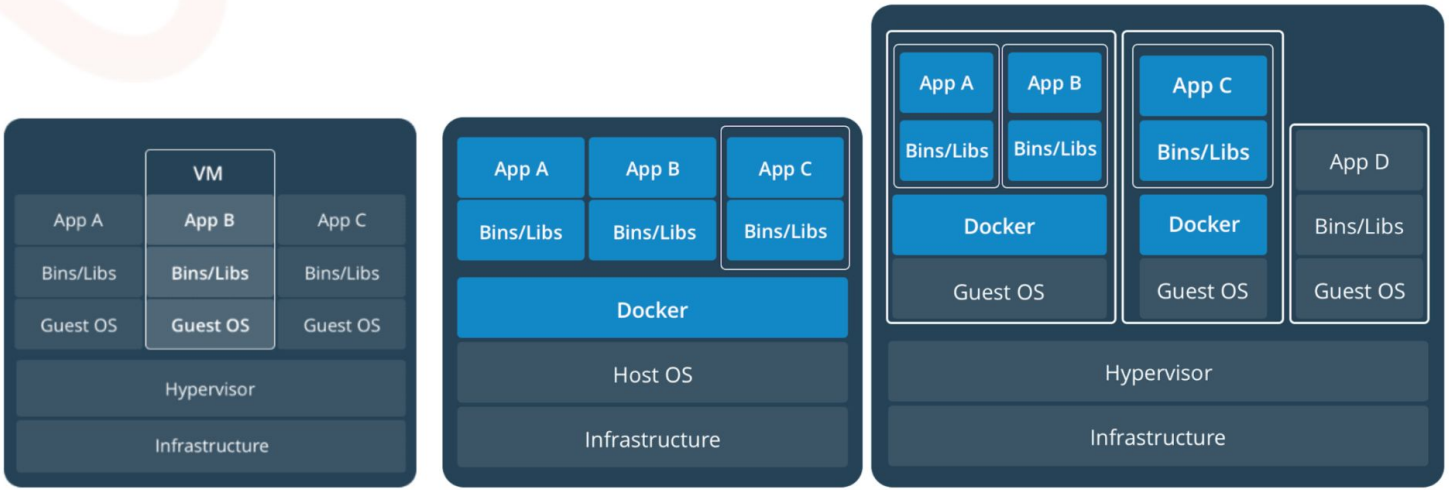
# CKA Course

By Nilesh Jayanandana

# Containers vs VMs




VM Container Hybrid

# Containers

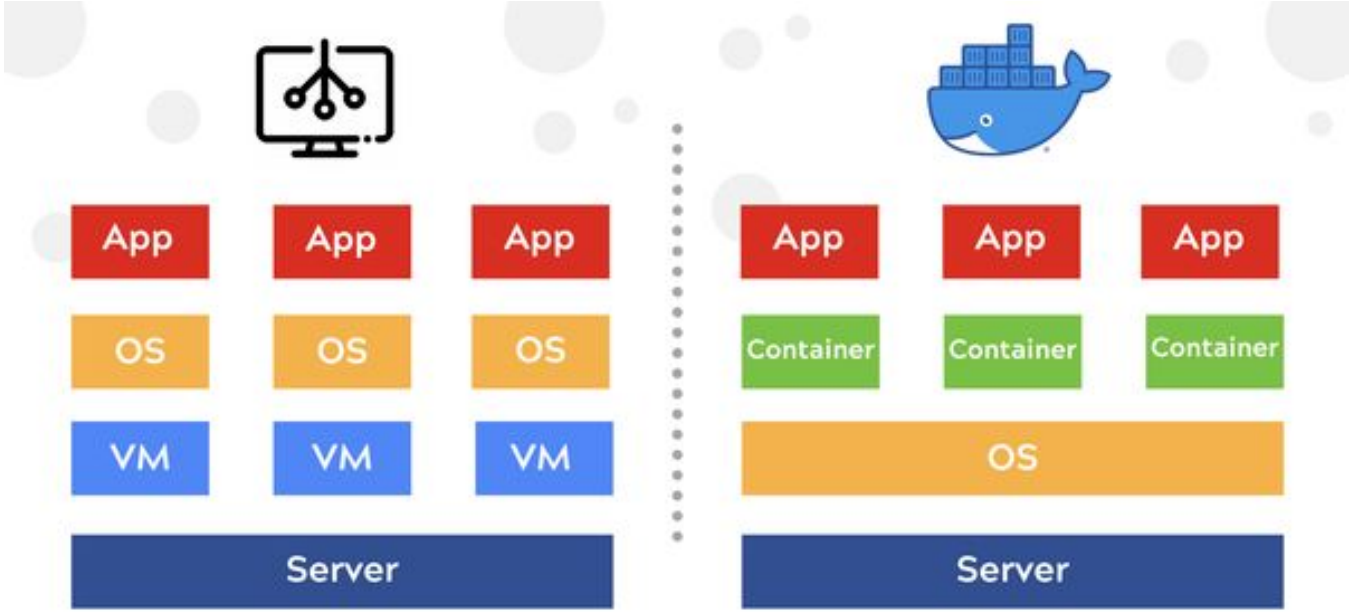
Containers are a solution to the problem of how to get software to run reliably when moved from one computing environment to another.

High Level Container Runtimes

- ContainerD
- RKT
- CRI-O

For more reference:
https://www.ianlewis.org/en/container-runtimes-part-1-introduction-container-r

# Containers and Microservices



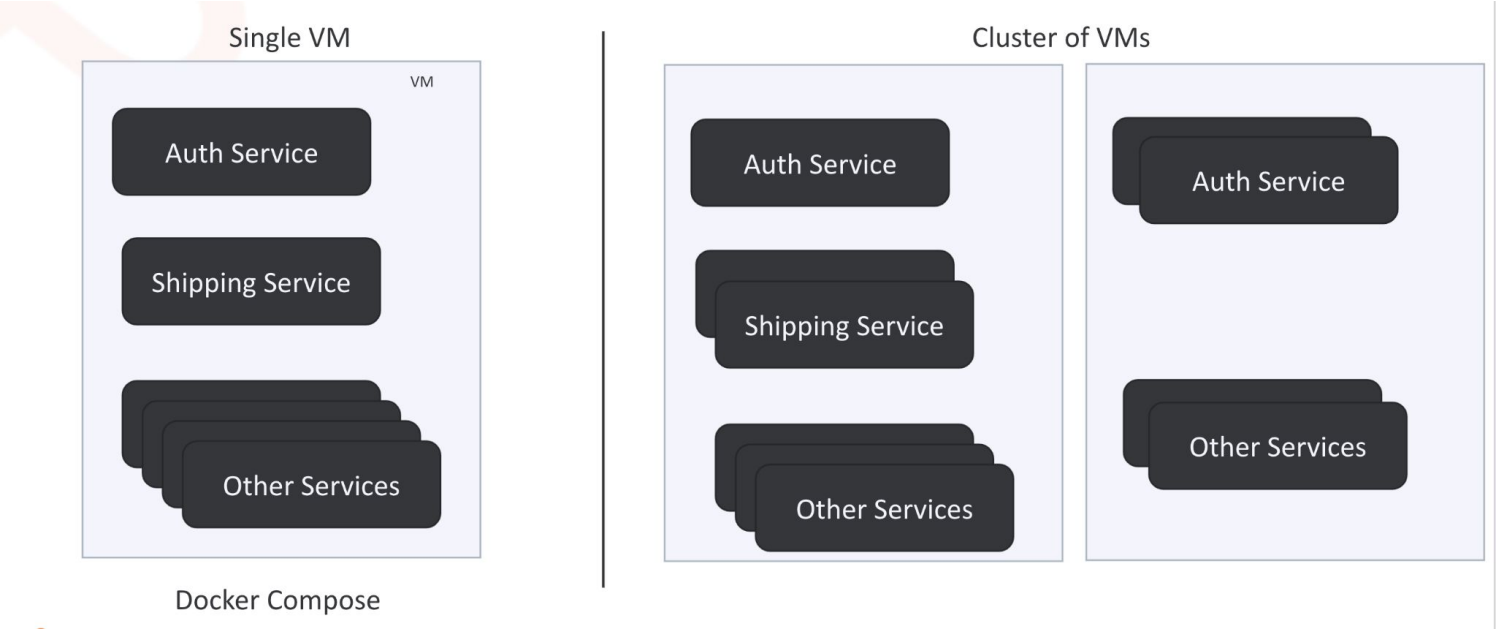
# New Problems with Micro Services

- Docker CLI Allows you to manage the lifecycle of containers  manually
- Docker Compose Allows you to run multi container systems

Is that enough??

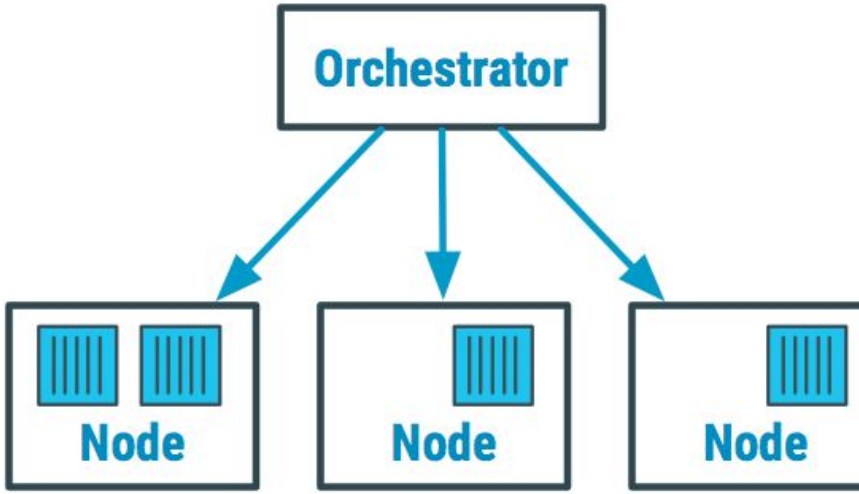
Think about running 1000s of containers and managing them through Docker CLI!!

