM or others and will lead to downtime.



Requests and Limits

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)

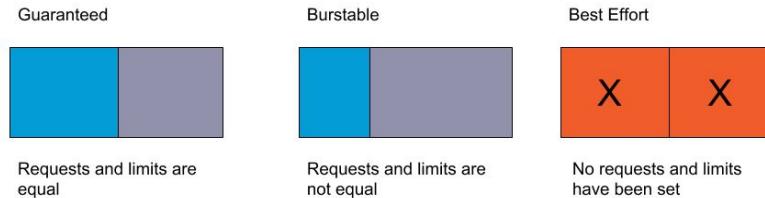
- Requests: Guaranteed to get.
- Limits: Makes sure that container does not take node resources above a specific value.



Requests and Limits

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.

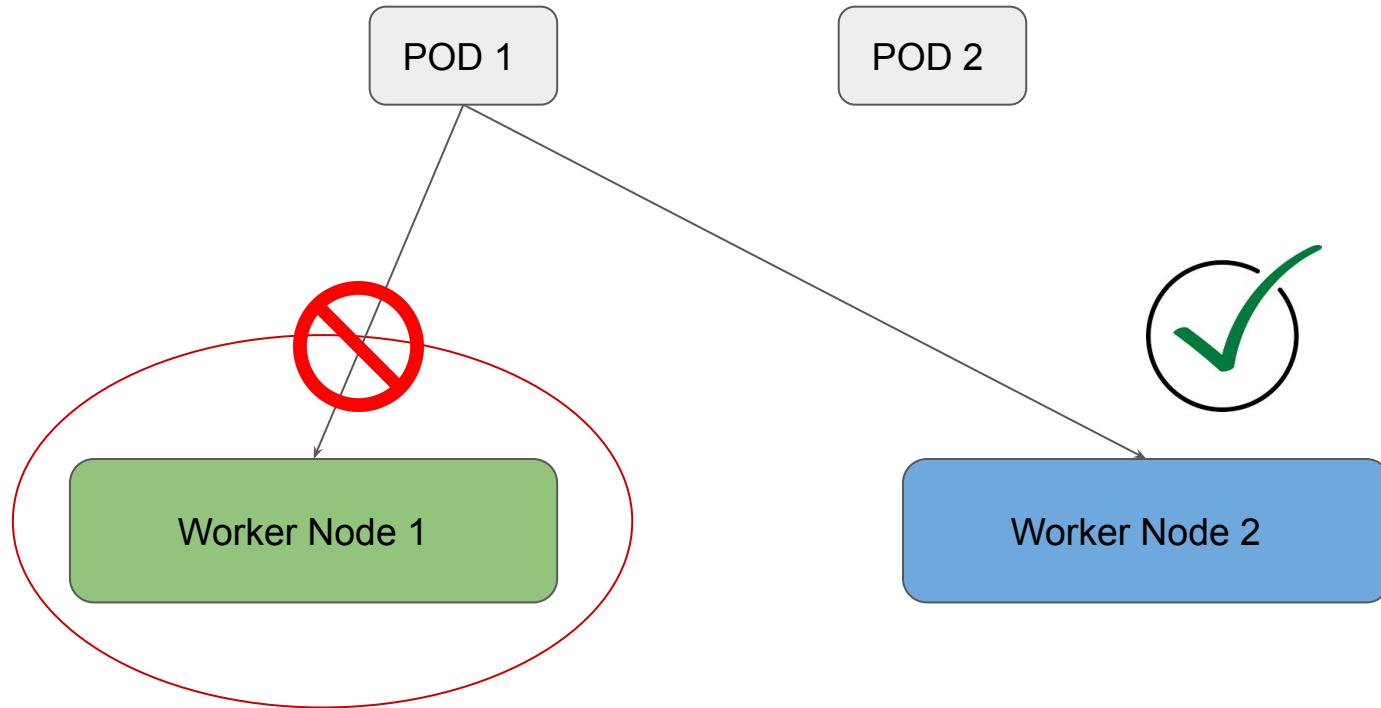


Taints and Tolerations

Advanced Scheduling

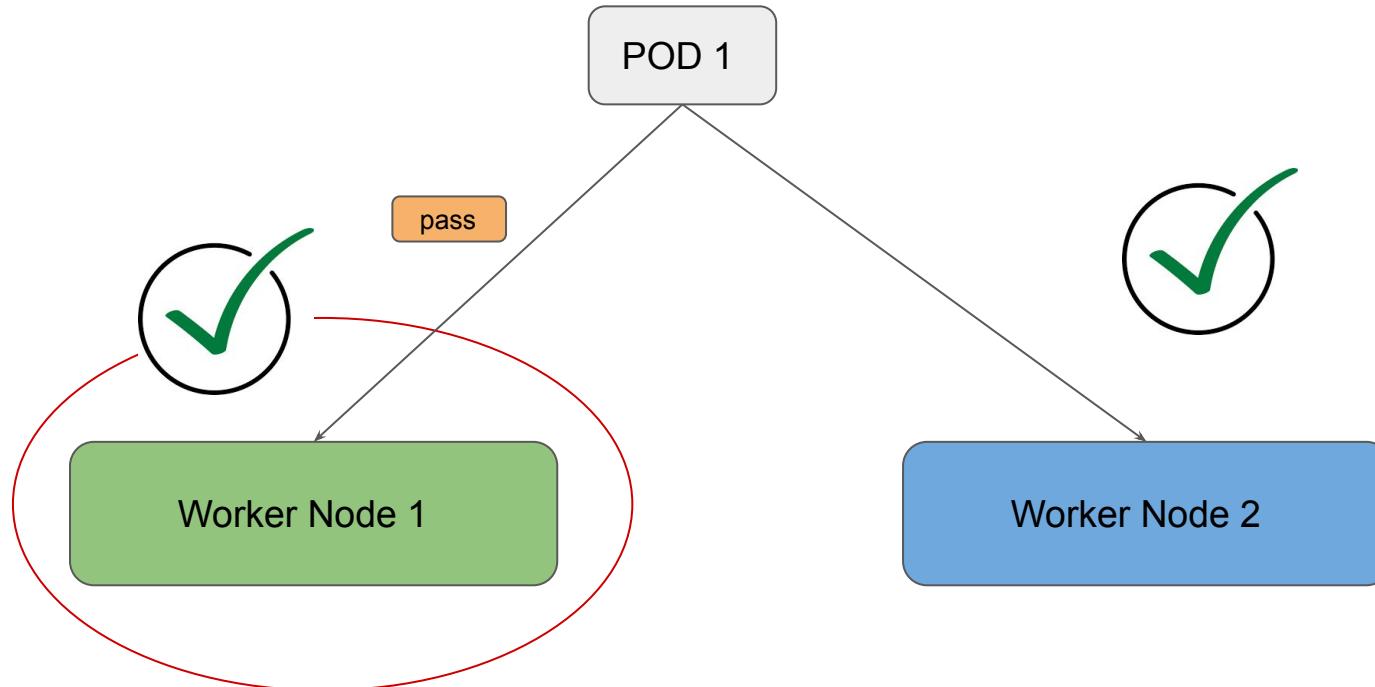
Understanding Taints

Taints are used to repel the pods from a specific nodes.



Understanding Tolerations

- In order to enter the taint worker node, you need a special pass.
- This pass is call toleration.

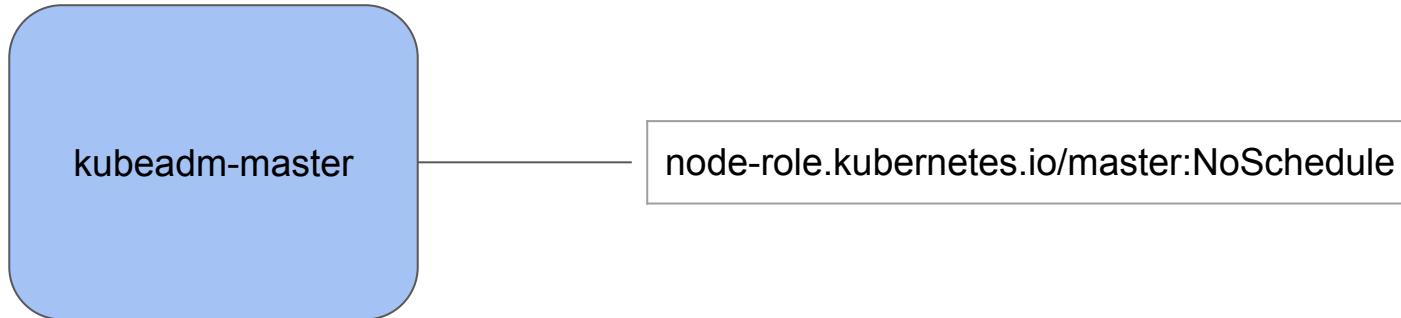


Our Setup

As of now, we have one server where we have kubeadm installed.

The master node has a Taint of NoSchedule

2 approaches: Create Pods with Toleration OR Remove the Taint.



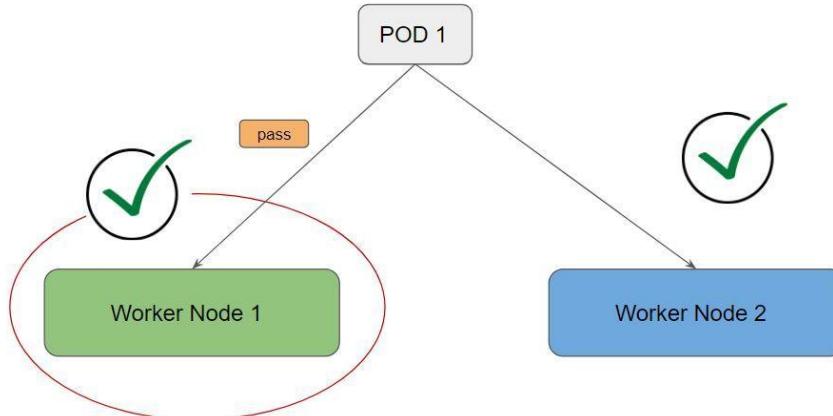
Taint and Toleration Components

Advanced Scheduling

Understanding Taints

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

We can apply toleration to a pod within the PodSpec



Components of Taints and Tolerations

Parameter	Description
key	A key is any string upto 253 characters.
value	The value is any string, up to 63 characters.
effect	<ul style="list-style-type: none">• NoSchedule• PreferNoSchedule• NoExecute
operator	<ul style="list-style-type: none">• Equal• Exist

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

Understanding Effects

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

Understanding Operator

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.

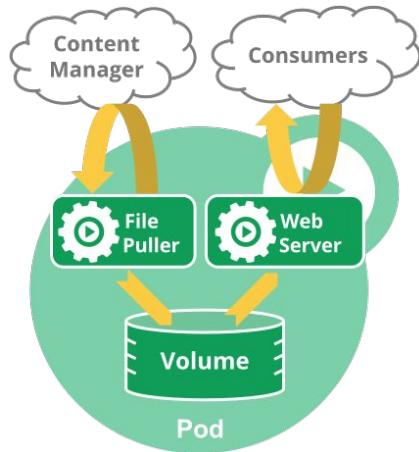
Multi-Container POD Patterns

Multi-Container Patterns

Overview of Sidecar Pattern

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

In below diagram 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.

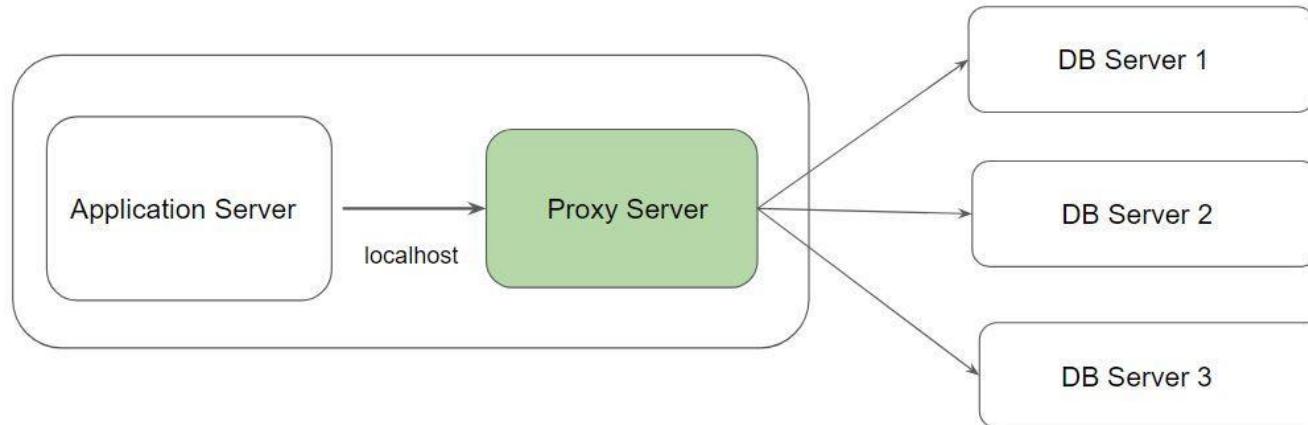


Overview of Sidecar Pattern



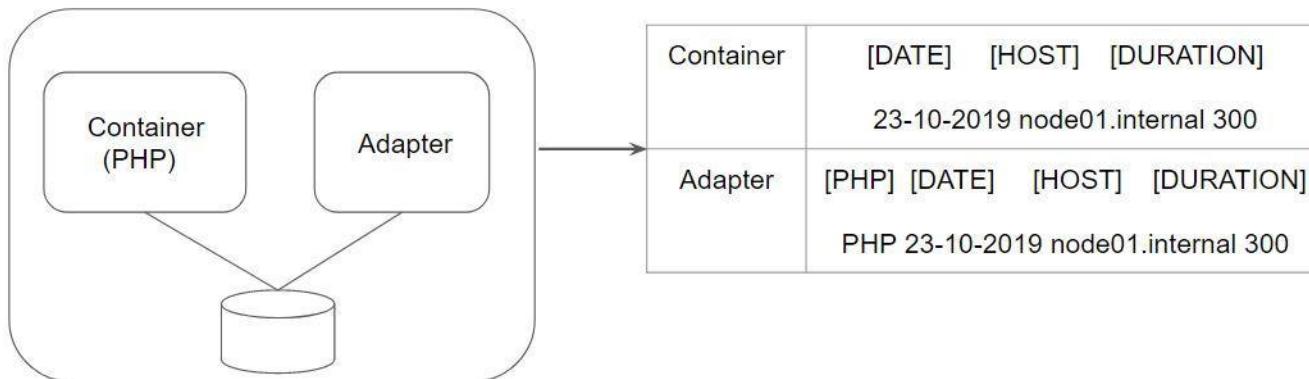
Ambassador Pattern

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



Sidecar Container - Adapter Pattern

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



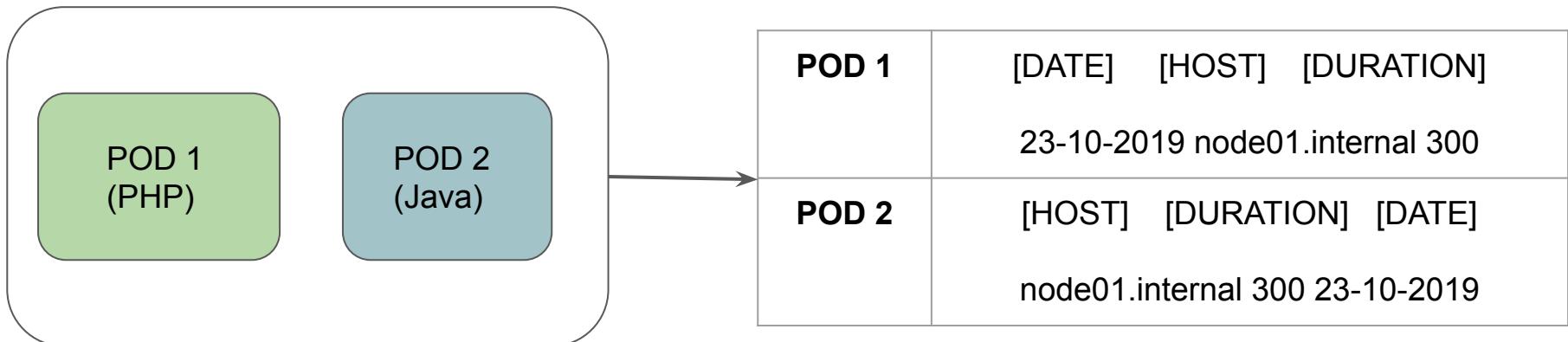
Adapter Pattern

Multi-Container Patterns



Overview of Adapter Pattern

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



Standardizing Log Format

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

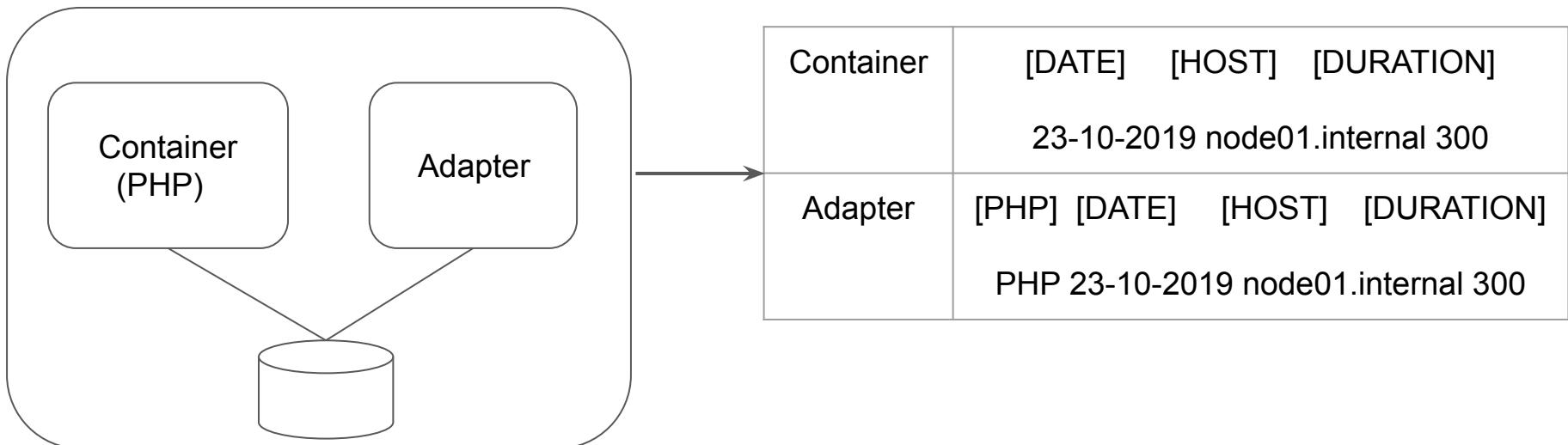
[PHP|JAVA] [DATE] [HOST] [DURATION]

PHP 23-10-2019 node01.internal 300

Transforming Logs

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

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

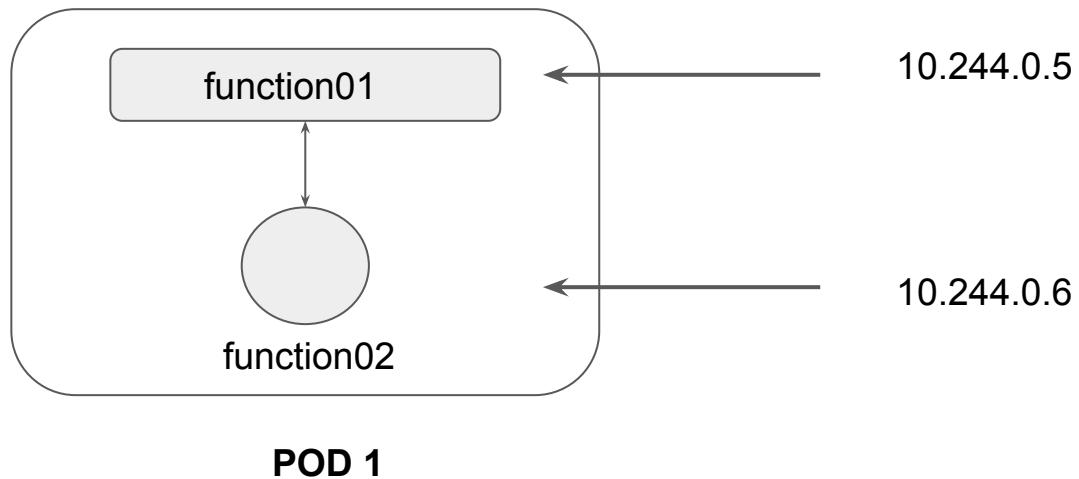


Overview of Service

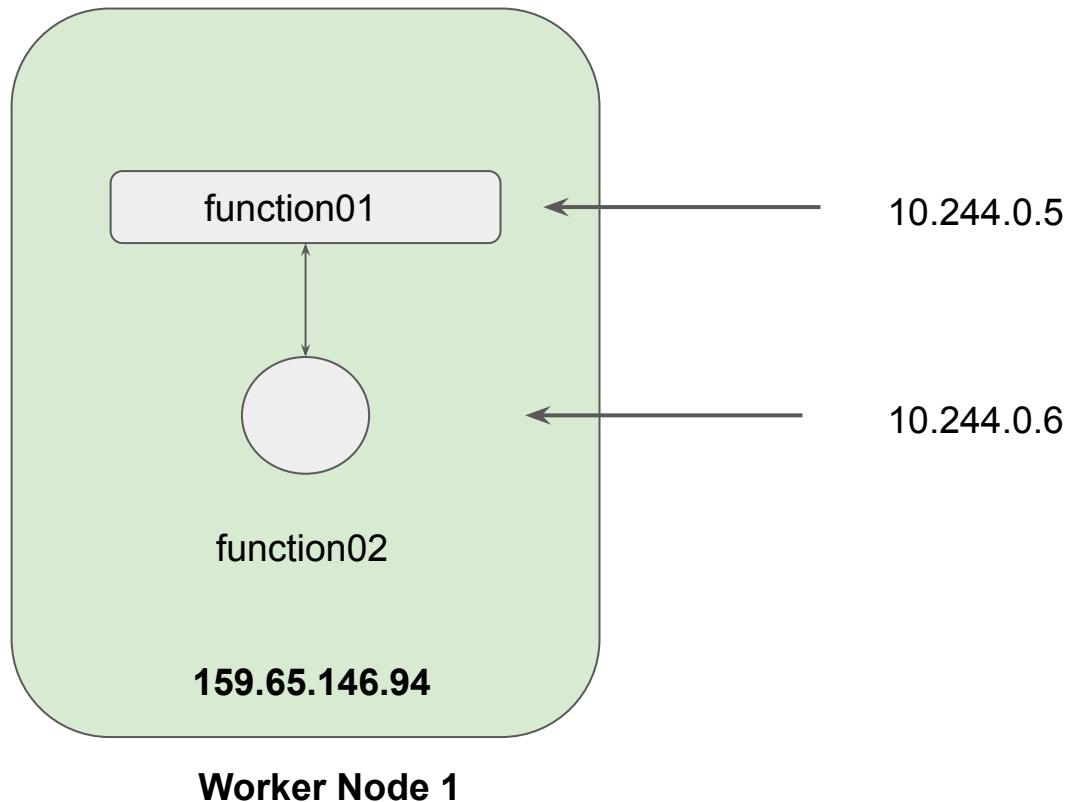
Let's get started

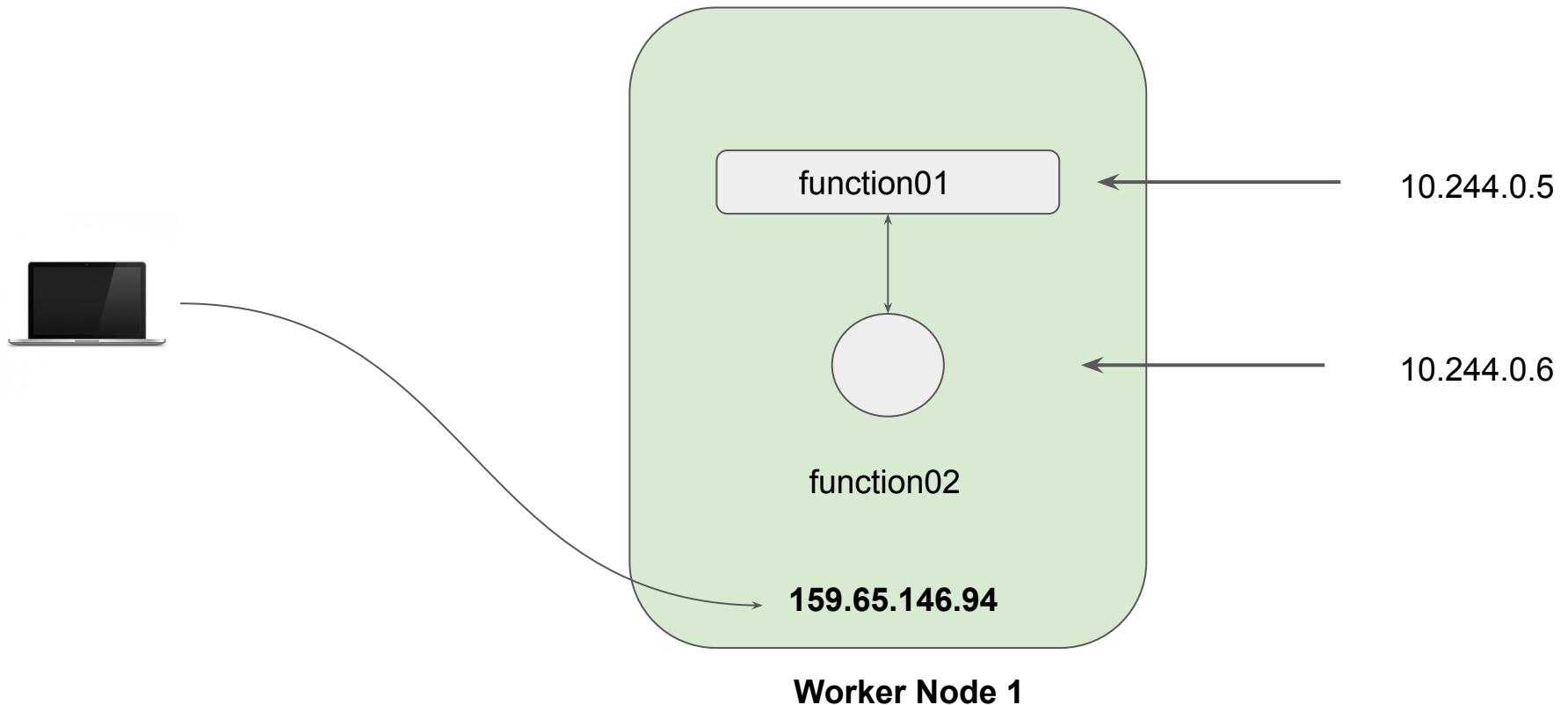
Overview of Kubernetes Pods

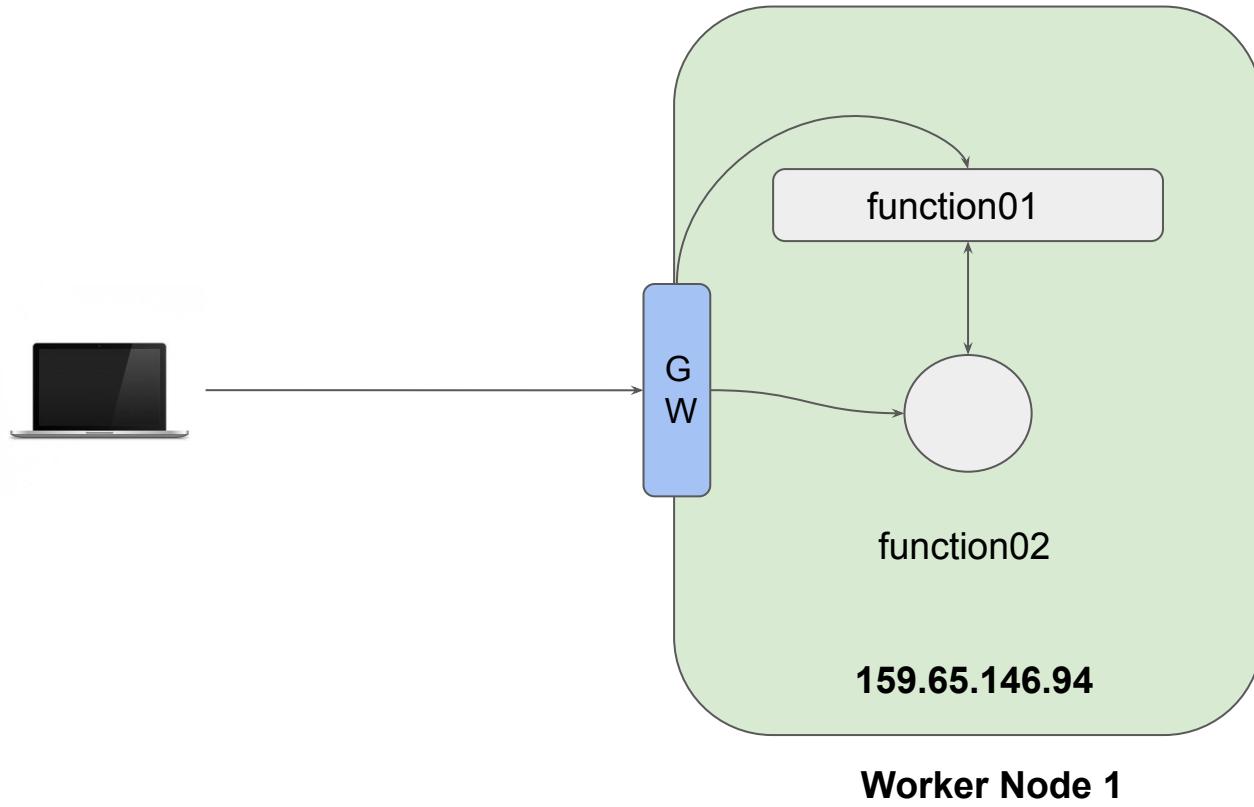
Whenever you create a Pod, the containers created will have Private IP addresses.

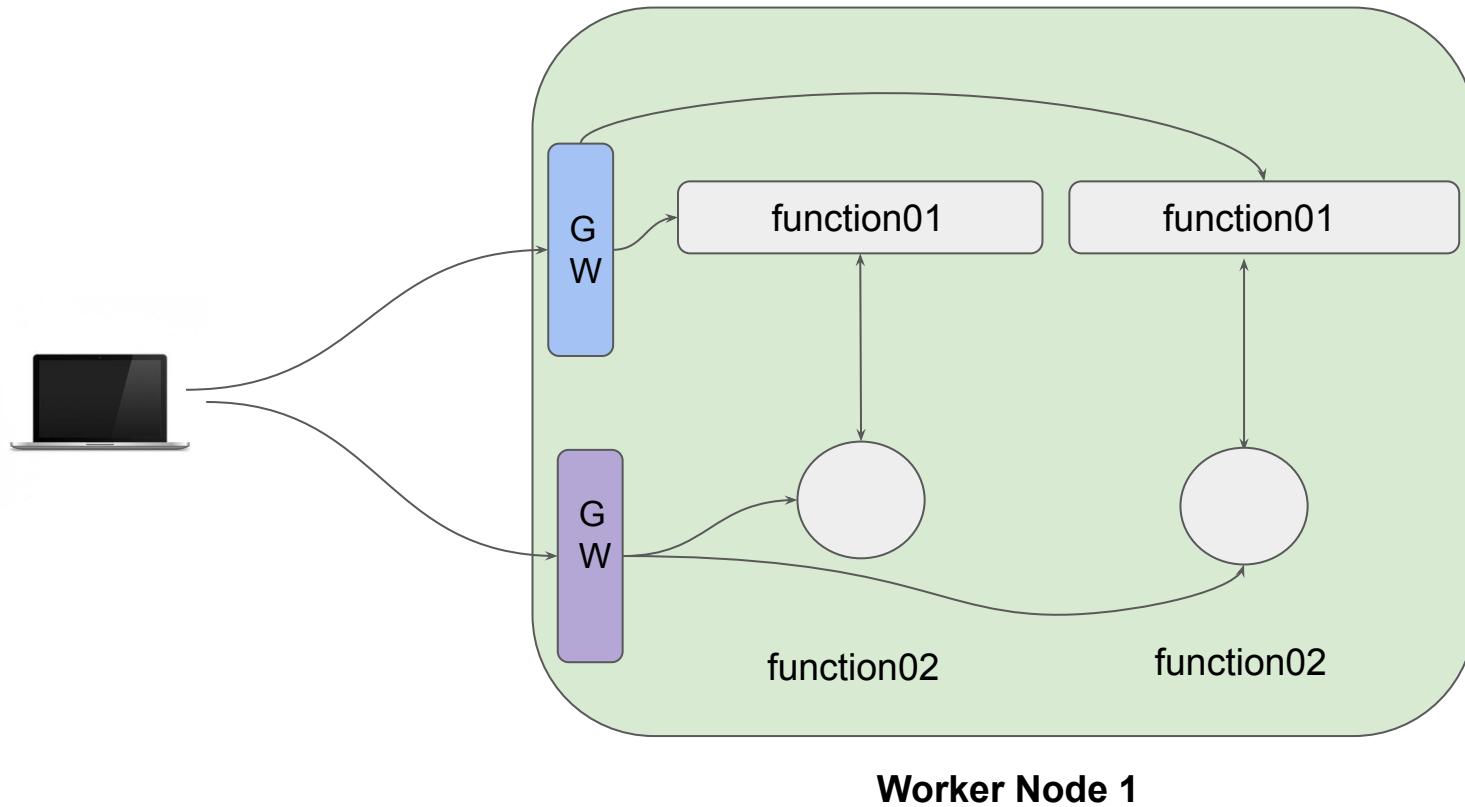


Bigger Network Picture









Worker Node 1

Importance of Service in Kubernetes

In a Kubernetes cluster, each Pod has an internal IP address.

Pods are generally ephemeral, they can come and go anytime.

We can make use of service which acts as a gateway and can get us connected with right set of pods.

Types of Kubernetes Service

Service is an abstract way of exposing application running in the pods as a network service.