# Docker Compose vs Orchestration



# Container Orchestration

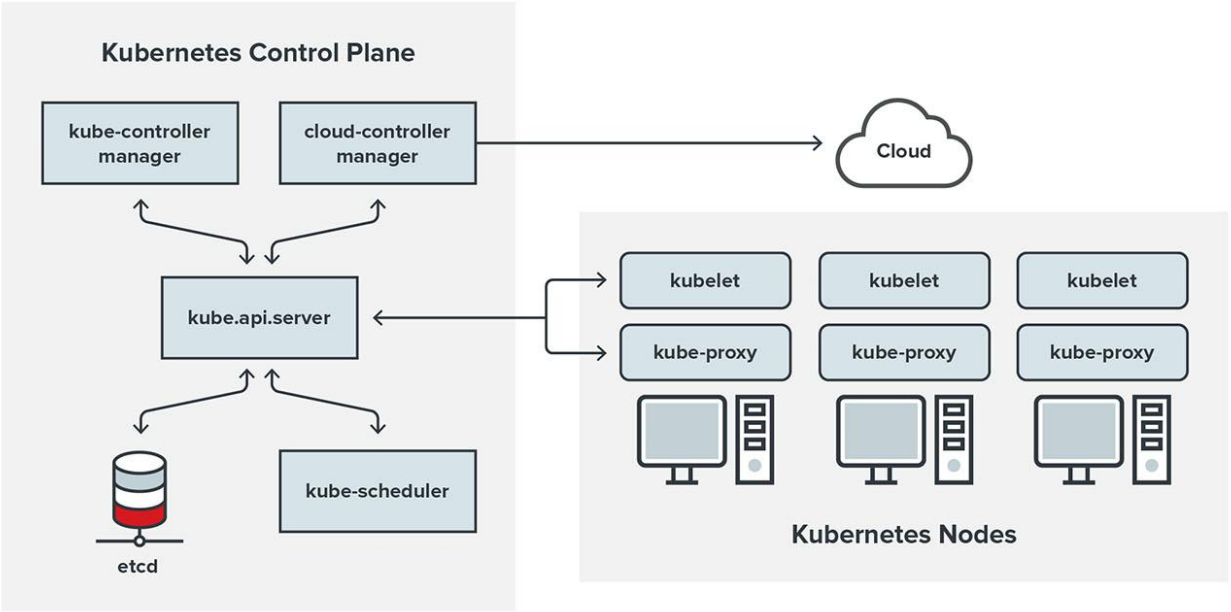
- Resource allocation
- Scheduling
- Scaling
- Load Balancing
- Service Discovery
- Manage Network & Access
- Track State
- Availability



# Kubernetes

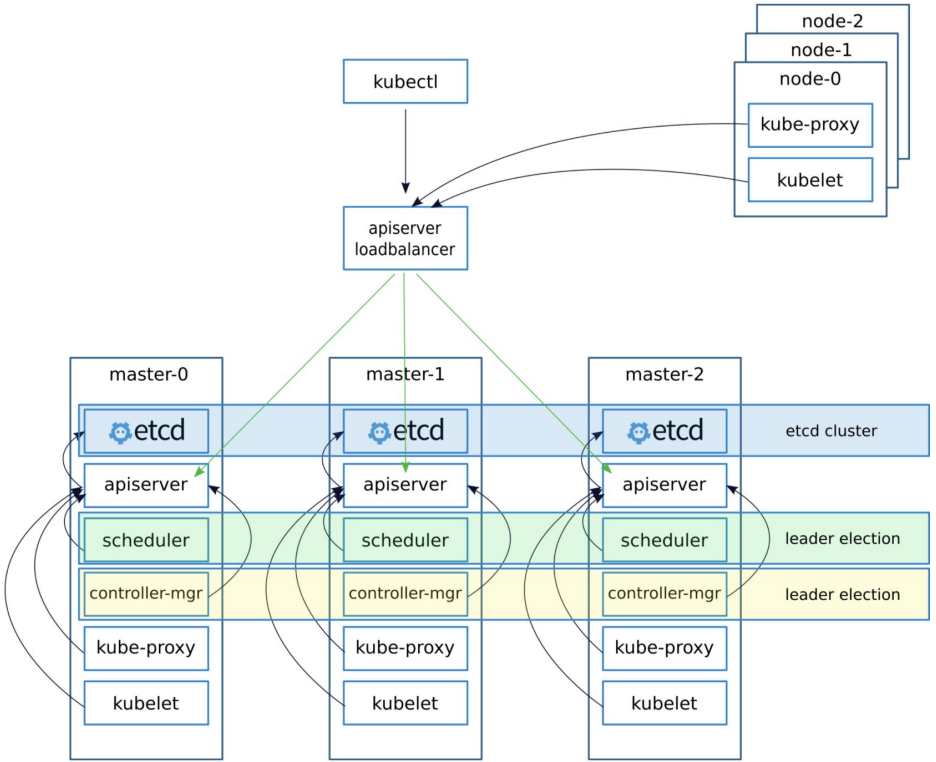
# Kubernetes Components

- etcd
- Kubelet
- Scheduler
- Controller Manager
- Kube-DNS (Core dns)
- Kube-Proxy
- Kube API Server



# Kubernetes Deployment

- Single Master Cluster
- HA with Stacked ETCD
- HA with external ETCD



# API Server

- Authenticate User
- Validate Request
- Retrieve data
- Update ETCD
- Scheduler 6
- Kubelet

# Controller Manager

- Watch Status
- Handle Life Cycle of resources

# ETCD

- Store State
- Key Value Store
- Data at rest encryption needs to be done at API server level
- Can be external to K8s

# Scheduler

- Make Scheduling decisions
- Can have custom schedulers

# Kubelet

- Manage Container Lifecycle
- Needs to run in every node

# Kube Proxy

- Manages internal Kubernetes network
- IP Forwarding and editing route tables
- A Kubernetes CNI is needed to be installed in order for networking to work properly.

# Setup a Kubernetes Cluster with Kubeadm

# Ports Needed to be Open in Kubernetes

- Master
  - 6443 - api server
  - 2379-2380 - etcd
  - 10250 - kubelet
  - 10251 - scheduler
  - 10252 - controller-manager
  - 10255 - kubelet read only
  - 8472 UDP  - kube proxy
  - 30000-32767  - node ports

- Worker
  - 10250 - kubelet
  - 10255 - kubelet readonly
  - 8472 UDP -  kube proxy
  - 30000-32767 - nodeports

# Setup Kubernetes Cluster

1. Install ContainerD/Docker
2. Install Kubeadm, Kubectl
3. Open Ports
4. Initialize Kubernetes master with Kubeadm
5. Initialize Kubernetes nodes with Kubeadm
6. Install CNI

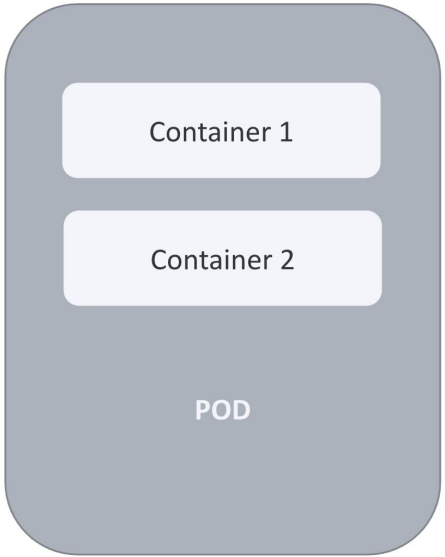
Install Script:
https://gist.githubusercontent.com/nilesh93/fe90c8d2137bc24d32479e4fae64c558/raw/9db75cbf4e211c23c9647dc41d1d29e45fc16f41/kubernetes-prerequisites-ubuntu.sh

CNI Reference: Additional Reading
https://www.slideshare.net/JurajHantak/4-cncf-kubernetes-comparison-ofexistingcnipluginsforkubernetes?from_action=save

# Pods and Deployments

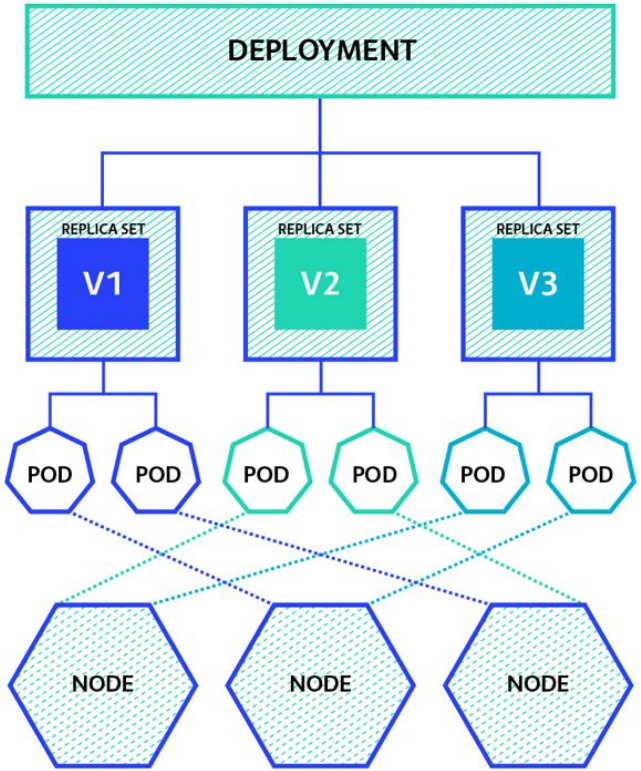
# Pods

- Atomic Unit in Kubernetes
- Similar to VM in a Cloud Environment
- Has an IP address and resources are shared among containers
- IMMUTABLE – Can only be created or deleted

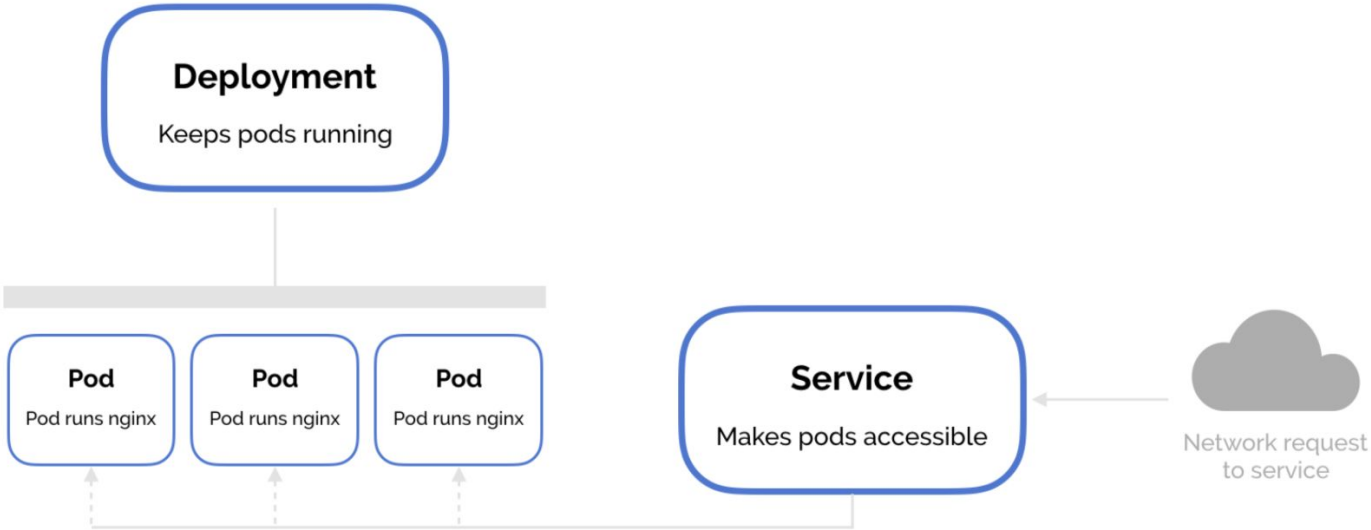


# Deployments

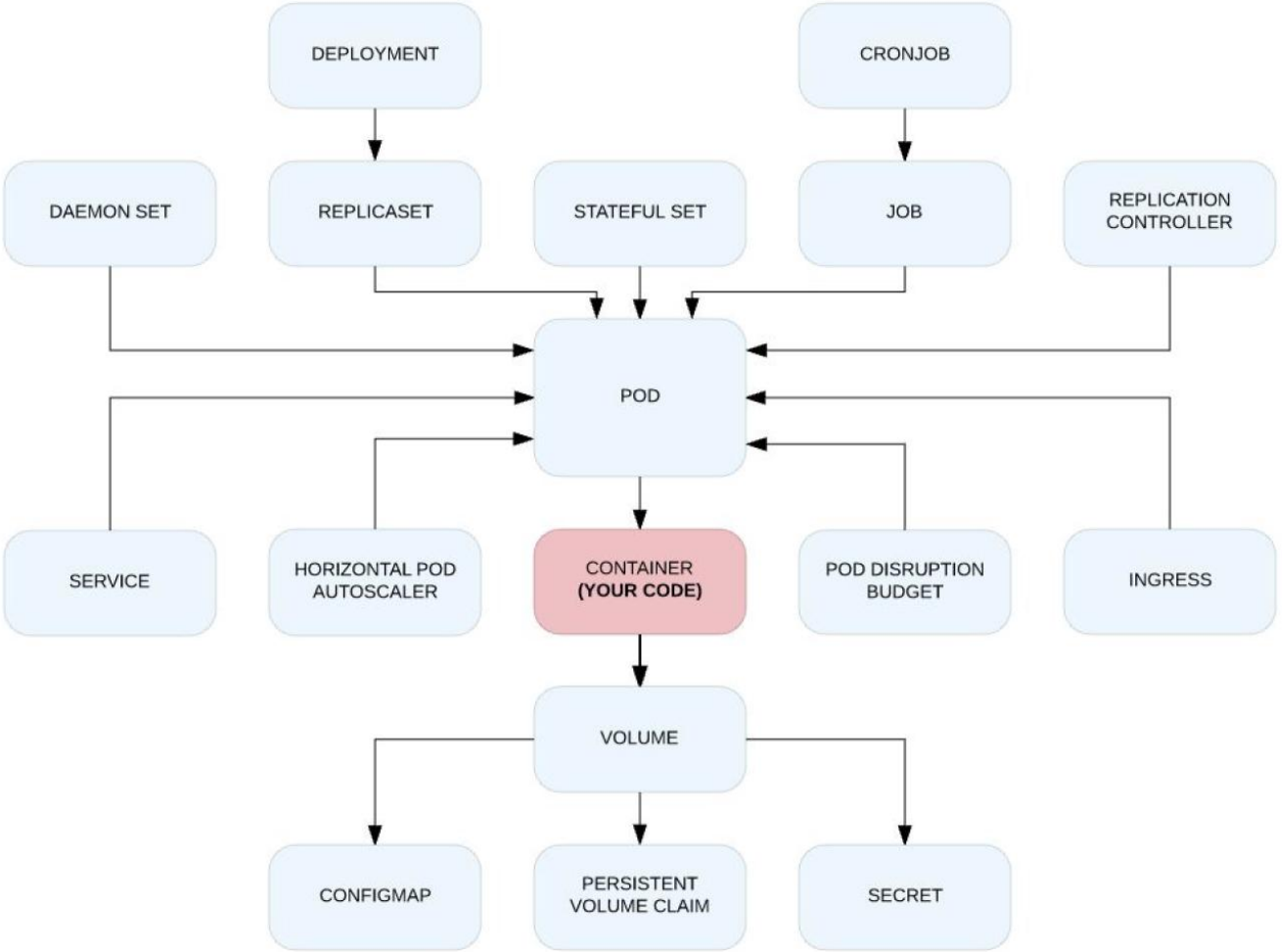
- Manages Pods
- Allows Rollbacks
- Rolling Updates
- **Better suited for Stateless applications**
- Replicaset is a snapshot of a deployment version



# Pods vs Deployments vs Services



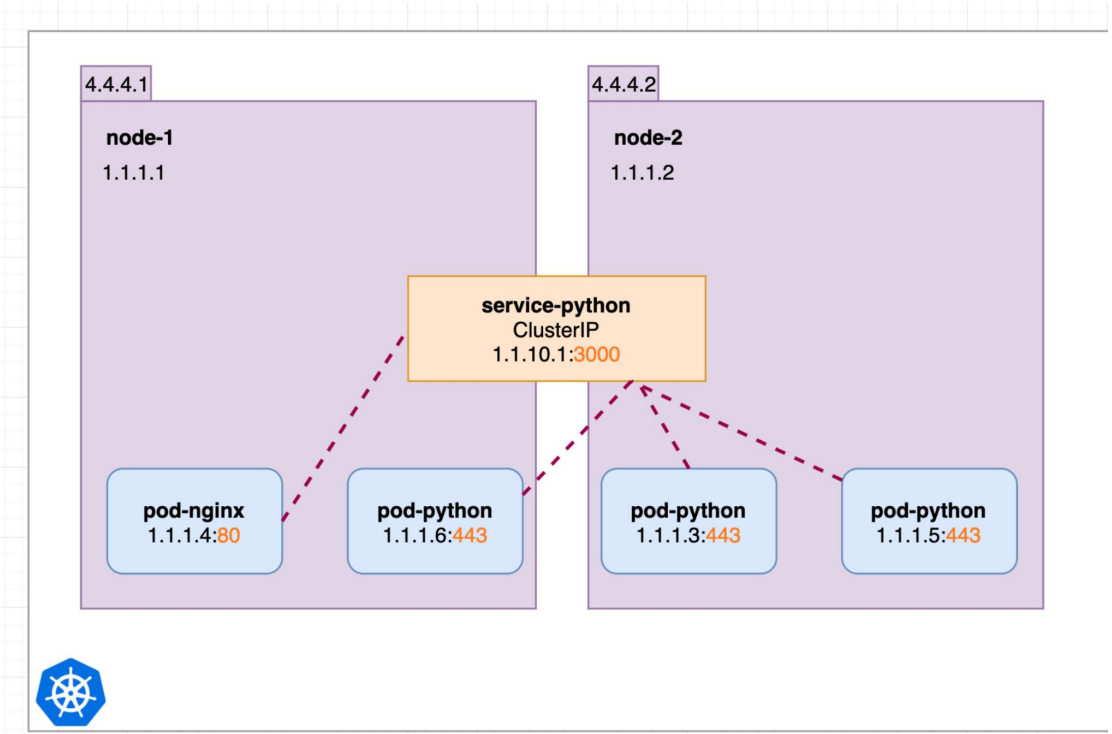
# Kubernetes Resources Brief Overview



# Services

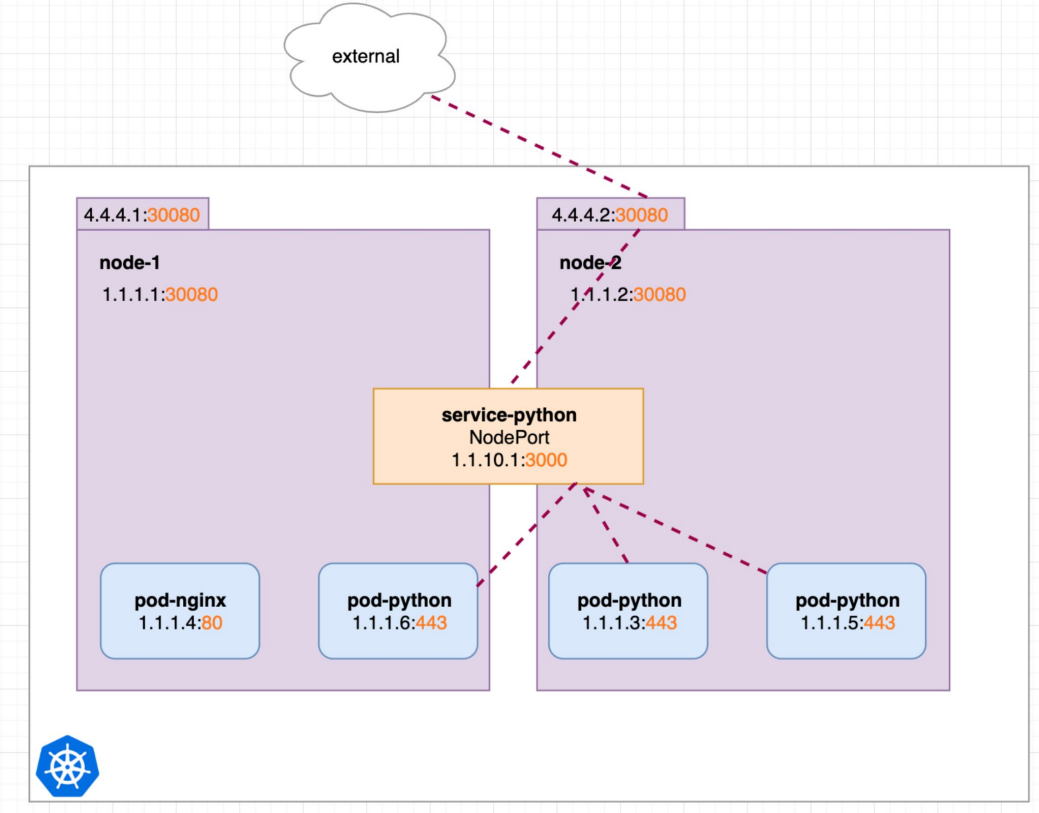
# Services - ClusterIP

- Service is accessible internally



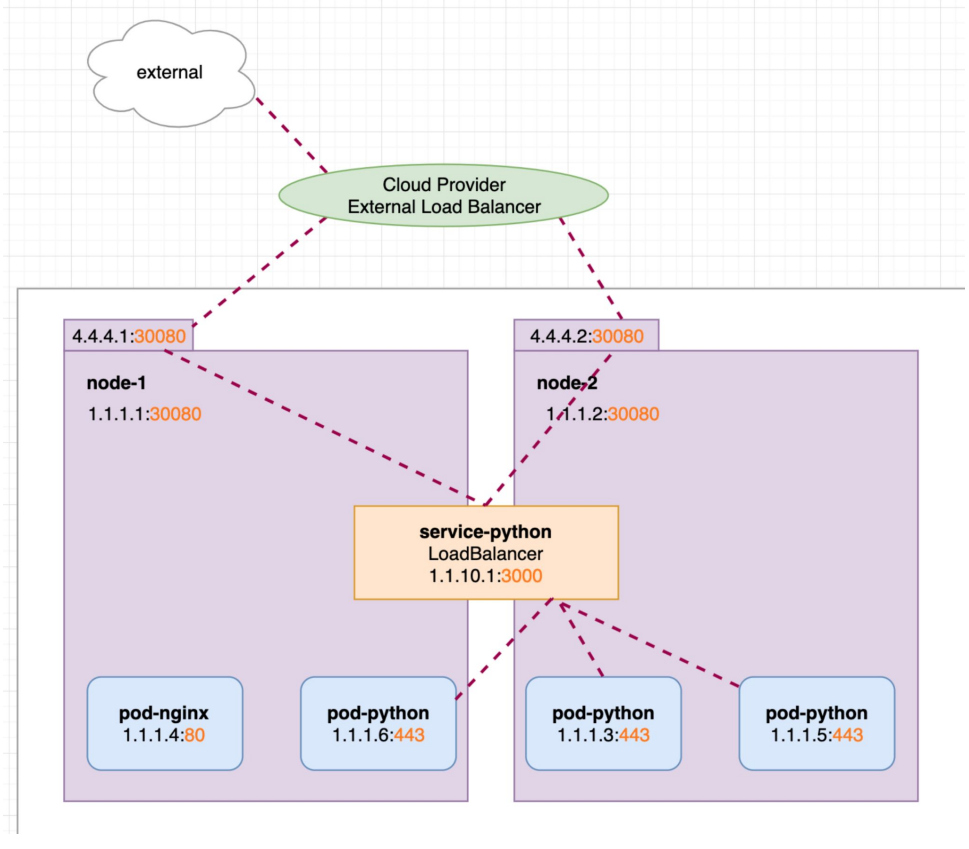
# Services - NodePort

- Expose services externally over TCP
- Same port is opened on all nodes and will forward to service