There are several types of Kubernetes Services which are available:

- NodePort
- ClusterIP
- LoadBalancer
- ExternalName

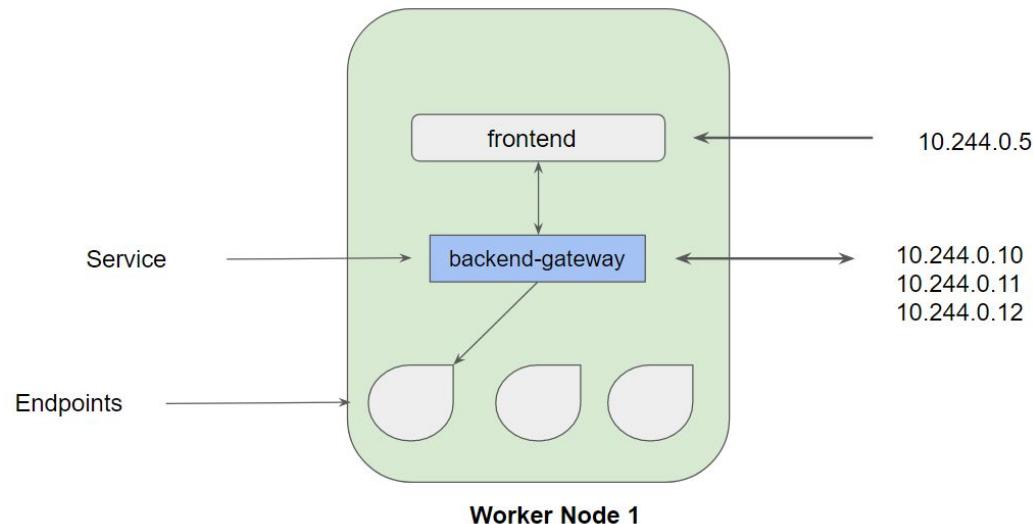
Creating Service and Endpoints

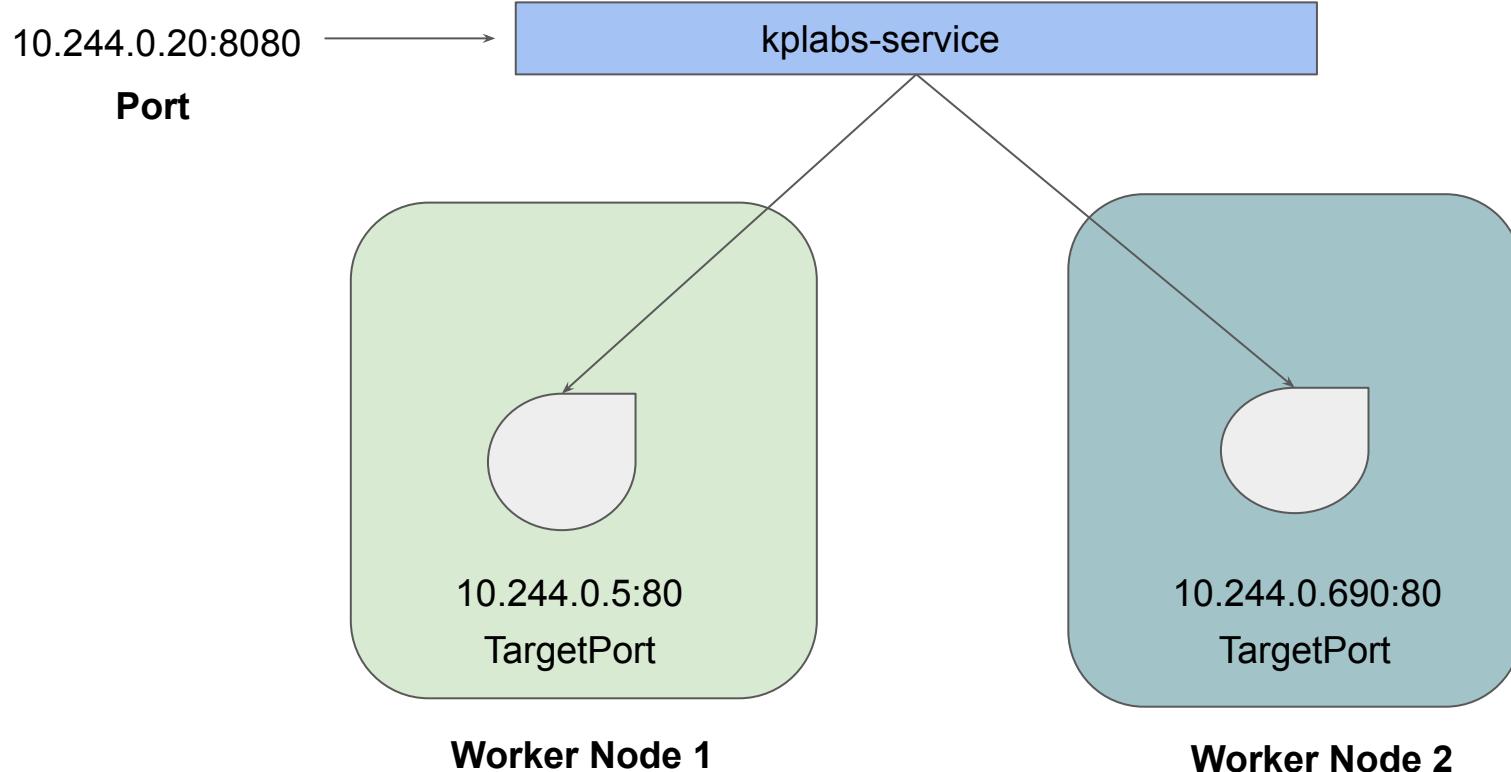
Networking Aspect

Service and Endpoints

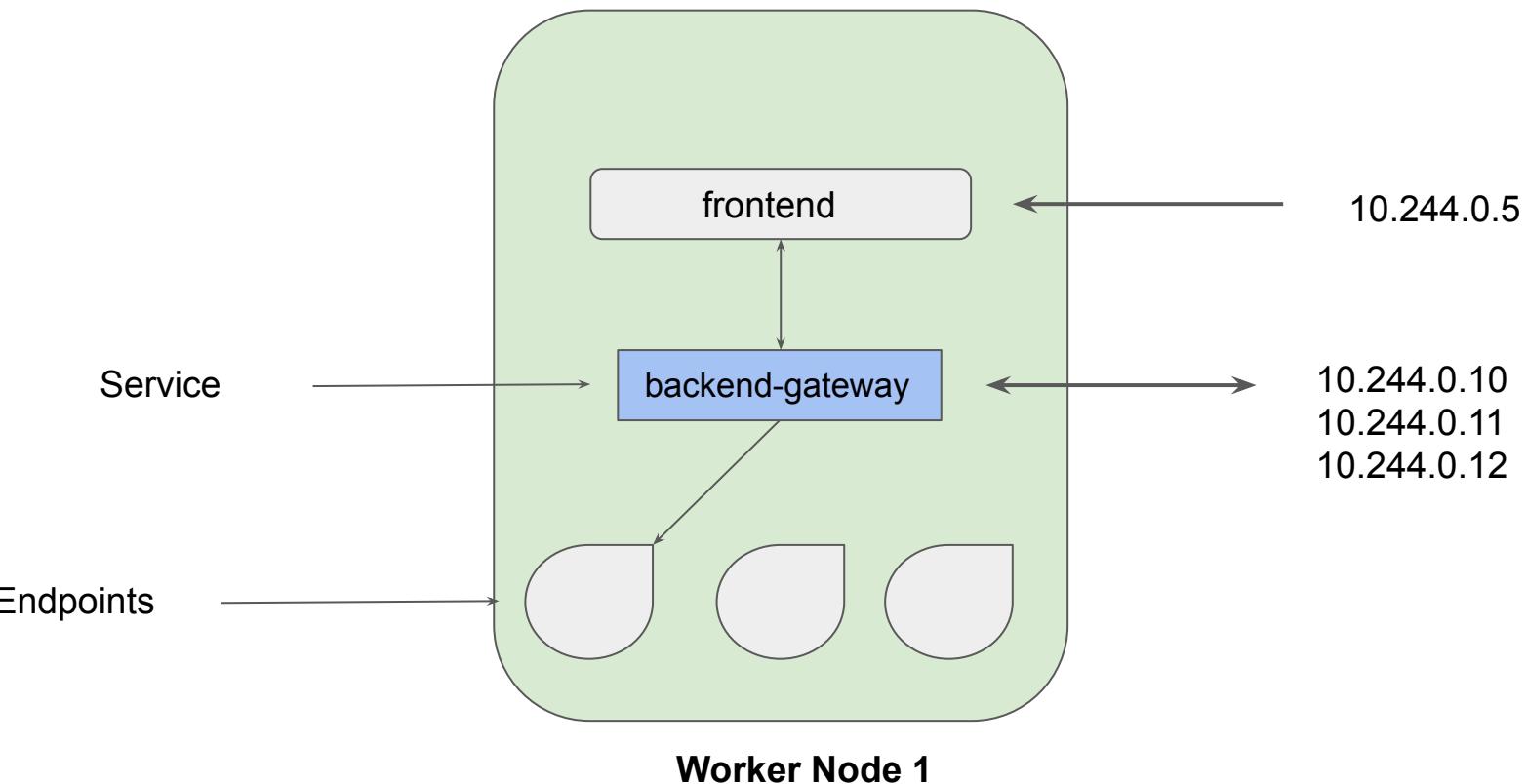
Service is a gateway that distributes the incoming traffic between its endpoints

Endpoints are the underlying PODS to which the traffic will be routed to.





Services and Endpoints



Service Type - ClusterIP

Networking Aspect

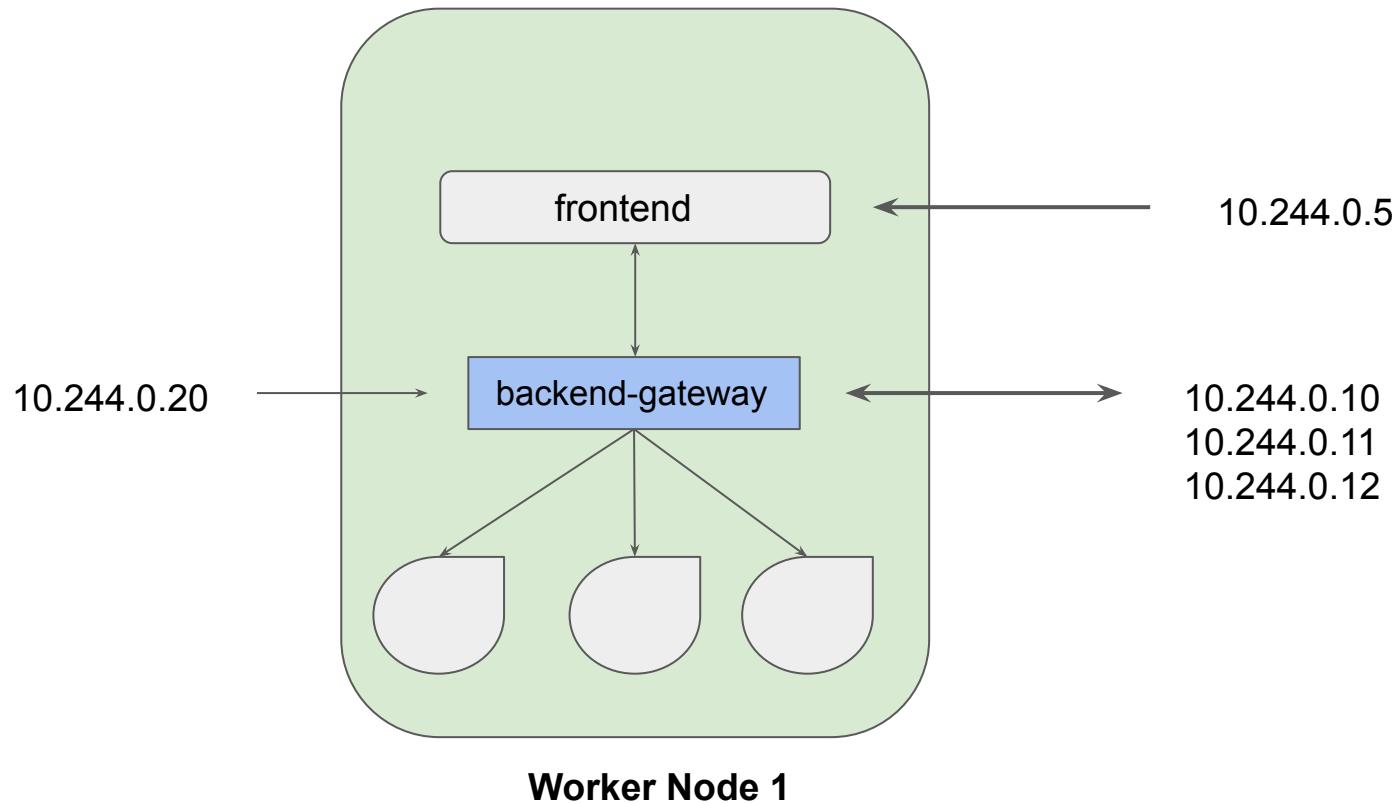
Understanding ClusterIP

Whenever service type is ClusterIP, an internal cluster IP address is assigned to the service.

Since an internal cluster IP is assigned, it can only be reachable from within the cluster.

This is a default ServiceType.

Overview of Services



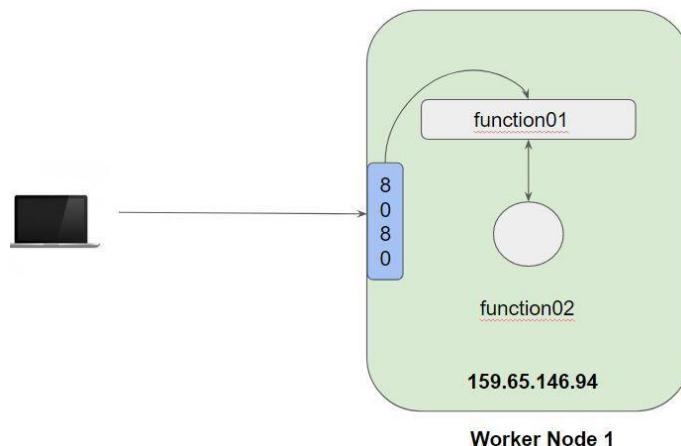
NodePort Service

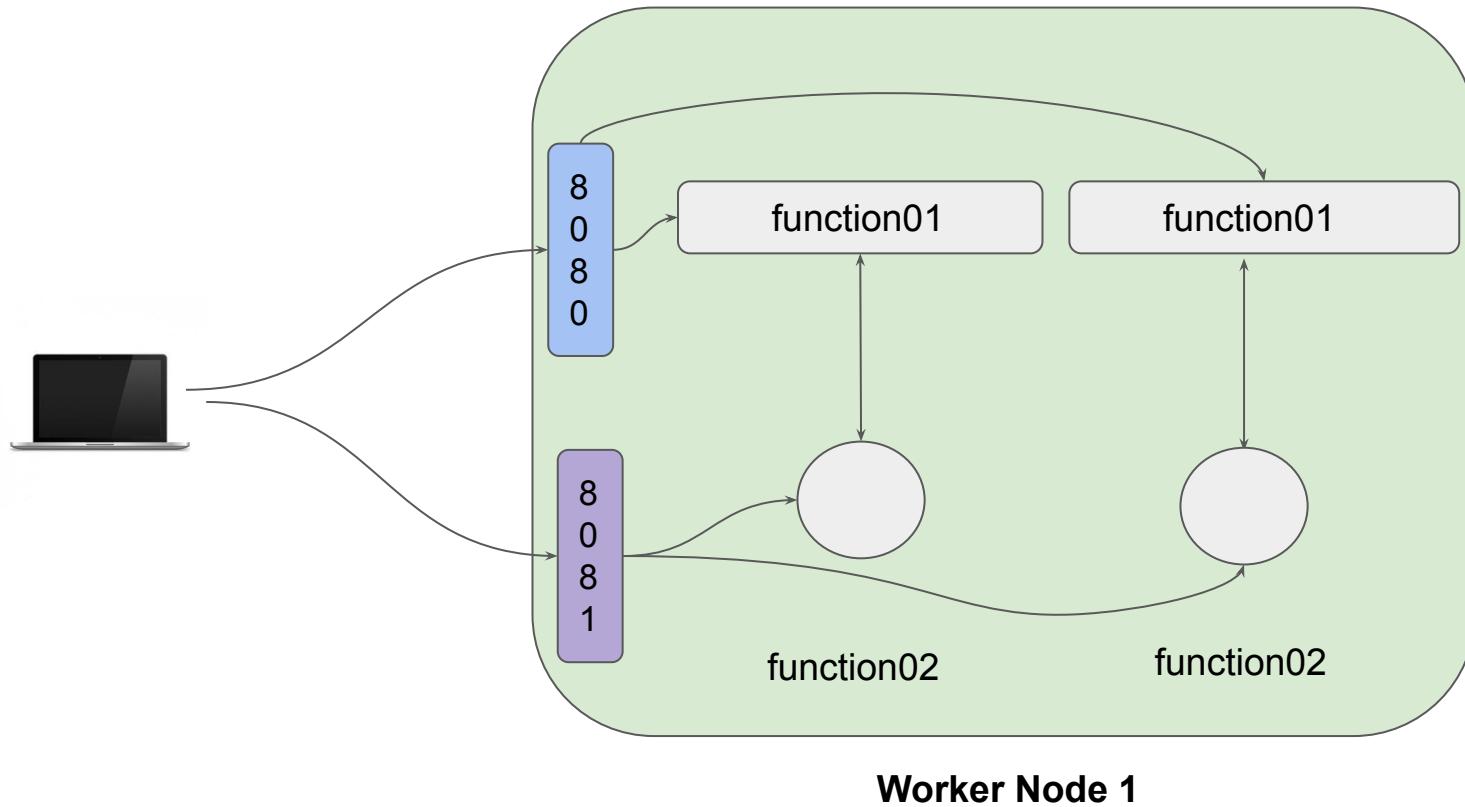
Let's get started

Overview of Kubernetes Pods

Exposes the Service on each Node's IP at a static port.

You'll be able to contact the NodePort Service, from outside the cluster, by requesting <NodeIP>:<NodePort>.

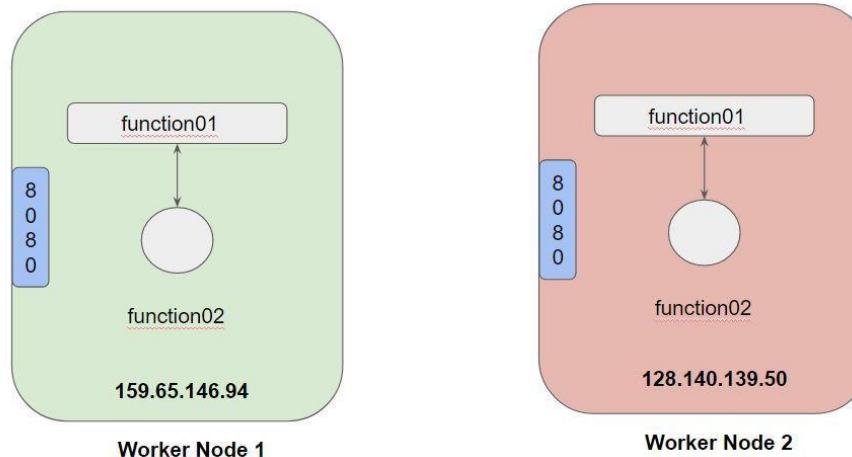




Types of Kubernetes Service

In a NodePort service type, Kubernetes will allocate a port from a range specified by --service-node-port-range flag (default: 30000-32767).

Each node proxies that port (the same port number on every Node) into your Service.



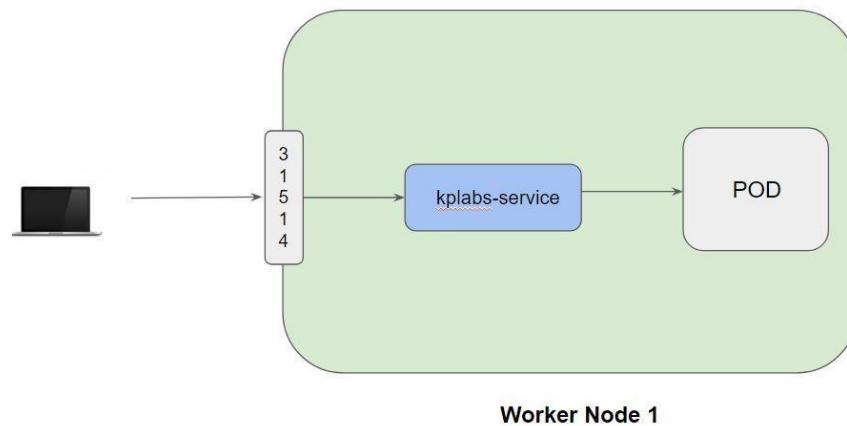
Service Type - LoadBalancer

Networking Aspect

Challenge with NodePort

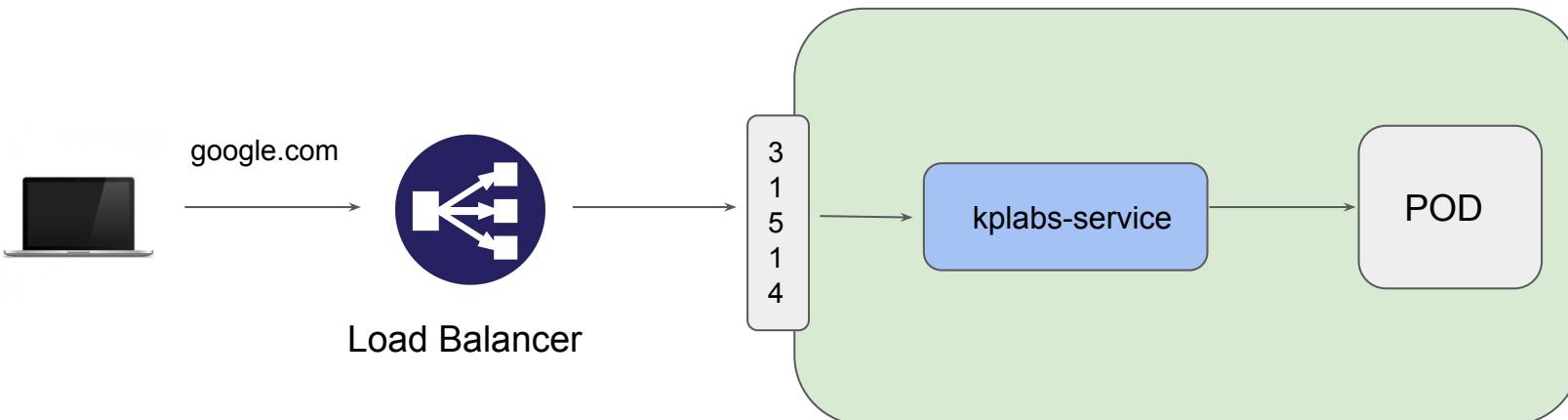
Challenge in NodePort: We need to access it via IP/DNS:Port

Example: google.com:31514



Understanding LoadBalancer Service Type

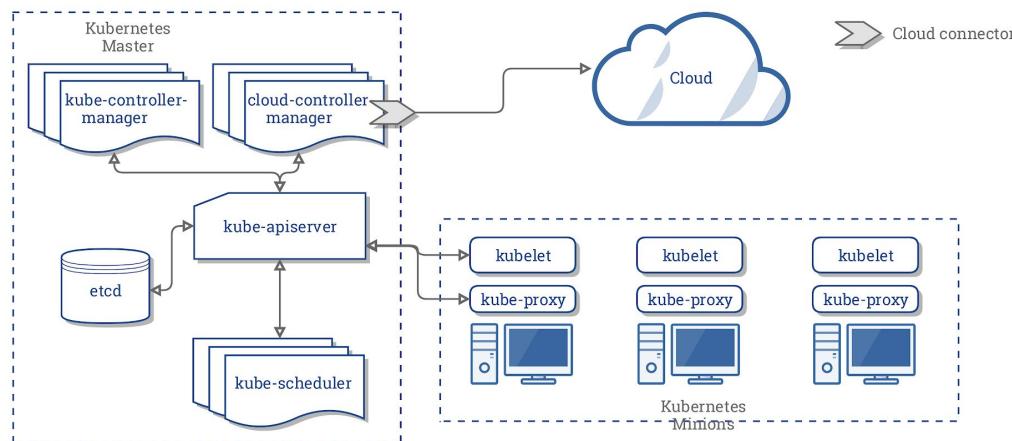
- LoadBalancer Service Type will automatically deploy an external load balancer.
- This load balancer takes care of routing requests to the underlying service.



Important Points to Remember

The overall implementation of LoadBalancer depends on your Cloud Provider.

If you plan to use it in bare-metal, then you will have to provide own load balancer implementation.



Generating Service Manifest - CLI

Creating Manifest Quickly

Service Manifest - POD

This will create a manifest file for service and add POD in its endpoints

Example Command:

```
C:\Users\Zeal Vora>kubectl expose pod nginx --name nginx-svc --port=80 --target-port=80 --dry-run=client -o yaml
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    run: nginx
  name: nginx-svc
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    run: nginx
```

Service Manifest - NodePort

Create a service manifest of type NodePort and add POD as part of it's endpoints.

Example Command:

```
C:\Users\Zeal Vora\Desktop\k8s\Networking>kubectl expose pod nginx --name nginx-svc --port=80 --target-port=80 --dry-run=client -o yaml --type=NodePort
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    run: nginx
    name: nginx-svc
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    run: nginx
  type: NodePort
```

Service Manifest - Deployment

This will create a manifest file for service and add deployment PODS in its endpoints

Example Command:

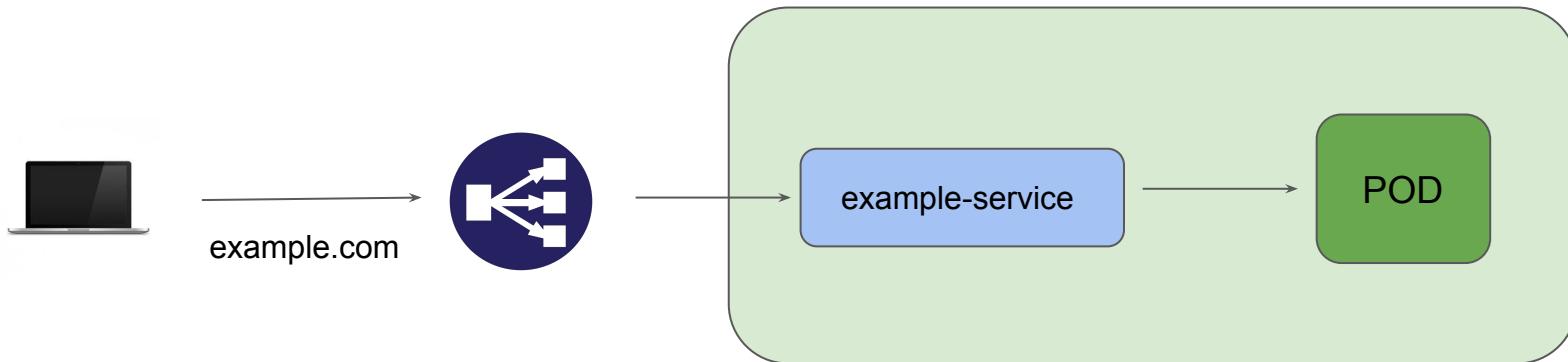
```
C:\Users\Zeal Vora\Desktop\k8s\Networking>kubectl expose deployment nginx-deployment --name nginx-dep-svc --port=80 --target-port=8000 --dry-run=client -o yaml
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    app: nginx
    name: nginx-dep-svc
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 8000
  selector:
    app: nginx
```

Ingress

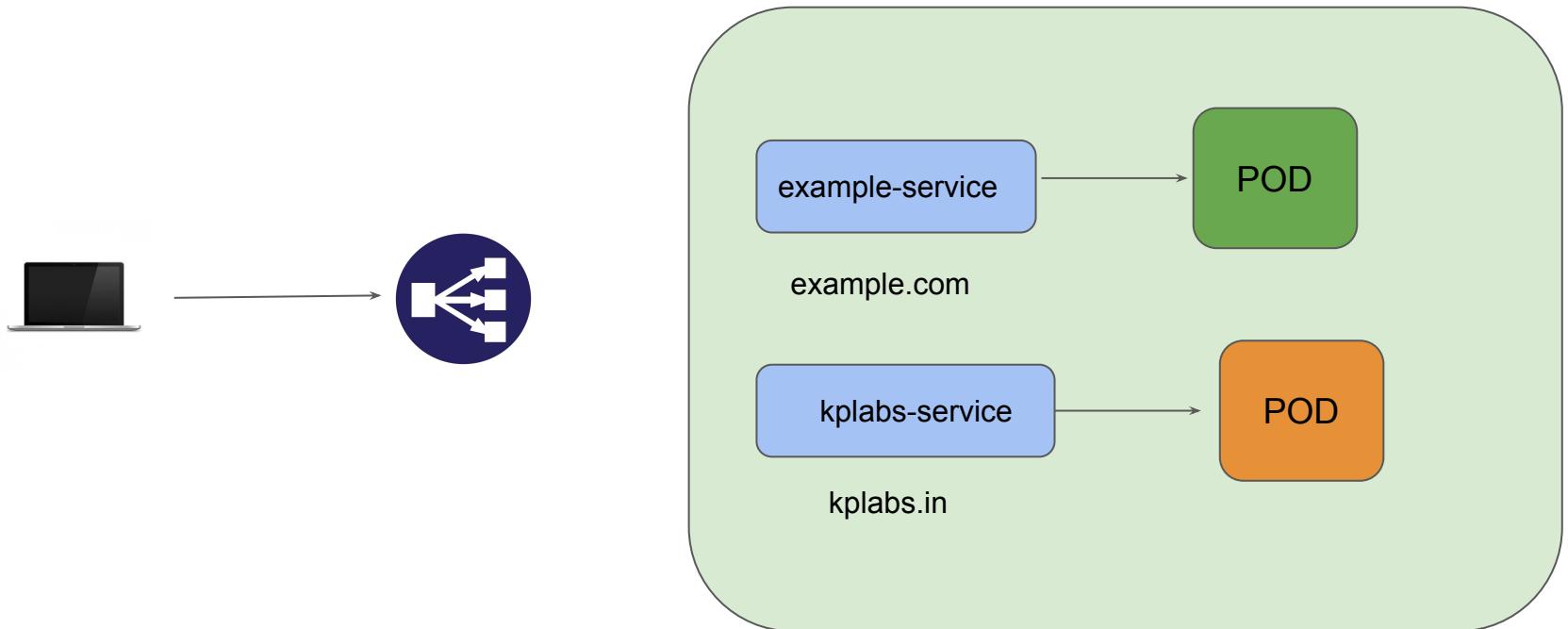
Networking Aspect

Challenge with Typical LoadBalancer Deployment

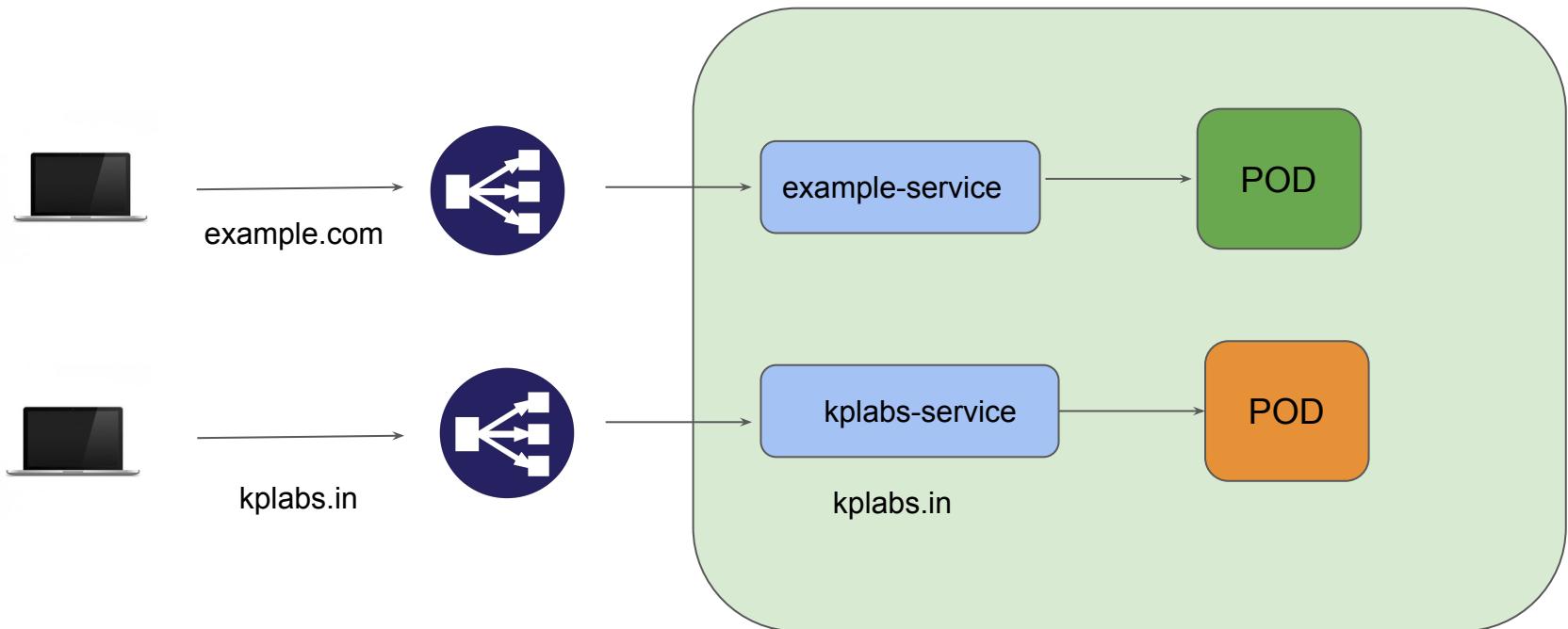
Whenever making use of LoadBalancer Service Type, out of box, you can make use of single website.



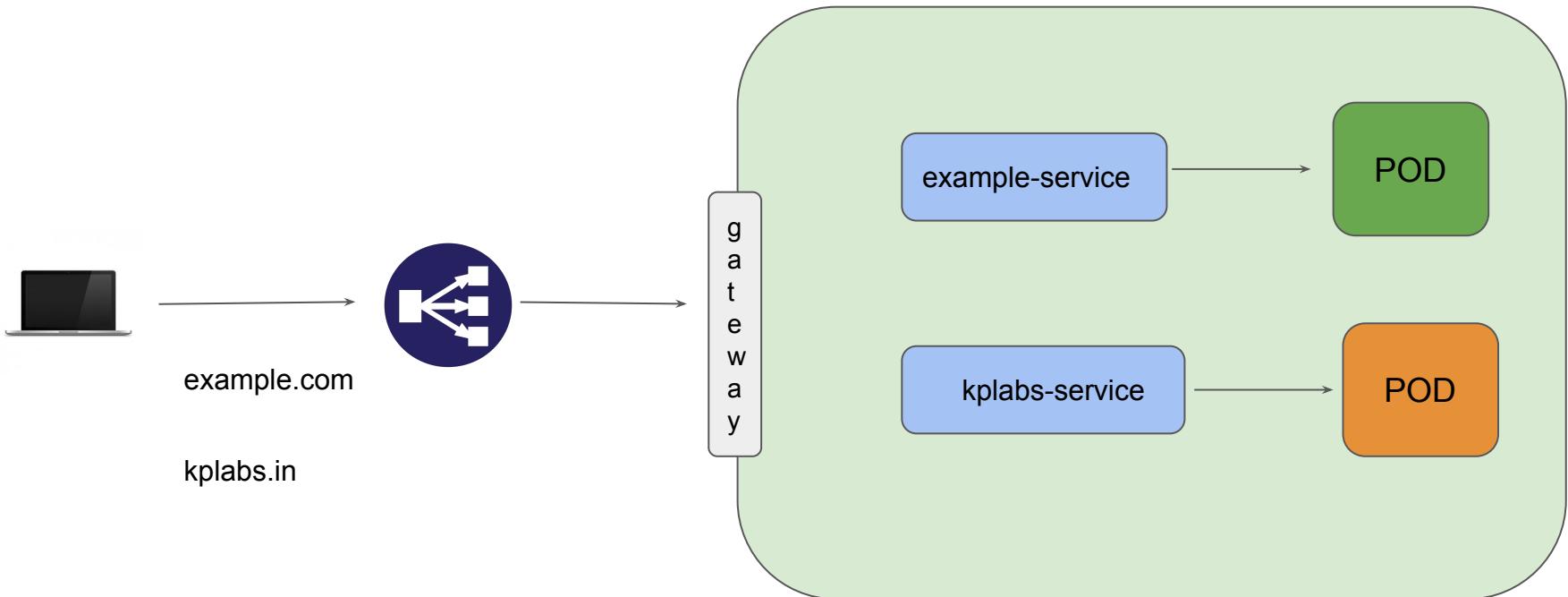
Challenge - Multiple Website Scenario



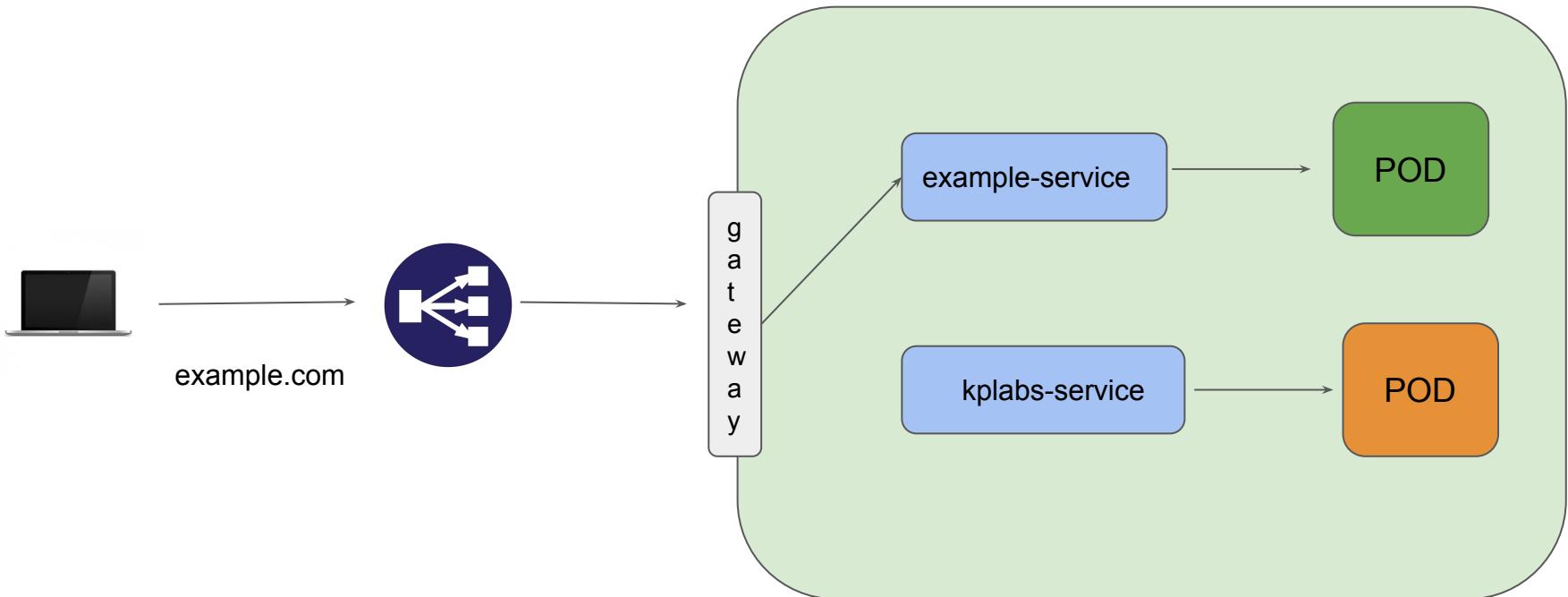
Multiple Website Scenario



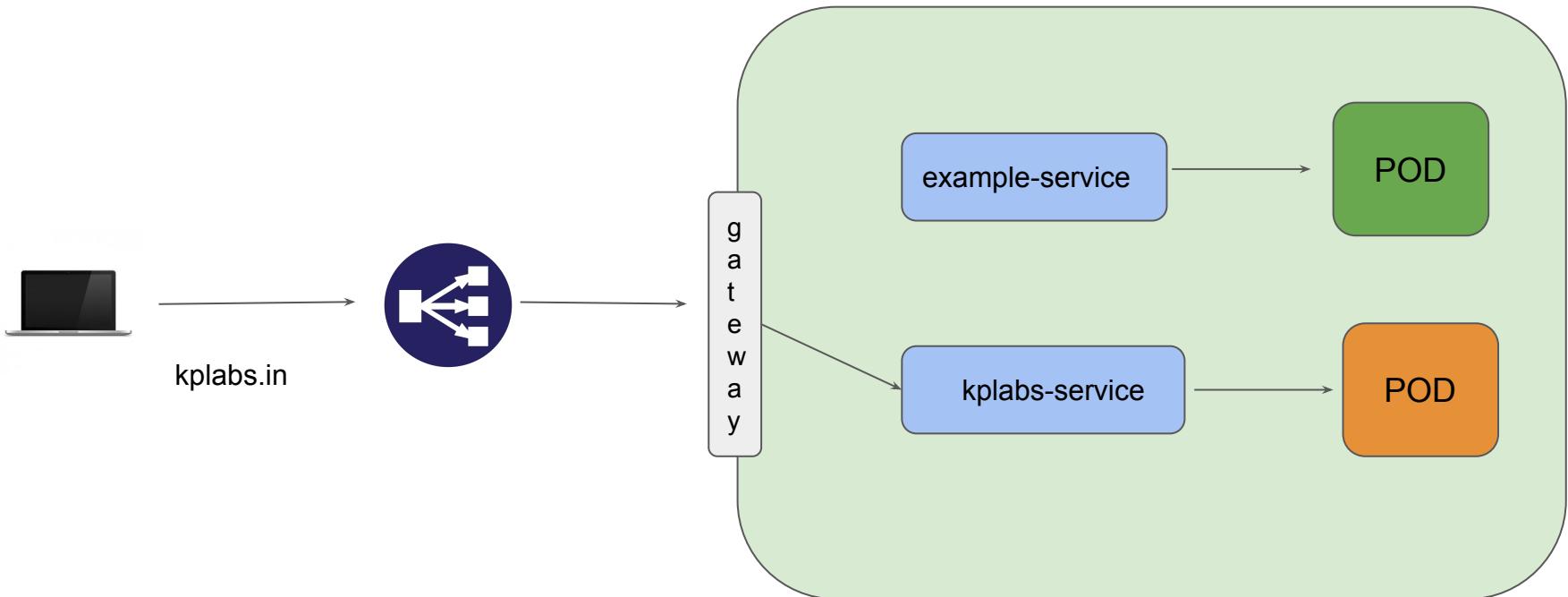
Overview of Ingress



Overview of Ingress



Overview of Ingress



Overview of Ingress

Kubernetes Ingress is a collection of routing rules which governs how external users access the services running within the Kubernetes cluster.