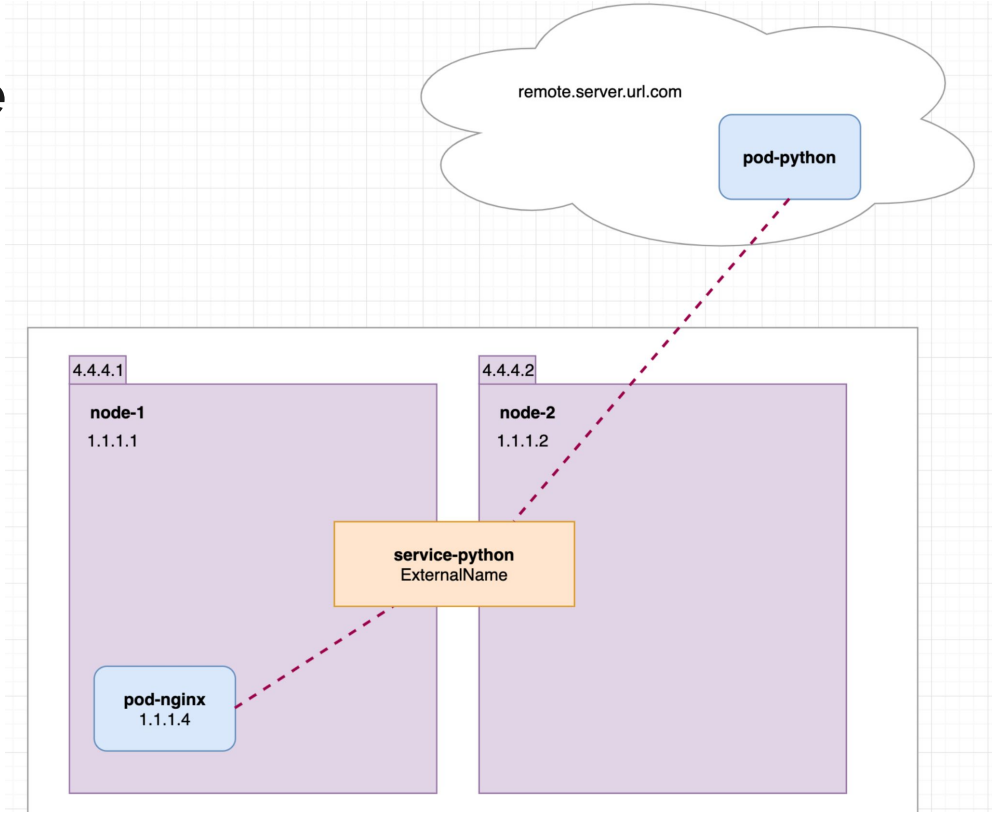
# Services - LoadBalancer

- Automatically provision Cloud LB resource and connect to nodeports



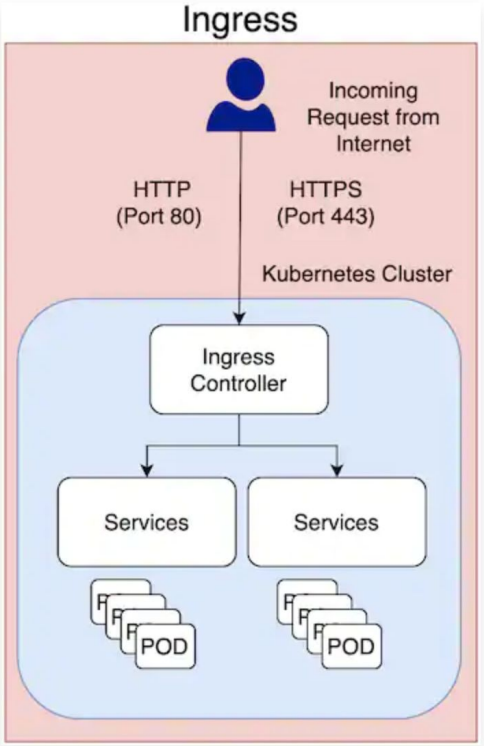
# Services - ExternalName

- Point to an external DNS as a native service

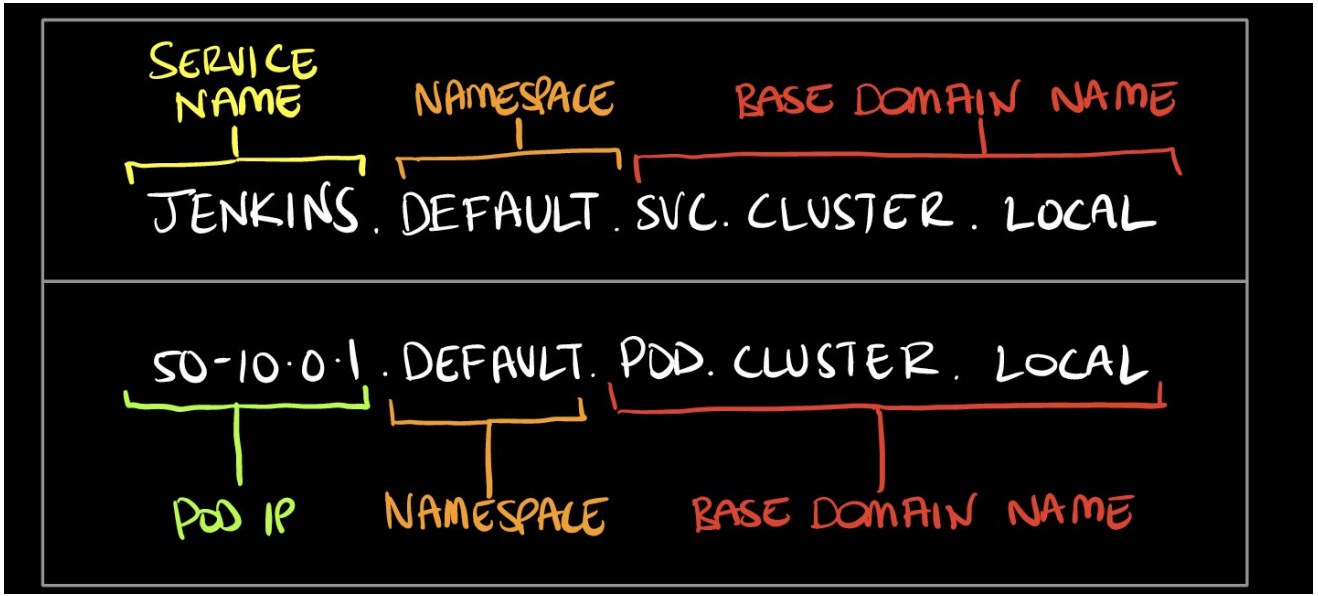


# Ingress

- Works at Layer 7
- Needs to Have an Ingress Controller
  - Nginx
  - HAProxy
  - Contour
  - Envoy
- Support TLS termination



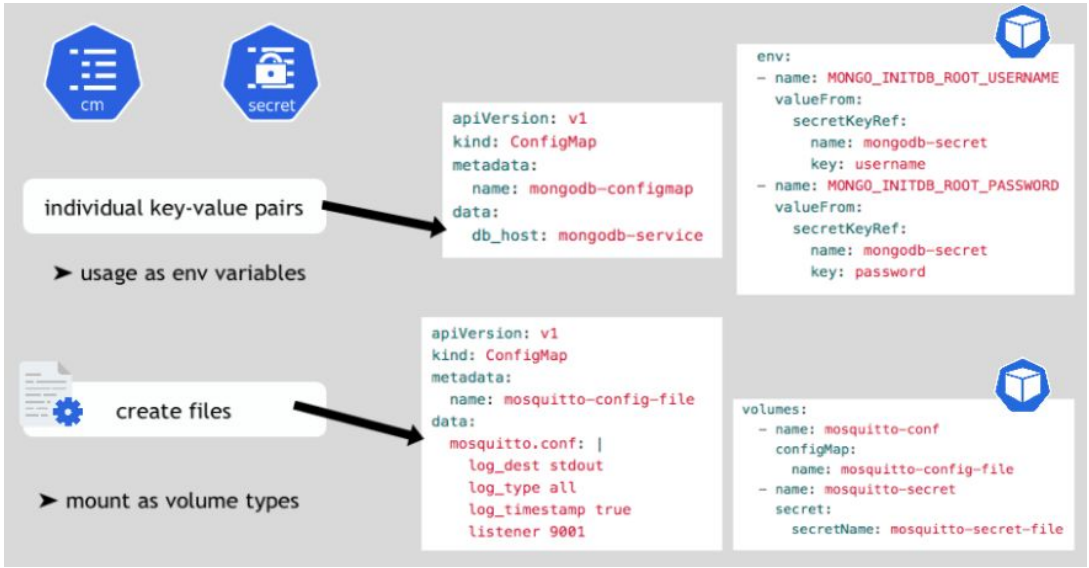
# Cluster DNS


SERVICE NAME
NAMESPACE
BASE DOMAIN NAME
JENKINS . DEFAULT . SVC . CLUSTER . LOCAL
50-10-0-1 . DEFAULT . POD . CLUSTER . LOCAL
POD IP
NAMESPACE
BASE DOMAIN NAME

# Configmaps and Secrets

# Configmaps

- Key Value Pairs
- Can be injected as environment variables
- Can be mounted as a directory or a file



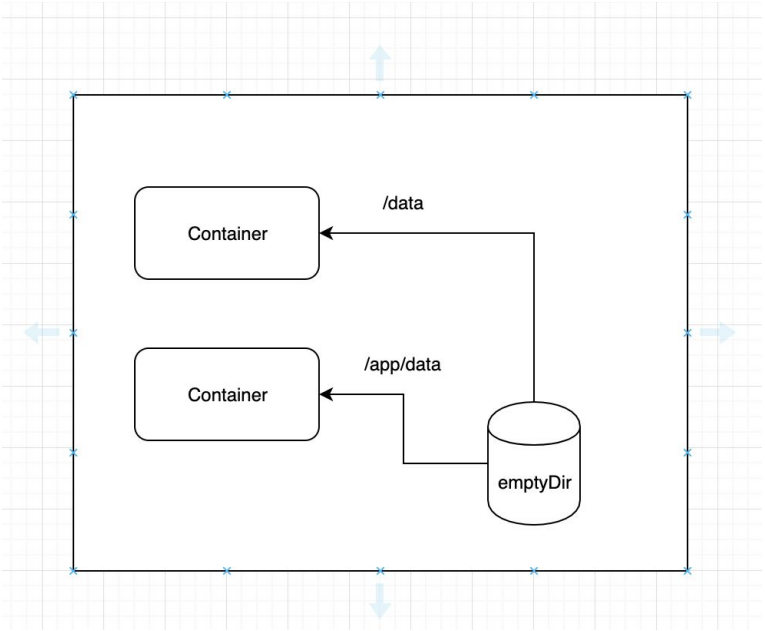
# Secrets

- SAME as Configmaps, but with the exception of data being saved as base64
- Different types
  - Opaque
  - Dockerconfig
  - Tls
  - Sshkey
- Mounted into the pod/container as text and NOT as base64