Ingress can provides various features which includes:

- Load Balancing
- SSL Termination
- Named-based virtual hosting

Ingress [Resource & Controllers]

Networking Aspect

Overview of Ingress

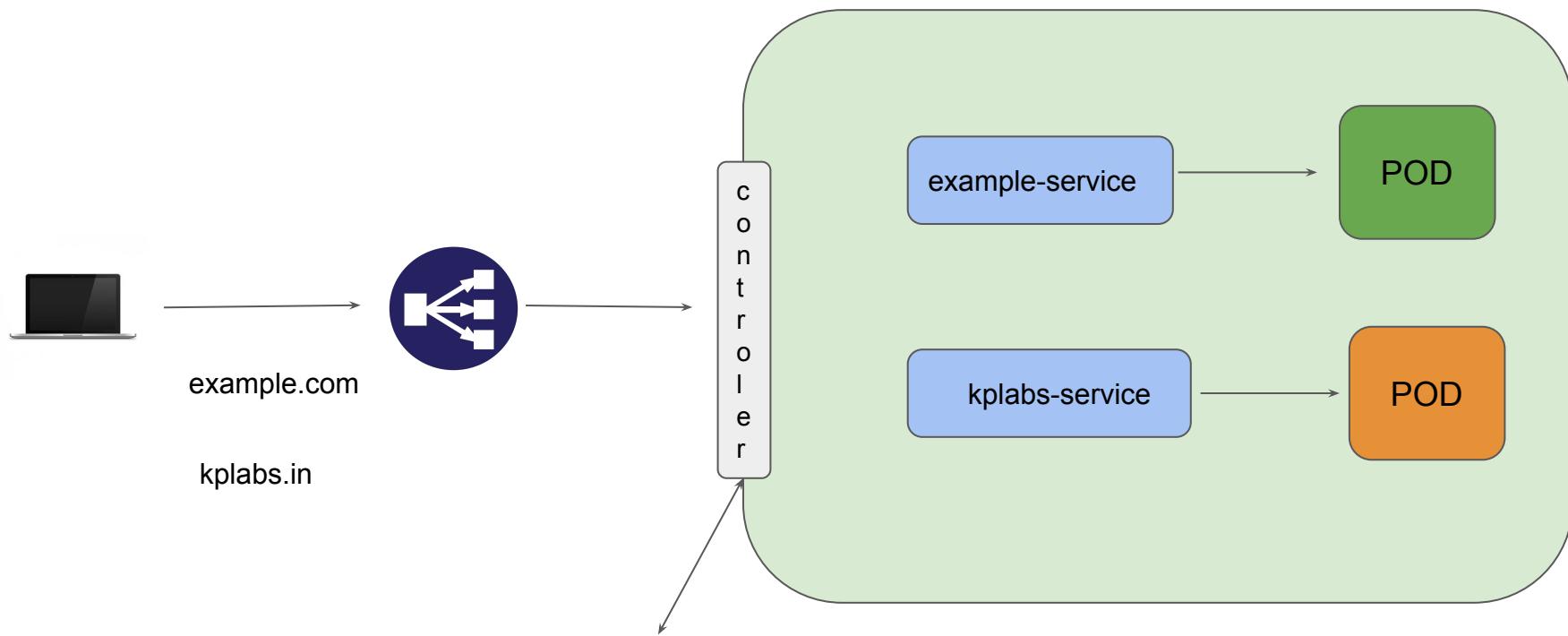
There are two sub-components when we discuss about Ingress:

- Ingress Resource
- Ingress Controllers

Ingress Resource contains set of routing rules based on which traffic is routed to a service.

Ingress Controller takes care of the Layer 7 proxy to implement ingress rules.

You must have an ingress controller to satisfy an Ingress. Only creating an Ingress resource has no effect



Rule	Host	Service
1	example.com	example-service
2	kplabs.in	kplabs-service

Overview of Helm

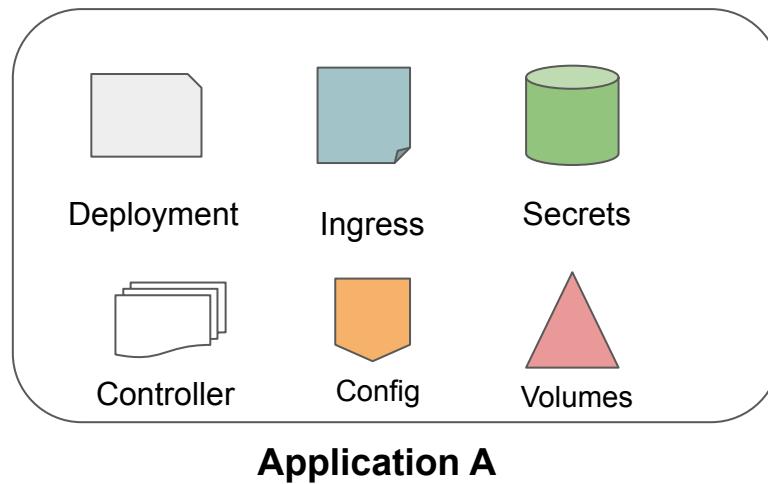
Package Manager for Kubernetes

Understanding Helm

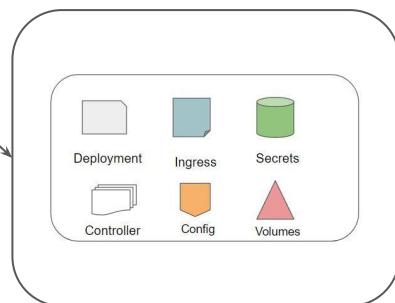
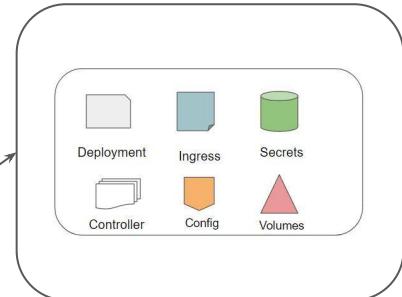
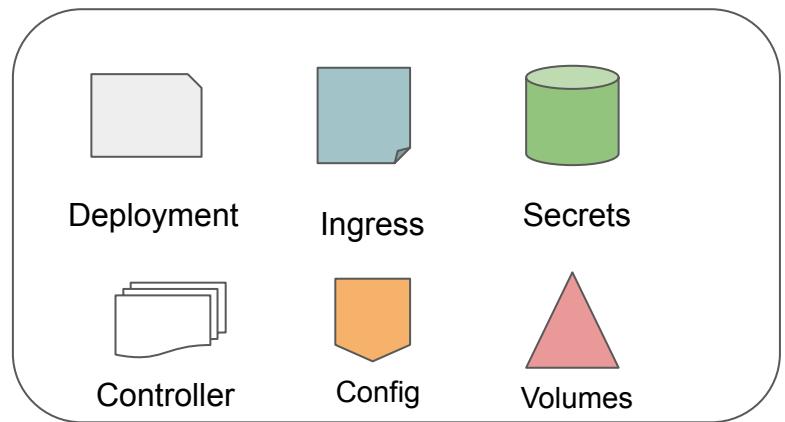
Helm is one of the package manager for Kubernetes.

Kubernetes application can contain lot of lot of objects like:

Deployments, Secrets, LoadBalancers, Volumes, services, ingress and others.

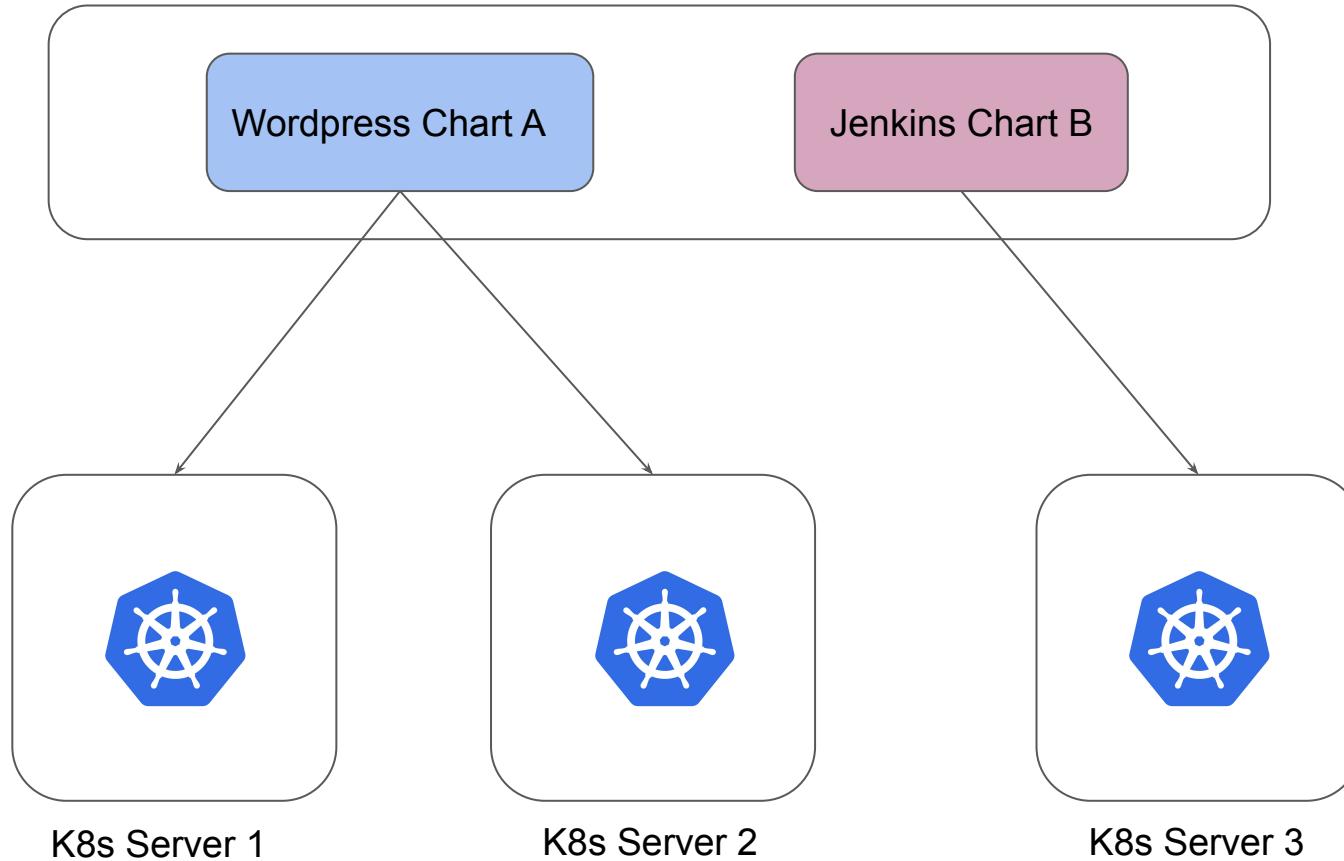


Deploying Helm Charts



Repository for Helm Charts

Public Repo



Revising Helm Concepts

A Chart is a Helm package.

It contains all of the resource definitions necessary to run an application, tool, or service inside of a Kubernetes cluster.

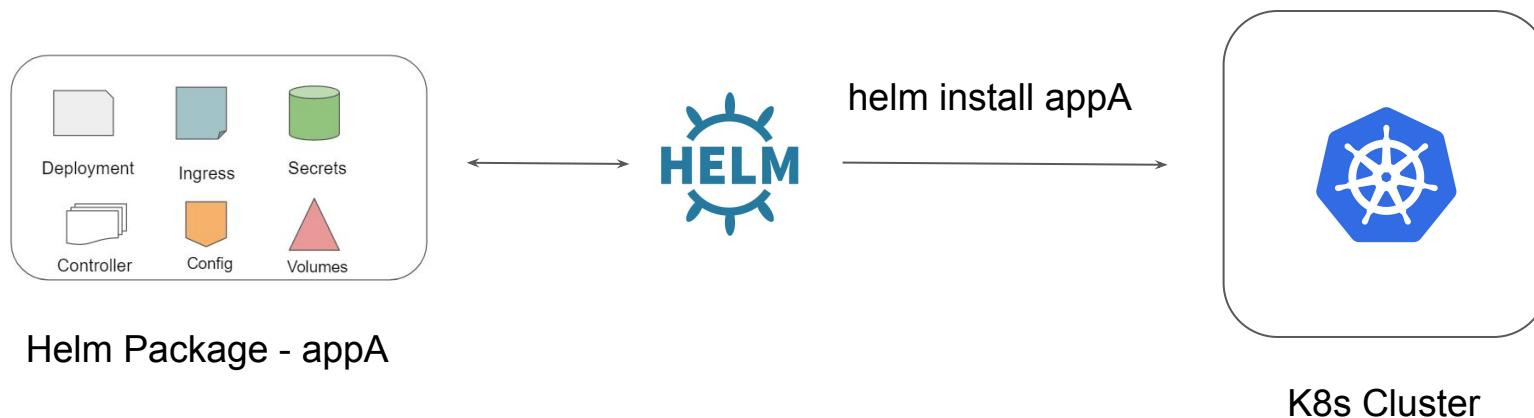
A Repository is the place where charts can be collected and shared

Deploying Helm Charts

Let's Deploy Applications

Installing Helm Chart

To install a new package, use the `helm install` command



Helm Package Location

Helm packages can be placed in local disk or can also be stored centrally in public or private repository.

Depending on the location of the package, there is a slight change in the installation step.

Public Repo

Wordpress Chart A

Jenkins Chart B

`helm repo add public-repo URL`



`helm install wordpress`



K8s Cluster

Important Notes

Every application packaged in Helm has its own set of requirements. Make sure to read the instructions carefully.

Install Helm Package only from trusted repositories.

Basic Helm Commands

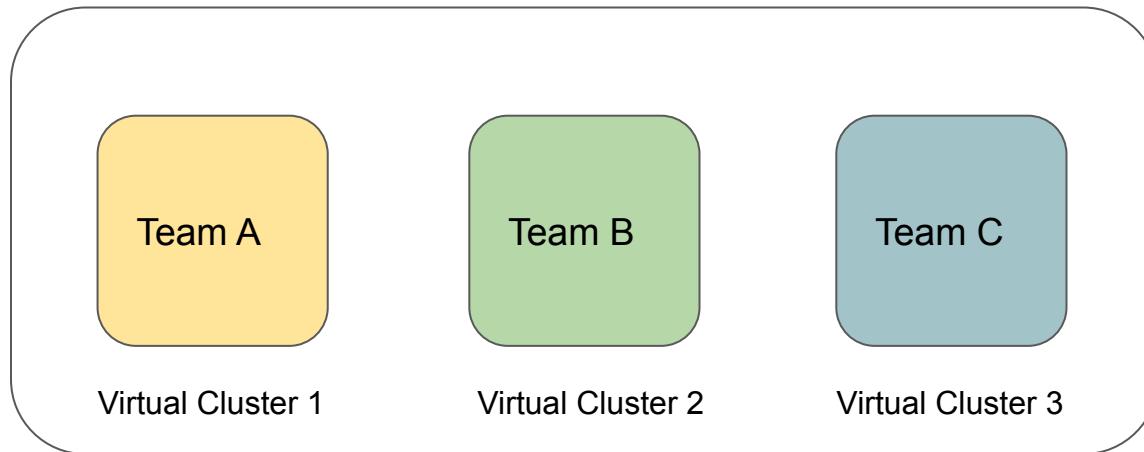
Commands	Description
helm repo add	Adds a chart repository.
helm install <chart>	Installs Helm Package
helm uninstall <release>	Uninstalls the release.
helm repo list	List repositories
helm list	List the releases

Namespaces

Security Aspect

Overview of Namespace

Kubernetes supports multiple virtual clusters backed by the same physical cluster. These virtual clusters are called namespaces.

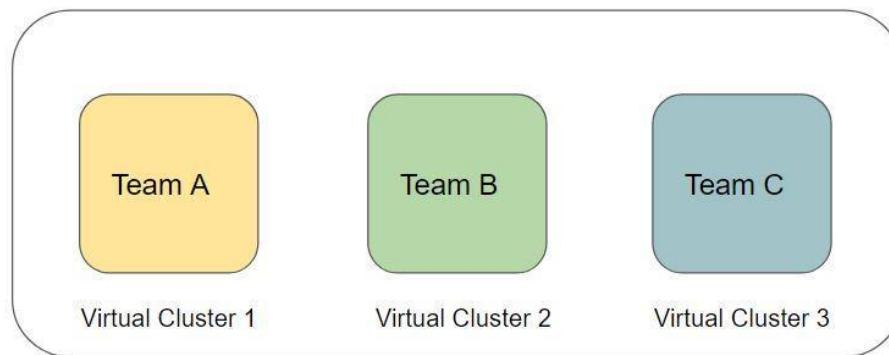


Benefits of Namespace

Example Use-Case:

You provide all the users within Team A full access on “Pods” resources [only for Team A NS]

You provide all the users within Team A full access on deployments [only for Team B NS]



List of Namespaces

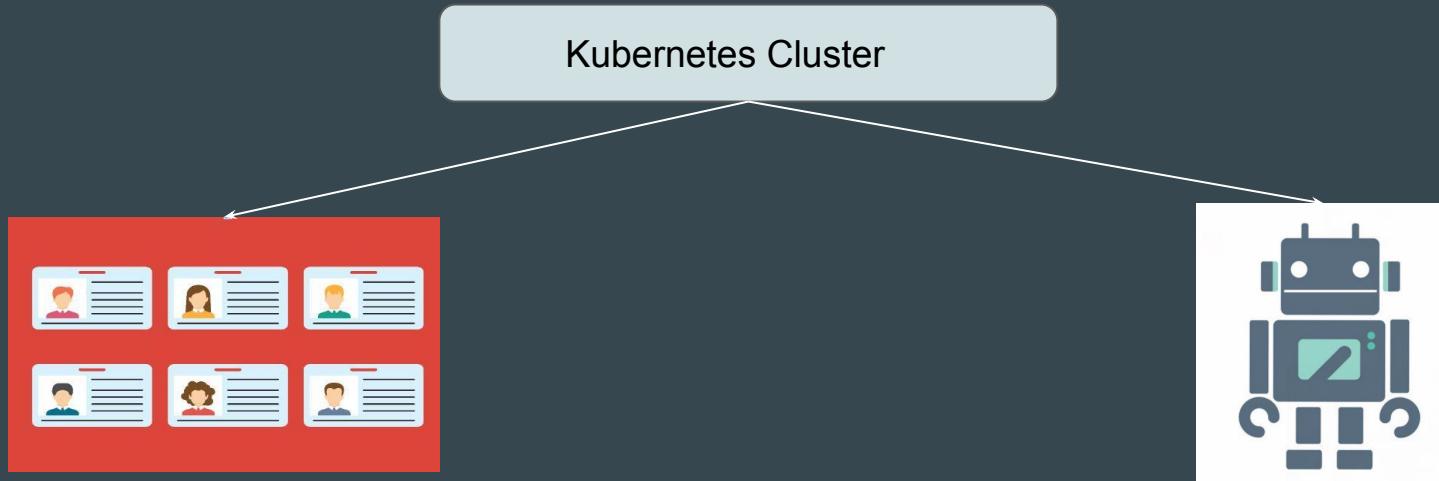
Namespace	Description
default	The default namespace for objects with no other namespace
kube-system	The namespace for objects created by the Kubernetes system
kube-public	This namespace is created automatically and is readable by all users. It contains information, like CA, that helps kubeadm join and authenticate worker nodes.
kube-node-release	The kube-node-lease namespace contains lease objects that are used by kubelet to determine node health.

Service Accounts

Understanding the basics

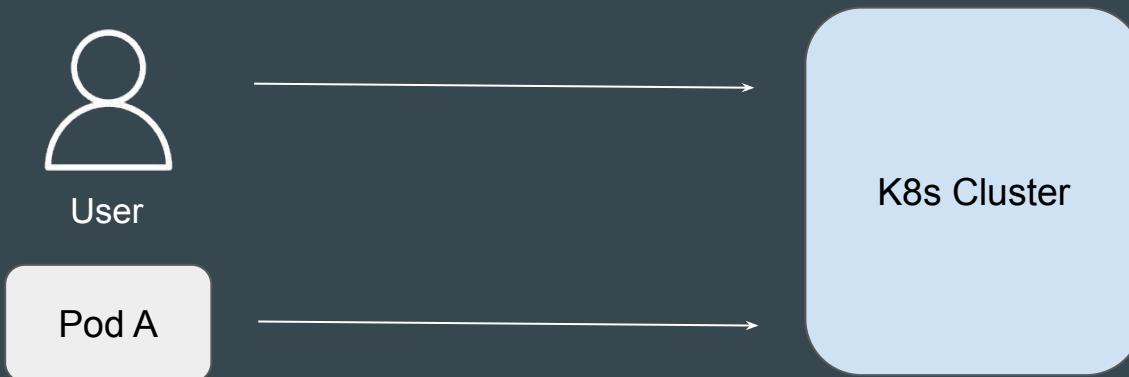
Kubernetes Clusters have two categories of accounts:

- User Accounts (For Humans).
- Service Accounts (For Applications)



Importance of Credentials

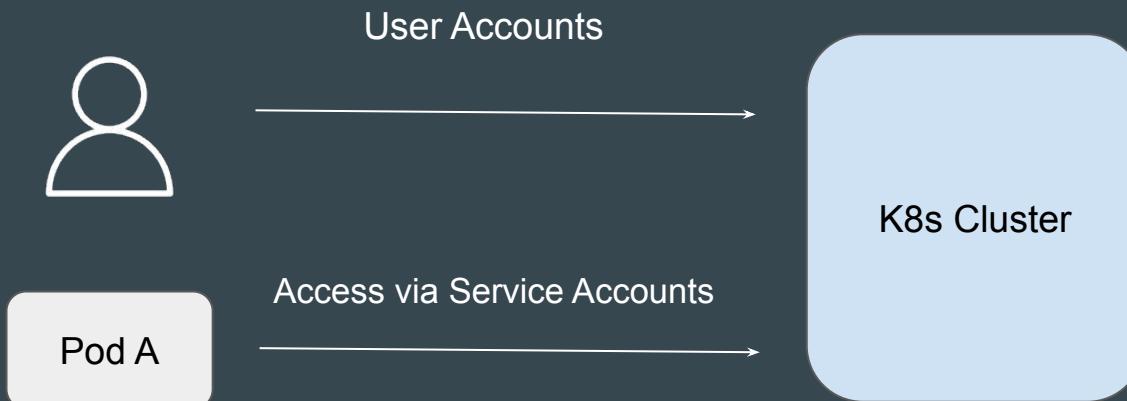
To connect to Kubernetes cluster, an entity needs to have some kind of authentication credentials.



Different Type of Credentials

Humans will use **User Accounts** to connect to Cluster.

Pods / Applications will use **Service Accounts** to connect to Cluster.



Service Accounts in K8s Cluster

Various different components of Kubernetes uses service accounts to communicate with the cluster

```
C:\Users\zealv>kubectl get serviceaccounts --all-namespaces
NAMESPACE      NAME          SECRETS   AGE
default        default       0         5m57s
kube-node-lease default       0         5m57s
kube-public    default       0         5m57s
kube-system   attachdetach-controller 0         5m57s
kube-system   certificate-controller 0         6m1s
kube-system   cilium        0         4m22s
kube-system   cilium-operator 0         4m22s
kube-system   cloud-controller-manager 0         5m54s
kube-system   cluster-autoscaler 0         5m10s
kube-system   clusterrole-aggregation-controller 0         5m57s
kube-system   coredns        0         102s
```

Service Accounts and Pods

Let's assume that a Service Account is associated with Pod A.

Pod A can use the token associated with the service account to perform some actions on Kubernetes cluster.



Service Accounts - Points to Note

Default Service Account

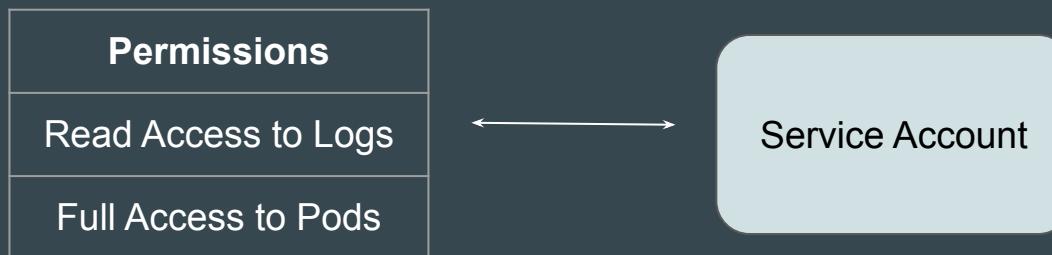
When you create a cluster, Kubernetes automatically creates a ServiceAccount object named **default** for every namespace in your cluster.

```
C:\Users\zealv>kubectl get serviceaccount
NAME      SECRETS   AGE
default    0          23m
```

Service Account and Permissions

Each Service Account in Kubernetes can be associated with certain permissions.

When Pod uses the service account, it can inherit the permissions.



Point to Note

The default service accounts in each namespace get no permissions by default other than the default API discovery permissions that Kubernetes grants to all authenticated principals if role-based access control (RBAC) is enabled

Assigning Pods to Service Accounts

If you deploy a Pod in a namespace, and you don't manually assign a ServiceAccount to the Pod, Kubernetes **assigns the default ServiceAccount** for that namespace to the Pod



Accessing Service Account Token

Service Account Token gets mounted inside the Pod inside the /var/run directory and can easily be accessed using cat command.

```
root@nginx:/# cat /var/run/secrets/kubernetes.io/serviceaccount/token
eyJhbGciOiJSUzIiNlIsImtpZCI6IiJLZWlNejJxdTh3NlY2SjhRZHvkSEZfMWxKazFoZmttb3JLNnNTNzTdxMifQ.eyJhdWQiolsic3lzdGVt0mtvbm5lY3Rpdm
l0eS1zZXJ2ZXIiXSwiZXhwIjoxNzI3NzU50TY2LCJpYXQiOjE2OTYyMjM5NjYsImlzcyI6Imh0dHBz0i8va3ViZXJuZXRlcyc5kZWZhdWx0LnN2Yy5jbHVzdGVyLmx
vY2FsIiwiia3ViZXJuZXRlcyc5pbI6eyJuYW1lc3BhY2Ui0iJkZWZhdWx0IiwickG9kIjp7Im5hbWUiOjJuZ2lueCIsInVpZCI6ImU3ZjUyYWY4LWM5MGUtNDY5Zi1i
MTg3LTc3MjliMmNmE5MyJ9LCJzZXJ2aWN1YWNjb3VudCI6eyJuYW1lIjoIZGVmYXVsdcIsInVpZCI6ImZlOTUzMTQ0LTThjNDYtNDF1My1iODFjLTMsMThkNTNhM
zAyOCJ9LCJ3YXJuYWZ0ZXIIoje2OTYyMjc1NzN9LCJuYmYiOjE2OTYyMjM5NjYsInN1Yi6In5c3R1bTpzZXJ2aWN1YWNjb3VudDpkZWZhdWx00mR1ZmF1bHQifQ
.i5jS2uEbNgj_1GA6LQh2YxQKEJsNjZbvoJh-Qjf-vH_f10w0KTvhmLPCL1UxCLfoI1NejT07Agckj3SFXYjEqmlvcbEYpLXt8eUjrHC8s6Ra105XGnpbt5uUy3GU
BYP4HnRE970zzU3HyU2gM14Kd8d8a9iiHb7yov82ph6moOPuuxCxBowNo5edWMWnNDWLKLNOFX168oxAmQo8ZQI5k37INIE1oN5N2bzp7Y40Gv3CURK1X904B0YuT
UN8gSYZsHaQaVq8FT-Q_xGfGQbvjqUzi0rRaKNc9QJ0BiIE3CKQN7PL6C0Lxyt_9LieJV438xPgwer2V_aNeHBWgU99oAroot@nginx:/#
```

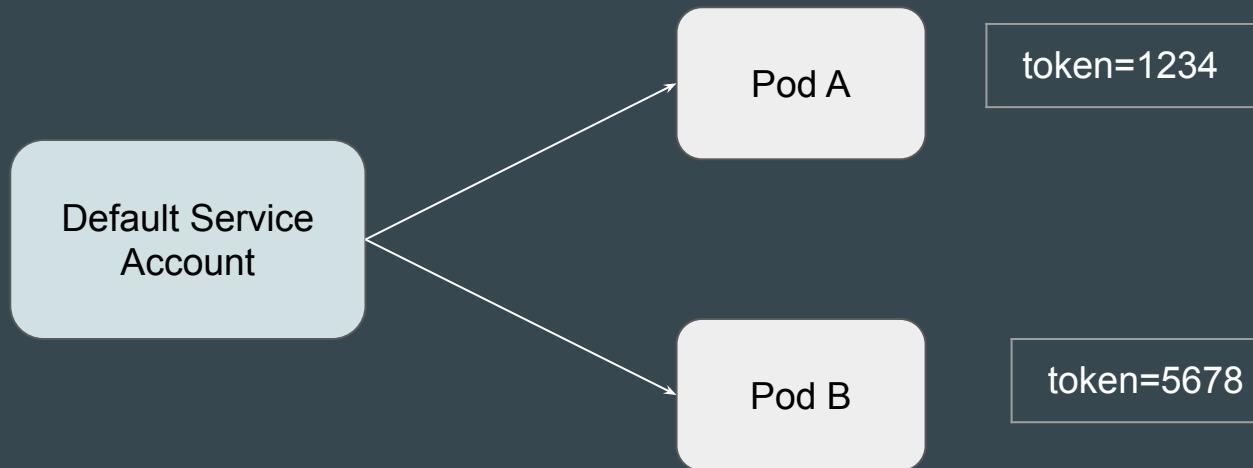
Connecting to K8s using Token

Using the Service Account Token, you can connect to the Kubernetes Cluster to perform operations.

```
root@nginx:/var/run/secrets/kubernetes.io/serviceaccount# curl -k -H "Authorization: Bearer $t" https://0f3570d8-03b7-45af-a  
f4-4c1b90504be3.k8s.ondigitalocean.com/api/v1  
{  
  "kind": "APIResourceList",  
  "groupVersion": "v1",  
  "resources": [  
    {  
      "name": "bindings",  
      "singularName": "binding",  
      "namespaced": true,  
      "kind": "Binding",  
      "verbs": [  
        "create"  
      ]  
    },  
  ],  
  "nonResourceURLs": []  
}
```

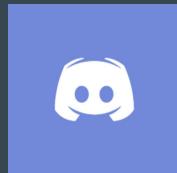
Points to Note

Even though 2 Pods use same service account, each Pod will receive different set of tokens.



Join us in our Adventure

Be Awesome



kplabs.in/chat



kplabs.in/linkedin

Service Accounts - Practical Scenarios

Creating Service Accounts

You can create new service account directly using kubectl.

```
C:\Users\zealv>kubectl create serviceaccount custom-token  
serviceaccount/custom-token created
```

Mounting Custom Service Account to Pod

You can use `serviceAccountName` to specify custom service account token as part of the Pod.

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  serviceAccountName: custom-token
```

Named Port

Networking Aspect

Named Port

We can also associate a name associated with the port

- This name must be unique within the pod.

```
spec:  
  containers:  
    - image: nginx  
      name: nginx  
      ports:  
        - containerPort: 80
```



```
spec:  
  containers:  
    - image: nginx  
      name: nginx  
      ports:  
        - containerPort: 80  
          name: http
```

Named Port Reference in Service File

Instead of Port Number, we can also specify the name while creating service.