# Volumes

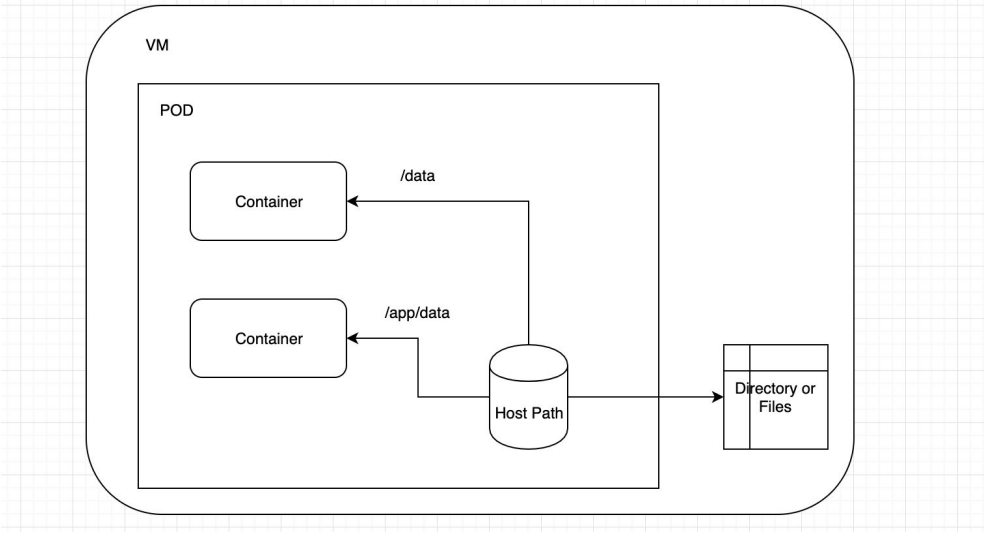
# Volumes - EmptyDir

- In memory
- Used to share data between 2 containers in the same pod
- Data is not persisted



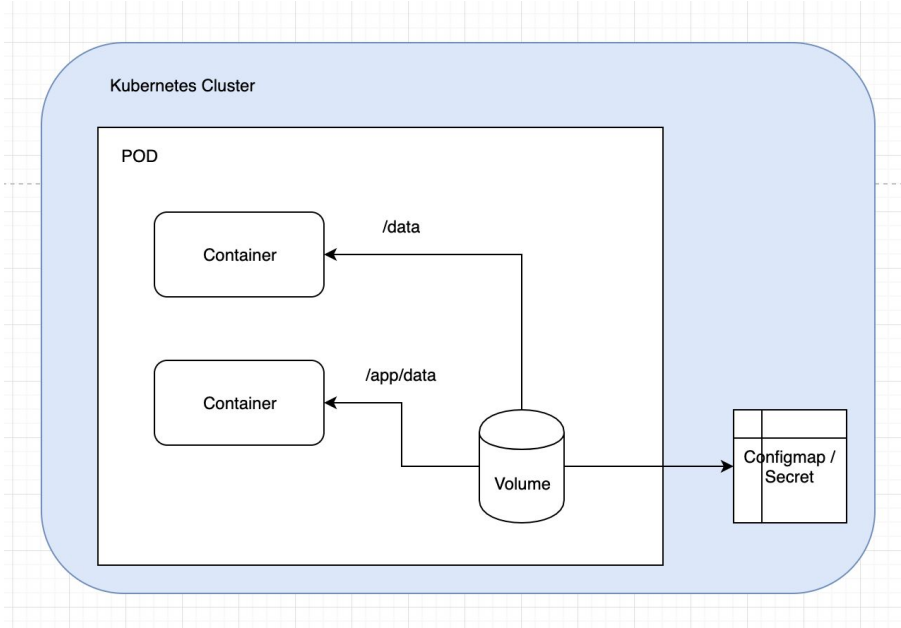
# Volumes - HostPath

- Mount a directory in host machine into containers
- Data is persisted



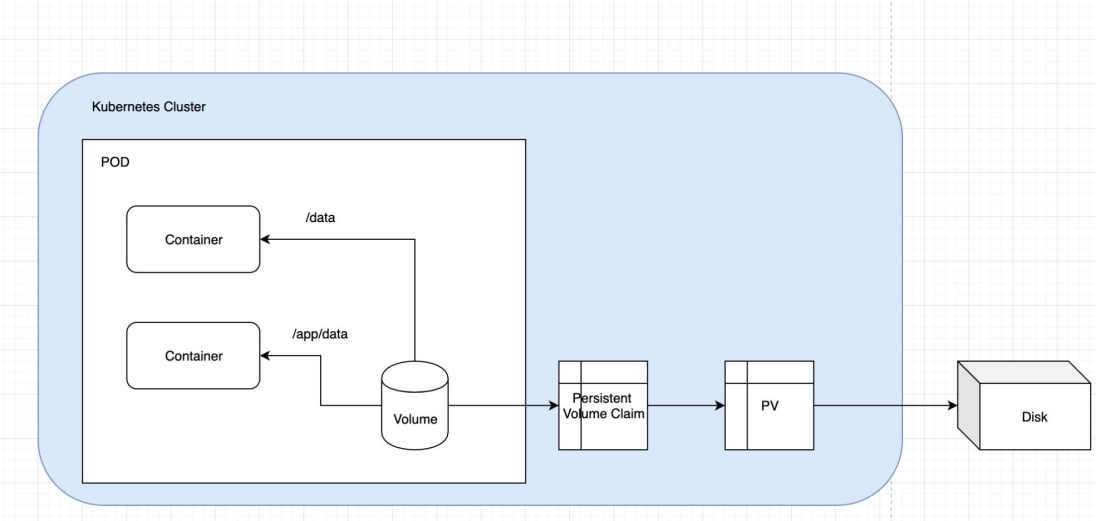
# Volumes - Configmap/Secret

- Read only
- Can be mounted as a file or a volume
- Changes to these resources are reflected inside the containers at realtime



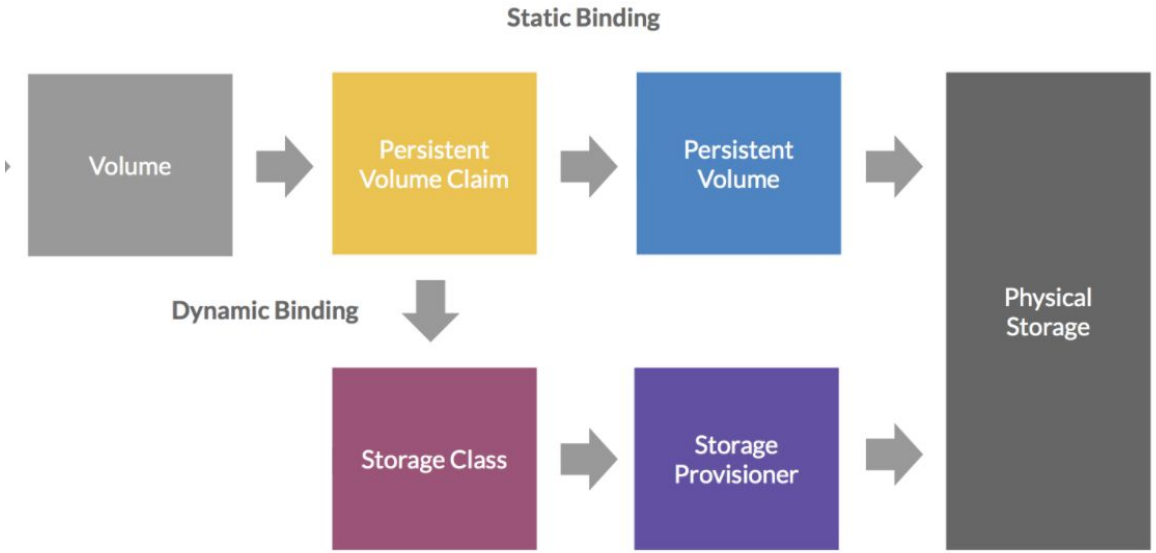
# Volumes - PVC and PV

- Persistent method
- 3 modes
  - ReadOnly
  - ReadWriteOnly (commonly supported)
  - ReadWriteMany (only special volumes support this)



# Storage Class

- Dynamically provision PV and underlying physical storage



# Cluster Upgrade and Maintenance

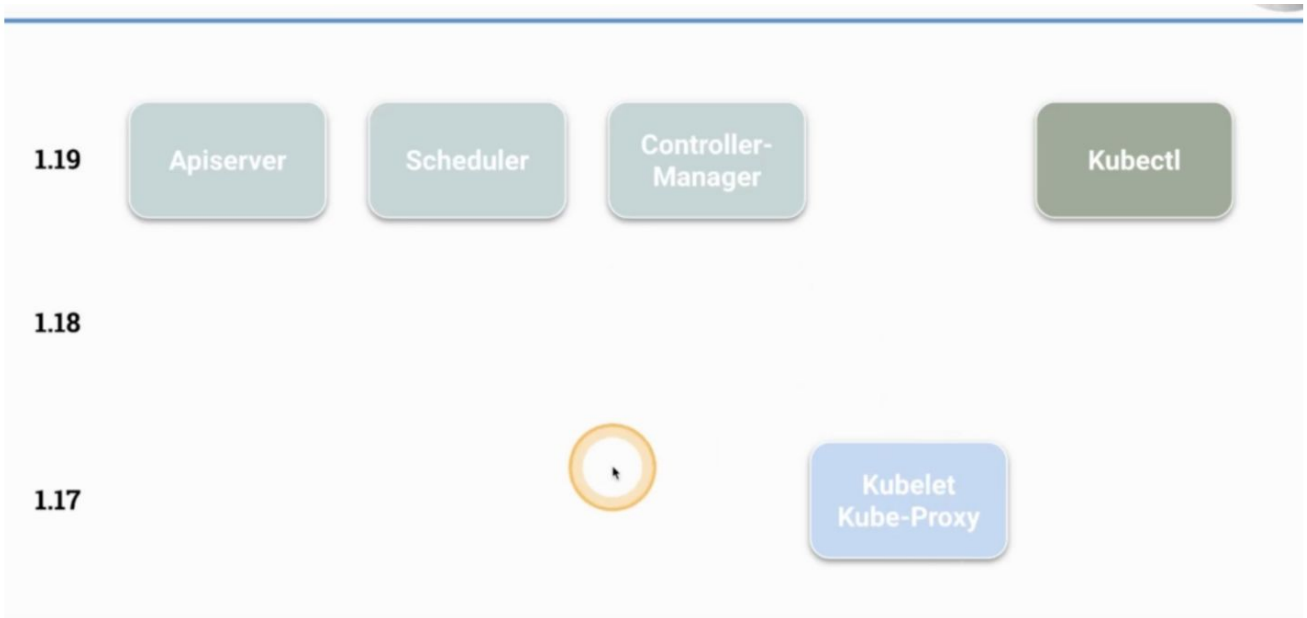
# Cluster Upgrade

- Upgrade master
  - Kubectl drain master --ignore-daemonsets
  - Update kubeadm, kubelet and kubectl
  - Kubeadm upgrade plan
  - Kubeadm upgrade apply
  - Kubectl uncordon master
- Then worker nodes
  - Kubectl drain worker --ignore-daemonsets
  - Update kubeadm
  - Kubeadm upgrade node
  - Update kubelet
  - Kubeadm uncrodon worker

# Possible different versions

Kubelet can be 2 minor versions under the API server.

But as a rule of thumb, always stick to same versions



# ETCD Backup and Restore

```
ETCDCTL_API=3 etcdctl --endpoints 10.2.0.9:2379 \
 --cert=/etc/kubernetes/pki/etcd/server.crt \
 --key=/etc/kubernetes/pki/etcd/server.key \
 --cacert=/etc/kubernetes/pki/etcd/ca.crt \
 <command>
```

Backup

```
snapshot save snapshotdb
```

Verify

```
--write-out=table snapshot status snapshotdb
```

https://kubernetes.io/docs/tasks/administer-cluster/configure-upgrade-etcd/

# Scheduling

# Scheduling

- Does node have adequate resources
- Is the node running out of resources (conditional taints)
- Does the pod request a specific node
- Does the node have a matching label
- If Pod requests a volume, can it be mounted
- Does the pod tolerate taints by the node
- Does the pod have affinity rules

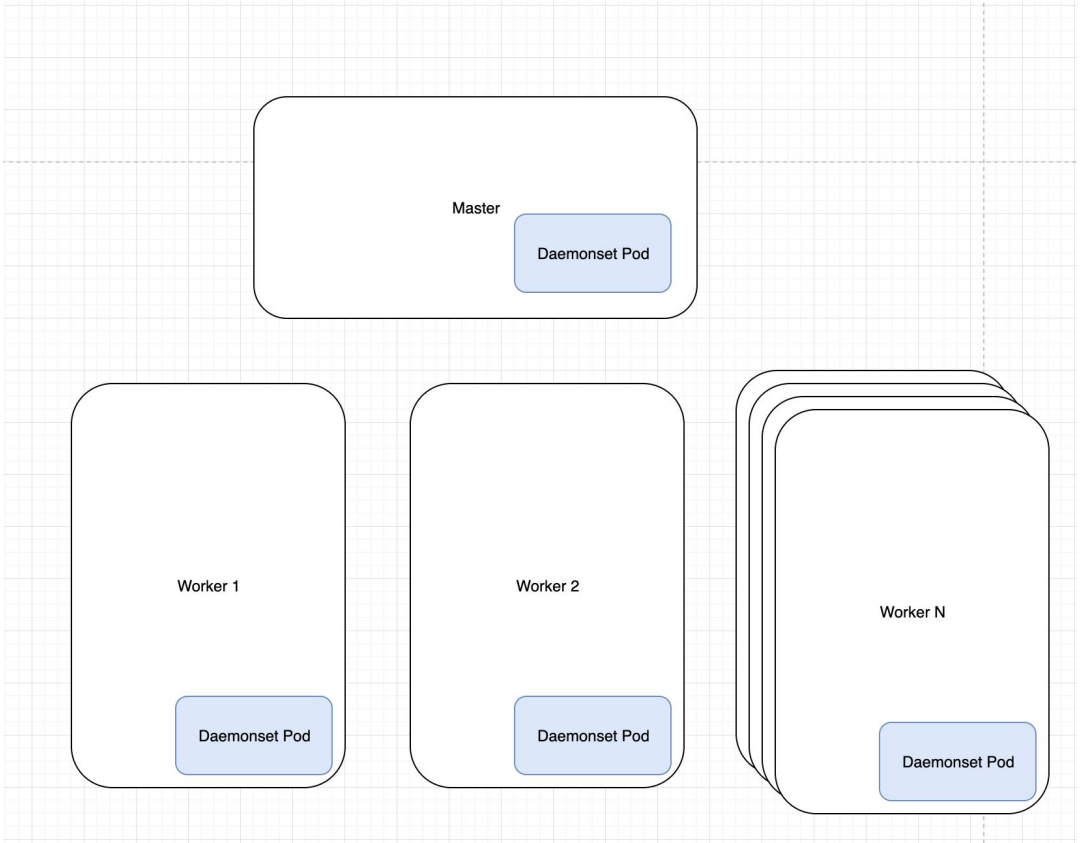
# Scheduling

- Pod
  - NodeSelectors - select from node labels
  - NodeName - specific node
  - Tolerations - override a node taint
  - Affinity - Set advanced rules on how pods can be scheduled
  - schedulerName - use custom schedulers
- Node
  - Taints - disable scheduling

StatefulSets
DaemonSets
Jobs
CronJobs
HPA

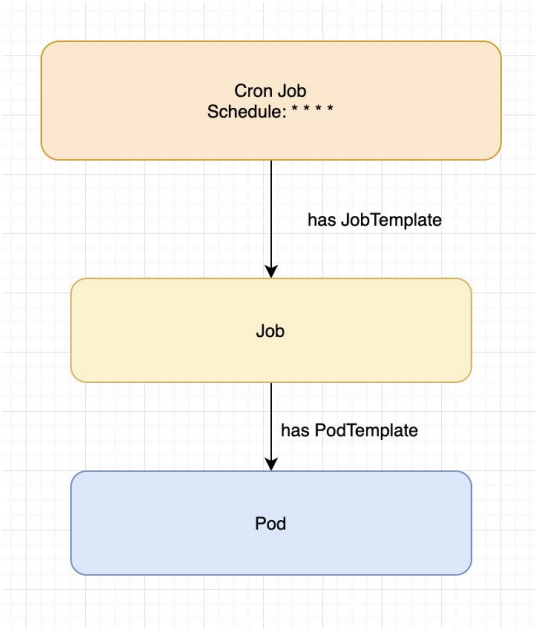
# DaemonSets

- No Replicas
- Runs a pod in Every current and Future Nodes
- Examples
  - Kube-Proxy
  - CNI Plugin
  - Logging Agent



# Job and CronJob

- A container with a process that has a termination
  - Batch Job
  - Database migration script
- JOBS are immutable just like pods
- CronJob Ref
  https://crontab.guru/#*/10 * * * *

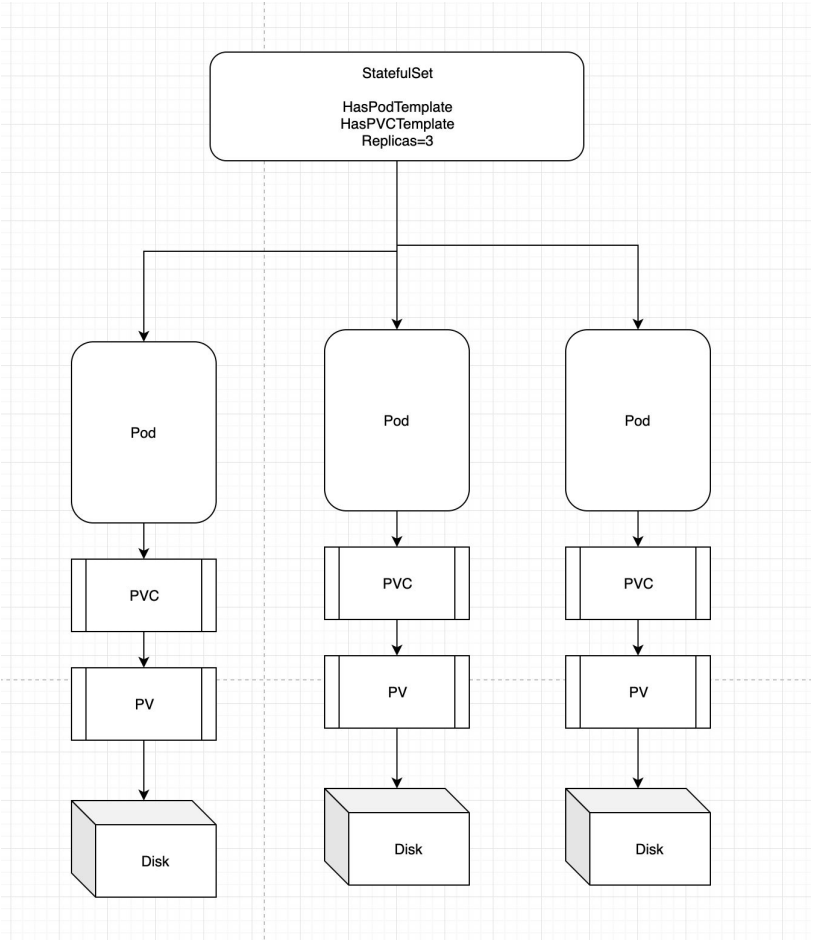


# StatefulSet

- Stable, unique network identifiers.
- Stable, persistent storage.
- Ordered, graceful deployment and scaling.
- Ordered, automated rolling updates.

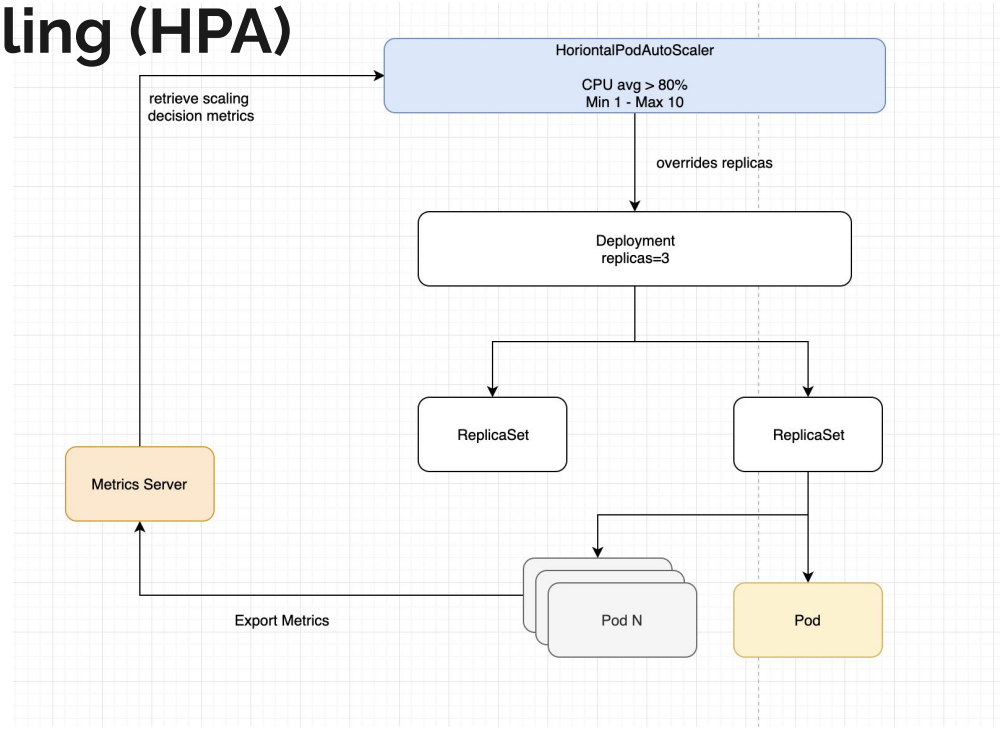
Needs a headless service to operate. (for inter pod discovery and communication)
<pod-name>.<svc-name>

***For headless Services, a cluster IP is not allocated, kube-proxy does not handle these Services, and there is no load balancing or proxying done by the platform for them. How DNS is automatically configured depends on whether the Service has selectors



# Horizontal Pod AutoScaling (HPA)

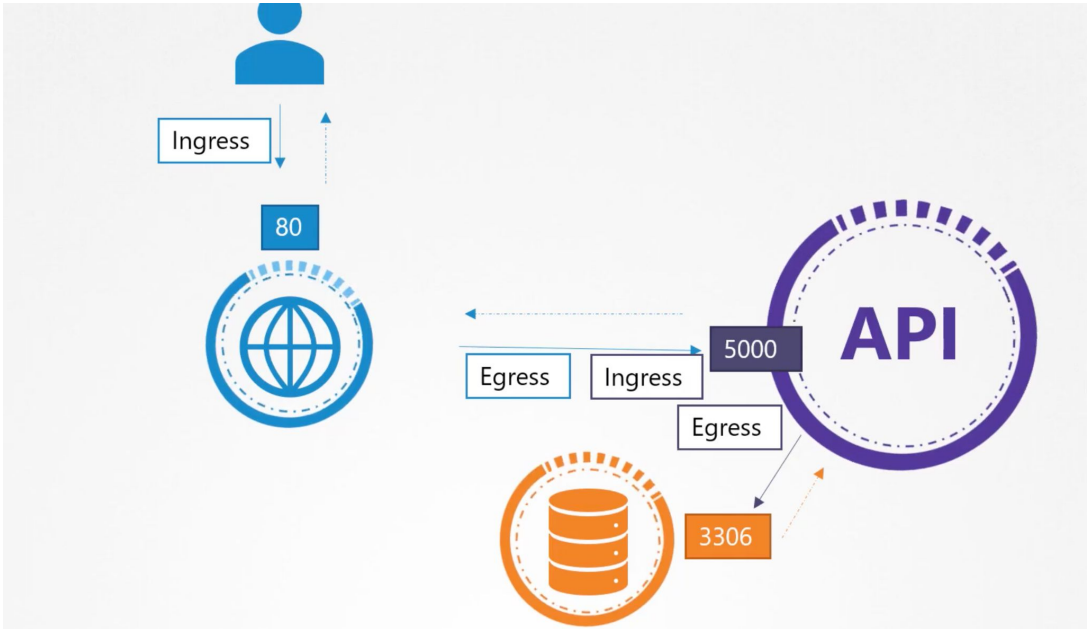
- Metrics server needs to be installed
- Overrides replica count of deployment
- CPU Utilization and Memory Utilization can be used as a threshold
- Min, Max needs to be provided
- Can’t scale to Zero