```
kubectl expose pod nginx --port=80 --target-port=http --name "kplabs-svc"
```

```
spec:  
  ports:  
    - port: 80  
      protocol: TCP  
      targetPort: http  
  selector:  
    run: nginx  
  type: NodePort
```

Authentication

Kubernetes Authentication

Basics of Authentication

Authentication is the process or action of verifying the identity of a user or process.



kubectl delete pod database



K8s Cluster

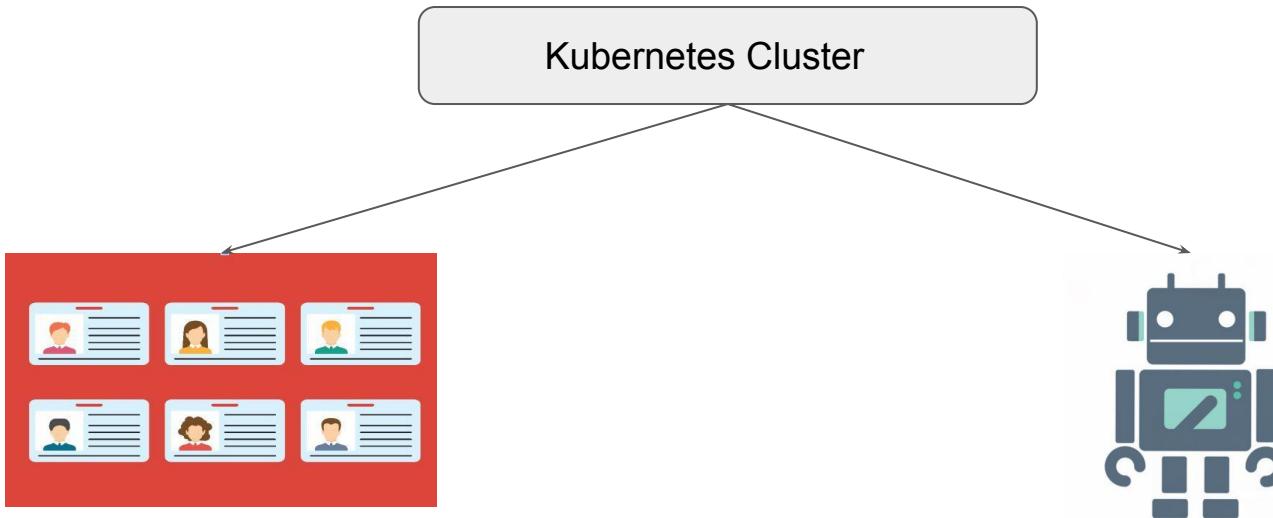


Who are you? Authenticate first.

Categories of Users

Kubernetes Clusters have two categories of users:

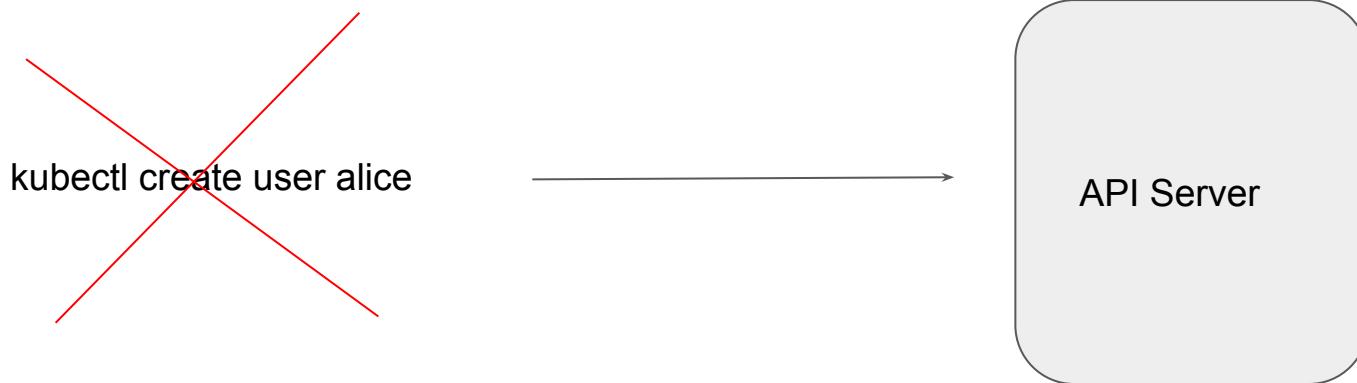
- Normal Users.
- Service Accounts



Important Aspect

Kubernetes does not have objects which represent normal user accounts.

Kubernetes does not manage the user accounts natively.



Authentication Strategies

There are multiple ways in which we can authenticate. Some of these include:

- Usernames / Passwords.
- Client Certificates
- Bearer Tokens
- etc

Let's take a demo for the same.

Authorization

Authorization Step

Basics of Authorization

Authorization is the process of giving someone permission to perform some operation.



New AWS User

Delete All the Servers



AWS Account

Authorization Modules

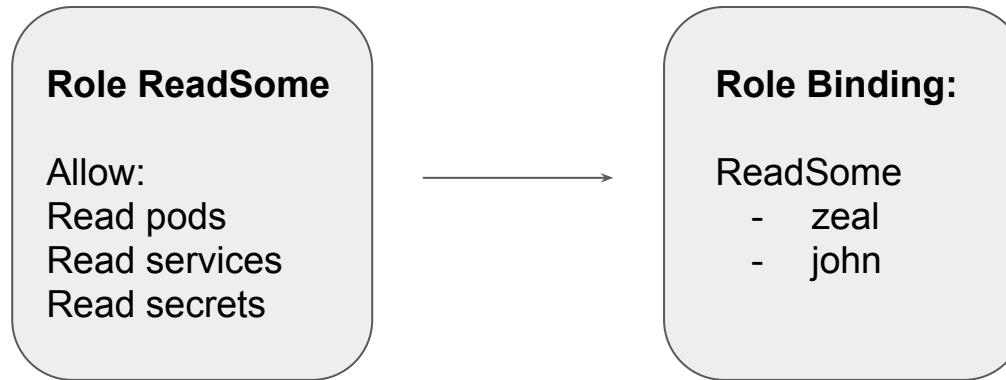
Kubernetes provides multiple authorization modules like:

- AlwaysAllow
- AlwaysDeny
- Attribute-Based Access Control (ABAC)
- **Role-based access control (RBAC)**
- Node
- WebHook

Two Concepts

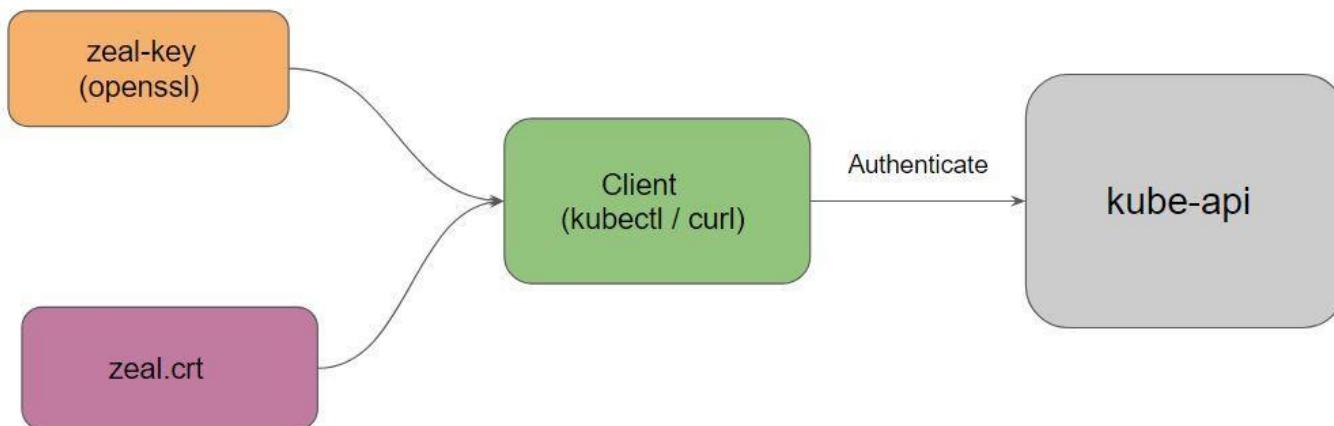
A **role** contains rules that represent a set of permissions

A **role binding** grants the permissions defined in a role to a user or set of users.



Understanding Authorization

By default, the user do not have any permission granted to them.



ClusterRole and ClusterRoleBinding

Security Aspect

Overview of ClusterRole

A Role can only be used to grant access to resources within a single namespace.

A ClusterRole can be used to grant the same permissions as a Role, but because they are cluster-scoped, they can also be used to grant access to:

- cluster-scoped resources (like nodes)
- namespaced resources (like pods) across all namespaces

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: default
  name: pod-reader
rules:
- apiGroups: []
  resources: ["pods"]
  verbs: ["get", "watch", "list", "create"]
```

Important Pointer

A RoleBinding may also reference a ClusterRole to grant the permissions to resources defined in the ClusterRole within the RoleBinding's namespace

This allows the administrator to have set of central policy which can be attached via RoleBinding so it is applicable at a per-namespace level.



Asymmetric Key Encryption

Right Architecture is the Key

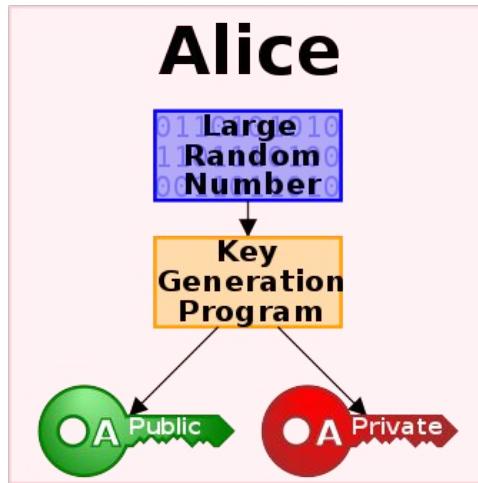
Overview of Asymmetric Key Encryption

Asymmetric cryptography, uses public and private keys to encrypt and decrypt data.

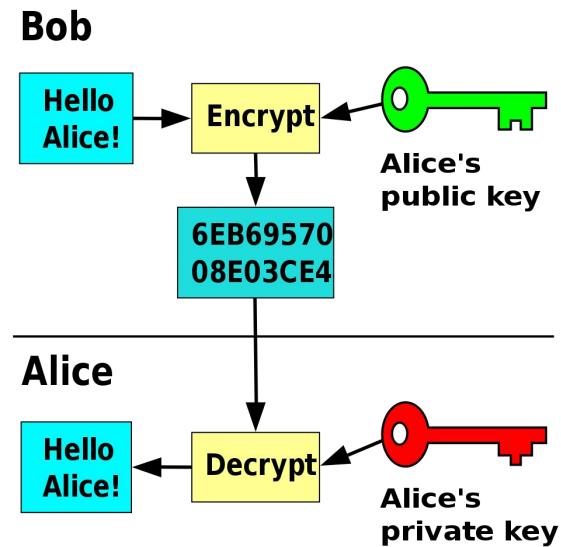
One key in the pair can be shared with everyone; it is called the public key. The other key in the pair is kept secret; it is called the private key.

Either of the keys can be used to encrypt a message; the opposite key from the one used to encrypt the message is used for decryption.

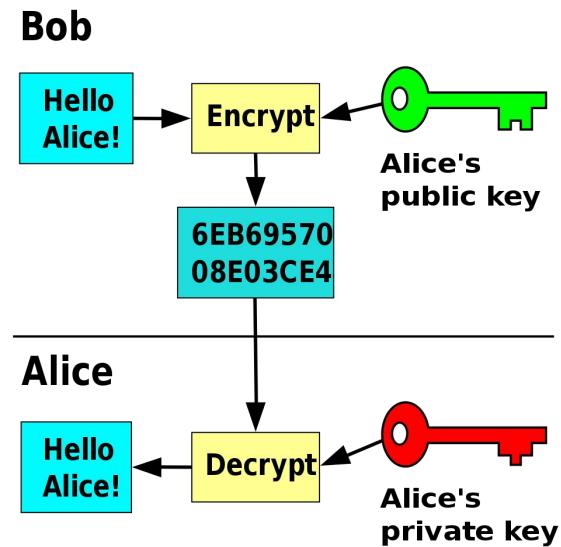
Step 1: Generation of Keys



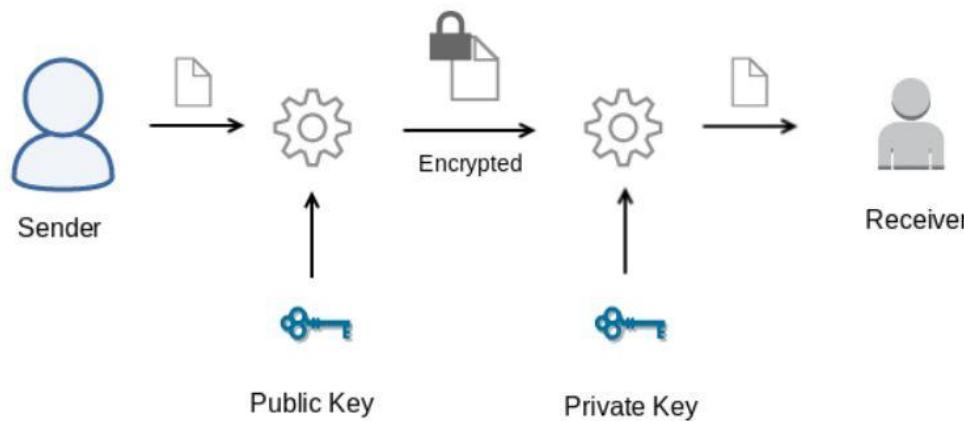
Step 2: Encryption and Decryption



Step 2: Encryption and Decryption

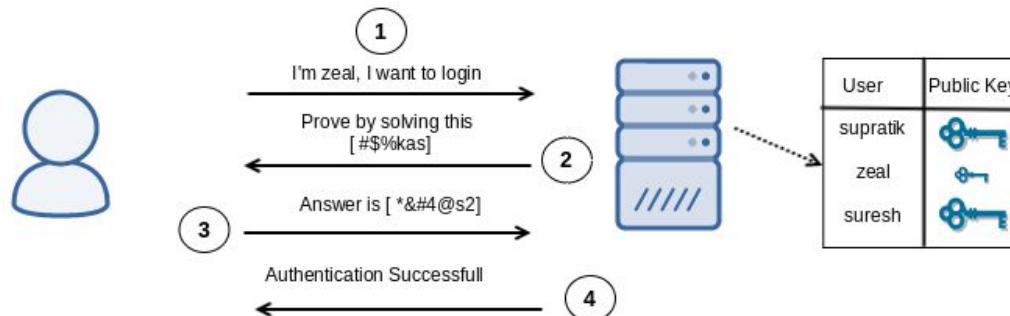


Step 2: Encryption and Decryption



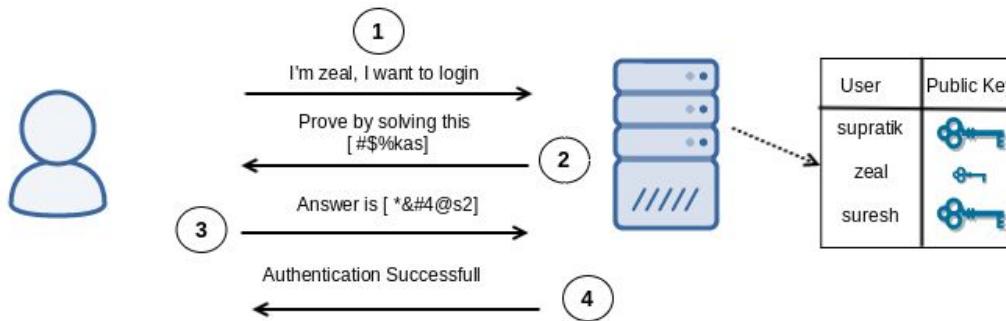
Use-Case of Asymmetric Key Encryption - Step 1

User zeal wants to log in to the server. Since the server uses a public key authentication, instead of taking the password from the user, the server will verify if the User claiming to be zeal actually holds the right private key.



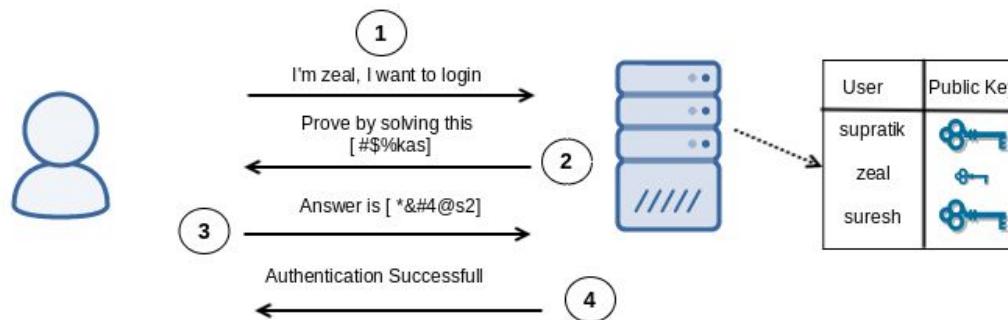
Use-Case of Asymmetric Key Encryption - Step 2

The server creates a simple challenge, $2+3=?$ and encrypts this challenge with the Public Key of the User and sends it back to the User. The challenge is sent in an encrypted format.



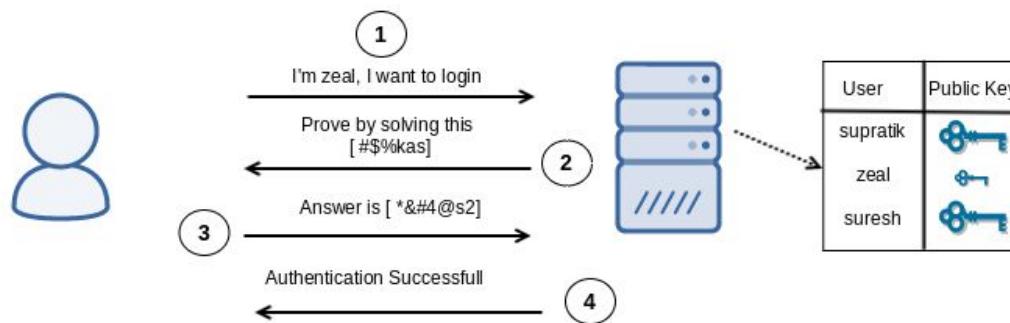
Use-Case of Asymmetric Key Encryption - Step 3

Since the user zeal holds the associated private key, he will be able to decrypt the message and compute the answer, which would be 5. Then, he will encrypt the message with the private key and send it back to the server.



Use-Case of Asymmetric Key Encryption - Step 4

The server decrypts the message with the user's Public Key and checks if the answer is correct. If yes, then the server will send an Authentication Successful message and the user will be able to log in.



Protocols

Because of the advantage that it offers, Asymmetric key encryption is used by variety of protocols.

Some of these include:

- PGP
- SSH
- Bitcoin
- TLS
- S/MIME

HTTPS

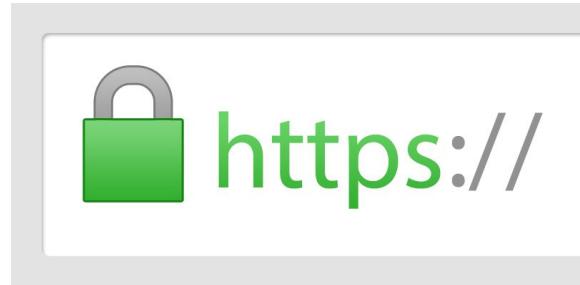
Secure Communication

Overview of HTTPS

HTTPS is an extension of HTTP.

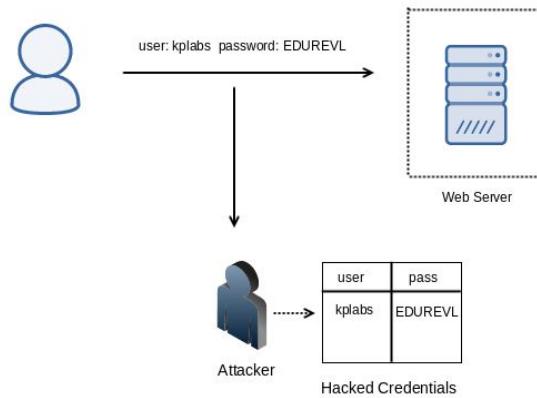
In HTTPS, the communication is encrypted using Transport Layer Security (TLS)

The protocol is therefore also often referred to as HTTP over TLS or HTTP over SSL.



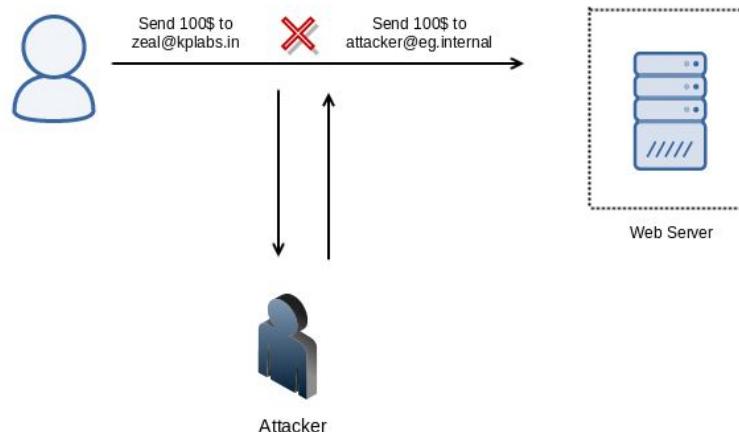
Scenario 1: MITM Attacks

- User is sending their username and password in plaintext to a Web Server for authentication over a network.
- There is an Attacker sitting between them doing a MITM attack and storing all the credentials he finds over the network to a file:



Scenario 2: MITM & Integrity Attacks

- Attacker changing the payment details while packets are in transit.



Introduction to SSL/TLS

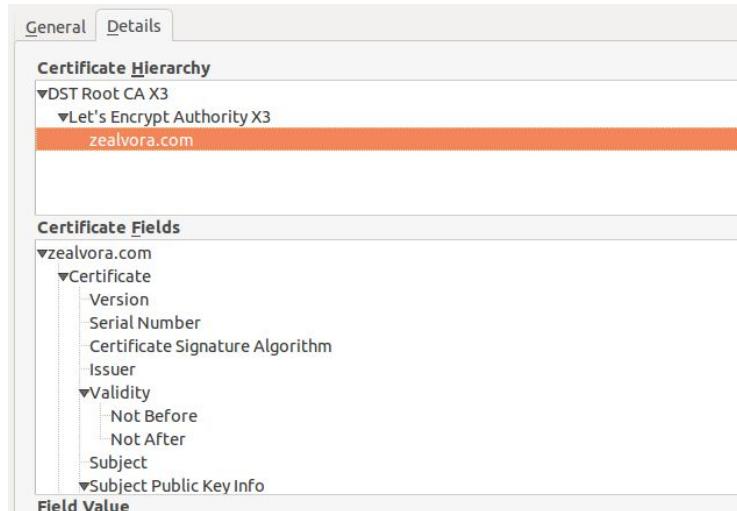
To avoid the previous two scenarios (and many more), various cryptographic standards were clubbed together to establish a secure communication over an untrusted network and they were known as SSL/TLS.

Protocol	Year
SSL 2.0	1995
SSL 3.0	1996
TLS 1.0	1999
TLS 1.1	2006
TLS 1.2	2008
TLS 1.3	2018

Understanding it in easy way

Every website has a certificate (like a passport which is issued by a trusted entity).

Certificate has lot of details like domain name it is valid for, the public key, validity and others.



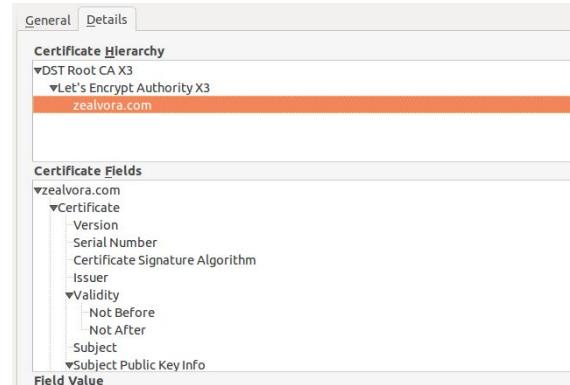
Understanding it in easy way

Browser (clients) verifies if it trusts the certificate issuer.

It will verify all the details of the certificate.

It will take the public key and initiate a negotiation.

Asymmetric key encryption is used to generate a new temporary symmetric key which will be used for secure communication.



Web Server Configuration

```
server {
    listen      80;
    server_name zealvora.com;
    return      301 https://$server_name$request_uri;
}

server {
    server_name zealvora.com;
    listen 443 default ssl;
    server_name zealvora.com;
    ssl_certificate /etc/letsencrypt/archive/zealvora.com/fullchain1.pem;
    ssl_certificate_key /etc/letsencrypt/archive/zealvora.com/privkey1.pem;

    location / {
        root /websites/zealvora/;
        include location-php;
        index index.php;
    }
    location ~ /.well-known {
        allow all;
    }
}
```

Certificate Based Auth

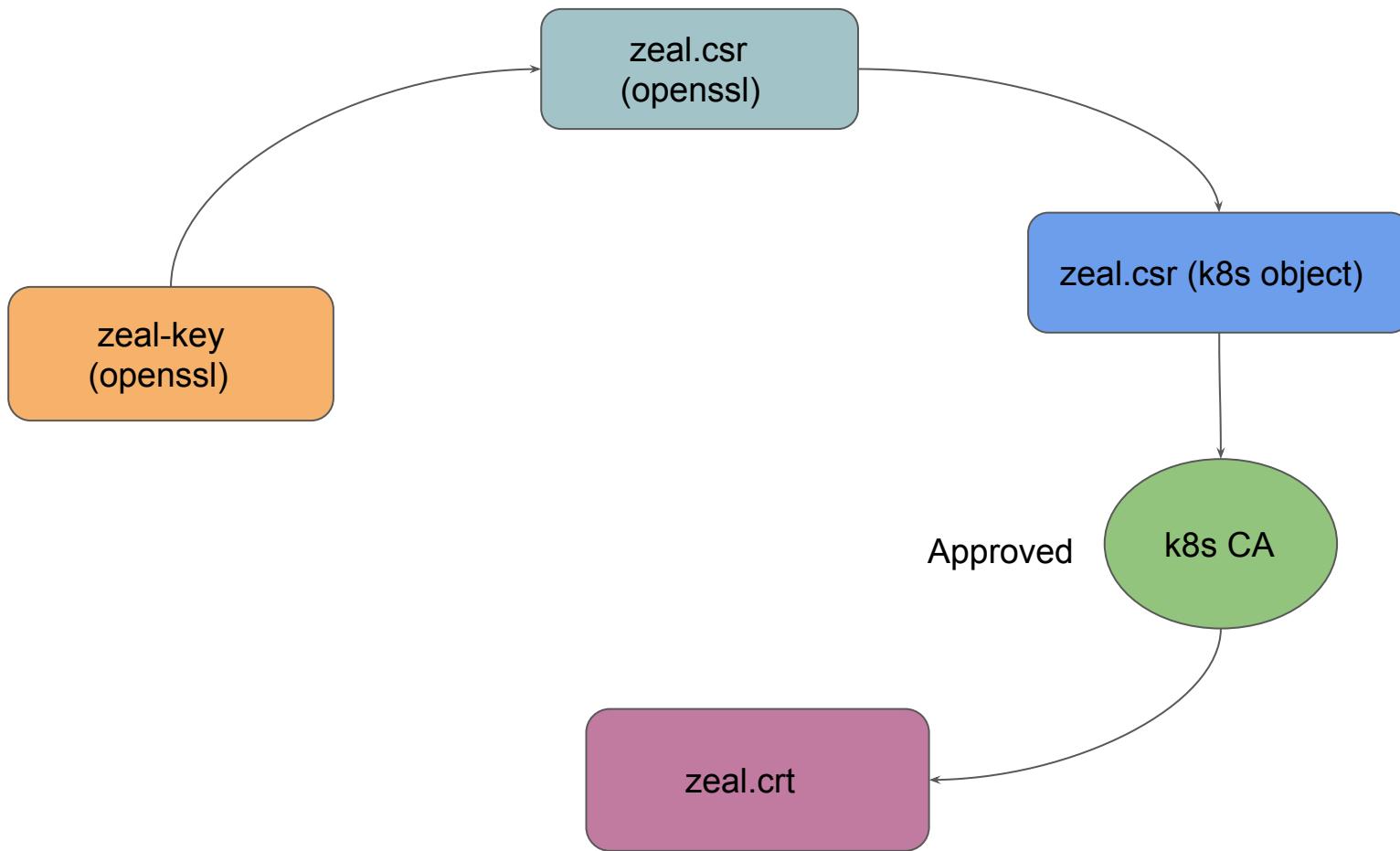
Kubernetes Authentication

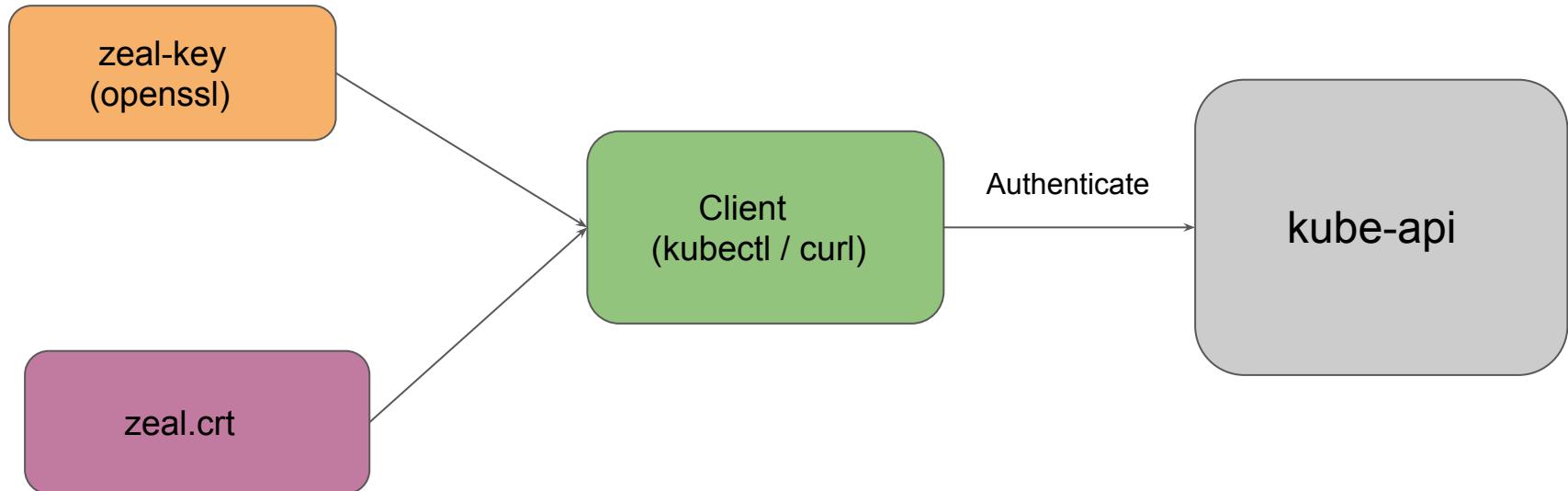
Our Goal for Today

Our goal is to setup a authentication based on X509 Client certificates.

User A should be able to authenticate with Kubernetes Cluster with his certificates.





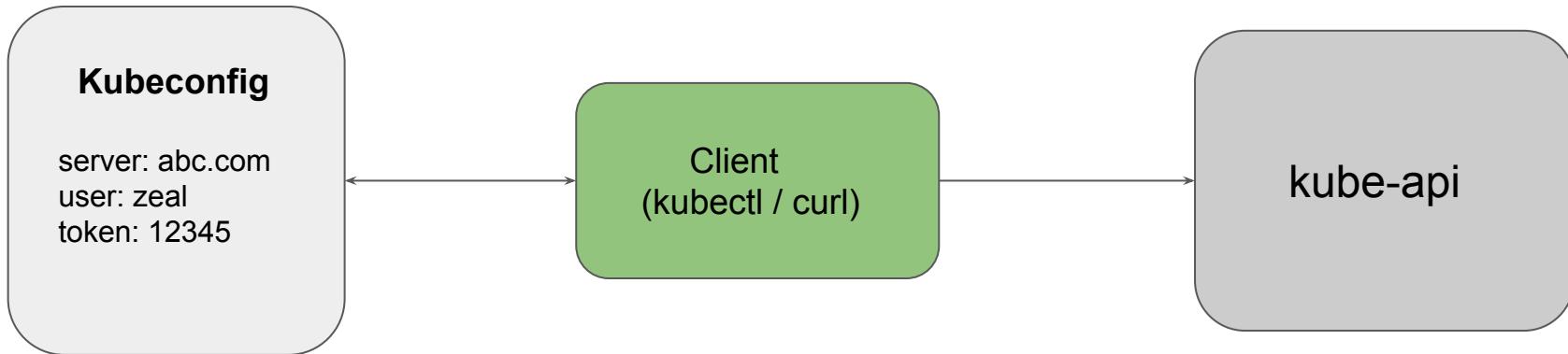


Kubeconfig

Security Aspect

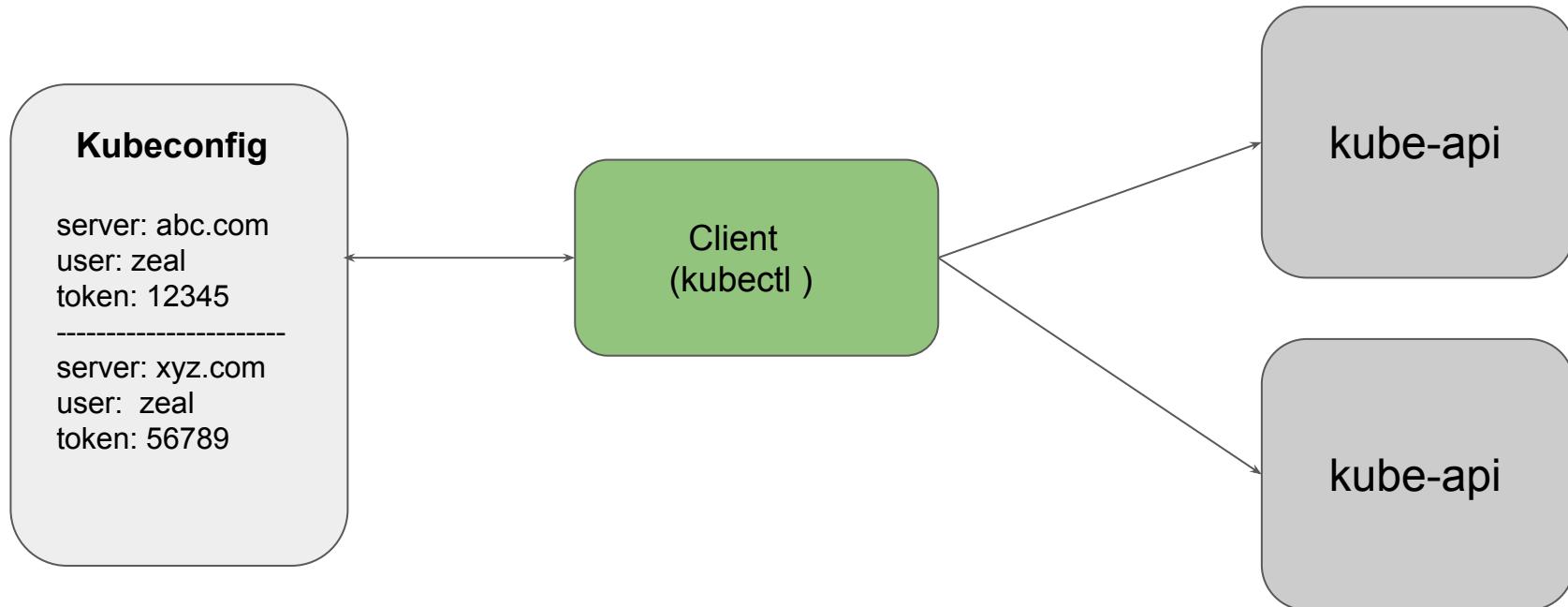
Understanding Kubeconfig

Kubectl command uses kubeconfig files to find the information about the cluster, username, authentication mechanisms and others.



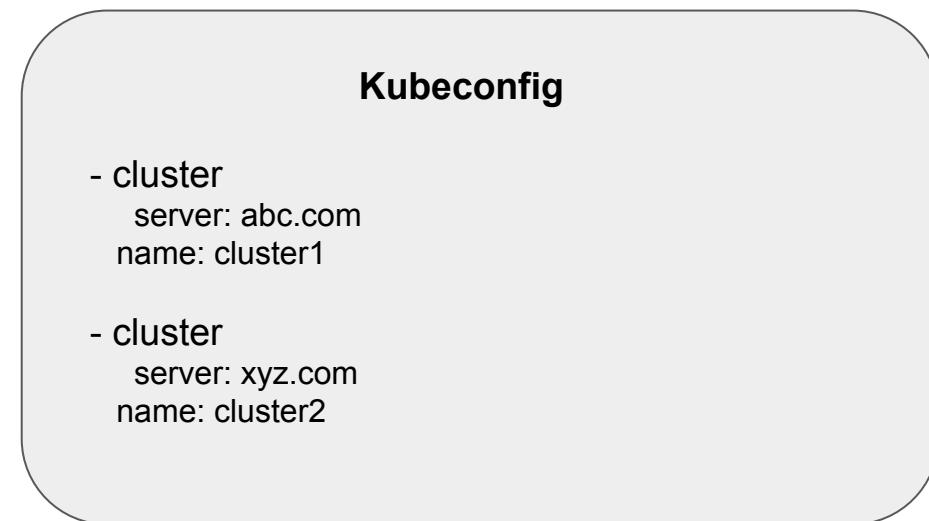
Access to Multiple Clusters

Kubeconfig file can also have details associated with multiple kubernetes clusters.



Step 1 - Understanding Clusters Field

Cluster field has details related to the URL of your Kubernetes Cluster and it's associated information.



cluster1

cluster2

Step 2 - Users Field

User field contains authentication specific information like username, passwords.

There can be different type of authentication mechanisms (user/pass, certificates, tokens etc)



Step 3 - Context Field

Context groups information related to cluster, user and namespace.

Kubeconfig

```
- cluster
  server: abc.com
  name: cluster1

- context:
  cluster: cluster1
  namespace: kplabs
  user: adminuser

users:
- name: adminuser
  user:
    token: 12345
```

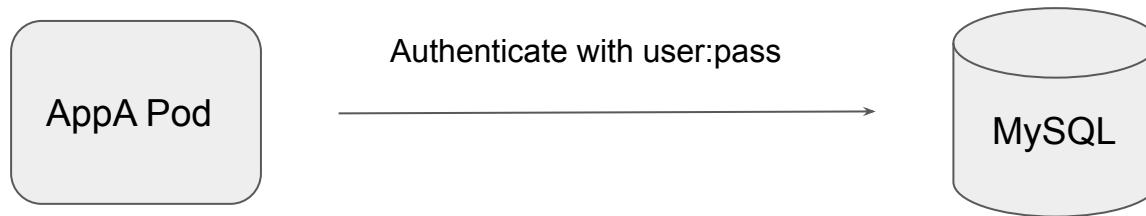
Kubernetes Secrets

Security Aspect

Let's understand the challenge

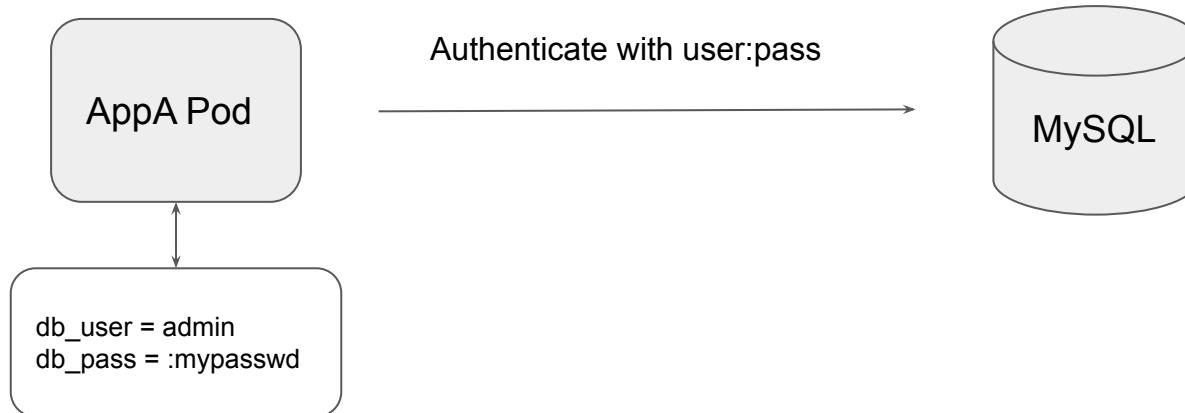
Use-Case:

AppA runs on a pod and it needs to connect to a MySQL DB to start working properly.



Common Approach: Hardcoding Credentials

Many a times you will find that developer have hardcoded credentials within their container image.



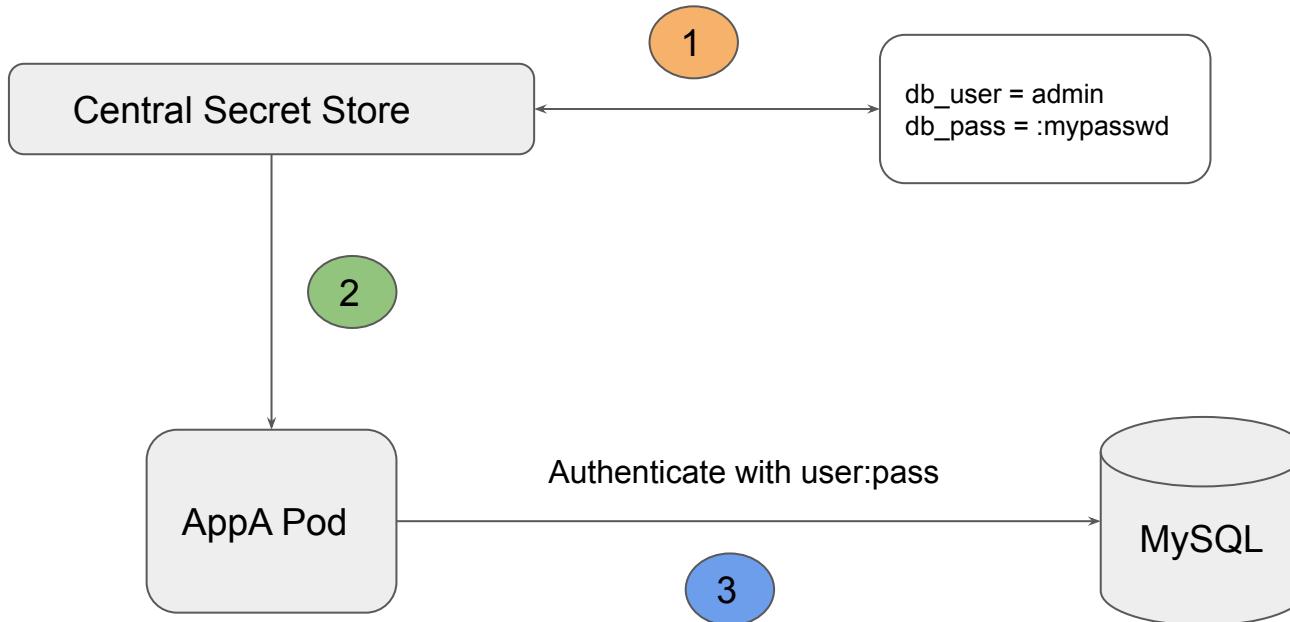
Common Approach: Risks

There are multiple risks of hard coding credentials:

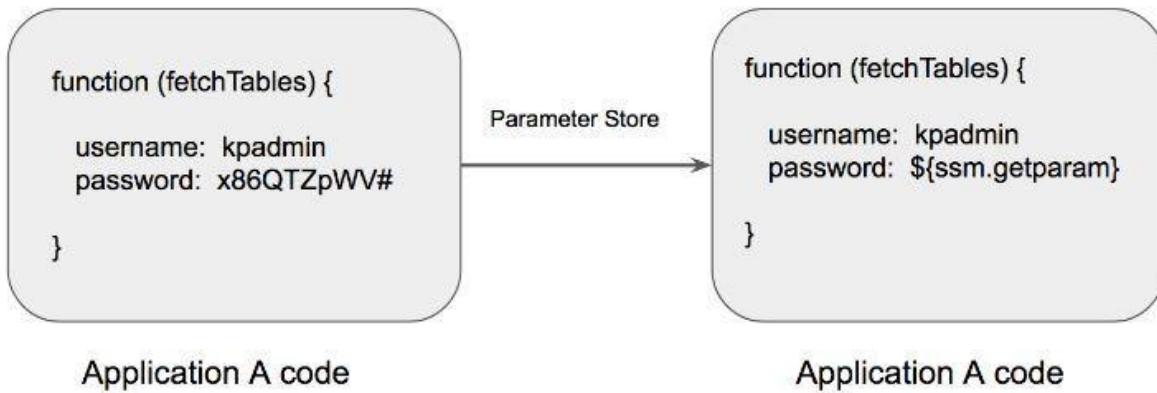
1. Anyone having access to the container repository can easily fetch the credentials.
2. Developer needs to have credentials of production systems.
3. Update of credentials will lead to new docker image being built.



Storing Credentials Centrally



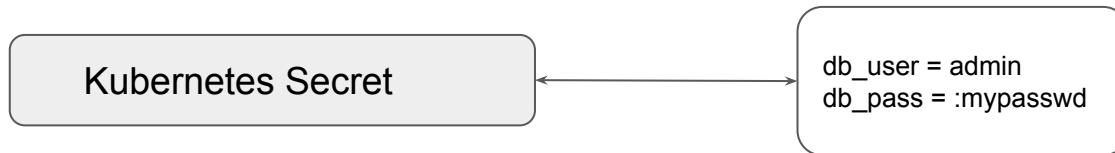
AppA Code Needs to be Modified



Overview of Kubernetes Secrets

A Secret is an object that contains a small amount of sensitive data such as a password, a token, or a key.

- Allows customers to store secrets centrally to reduce risk of exposure.
- Stored in ETCD database.



CLI Syntax for Creating Secret

```
kubectl create secret [TYPE] [NAME] [DATA]
```

Elaborating Type:

i) Generic:

- File (--from-file)
- directory
- literal value

ii) Docker Registry

iii) TLS

Mounting Secrets in Containers

Security Aspect

Mounting Secrets in Containers

Once a secret is created, it is necessary to make it available to containers in a pod.

There are two approaches to achieve this:

1. Volumes
2. Environment Variables.

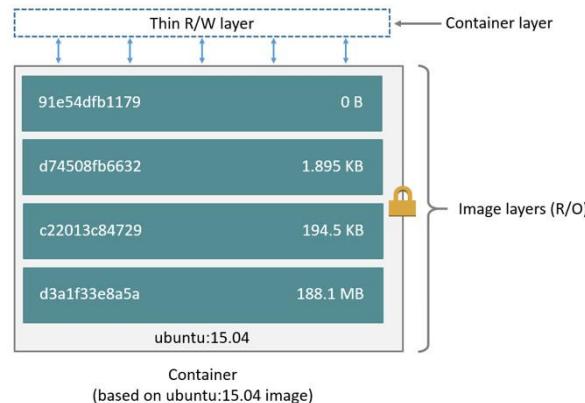
Docker Volumes

Build once, use anywhere

Challenges with files in Container Writable Layer

By default all files created inside a container are stored on a writable container layer. This means that:

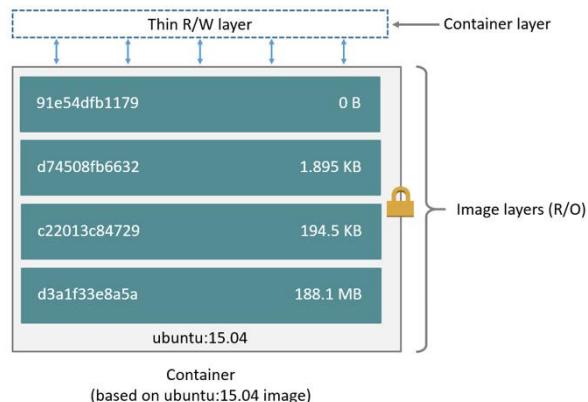
The data doesn't persist when that container no longer exists, and it can be difficult to get the data out of the container if another process needs it.



Challenges with files in Container Writable Layer - Part 2

Writing into a container's writable layer requires a storage driver to manage the filesystem. The storage driver provides a union filesystem, using the Linux kernel.

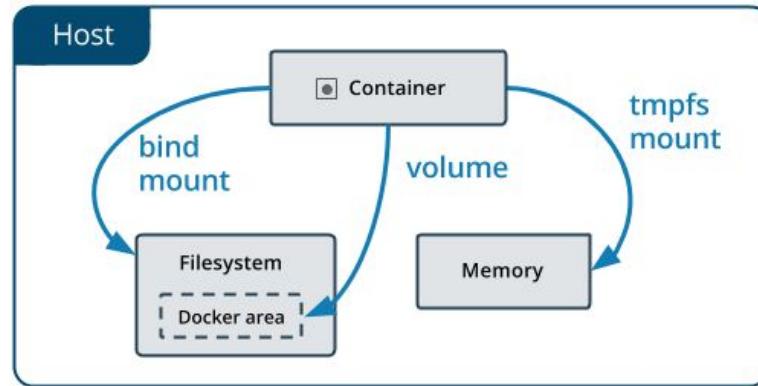
This extra abstraction reduces performance as compared to using data volumes, which write directly to the host filesystem.



Ideal Approach for Persistent Data

Docker has two options for containers to store files in the host machine, so that the files are persisted even after the container stops: volumes, and bind mounts.

If you're running Docker on Linux you can also use a tmpfs mount.



Important Pointers to Remember

A given volume can be mounted into multiple containers simultaneously.

When no running container is using a volume, the volume is still available to Docker and is not removed automatically.

When you mount a volume, it may be named or anonymous. Anonymous volumes are not given an explicit name when they are first mounted into a container, so Docker gives them a random name that is guaranteed to be unique within a given Docker host.

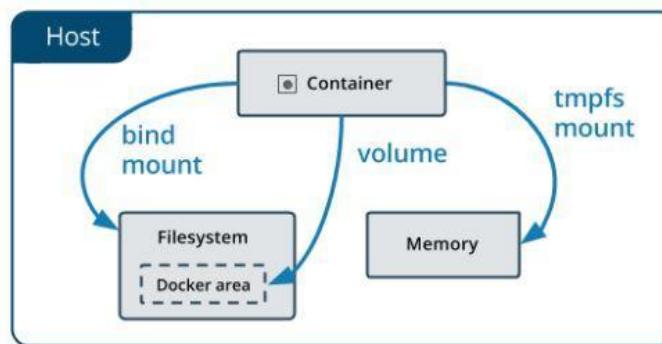
Volumes in Kubernetes

Security Aspect

Two Challenges

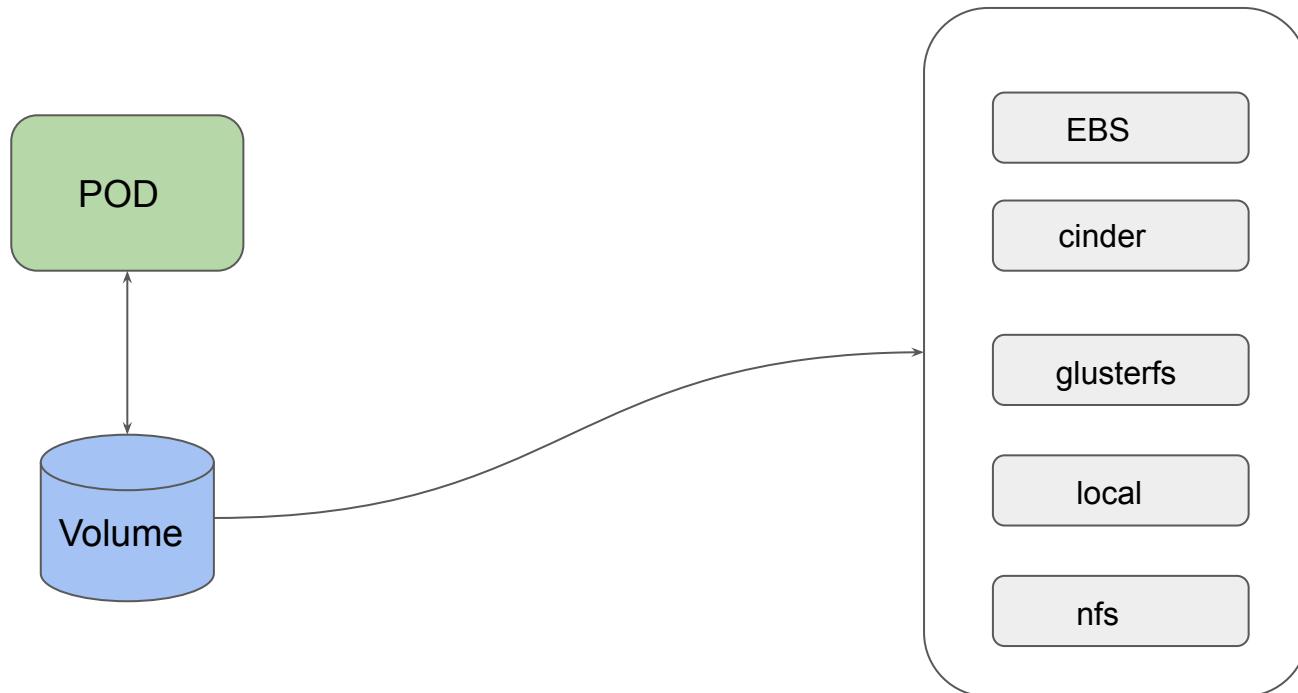
On-disk files in a Container are ephemeral.

When there are multiple containers who wants to share same data, it becomes a challenge.



Volumes in Kubernetes

One of the benefits of Kubernetes is that it supports multiple types of volumes.



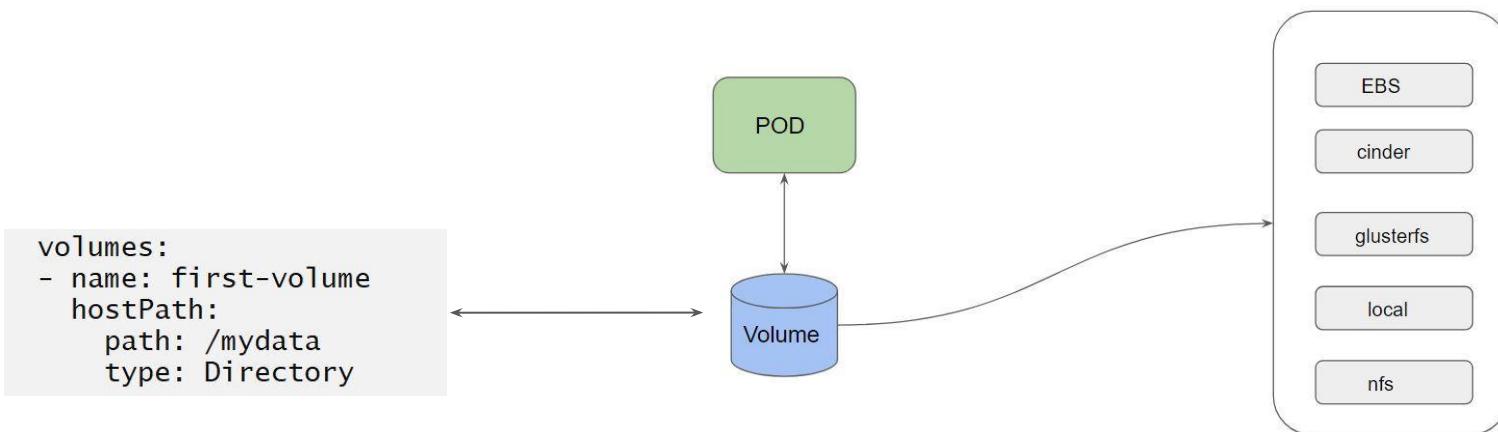
PersistentVolume and PersistentVolumeClaim

Storage Aspect

Volumes in Kubernetes

Generally developers creates the YAML files associated with pods that they want to deploy.

In a scenario of directly referencing to volumes, developer need to be aware of configurations



Understanding the Challenge



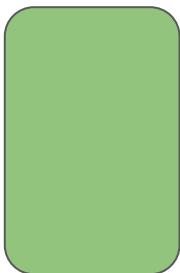
I am a Developer. I just want to write my code and pod.yaml file. I don't want to take care of storage provisioning, and its configurations.



I am a Storage Administrator. I can take care of provisioning storage and its associated configurations. Developers can reference the storage within their podspec.

Persistent Volumes

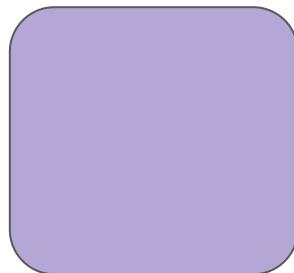
A PersistentVolume (PV) is a piece of storage in the cluster that has been provisioned by an administrator or dynamically provisioned using Storage Classes



Volume 1
Size: Small
Speed: Fast



Volume 2
Size: Medium
Speed: Fast



Volume 3
Size: Big
Speed: Slow



Volume 4
Size: VSmall
Speed: Ultra Fast



Volume 5
Size: Very Big
Speed: Very Slow

Different Types of Persistent Volumes

- Every Volume which is created can be of different type.
- This can be taken care by the Storage Administrator / Ops Team



Volume 1
Size: Small
Speed: Fast



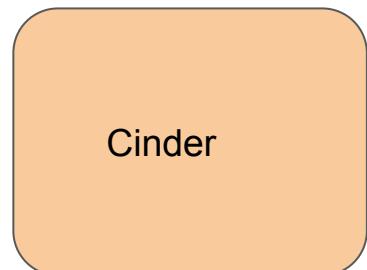
Volume 2
Size: Medium
Speed: Fast



Volume 3
Size: Big
Speed: Slow



Volume 4
Size: VSmall
Speed: Ultra Fast



Volume 5
Size: Very Big
Speed: Very Slow

PersistentVolumeClaim

A PersistentVolumeClaim is a request for the storage by a user.

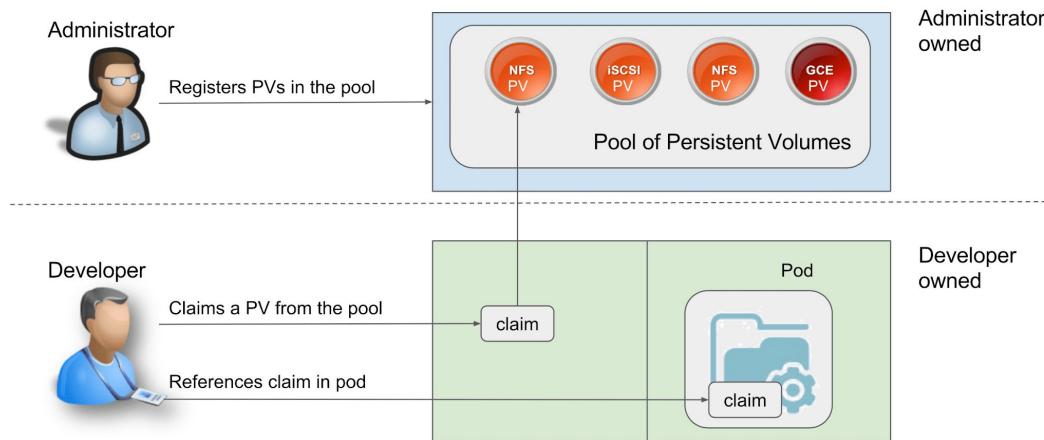
Within the claim, user need to specify the size of the volume along with access mode.

Developer:

I want a volume of size 10 GB which is has speed of Fast for my pod.

Overall Working Steps

1. Storage Administrator takes care of creating PV.
2. Developer can raise a “Claim” (I want a specific type of PV).
3. Reference that claim within the PodSpec file.



Static vs Dynamic Provisioning

PersistentVolume Aspect

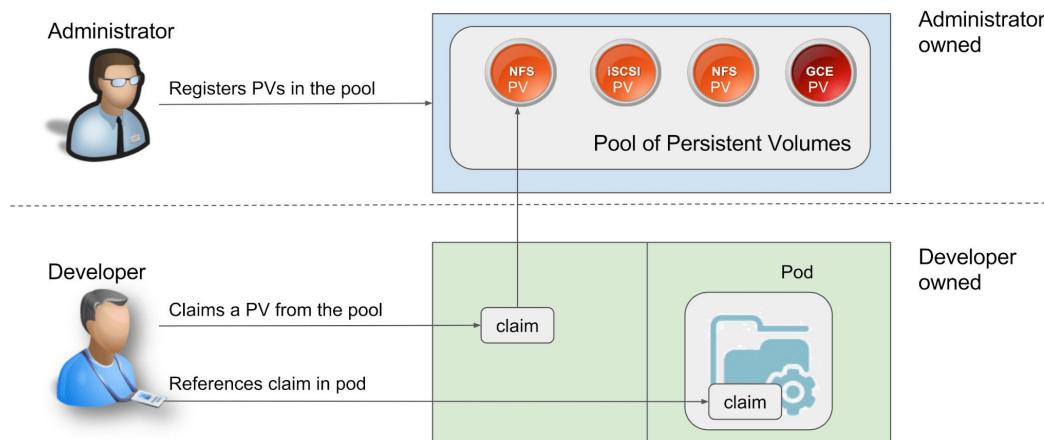
Types of PV Provisioning

There are two ways PVs may be provisioned: statically or dynamically.

Types of PV Provisioning	Description
Static	A cluster administrator creates a number of PVs.
Dynamic	When none of the static PVs the administrator created matches a user's PersistentVolumeClaim, the cluster may try to dynamically provision a volume specially for the PVC.

Overall Working Steps

1. Storage Administrator takes care of creating PV.
2. Developer can raise a “Claim” (I want a specific type of PV).
3. Reference that claim within the PodSpec file.



ConfigMaps

Storing Configurations Centrally

Sample Use-Case

We have an AppA container and depending on the environment, its settings needs to be differed.

Example Properties:

Dev Environment: app.env=dev app.mem=2048m app.properties=dev.env.url

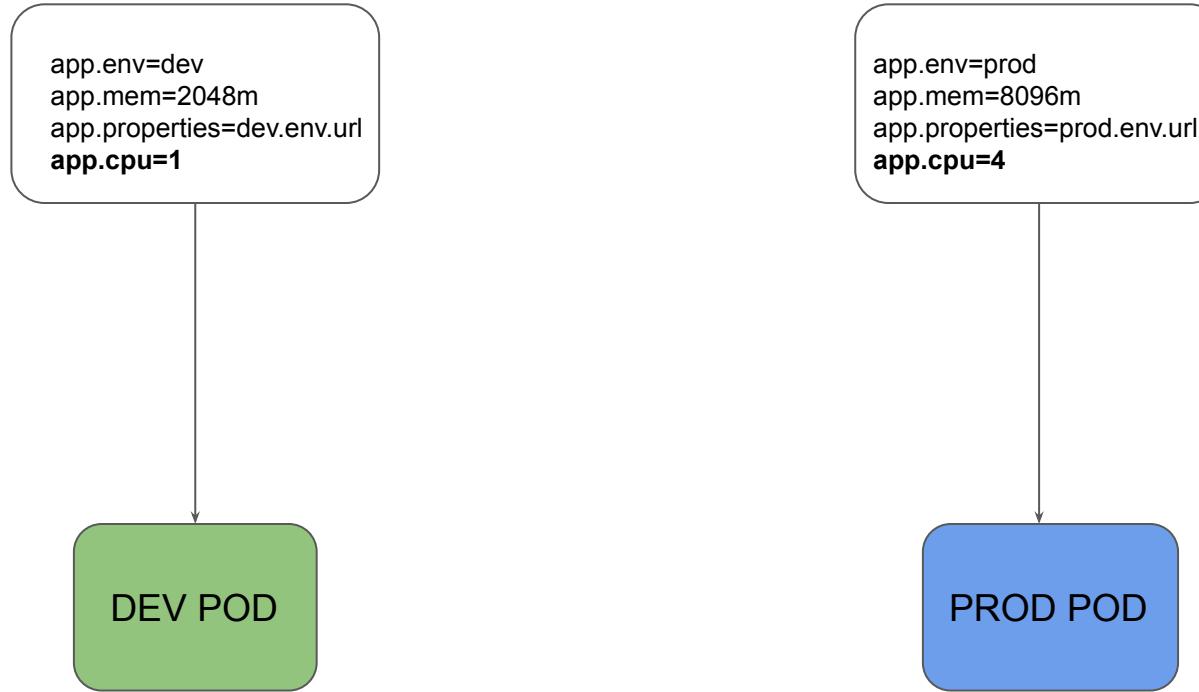
Prod Environment: app.env=prod app.mem=8096m app.properties=prod.env.url



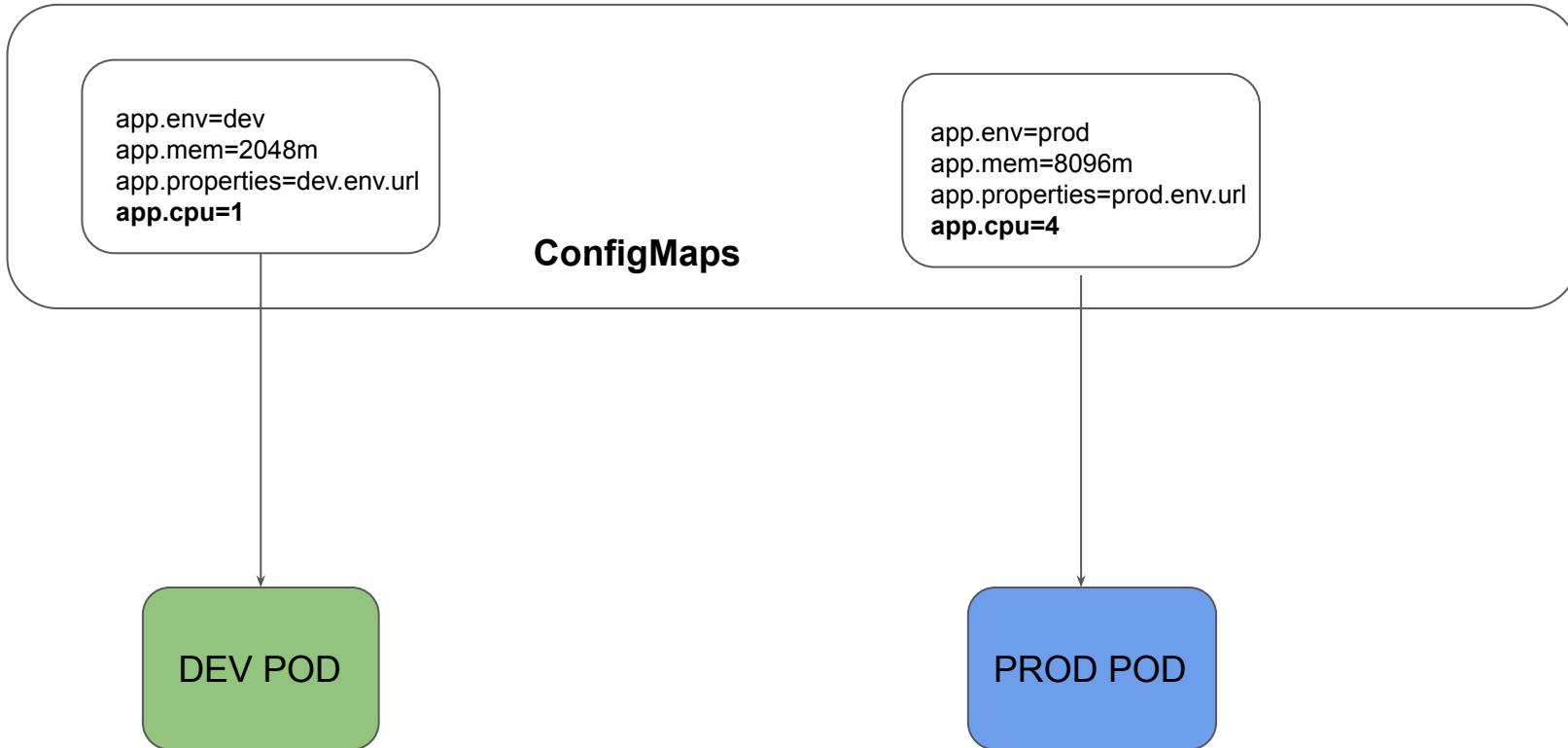
Centrally Store Data



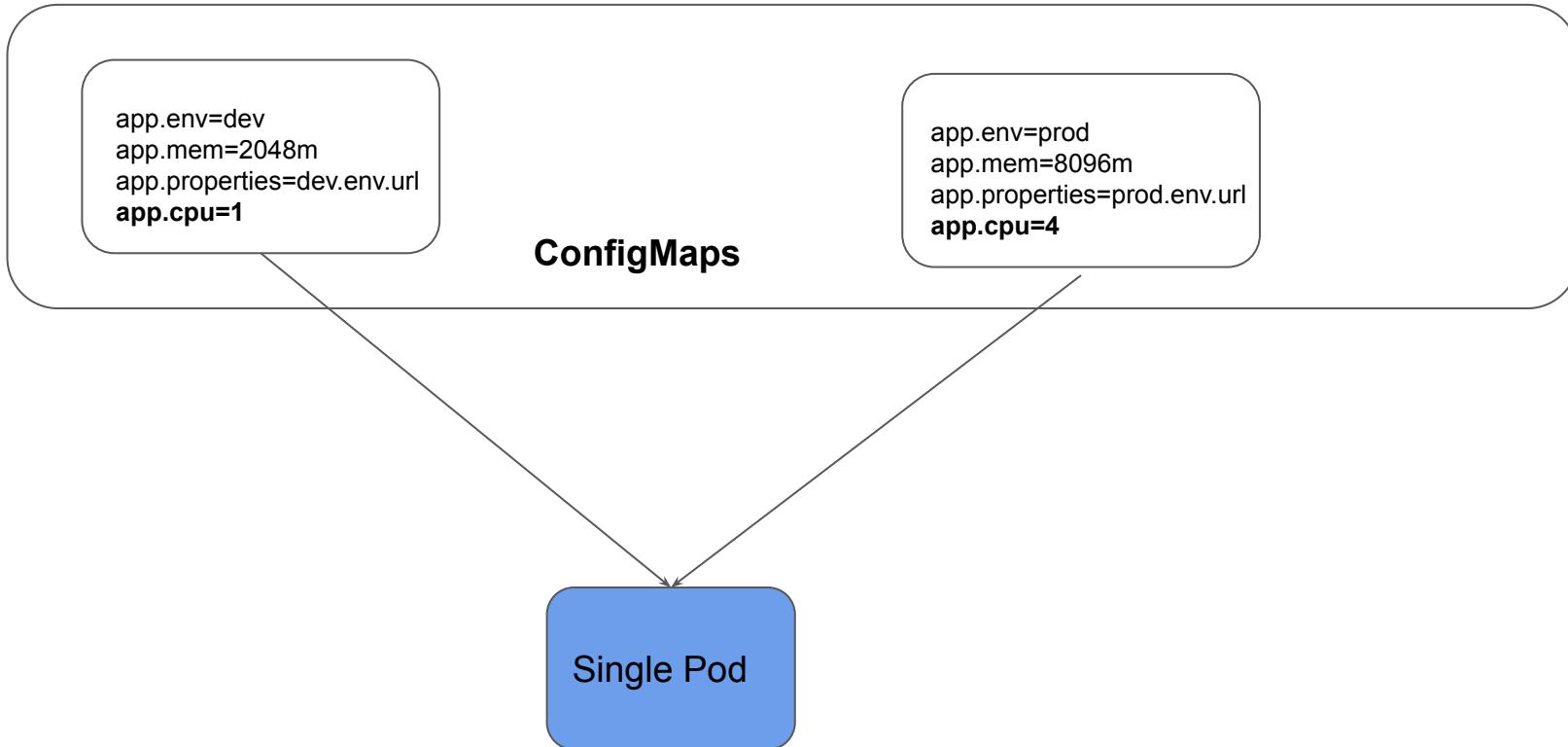
Dynamically Change the Values



ConfigMaps



ConfigMaps



CLI Syntax for Creating ConfigMap

```
kubectl create configmap [NAME] [DATA-SOURCE]
```

- File
- directory
- literal value

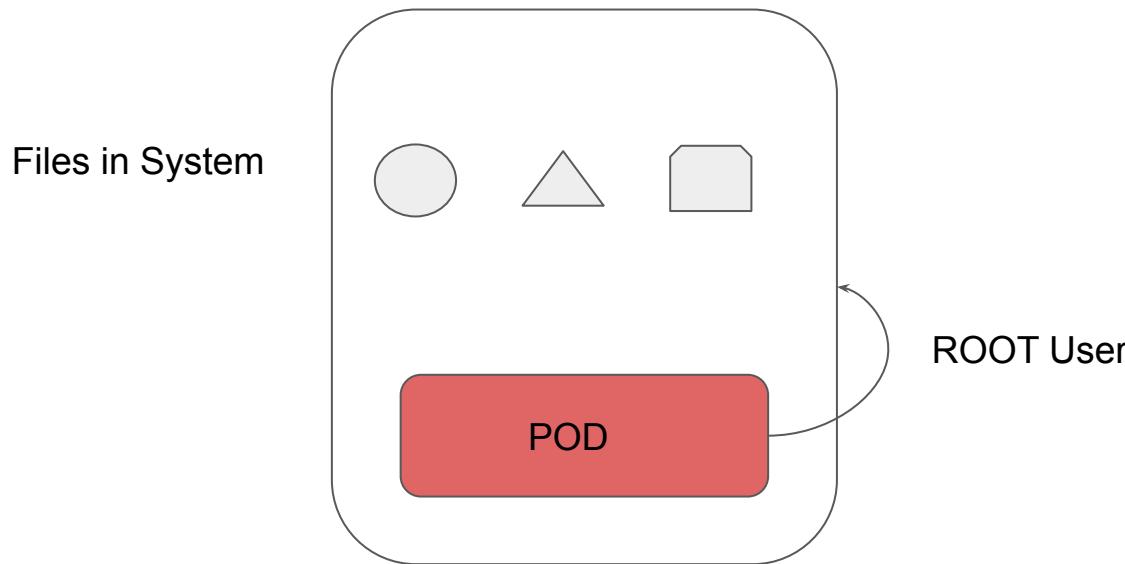
Security Context

Security Aspect

Let's understand the challenge

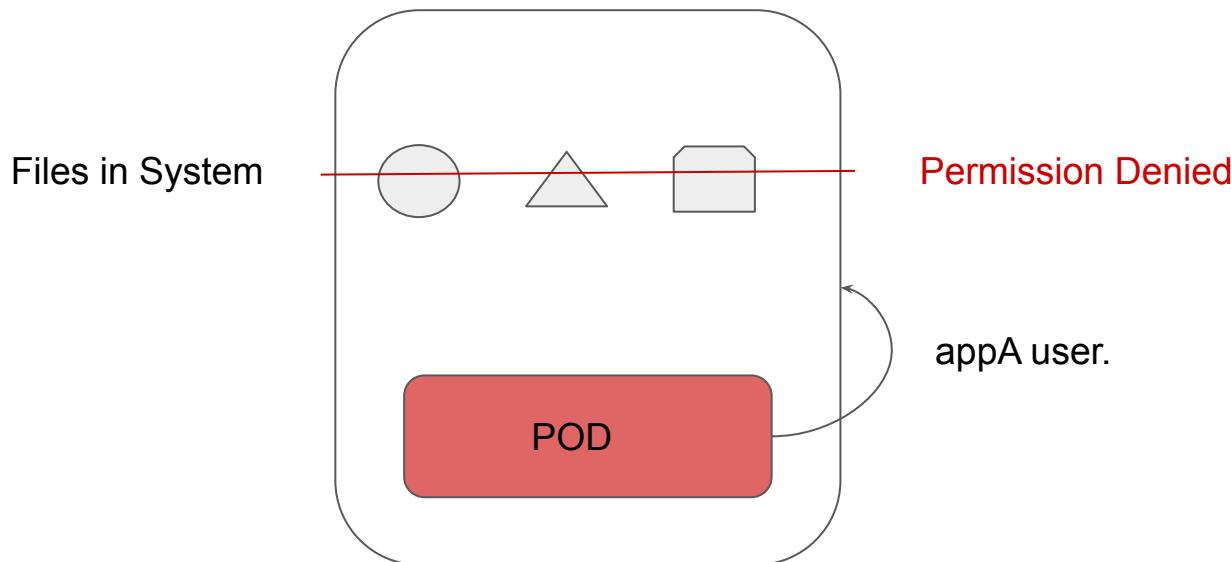
When you run a container, it runs with the UID 0 (Administrative Privilege)

In-case of container breakouts, attacker can get root privileges to your entire system.