# Network Policies

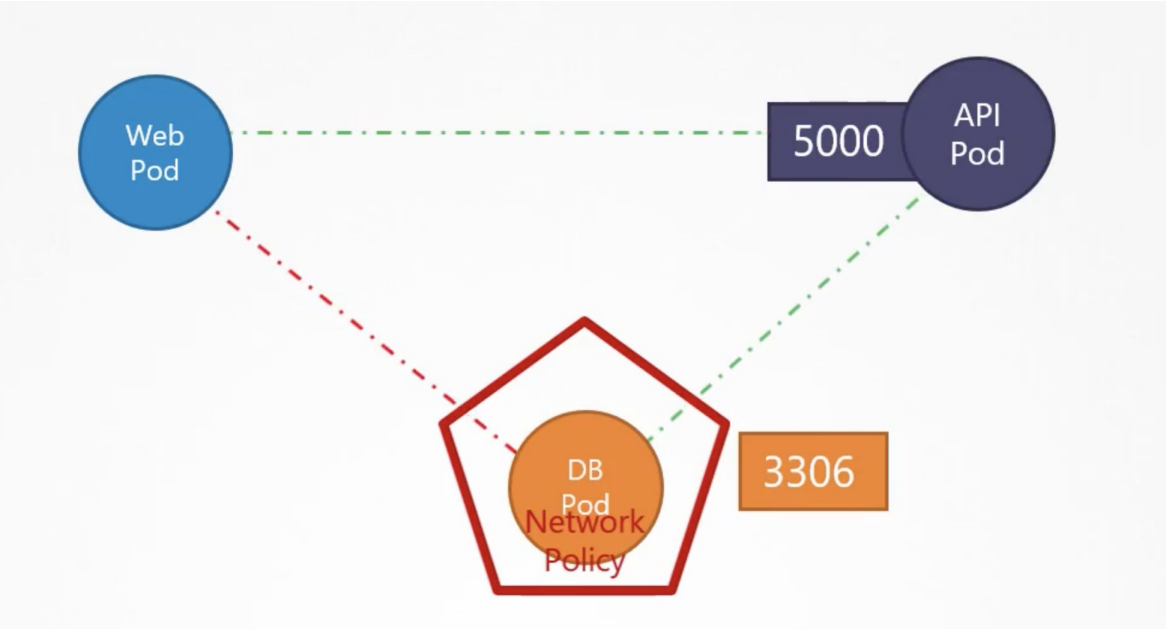
# Network Policies

- Ingress - Traffic coming in to the pod
- Egress - Traffic going out of the pod
- Can limit via
  - Pod Selectors
  - Namespace selectors
  - IP Ranges



# Traffic Rules

# Traffic Rules with Network Policy




```yaml
policyTypes:
- Ingress
ingress:
- from:
  - podSelector:
      matchLabels:
        name: api-pod
  ports:
  - protocol: TCP
    port: 3306
```

# Network Policy Full Example

Additional Information:
https://kubernetes.io/docs/concepts/services-networking/network-policies/