Running with Least Privilege User

We can run POD and container with limited privilege user instead of the ROOT user.



Three Important Permission Aspects

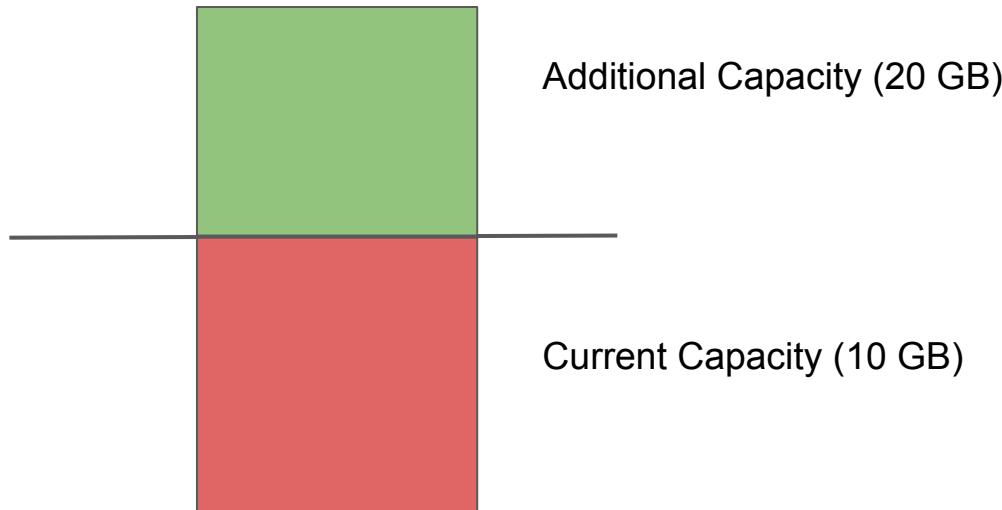
SecurityContext	Description
runAsUser	Specifies the user of the running process in containers.
runAsGroup	Specifies the primary group for all process within containers.
fsGroup	Applies the settings to the volumes. Volumes which support ownership management are modified to be owned and writable by the GID specified in fsGroup

Volume Expansion in K8s

Mastering K8s

Understanding the Challenge

It can happen that your persistent volume has become full and you need to expand the storage for additional capacity.



Step 1 - Enable Volume Expansion

1. Enabling Volume Expansion in the Storage Class.

```
bash-4.2# kubectl describe storageclass do-block-storage
Name:           do-block-storage
IsDefaultClass: Yes
Annotations:   kubectl.kubernetes.io/last-applied-configuration={"allowVolumeExpansion":true,"apiVersion":"storage.k
8s.io/v1","kind":"StorageClass","metadata":{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"},"name
":"do-block-storage"},"provisioner":"dobs.csi.digitalocean.com","reclaimPolicy":"Delete"}
               ,storageclass.kubernetes.io/is-default-class=true
Provisioner:    dobs.csi.digitalocean.com
Parameters:     <none>
AllowVolumeExpansion: True
MountOptions:   <none>
ReclaimPolicy: Delete
VolumeBindingMode: Immediate
Events:        <none>
```

Step 2 - Resizing the PVC

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"PersistentVolumeClaim",
       "metadata":{"name":"csi-pvc","namespace":"default","uid":2ec4833a-bdb7-49b1-8c32-6b73aaafea04},
       "spec":{"accessModes":["ReadWriteOnce"],"resources":{"requests":{"storage":15Gi}}}}
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: dobs.
  creationTimestamp: "2020-08-30T07:08:20Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: csi-pvc
  namespace: default
  resourceVersion: "2154"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/csi-pvc
  uid: 2ec4833a-bdb7-49b1-8c32-6b73aaafea04
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 15Gi
```

Step 3 - Restart the POD

Once PVC object is modified, you will have to restart the POD.

```
kubectl delete pod [POD-NAME]
```

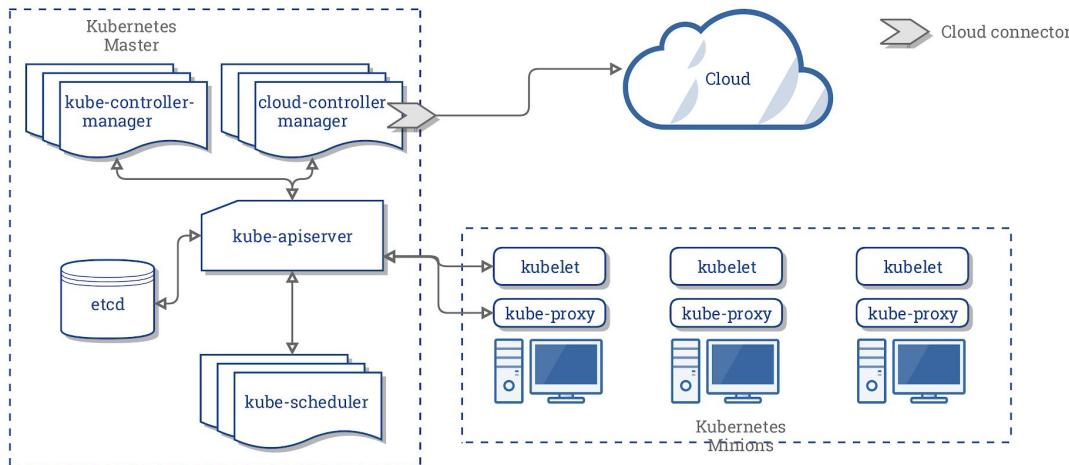
```
kubectl apply -f pod-manifest.yaml
```

Provisioning Kubernetes Cluster with kubeadm

Setting K8s Cluster

Overview of kubeadm

Kubeadm allows us to quickly provision a secure Kubernetes cluster.



Upgrading kubeadm Clusters

Upgrading K8s Clusters

High-Level Steps

Following are the high-level steps required to perform kubeadm upgrade.

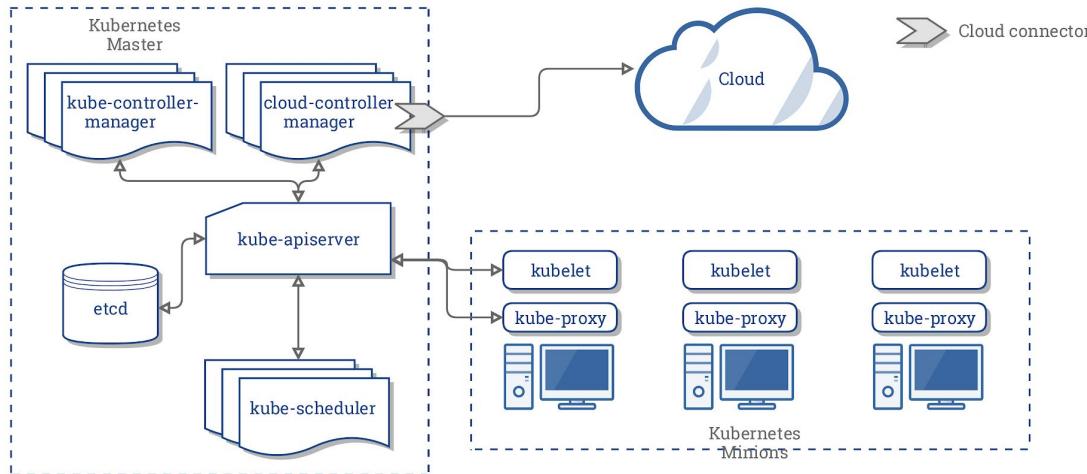
1. Find the latest version to upgrade to.
2. Unhold the kubeadm binary
3. Download the latest kubeadm binary (apt-get install)
4. Perform the Upgrade Plan
5. Choose version to upgrade and run appropriate command
6. Upgrade the kubelet and kubectl binaries
7. Restart service

Kubernetes Cluster from Scratch

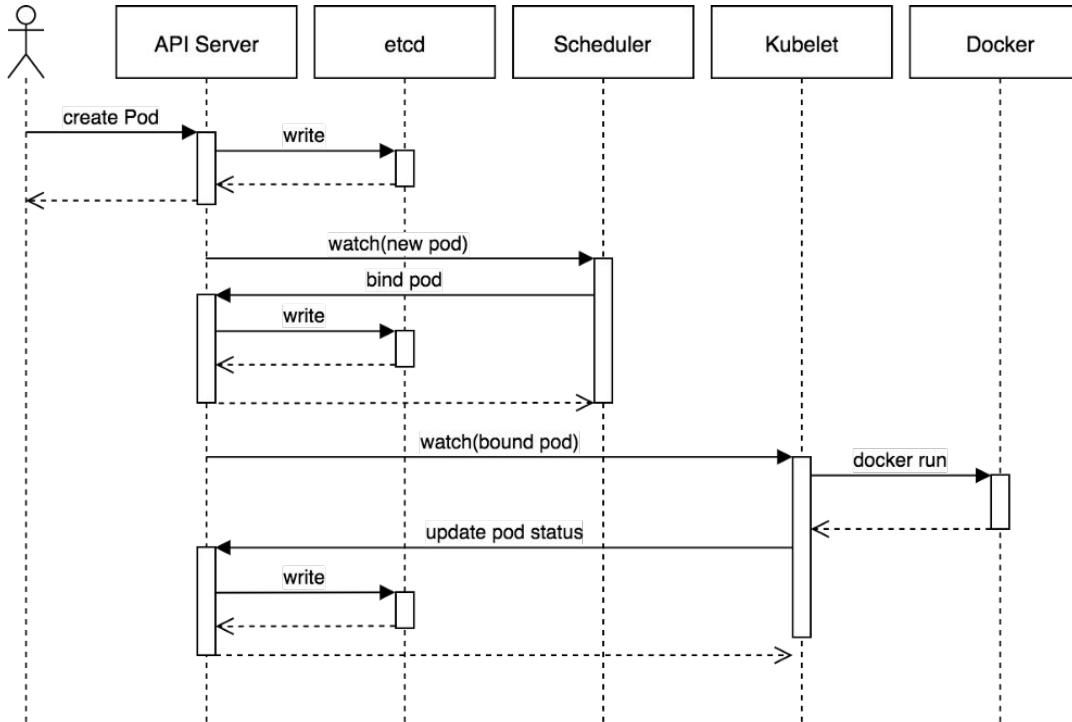
Let's Learn!

Goal of The Section

We will be configuring Kubernetes Master and Kubernetes Worker Node from Scratch.

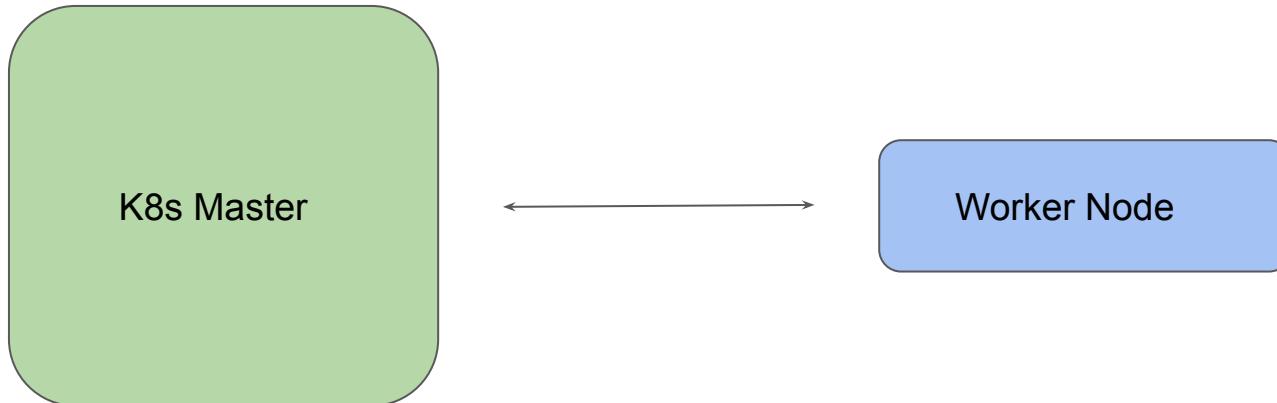


Remember the Workflow



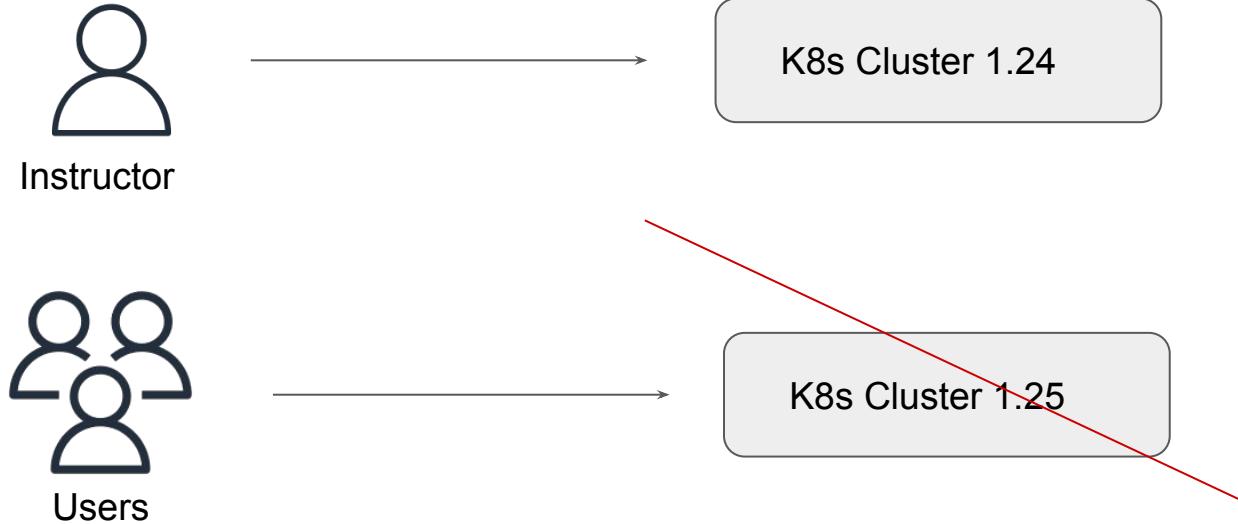
Our Architecture

We will make use of two servers, one will be for Master and second for the worker node.



Version Constraints

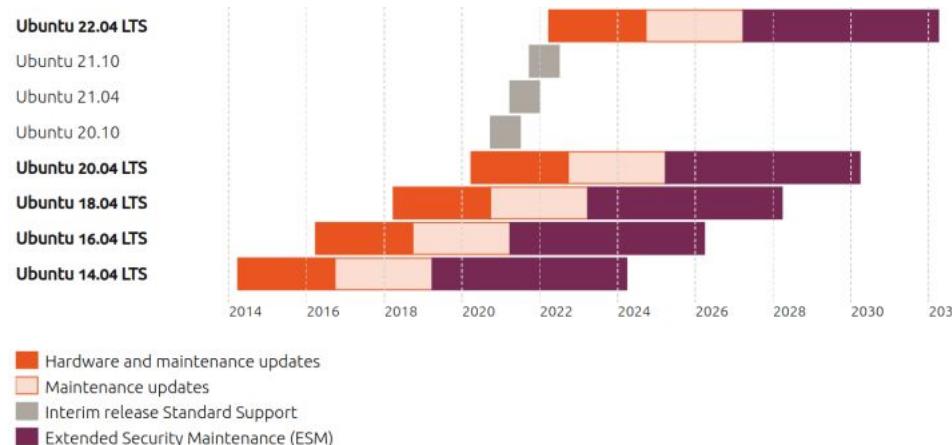
New Kubernetes versions are released quite frequently.



Operating System Constraints

It is recommended to use the same OS version as that of the demo.

We have intentionally used Ubuntu LTS (Long Term Support)



Exam Perspective

In exams, you won't have to configure K8s cluster from scratch.

You can expect small troubleshooting related questions.

Your high-level understanding about K8s Cluster deployment will help you solve the questions.

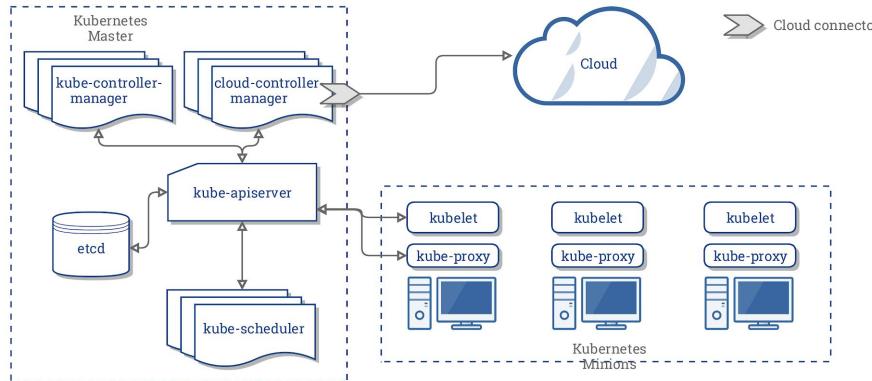
Common Patterns - K8s Cluster from Scratch

Secure Cluster Communication

Communication Should Be Secured

While configuring each components, there are some common set of steps that we will performing at a individual component level.

This can include, creating Certificates, Kubeconfig files.



Common Patterns 1 - Certificates

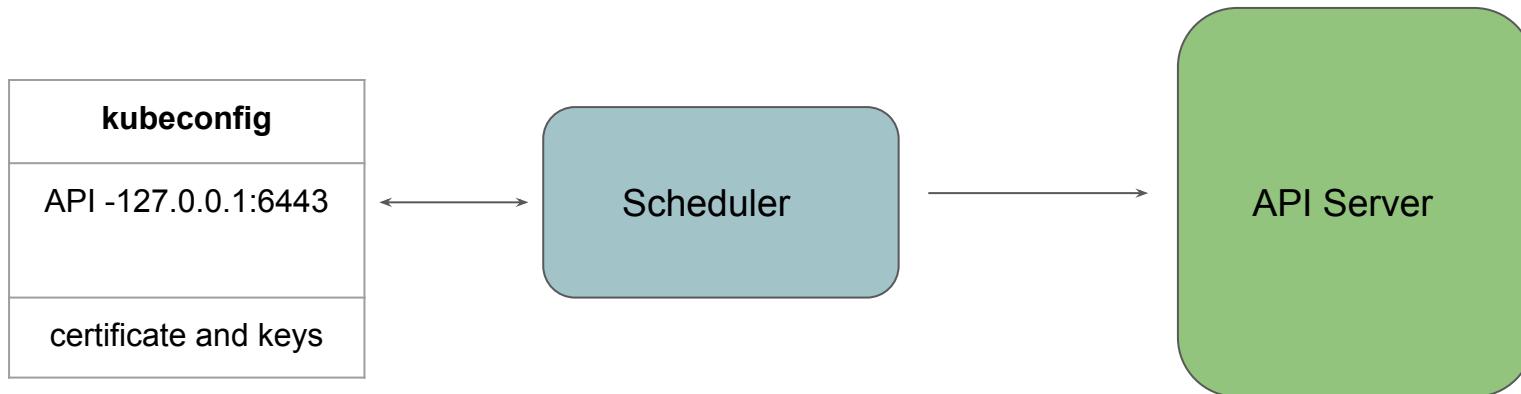
We will be generating a set of certificates and keys for each component as part of the cluster for secure communication.



Common Patterns 2 - Kubeconfig

Multiple components need to interact with the API server.

For this, we will have to create kubeconfig file for individual components.



Common Patterns 3 - Configuration for Components

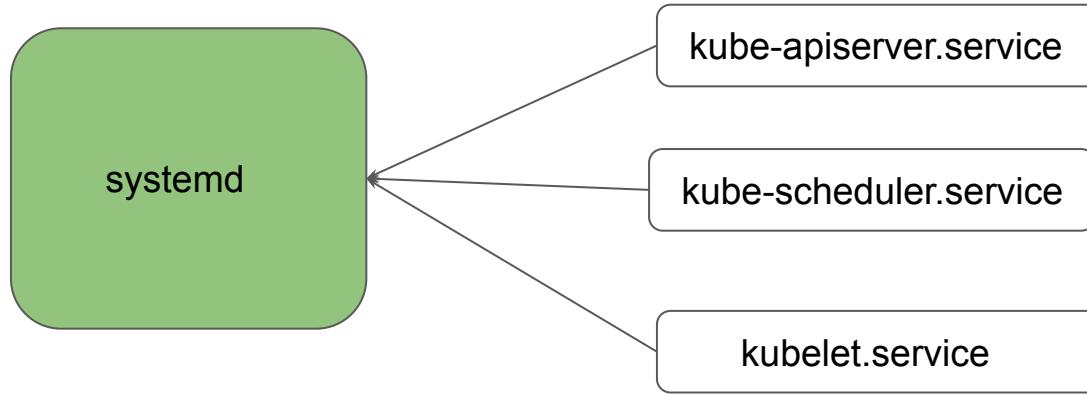
Each component of a cluster will have a unique set of configuration options that controls the functionality..



Common Patterns 4 - Systemd File

We will also create a systemd unit configuration file that encodes information about a process controlled and supervised by systemd

Systemd also can ensure to start the component automatically when system reboots.

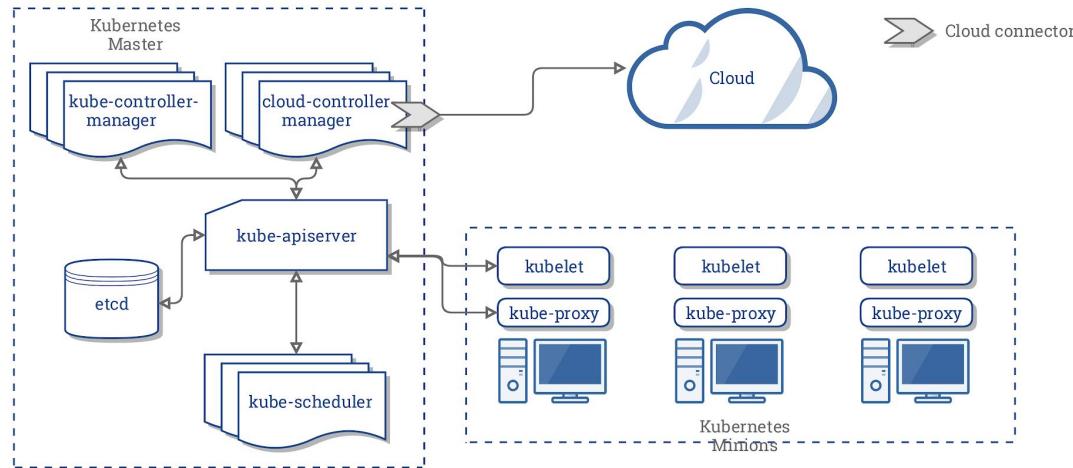


Downloading Release Binaries

Let's Learn!

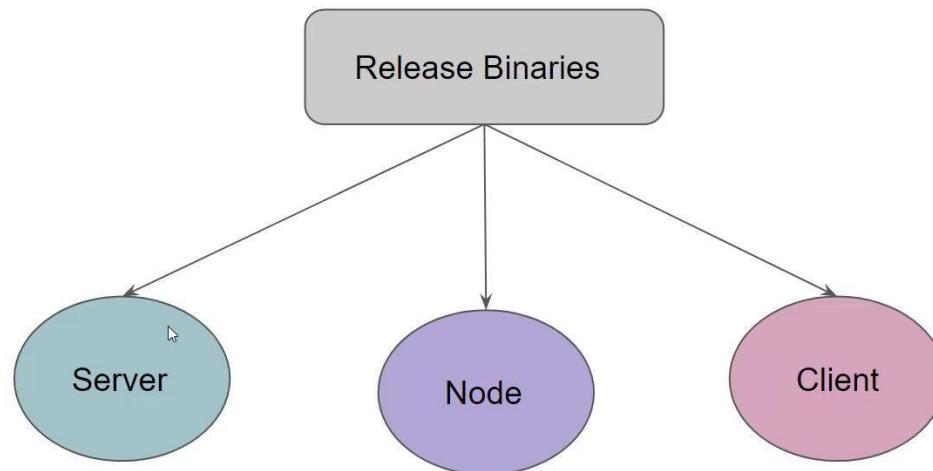
Goal of This Section

Before we start configuring the cluster, we have to first download the binaries associated with all the components.



Release Binaries

Kubernetes Release Binaries are divided into three categories



Release Binaries

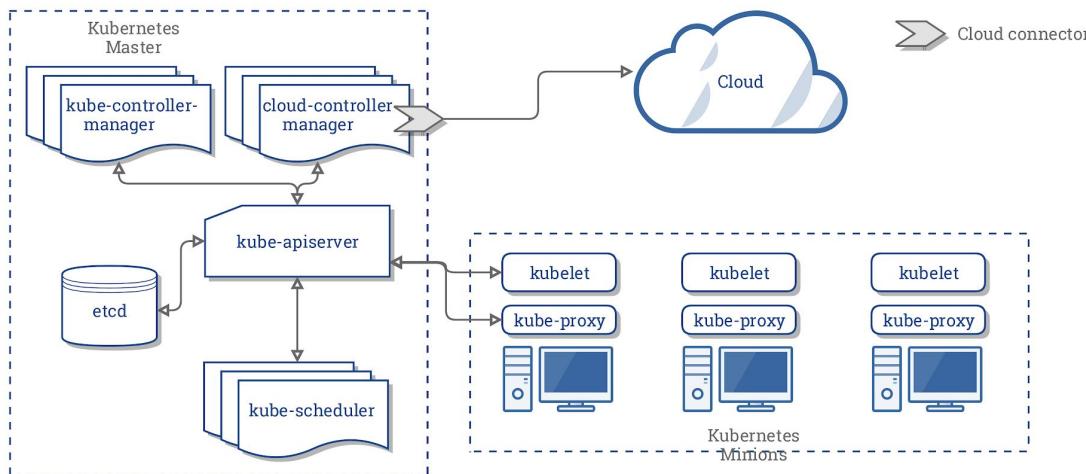
Release Binaries	Description
Server	Contains binaries for master server components like API Server, Controller Manager, Scheduler and others.
Node	Containers Binaries required by worker nodes. kubelet, kube-proxy
Client	Binaries required by client, kubectl.

Certificate Authority

Secure Cluster Communication

Communication Should Be Secured

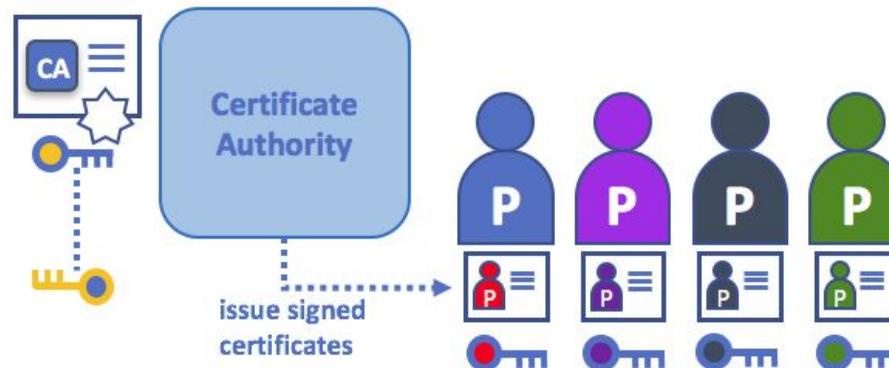
- There are multiple components which will be communicating with each other.
- When communication is in plain-text, it is prone to lot of attacks.
- Hence it is important to have secure communication between multiple components.
- This can be achieved with the help of certificates.



Certificate Authority

Certificate Authority is an entity which issues digital certificates.

Key part is that both the receiver and the sender trusts the CA.



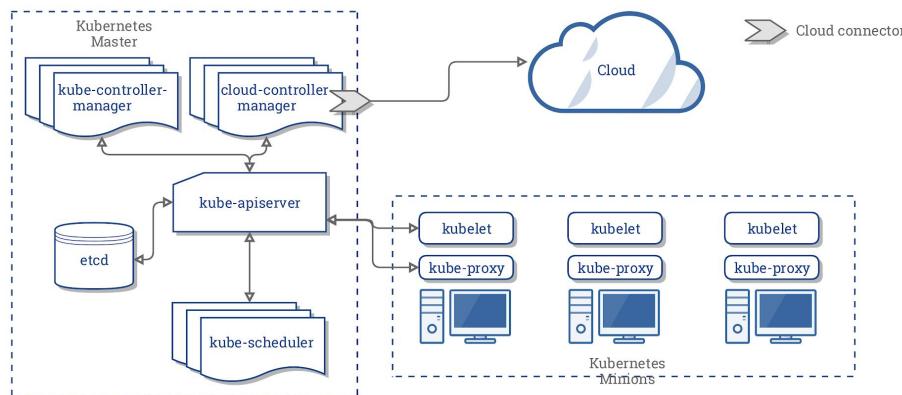
Configuring ETCD

Let's Learn!

Understanding the etcd

etcd is a distributed reliable key-value store.

etcd reliably stores the configuration data of the Kubernetes cluster, representing the state of the cluster at any given point of time.



Remember Secure Communication

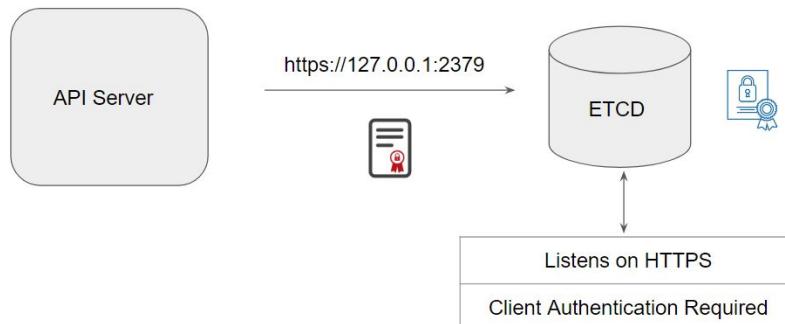


Configuring API Server

Let's Authenticate Securely

Client Authentication with ETCD

Options	Description
--etcd-cafile	CA file used to secure etcd communication.
-etcd-certfile	Client Certificate file to connect with ETCD
--etcd-keyfile	Client Key file to connect with ETCD
--etcd-servers	List of etcd servers to connect with



Certificate Provisioning

Every user who needs to access the website, will need to have a signed certificate.



Configuring Controller Manager

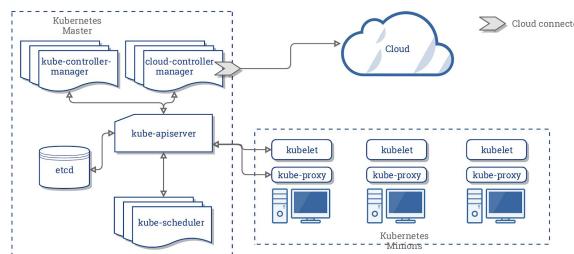
Let's Learn!

Understanding the Controller Manager

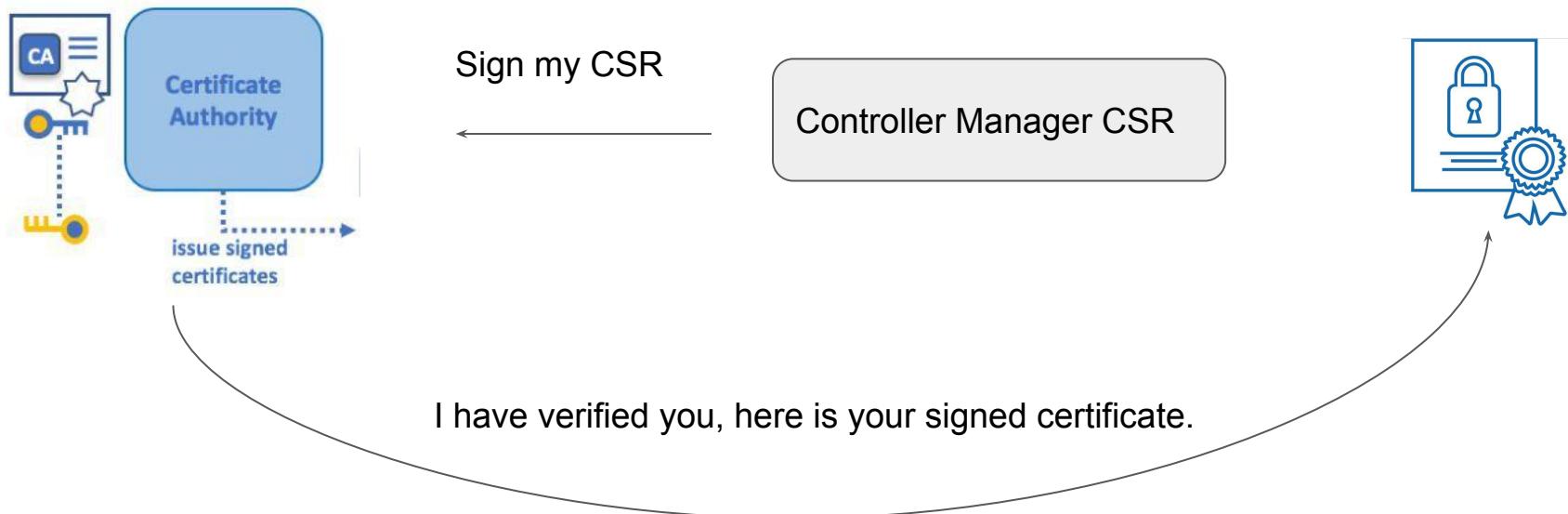
Controller Manager interacts with API server to move the current state to the desired state.

A controller tracks at least one Kubernetes resource type. These objects have a spec field that represents the desired state. The controller(s) for that resource are responsible for making the current state come closer to that desired state.

Node Controllers, Endpoint Controller, Replication Controller, Service Account & Token Controllers.



Remember Secure Communication



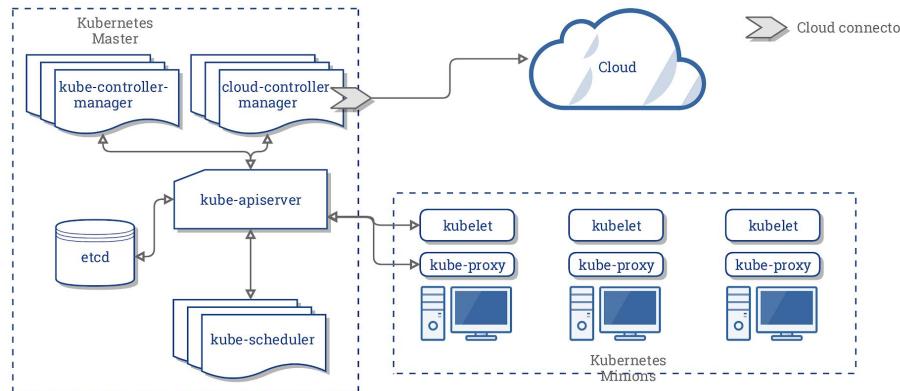
Configuring Scheduler

Let's Learn!

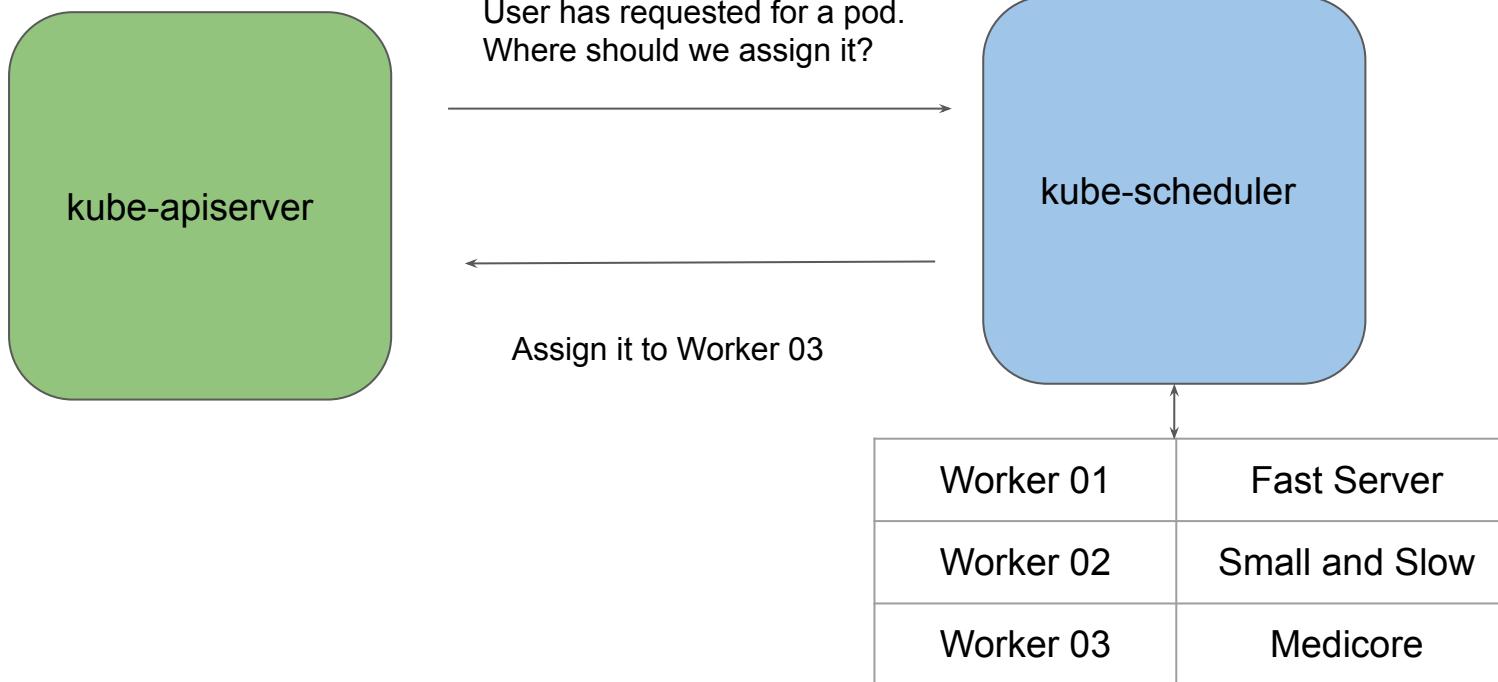
Understanding the Scheduler

A **scheduler** watches for newly created Pods that have no Node assigned.

For every Pod that the scheduler discovers, the scheduler becomes responsible for finding the best Node for that Pod to run on.



Understanding the kube-scheduler



Factors For Scheduling

There are several factors which are taken into consideration before a pod is scheduled to a node.

Some of these includes:

- Resource Requirements
- Hardware/Software policy constraints
- Affinity & Anti-Affinity
- Data Locality

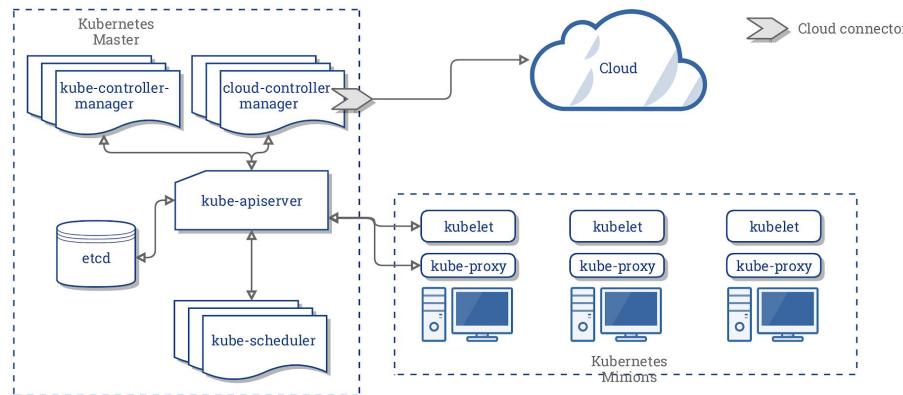
Validating Cluster Status

Let's Validate!

Finally!

We have configured all the required components as part of Kubernetes Master

Components: ETCD, API Server, Controller Manager and Scheduler.



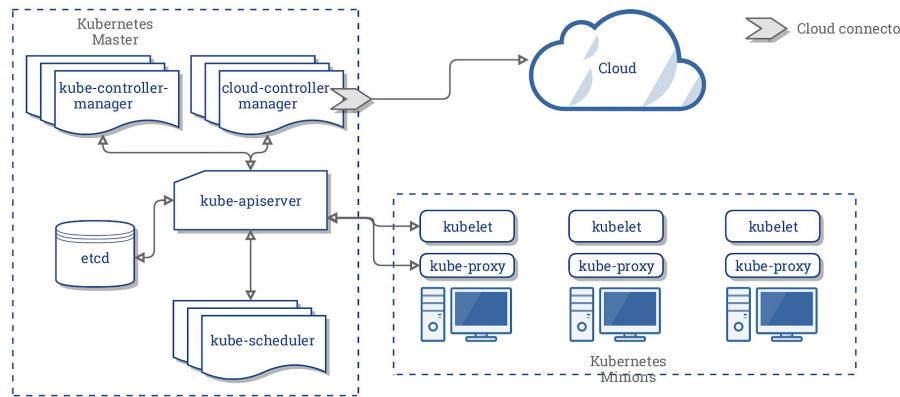
Worker Node Configuration

Second Part

Worker Node Configuration

There are two important components to configure as part of worker node.

Components: kubelet and kube-proxy



Worker Node Configuration

Components	Description
kubelet	An agent that runs on each node in the cluster. It makes sure that containers are running in a Pod.
kube-proxy	Maintains network rules on nodes. These network rules allow network communication to your Pods from network sessions inside or outside of your cluster.

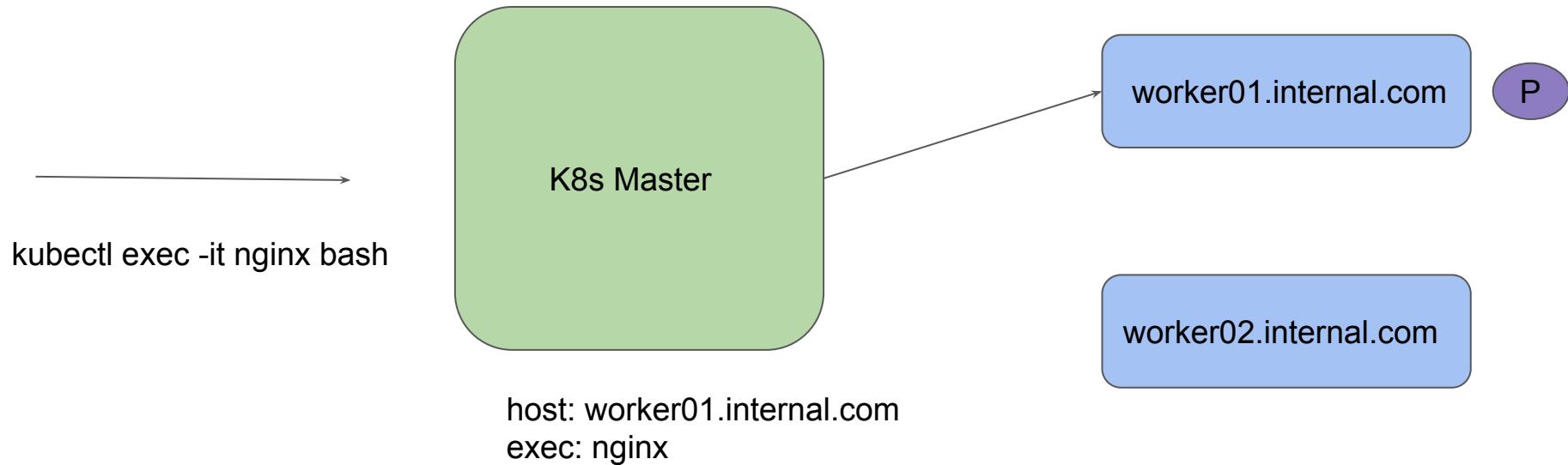
Kubelet Preferred Address Types

Let's Learn!

Overview of the Connection

On [kubectl-exec](#), kube-apiserver initiate the connection with the kubelet running on the host.

In order to communicate, kube-apiserver uses ways like IP addresses, hostnames of the nodes.



Breakdown Learnings - K8s From Scratch

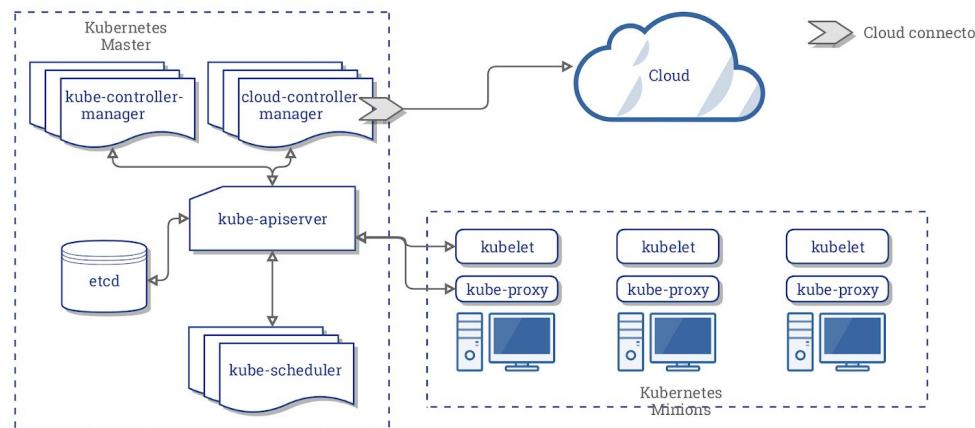
Let's Secure



Learning for Today

Each component in the Kubernetes cluster has an important role.

For testing, we will bring down certain components and see the results.



Bring Down Following Components

Components	Description
Scheduler	Watches for newly created Pods with no assigned node, and selects a node for them to run on.
kubelet	Makes sure that containers are running in a Pod.
Controller Manager	Multiple Controller types, some include: Service Account & Token controllers: Create default accounts and API access tokens for new namespaces

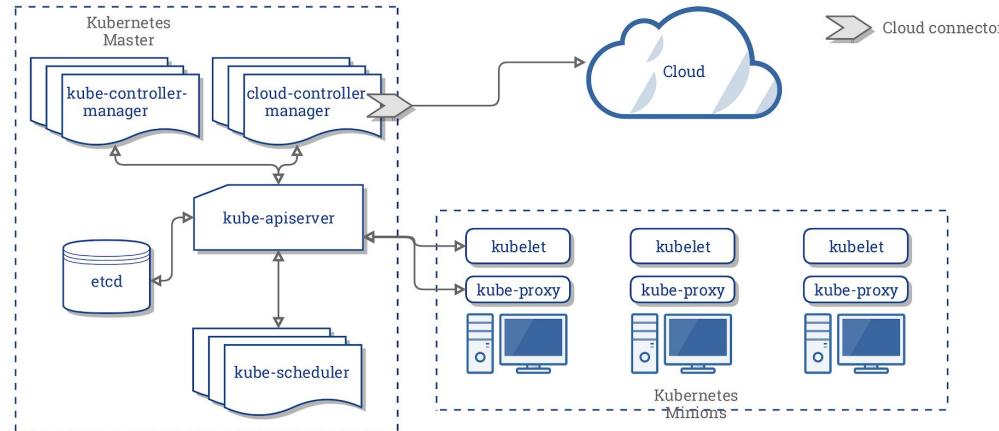
Backup & Restore ETCD

Let's Backup!

Backing ETCD

All Kubernetes objects are stored on etcd.

It is very important to backup ETCD if it's used as a backing store for our K8s cluster.



Backing ETCD

Backing up an etcd cluster can be accomplished in two ways:

- etcd built-in snapshot
- volume snapshot.



Network Policies

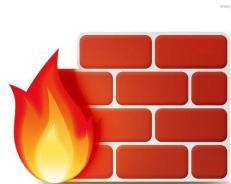
Security Aspect

Understanding the Challenge

By default, pods are non-isolated; they accept traffic from any source.

Example:

- Pod1 can communicate with Pod 2.
- Pod 1 in namespace DEV can communicate with Pod 3 in namespace Staging.



Overview of Network Policy

A network policy is a specification of how groups of pods are allowed to communicate with each other and other network endpoints.

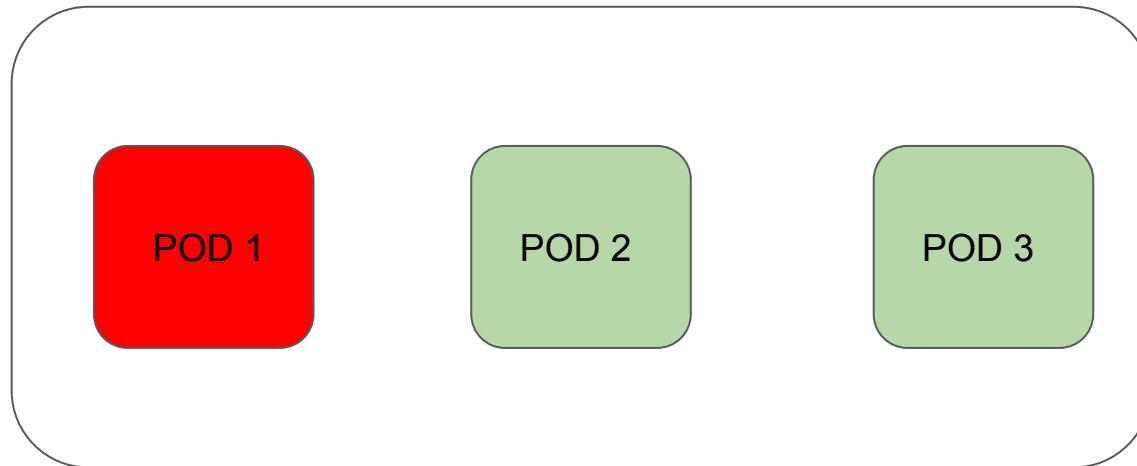
Example:

- POD 1 can only communicate with Pod 5 in the same namespace.
- POD 2 can only communicate with Pod 10 residing in namespace Security.
- No one should be able to communicate with Pod 3.

Use-Case 1 - Pod Compromise

There is a POD named AppA which is compromised.

Security Policy says that the pod should be isolated so that no communication happens.

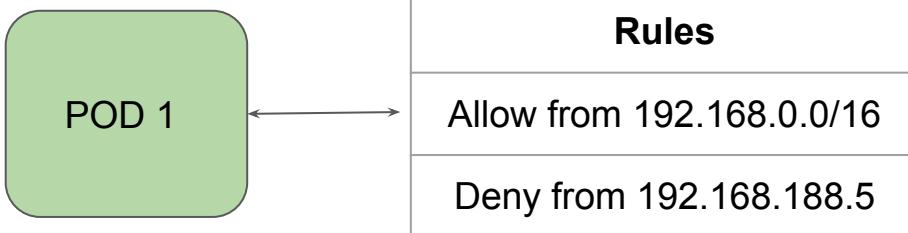


Network Policies - Part 02

Setting Right Rules

Ingress - From Selector

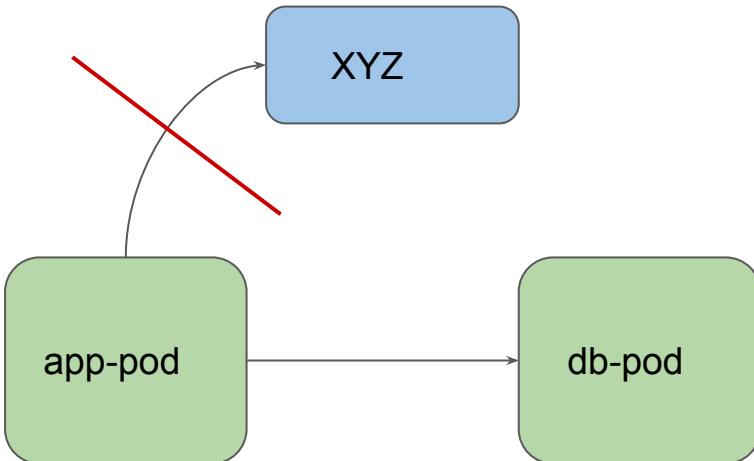
Each rule allows traffic which matches both the from and ports sections



```
---  
apiVersion: networking.k8s.io/v1  
kind: NetworkPolicy  
metadata:  
  name: ingress-from-ips  
spec:  
  podSelector:  
    matchLabels:  
      role: secure  
  ingress:  
  - from:  
    - ipBlock:  
        cidr: 192.168.0.0/16  
    except:  
    - 192.168.182.197/32  
  policyTypes:  
  - Ingress
```

Egress - To Selector

Allows outgoing connections only to a certain set of network.



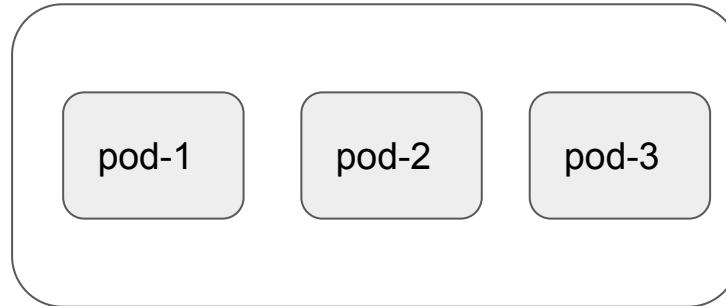
```
---
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: egress-to-ips
spec:
  podSelector:
    matchLabels:
      role: secure
  egress:
  - to:
    - ipBlock:
        cidr: 192.168.182.197/32
  policyTypes:
  - Egress
```

Namespace Selector

Allow connections based on namespace.

Example:

Allow ingress connections to FTP POD from namespace app with pods labeled reconcile.



Kubernetes Events

Let's Validate!

Overview of k8s Events

Kubernetes Events are created when other resources have state changes, errors, or other messages that should be broadcast to the system.

It provides insight into what is happening inside a cluster, such as what decisions were made by scheduler or why some pods were evicted from the node.

Events:				
Type	Reason	Age	From	Message
Normal	Scheduled	9m33s	default-scheduler	Successfully assigned default/nginx-7bb7cd8db5-nzpzm to secondary-nodes-btws
Normal	Pulling	9m30s	kubelet, secondary-nodes-btws	Pulling image "nginx"
Normal	Pulled	9m26s	kubelet, secondary-nodes-btws	Successfully pulled image "nginx"
Normal	Created	9m26s	kubelet, secondary-nodes-btws	Created container nginx
Normal	Started	9m26s	kubelet, secondary-nodes-btws	Started container nginx

Important Pointer

All the events are stored in the master server.

To avoid filling up master's disk, a retention policy is enforced: events are removed one hour after the last occurrence.

To provide longer history and aggregation capabilities, a third party solution should be installed to capture events.

Events and Namespaces

Events are namespaced.

Hence if you want event of a pod in “kplabs-namespace” then you will have to explicitly specify the --namespace kplabs-namespace.

To see events from all namespaces, you can use the --all-namespaces argument.

Field Selector

Let's Search!

Overview of Field Selector

Field selectors let you select Kubernetes resources based on the value of one or more resource fields

C:\Users\Zeal Vora>kubectl get pods --all-namespaces --field-selector metadata.namespace!=default					
NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	cilium-jg5vs	2/2	Running	0	3d
kube-system	cilium-mdm7m	2/2	Running	0	3d
kube-system	cilium-nj598	2/2	Running	0	11d
kube-system	cilium-operator-69676856c9-nvfjr	1/1	Running	2 (7d15h ago)	11d
kube-system	coredns-566f6cc75f-b4vmx	1/1	Running	0	11d
kube-system	coredns-566f6cc75f-cxsxq	1/1	Running	0	11d
kube-system	cpc-bridge-proxy-bc762	1/1	Running	0	3d
kube-system	cpc-bridge-proxy-svs6d	1/1	Running	0	3d
kube-system	cpc-bridge-proxy-vbzzc	1/1	Running	0	11d
kube-system	csi-do-node-vc7r4	2/2	Running	0	11d

Default Configuration

By default, no selectors/filters are applied, meaning that all resources of the specified type are selected.

This makes the kubectl queries kubectl get pods and kubectl get pods --field-selector "" equivalent.

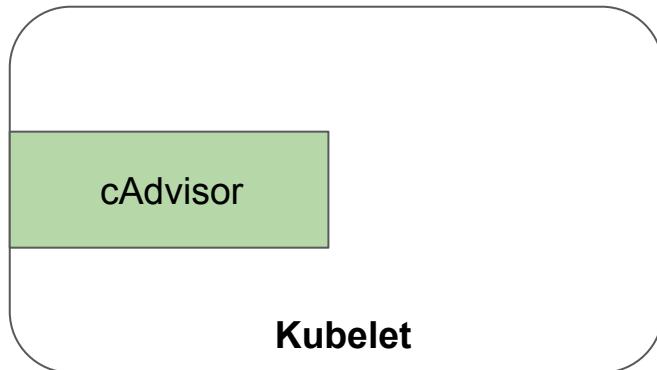
Monitoring Cluster Components

Let's Monitor!

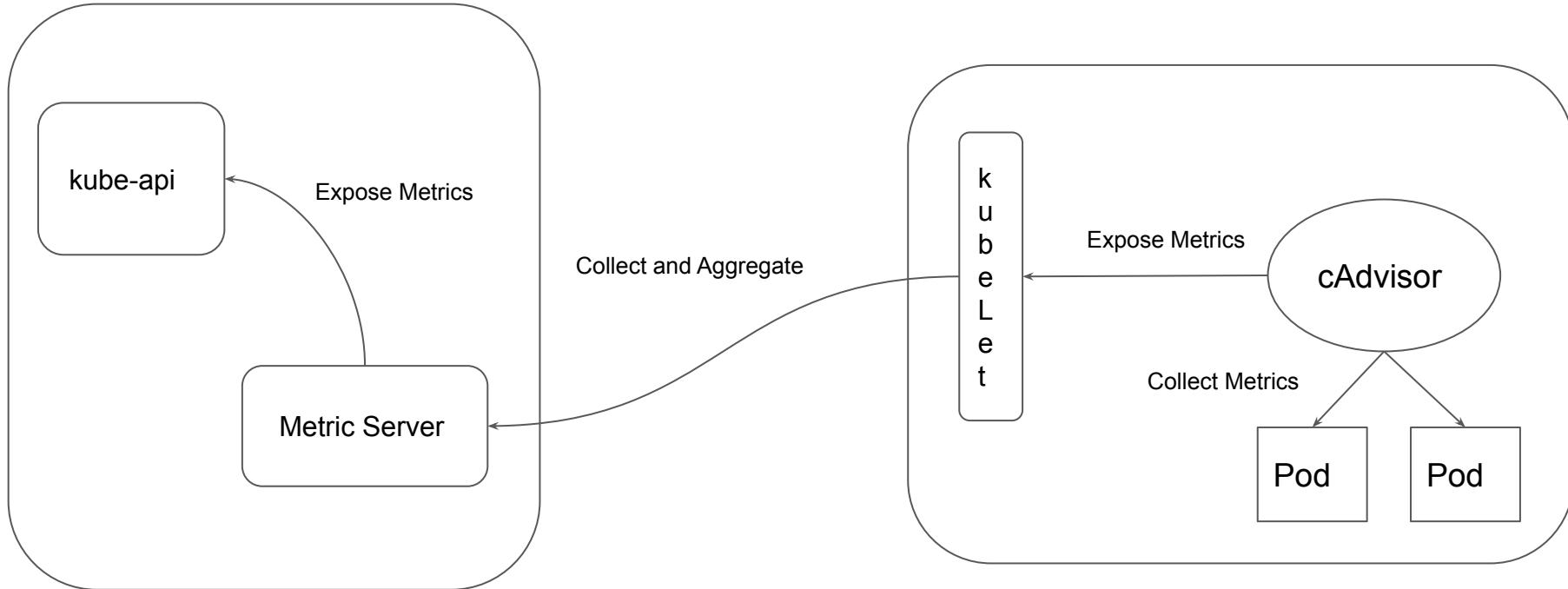
Overview about cAdvisor

One of the important functionalities of a Kubelet is to retrieve metrics aggregate and expose them through the Kubelet Summary API.

This is achieved with cAdvisor.



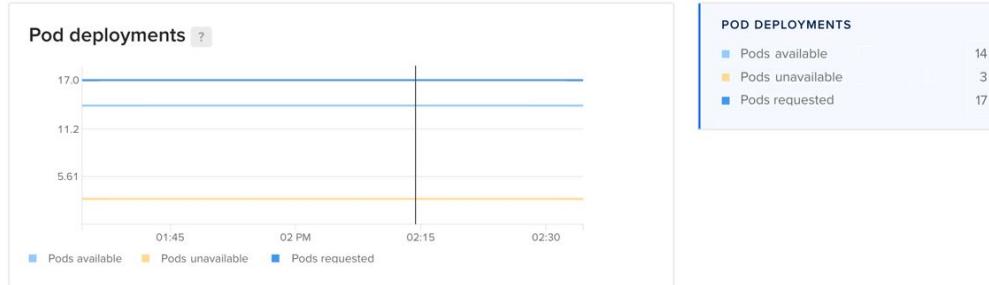
Overall Workflow



Overview about kube-state-metrics

kube-state-metrics is a simple service that listens to the Kubernetes API server and generates metrics about the state of the objects.

It is not focused on the health of the individual Kubernetes components, but rather on the health of the various objects inside, such as deployments, nodes and pods.

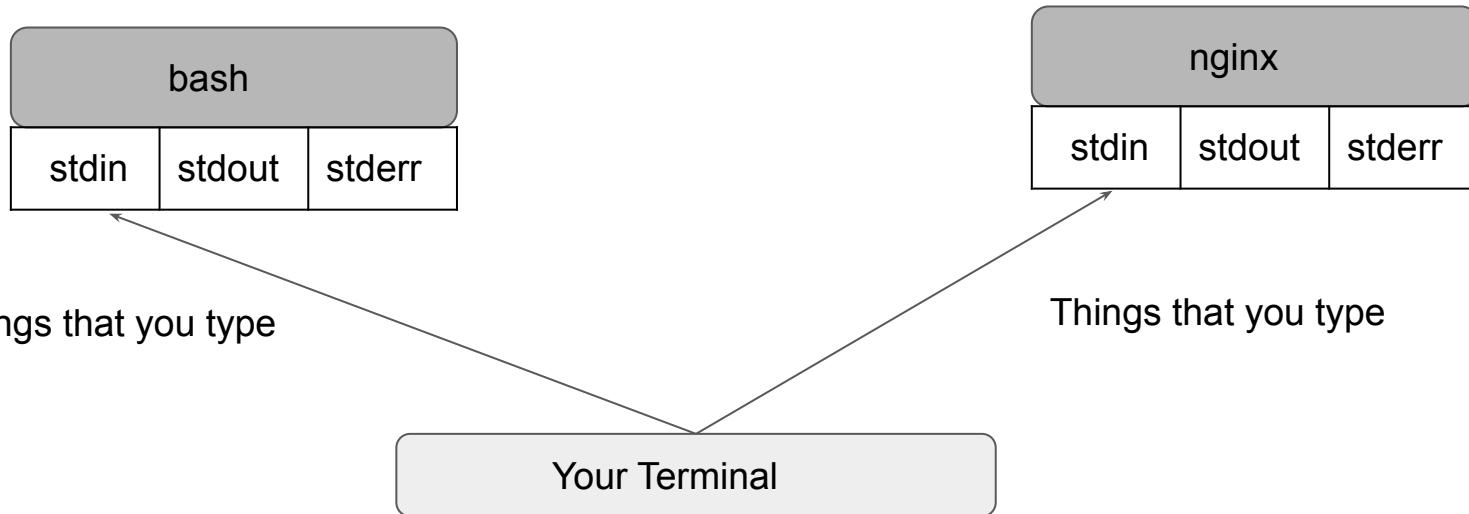


Logging Drivers

Build once, use anywhere

Getting Started

UNIX and Linux commands typically open three I/O streams when they run, called **STDIN**, **STDOUT**, and **STDERR**



Logging Drivers in Docker

There are a lot of logging driver options available in Docker, some of these include:

- json-file
- none
- syslog
- local
- journald
- splunk
- awslogs

The docker logs command is not available for drivers other than json-file and journald.

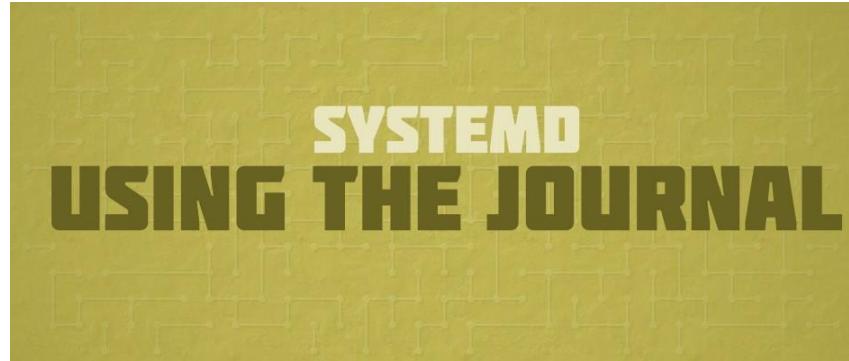
Cluster Component Logs

Let's Troubleshoot!

Component Logs

If we are making use of systemd, then we can view the component level logs with `journalctl`

It provides insight into what is happening inside a cluster, such as what decisions were made by scheduler or why some pods were evicted from the node.



Troubleshooting Application Failure

Let's Troubleshoot!

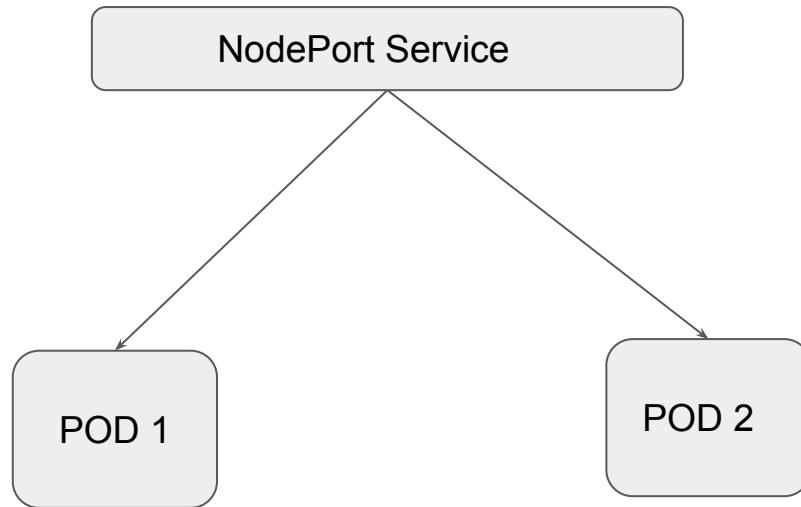
Troubleshooting Application Failure

An application can make use of various resource objects like:

- Pods
- Services
- Secrets
- ConfigMaps
- Roles, RoleBindings
- Networking Aspect etc

If one of these intermediary things are not working, your application can behave unexpectedly.

Scenario 1: Fix this up!

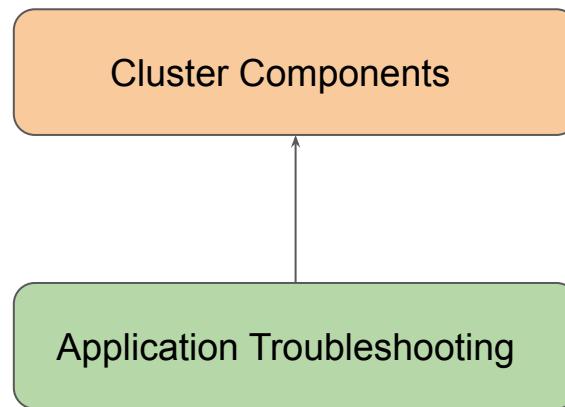


Troubleshooting Control Plane Failure

Let's Troubleshoot!

Keeping Record Straight

At this step of Cluster Troubleshooting, we assume you have already ruled out your application as the root cause of the problem you are experiencing.



Step 1: Verify Cluster Components

Verify if all the cluster level components are healthy.

```
[root@kplabs-cka-master ~]# kubectl get componentstatus
NAME           STATUS   MESSAGE             ERROR
scheduler      Healthy  ok
controller-manager  Healthy  ok
etcd-0         Healthy  {"health":"true"}
```

Step 2: Verify and Dump Cluster Info

Verify if the master and DNS servers are running.

To get more details, run the cluster-info dump to dump lots of relevant info for debugging and diagnosis

```
[root@kplabs-cka-master ~]# kubectl cluster-info  
Kubernetes master is running at https://134.209.158.242:6443  
KubeDNS is running at https://134.209.158.242:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.

Step 3: Check Logs of Individual Components

Journalctl allows us to look into the logs for components deployed via systemd.

```
journalctl -l kube-apiserver
```

```
-- Logs begin at Sun 2019-09-15 15:24:37 UTC, end at Wed 2019-09-18 13:28:44 UTC. --
kplabs-cka-master systemd[1]: Started Kubernetes API Server.
kplabs-cka-master kube-apiserver[18477]: Flag --insecure-port has been deprecated, This flag will be removed in a
kplabs-cka-master kube-apiserver[18477]: I0915 16:59:34.757359 18477 server.go:560] external host was not speci
kplabs-cka-master kube-apiserver[18477]: I0915 16:59:34.758460 18477 server.go:147] Version: v1.15.0
kplabs-cka-master kube-apiserver[18477]: I0915 16:59:35.736575 18477 plugins.go:158] Loaded 10 mutating admissi
kplabs-cka-master kube-apiserver[18477]: I0915 16:59:35.736616 18477 plugins.go:161] Loaded 6 validating admiss
kplabs-cka-master kube-apiserver[18477]: E0915 16:59:35.739172 18477 prometheus.go:55] Failed to register depth
kplabs-cka-master kube-apiserver[18477]: E0915 16:59:35.739235 18477 prometheus.go:68] Failed to register adds
```

Version Skew Support Policy

Important Pointer During Upgrades

Overview of Kubernetes Releases

Kubernetes versions are expressed as **x.y.z**, where x is the major version, y is the minor version, and z is the patch version,

Example Release: v1.15.0

Major Version	Minor Version	Patch Version
1	15	0

Supported Version Skew

Component	Description about Supported Version
kube-apiserver	<p>In highly-available (HA) clusters, the newest and oldest kube-apiserver instances must be within one minor version.</p> <p>newest kube-apiserver is v1.15 Other supported kube-apiserver instance is either v1.15 or v1.14</p>
kubelet	<p>kubelet must not be newer than kube-apiserver, and may be up to two minor versions older.</p> <p>Kube-apiserver is v1.15</p> <p>Supported kubelet version: 1.15, 1.14, 1.13</p>

Important Use-Case

Let's say that you have two kube-apiserver running in HA environment.

1st kube-apiserver version: 1.15

2nd kube-apiserver: 1.14

Supported Kubelet Version: 1.14, 1.13

1.15 kubelet version is not supported because it would be newer than 2nd kube-apiserver.

Supported Version Skew

Component	Description about Supported Version
kube-controller-manager kube-scheduler cloud-controller-manager	<p>Must not be newer than the kube-apiserver.</p> <p>Expected to match the kube-apiserver minor version.</p> <p>Can be upto one minor version older (specficially during upgrades)</p>
kubectl	<p>kubectl is supported within one minor version (older or newer) of kube-apiserver.</p> <p>kube-apiserver is at 1.13</p> <p>kubectl is supported at 1.14, 1.13, and 1.12</p>

Component Version Upgrade Order

In a single-instance cluster, the existing kube-apiserver instance is 1.n [1.15]

In an HA cluster, all kube-apiserver instances are at 1.n or 1.(n+1) (this ensures maximum skew of 1 minor version between the oldest and newest kube-apiserver instance)

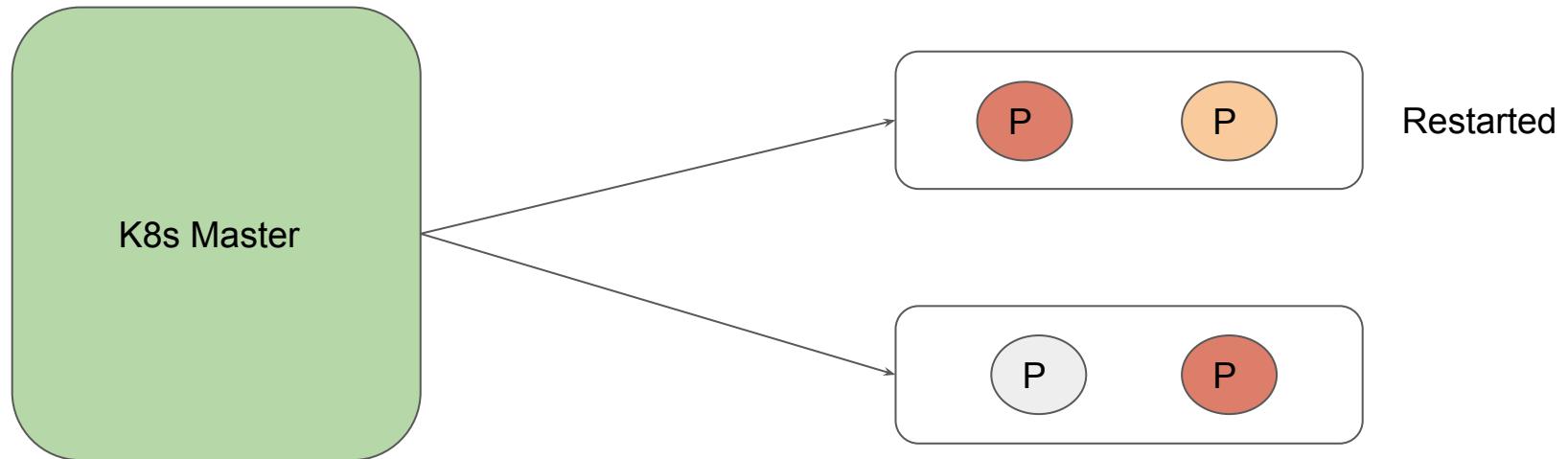
The kube-controller-manager, kube-scheduler, and cloud-controller-manager instances that communicate with this server are at version 1.n (this ensures they are not newer than the existing API server version, and are within 1 minor version of the new API server version)

Taint Based Evictions

Advanced Scheduling

Scenerio 1.1- Immediate Restart

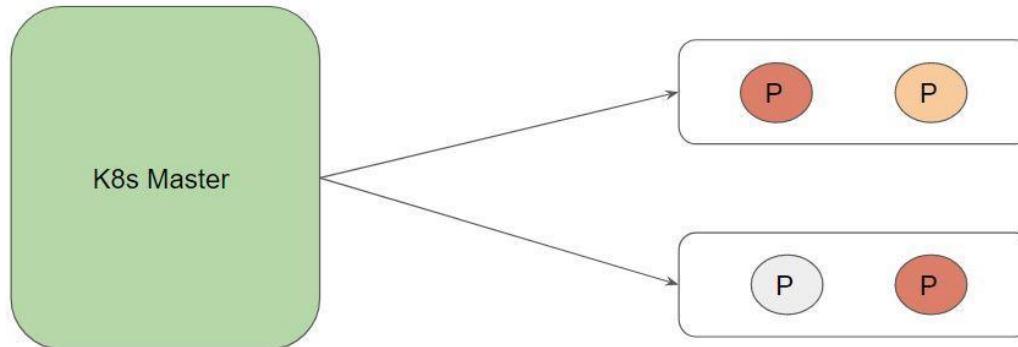
If node starts immediately, the kubelet process will start the pods and node is back online.



Scenario 1.2 - Brief Downtime

If reboot takes longer (default time is 5 minutes) then the node controller will terminate the pods that are bound to the unavailable node.

If there is a corresponding replica set, then a new copy of the pod will be started on a different node.



Understanding Taints

The worker nodes are automatically tainted in Kubernetes based on specific node conditions:

- node.kubernetes.io/not-ready
- node.kubernetes.io/unreachable
- node.kubernetes.io/out-of-disk
- node.kubernetes.io/memory-pressure
- node.kubernetes.io/disk-pressure
- node.kubernetes.io/network-unavailable
- node.kubernetes.io/unschedulable

Registering for Kubernetes Exams

It's time to Rock!

Exam Registration Pointers

- Both the CKA and CKAD exams are provided by Linux Foundation.
- The cost and the registration process of both the exams are the same.

Cost : 300\$

- One free-attempt is provided if you do not make it in the first attempt.
- Make sure to register officially from Linux Foundation Website.



Payment Invoice

Invoice



ORDER SUMMARY

Account No: 6432

Order Date: 5 October, 2019

Web Order No: 585427

Email Address: sunzealvora@gmail.com

Shipping Address:

The Linux Foundation:

1 Lettermann Drive
Building D, Suite D4700
San Francisco, CA 94129-1494
Tax ID: 46-0503801

TITLE	UNIT PRICE	QUANTITY	TOTAL
Certified Kubernetes Administrator (CKA)	\$252.00	1	\$252.00

PAYMENTS

Card type: Diners Club
Last 4 digits: 9683
Amount billed: \$252

Thank you for your purchase!

Scheduling an Exam

Schedule an Exam

You are signed in as sunzealvora@gmail.com [sign off](#)

Select a Test Site and Time

Sponsor & Exam [change](#)

Linux Foundation - Certified
Kubernetes Application Developer

Exam Code CKAD

[Handbook Link](#)

Exam Duration 120 mins

Language English

Search Date & Time [change](#)

Time (UTC+05:30) Chennai,
Zone Kolkata, Mumbai, New Delhi

Date 10/11/2019

Time 6:00 am

Search Date & Time [change](#)

Time (UTC+05:30) Chennai,
Zone Kolkata, Mumbai, New Delhi

Date 10/11/2019

Time 6:00 am

Date

October 2019

Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa	Su
11	12	13	14	15	16	17	18	19	20

Time

am ▶ 05 • 06 • 07 • 08

pm ▶ 06 • 07 • 08 • 10

Time Slot

6:00 am - 8:00 am | 6:15 am - 8:15 am

Scheduling an Exam

Schedule an Exam

You are signed in as sunzealvora@gmail.com [sign off](#)

Sponsor & Exam [change](#)

Linux Foundation - Certified Kubernetes Application Developer	
Exam Code	CKAD
Handbook Link	
Exam Duration	120 mins
Language	English

Selected Time Slot [change](#)

Linux Foundation Web Delivery.	
Date	10/11/2019
Time Slot	6:00 am to 8:00 am

Reservation

Please confirm the Sponsor and Exam information to the left is correct to complete the scheduling process for your exam.

Cancellation and Refund Policy

If you need to cancel or reschedule your exam, please use the cancellation options found at <https://www.examslocal.com/Candidate>, email us at examsupport@psionline.com, or call +1-888-504-9178, +1-702-904-7342 for additional support.

To receive a full refund you may cancel or reschedule the examination 24 hours prior to the scheduled time.

If you do not cancel the exam within 24 hours of your scheduled appointment, you will be charged the full original fee and you will not be issued a refund. In addition your appointment will be marked as a "no show".

If you do not arrive on time to the appointment and you do not start your exam within 10 minutes of your scheduled start time, the system will automatically mark you as a No Show and you will not be able to receive a refund for your appointment.

Confirm Reservation

Start The Exam

Once your exam is successfully registered, you will receive an email from Linux Foundation with all the details and instructions to start the exams.

LF allows students to start exams 15 minutes prior to appointment time.

Start Your Exam on Time: On your appointment day, please use your user credentials to log into <https://www.examslocal.com/linuxfoundation> to start your exam. Go to "My Exams" and select the exam you wish to start. You may start your exam up to *15 minutes prior to your scheduled appointment time*. **You MUST start your exam no later than 15 minutes after your scheduled appointment time.** Failing to start your exam within 15 minutes of your scheduled appointment time will result in the system marking you as a No-Show and you will not be able to take your exam.

Scheduling an Exam

Reminder for your Linux Foundation Exam

My Exams

You are signed in as sunzealvora@gmail.com [sign off](#)

Pending Exams

Click the 'View Details' button to Launch, Cancel, Reschedule or Check Compatibility for your exam.

Linux Foundation
Certified Kubernetes Application Developer
Conf Code: 69C-BF2
Exam Date: 10/10/2019

[View Details](#)

[Refresh Exams](#)

[+ history](#)

Exam Info

Scheduling Options

Not Available

[Compatibility Test](#)

Exam Countdown

00 : 00 : 41 : 25

Days Hours Minutes Seconds

Linux Foundation
Certified Kubernetes Application Developer
CKAD - English
Launch Status: Exam is not deliverable until 10/11/2019

10/11/2019

6:00 AM - 8:00 AM - India Standard Time

Confirmation Code
69C-BF2

Candidate Id
4168071716

Chat Now!

Online - Click here to get help +

Type here to search

Start Your Exam

The screenshot shows a web browser window titled "Reminder for your Linux Foundation Exam". The URL is "examslocal.com/Candidate". The page has a header with the "psi" logo and navigation links: "Schedule an Exam", "My Exams", "My Account", "Support", "Compatibility Tool", and "Exam FAQs". A sign-off link "sign off" is also present.

The main content area is titled "My Exams". It displays a "Pending Exams" section with a message: "Click the 'View Details' button to Launch, Cancel, Reschedule or Check Compatibility for your exam." Below this is a card for a "Linux Foundation Certified Kubernetes Application Developer" exam. The card includes the "Conf Code: 69C-BF2", "Exam Date: 10/10/2019", and a "View Details" button. There are also "Refresh Exams" and "+ history" buttons.

To the right, under "Exam Info", is a "Scheduling Options" section stating "Not Available". It includes a "Compatibility Test" link and a prominent green "Launch Exam" button. Below this, the exam details are listed: "Linux Foundation Certified Kubernetes Application Developer CKAD - English", "Date: 10/11/2019", and "Time: 6:00 AM - 8:00 AM - India Standard Time".

At the bottom left, there is a "Chat Now!" button with a speech bubble icon and a "psi" logo. A footer bar at the bottom says "Online - Click here to get help" with a plus sign icon.

Begin your Exam - 15 minutes early

The screenshot shows a web browser window for the PSI My Exams page at examlocal.com/Candidate. The PSI logo is at the top left. The main header says "My Exams". A navigation bar includes links for "Schedule an Exam", "My Exams", "My Account", "Support", "Compatibility Tool", and "Exam FAQs". The user is signed in as sunzealvora@gmail.com.

Pending Exams:

- Linux Foundation**
Certified Kubernetes Application Developer
Conf Code: 69C-BF2
Exam Date: 10/10/2019

Buttons: "View Details", "Refresh Exams", "+ history".

Exam Info:

Scheduling Options:
Not Available

Compatibility Test:

Launch Exam button.

Linux Foundation
Certified Kubernetes Application Developer
CKAD - English

Date: 10/11/2019
Time: 6:00 AM - 8:00 AM - India Standard Time

Confirmation Code: 69C-BF2
Candidate Id: 4168071716
Transaction Date: 10/8/2019

Chat Now! button and "Online - Click here to get help" link.

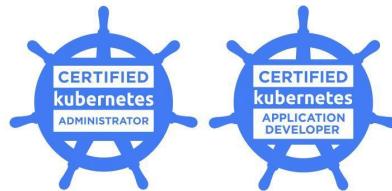
Candidate Handbook

It's time to Rock!

Candidate Handbook

Candidate Handbook provides overview information about the exam environment, system requirements as other important pointers.

Make sure to read it before sitting for the exams.



Certified Kubernetes Administrator
AND
Certified Kubernetes Application
Developer
Candidate Handbook

Important Pointer - Part 01

Make sure your system requirement fulfills according to specification mentioned.

Important Requirements:

- Chrome or Chromium Browser
- Reliable Internet Access
- Webcam
- Microphone

Verify things from compatibility checker tool before exams.

Important Pointer - Part 02

Make sure to provide government-issued photo identification.

Example Identification

- Passport
- Driving License.
- National / State Identity card.

Important Pointer - Part 03

Keep your desk clean and ensure no one else is present in the room.

Keep phone in silent mode, make sure you sit in silent room.

Follow the instruction which protector provides.

Exam Console Format

The screenshot shows a software interface for an exam console. At the top, there is a navigation bar with icons for Online, Camera, Desktop, Live Chat, Exam Controls, and Exam Information. A "Notepad" window is open in the center, displaying the text "This is Notepad". Below the Notepad, there are two text blocks:

In order to begin please press the
in the u

Use the Live Chat feature in the
You must ACCEPT A

Exam Console Format

The screenshot shows a user interface for an exam console. At the top, there is a navigation bar with icons for Online, Camera, Desktop, Live Chat, Exam Controls, and Exam Information. A "Notepad" window is open in the center, displaying the text "This is Notepad". Below the Notepad, there are two sets of instructions:

In order to begin please press the button in the upper right corner of the window.

Use the Live Chat feature in the upper right corner of the window.
You must ACCEPT the terms and conditions before you can begin the exam.

Close Not-Required Programs

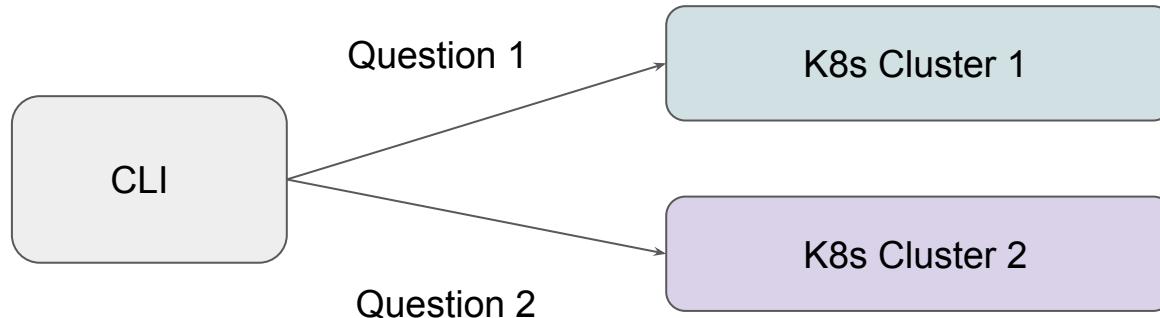
You can only have your chrome browser running and all windows should be closed.

There can only be two tabs open within your browser:

1. Actual Exam Session Tab
2. Kubernetes Documentation

Working with Multiple Clusters

- Exams will have multiple clusters.
- Make sure to switch to right context before performing the practical.
- CKAD exam typically has the kubectl set-context command
- 6 K8s cluster for CKA and 3 for CKAD



Make Backups ready

Always ensure that your laptops are fully charged before the exams.

Keep mobile phone fully charges in-case you need to use it's hotspot as a backup.



Exam Tips

- Every question has a score associated with it.
- Prioritize questions according to the time and its score.
- Add the question that you skip and its score in notepad to prioritize it later.

Example:

Question Number	Score	Estimate Time to Complete
4	3	10 minutes
6	8	12 minutes

Exam Results

Exam Results are provided within 36 hours of your examination.

- Minimum passing score of 66% is required for CKAD exam.
- Minimum passing score of 74% is required for CKA exam.

You will receive an email with your score and also your certificate.

Linux Foundation provides 1 free attempt if registered from them. So no worries!

Important Tips for Exams

Primary Focus is Time Management

Practice Based On Current K8s Version

Kubernetes releases new version quite frequently.

We had a student, who started at Kubernetes version with 1.16, by the time he completed the course, the exam was running the Kubernetes 1.18

Environments of Managed Service providers are not that face paced.

Use latest version of tools like Minikube for final exam preparation and practice tests.

Always use short names

Prefer making use of short-names in exams as it will save you a lot of time.

Full Command	Short Command
kubectl get networkpolicies	kubectl get netpol
kubectl get componentstatuses	kubectl get cs
kubectl get certificatesigningrequests	kubectl get csr

Use the command `kubectl api-resources` to find all the short-names.

Alias the Kubectl

During the exams, you might be running the kubectl command a hundred times.

Create alias of k instead of kubectl

- k get pods
- k get deployments

To create an alias, run following command:

```
alias k=kubectl
```

Kubectl auto-complete

Auto-Complete will save you a lot of time.

Without auto-complete, you will have to copy paste the name of object within your CLI.

For example:

```
k describe netpol net3s42sfds
```

Changes to :

```
k describe netpol net[TAB]
```

Create Manifests via CLI

Creating manifests via CLI will save you a lot of time during exams.

Question: Create a pod from the nginx image. Expose the port 80.

Approach 1:

Go to documentation, find pod manifest, copy and paste it to your CLI, edit it.

Approach 2:

```
kubectl run nginx --image=nginx --port=80 --dry-run=client -o yaml > pod.yaml
```

Know Documentation Well

You should be aware which document contains which examples.

Sometimes the entire exam questions are similar to examples shown in the documentations.

Focus primarily on “Tasks” related docs as they directly show you the commands and examples.

Prioritize Exam Questions

You will need to prioritize exam questions based on it's score and amount of time it will take to solve it.

Write details about it in notepad so that you can look into it at later stage.

1. Q1 - 4 points - 8 minutes.
2. Q3 - 8 points - 10 minutes

If you see a question which will take more than 5 minutes and is of just 2 points, keep it for the last.

If you are not able to solve the question, move on, immediately.

Avoid Taking Breaks

Time Management is very important.

Although you are allowed to request for breaks from proctor, avoid doing that.

I generally avoid drinking water 1 hour before the exams even though I'm thirsty.

Practice in Ubuntu environment

Since exams will be based on Ubuntu environment, go ahead and do all the practice tests with Ubuntu environment.

This is very important if you are using Windows.

Apply via Manifest Files

Always prefer to create new object via manifest files.

This will help you have a document and quickly modify things if required.

Since exam is time sensitive, you might have to skip a question half way and return to it at later stage.

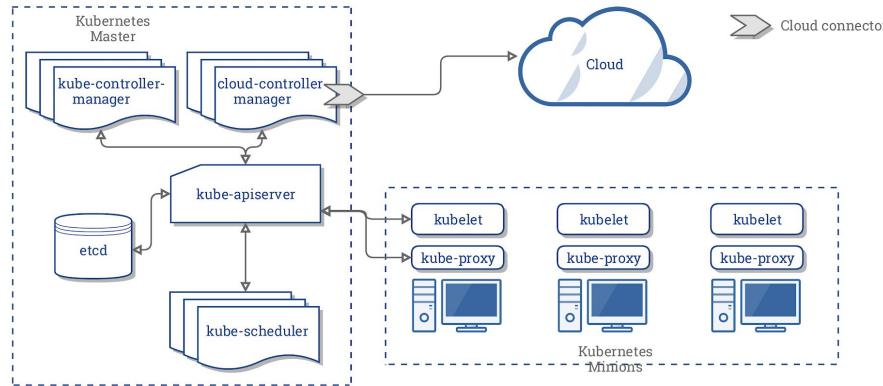
Important Pointers - Domain 1

Workloads & Scheduling

PODS

Know on how to create PODS [single container and multi-container]

Be aware about the architecture of Kubernetes and importance of the components.



Multi-Container PODS

Know on how you can launch multiple containers in a given POD.

Example Question:

- Create a pod with the name kplabs-multicontainer
- The pod should have three containers named nginx, consul and redis

Labels

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

List nodes within the K8s cluster along with labels:

```
kubectl get nodes --show-labels
```

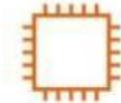
NAME	STATUS	ROLES	AGE	VERSION	LABELS
worker0	Ready	<none>	1d	v1.13.0	..., kubernetes.io/hostname=worker0
worker1	Ready	<none>	1d	v1.13.0	..., kubernetes.io/hostname=worker1
worker2	Ready	<none>	1d	v1.13.0	..., kubernetes.io/hostname=worker2

Selectors

Selectors allows us to filter objects based on labels.

1. Show me all the pods which has label where **env: production**

kubectl get pods -l env=production



name: kplabs-gateway
env: production



name: kplabs-db
env: production



name: kplabs-elb
env: production

Command and Arguments

Be familiar with commands and arguments for PODS.

You should be able to start a given POD with any given command / argument.

Example Question:

Launch a Pod Busybox with the image of 1.28. Pod should sleep for 3600 seconds and exit.

Deployments

Be very very familiar with Deployments.

Topics:

- Rolling Updates,
- Rollbacks
- Scaling
- Record Instruction

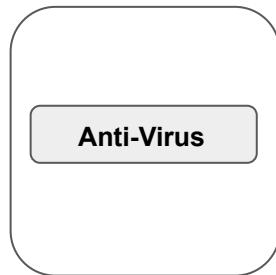
Tip: Most of the commands you will find within the kubectl cheat sheet.

DaemonSets

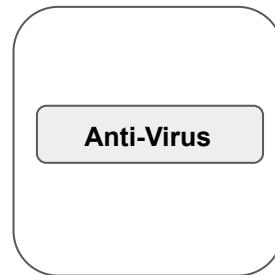
A DaemonSet can ensure that all Nodes run a copy of a Pod.

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

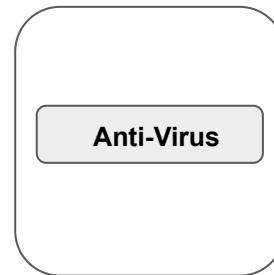
Worker Nodes



Node 01



Node 02



Node 03

NodeSelectors

You should be able to schedule a pod on a specific node.

Hint: NodeSelector

Having an understanding of labels and selectors is important.

Q: Schedule a POD on a node which has disk=ssd

Taints and Tolerations

Taints are used to repel the pods from a specific nodes.

In order to enter the taint worker node, you need a special pass. This is referred as Toleration.

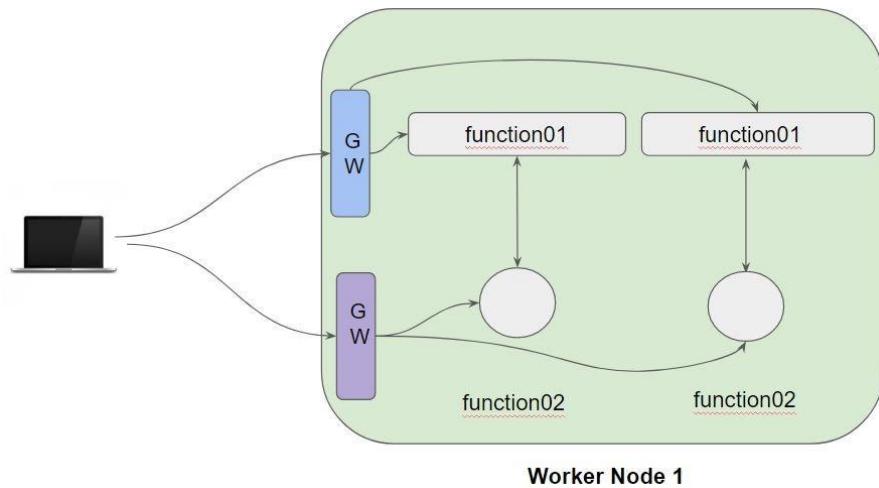
Taints	Description
NoSchedule	No pod will be able to schedule onto node1 unless it has a matching toleration. Pods that are already running will continue to run.
NoExecute	Pod is evicted from the node if it is already running on the node

Important Pointers - Domain 2

Services and Networking

Kubernetes Service

In Kubernetes, a Service is an abstraction which defines a logical set of Pods and a policy by which to access them



Service Type Cluster IP

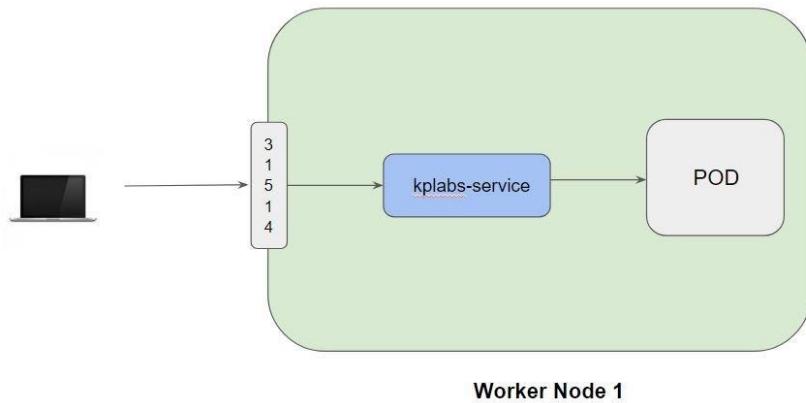
Whenever service type is ClusterIP, an internal cluster IP address is assigned to the service.

Since an internal cluster IP is assigned, it can only be reachable from within the cluster.

This is a default ServiceType.

Service Type Node Port

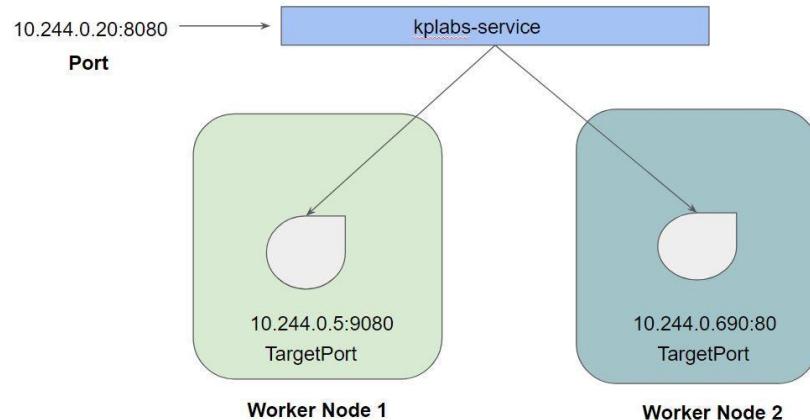
If `servicetype` is `NodePort`, then Kubernetes will allocate a port (default: 30000-32767) on every worker node.



Port vs Target Port - Service

Port refers to the service port.

Target Port refers to port of the container.



Named Port

We can also associate a name associated with the port

This name must be unique within the pod.

```
spec:  
  containers:  
    - image: nginx  
      name: nginx  
    ports:  
      - containerPort: 80
```



```
spec:  
  containers:  
    - image: nginx  
      name: nginx  
    ports:  
      - containerPort: 80  
        name: http
```

Named Port Reference in Service File

Instead of Port Number, we can also specify the name while creating service.

```
kubectl expose pod nginx --port=80 --target-port=http --name "kplabs-svc"
```

```
spec:  
  ports:  
    - port: 80  
      protocol: TCP  
      targetPort: http  
  selector:  
    run: nginx  
  type: NodePort
```

Ingress

Know the basics of ingress and how ingress resource can be created.

Example Question:

- Create an ingress for service kplabs-svc.
- Path should be /hello
- Port should be 8000
- Namespace: kplabs-internal

Service Accounts

Service Accounts are namespaced.

Default service account gets automatically created when you create a namespace

PODS are automatically mounted with the default service accounts.

Know on how you can create custom service accounts for a given namespace/

Important Pointers - Domain 3

Cluster Architecture, Installation & Configuration

Important Pointer - kubeadm

Know very well on how to set up cluster with kubeadm.

- Setting up Master Node.
- Setting up Worker Node.

Be very thorough with the upgrade steps (kubeadm 1.18 to kubeadm 1.19.X)

ETCD - Backup & Restore

You should know on how to backup and restore ETCD based on certificates

Following things will be provided:

- CA Certificate
- ETCD Certificate
- ETCD Key

Store the backup in a specific location.

Restore backup from specific location

Important Note - ETCD Restore

It can happen that ETCD systemd file is configured with a specific user, etcd

If you perform restore, all the files/folders will have permission of user who restored it (root/student).

Make sure to change the permission so that ETCD can read the contents.

```
chown -R etcd.etcd /etc/etcd-data
```

Network Security Policy

A network policy is a specification of how groups of pods are allowed to communicate with each other and other network endpoints.

Example:

- POD 1 can only communicate with Pod 5 in the same namespace.
- POD 2 can only communicate with Pod 10 residing in namespace Security.
- No one should be able to communicate with Pod 3.

Be prepared to answer question on Network Security Policy with a specific requirement.

Authorization

You should know on how to create following:

- Role / Cluster Role
- Role Binding / Cluster Role Binding

Know on how to bind a ClusterRole with a specific Service Account

Namespace

Have an understanding about namespace.

- Know how to create a namespace.
- Know how to launch objects in a given namespace.
- Know how to search object from a given namespace

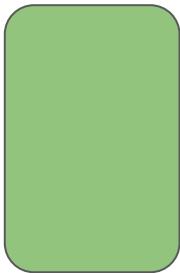
Description	Commands
Create a namespace	kubectl create namespace production
Search objects from a namespace	kubectl get pods -n production

Important Pointers - Domain 4

Storage

Persistent Volumes

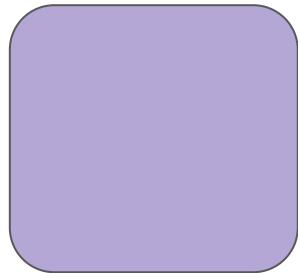
A PersistentVolume (PV) is a piece of storage in the cluster that has been provisioned by an administrator or dynamically provisioned using Storage Classes



Volume 1
Size: Small
Speed: Fast



Volume 2
Size: Medium
Speed: Fast



Volume 3
Size: Big
Speed: Slow



Volume 4
Size: VSmall
Speed: Ultra Fast

PersistentVolumeClaim

A PersistentVolumeClaim is a request for the storage by a user.

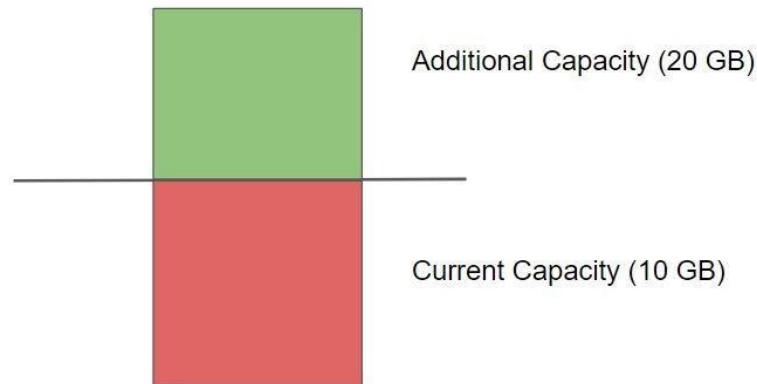
Within the claim, user need to specify the size of the volume along with access mode.

Developer:

I want a volume of size 10 GB which is has speed of Fast for my pod.

Volume Expansion Steps

1. Enable Volume Expansion in Storage Class (allowVolumeExpansion: true)
2. Resize the PersistentVolumeClaim
3. Restart the POD.



Important Note - Volume Expansion

If questions asks you to expand an existing volume and also record the changes, then it is required to make use of --record

example:

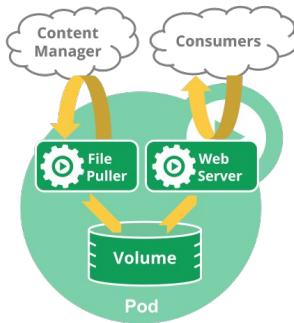
```
kubectl edit pvc my-pvc --record
```

Sidecar Pattern

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

Example Question:

Add a sidecar container for an existing POD busybox. Make sure both of the pods share same mount-points of /var/log.

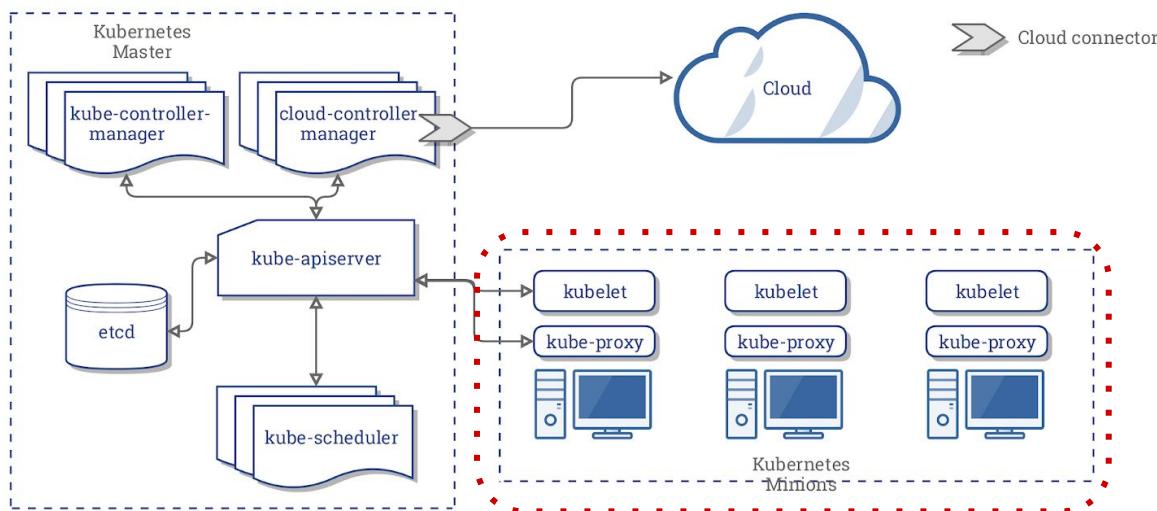


Important Pointers - Domain 5

Troubleshooting

Kubernetes Components

It is important to be familiar with the K8s components.



Important Pointer - Part 01

Know on how you can fetch the logs of a specific pod.

- kubectl get logs pod-name

Be familiar with basic grep functionality.

```
kubectl get logs pod-name | grep "access-denied" > /opt/pod-logs.txt
```

Important Pointer - Part 02

Be aware on how you can drain a worker node.

Example Question:

Drain a worker node kpabs-worker-01. Ensure that all the pods are migrated to a newer node.

kubectl drain node-name << **This will not work.**

kubectl drain kplabs-worker-01 --ignore-daemonsets --delete-local-data

Important Pointer - Part 03

Be familiar with Kubernetes Metric Server

- Know on how to get top utilization metrics of a pod
 - Know on how to get top utilization metrics of a node
-
- kubectl top pods
 - kubectl top nodes

Metric Server will be pre-installed so no need to worry.

Important Pointer - Part 04

Be aware on how you can observe a node based on taints.

Example Question:

Find all the node which do not have taints associated with them. Store the value in /opt/taint.txt

Easiest way:

```
kubectl describe node [node-name] << observer for taints.
```

Sample Question 1 - Kubernetes the Hard Way

Q. Fix a Broken Kubernetes Cluster

1. Worker Node is in NotReady state
2. Fix it.