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: db-policy
spec:
   podSelector:
     matchLabels:
       role: db
   policyTypes:
   - Ingress
   ingress:
   - from:
     - podSelector:
         matchLabels:
           name: api-pod
     ports:
     - protocol: TCP
       port: 3306
```

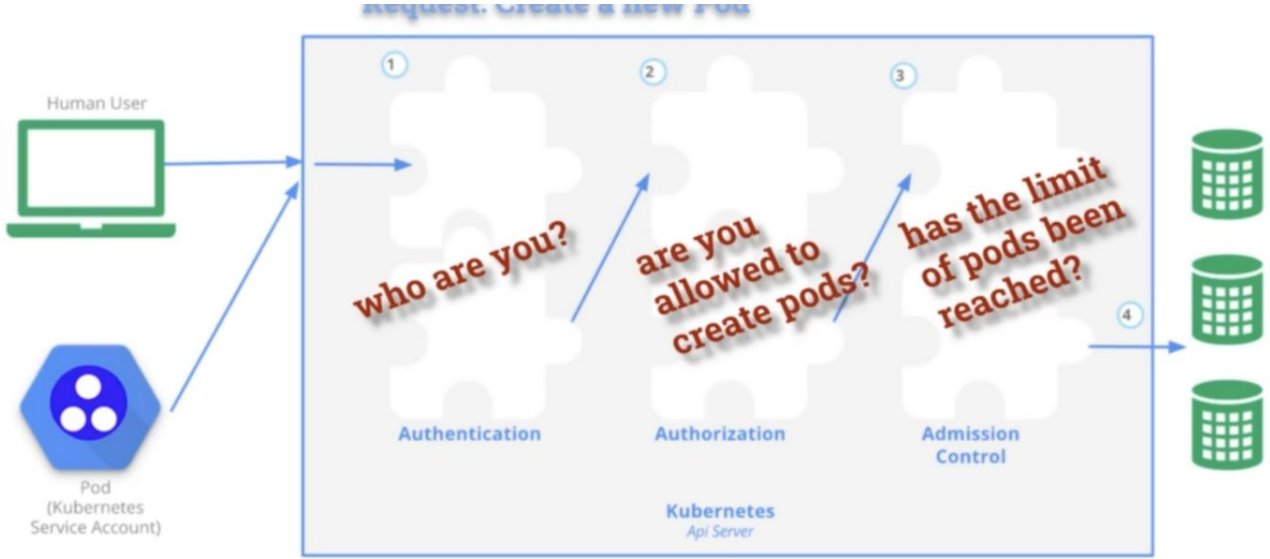
# CNIs that Support Network Policies

- Calico
- Cillium
- Kube Router
- Canal
- Weavenet

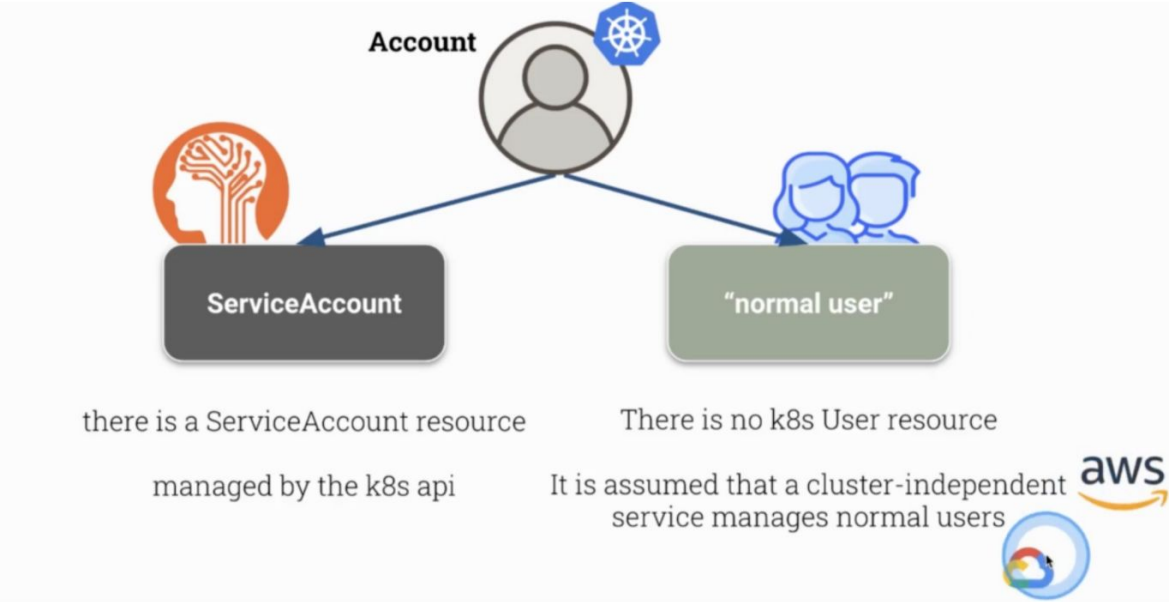
**Flannel Does not support Network Policies

# RBAC

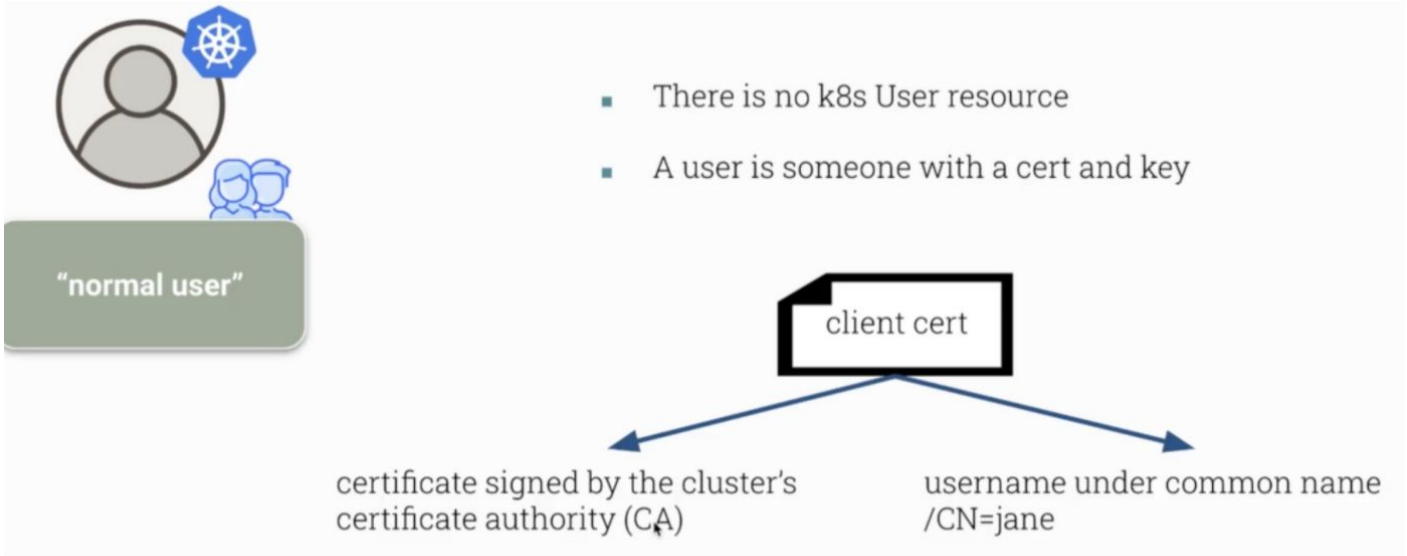
# API Request Flow



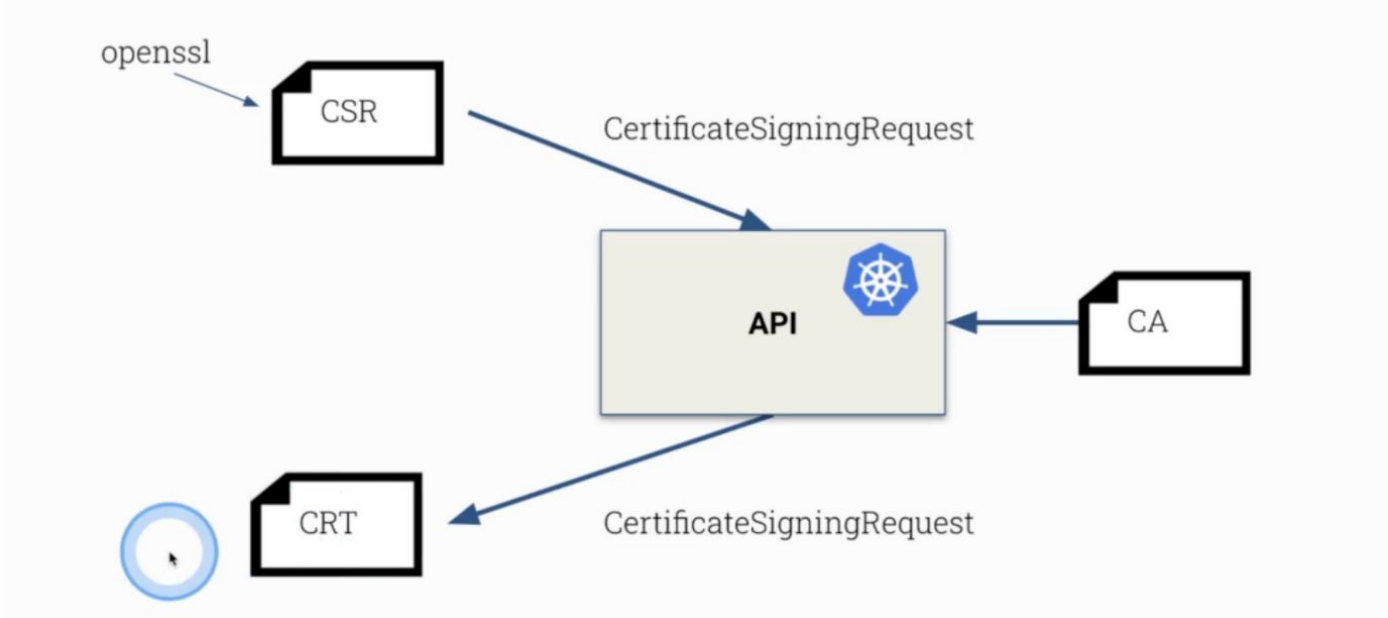
# Kubernetes Users



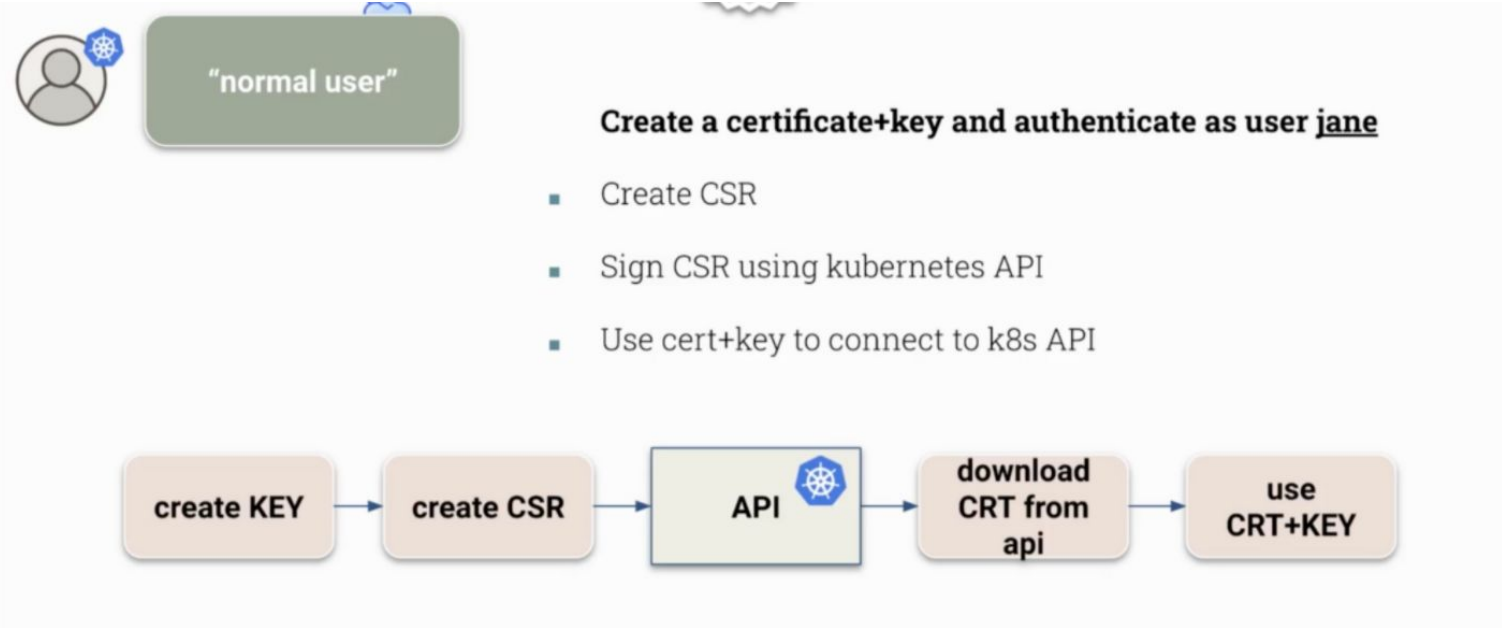
# User Certificates



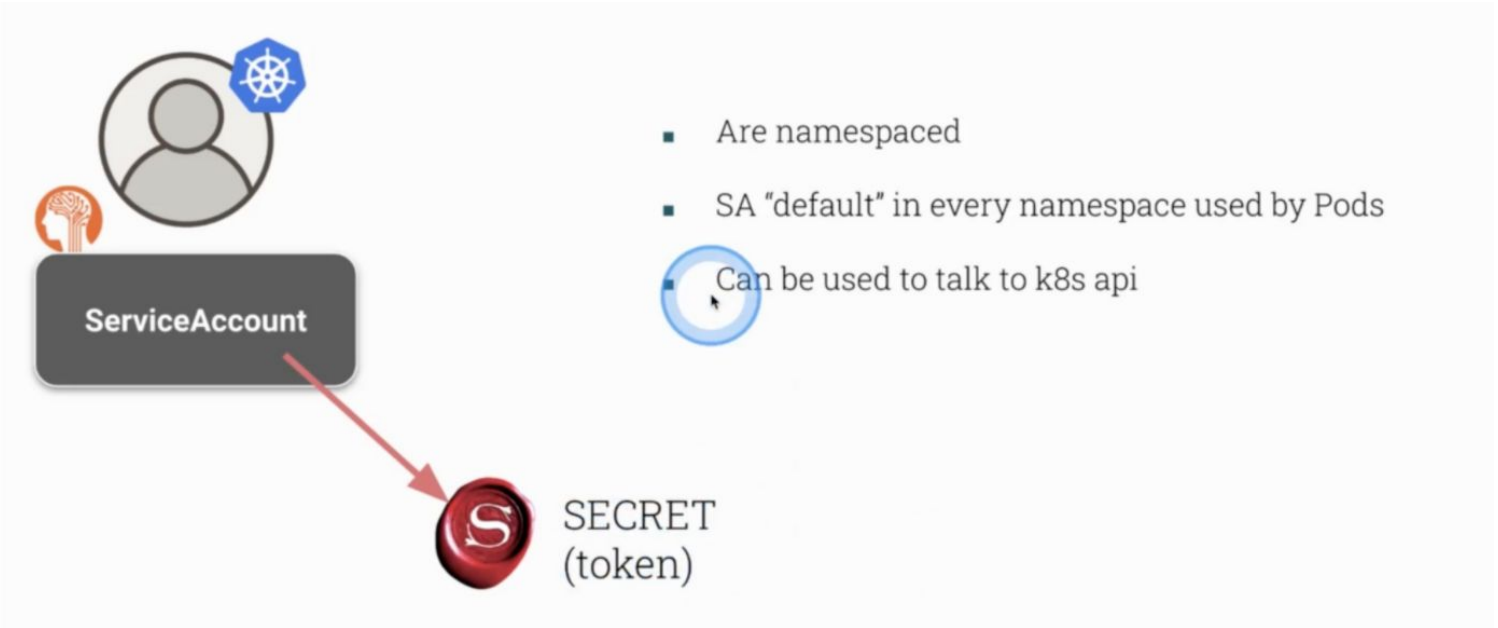
# Certificate Signing



# Users and Certificates



# Service Accounts



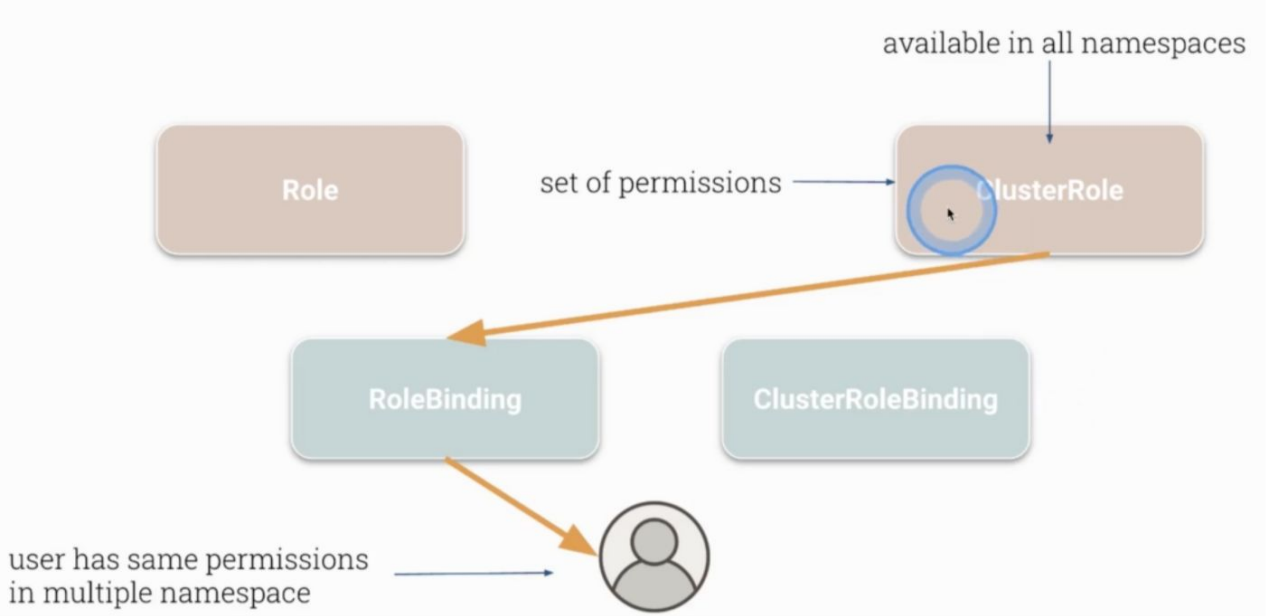
# Roles

ClusterRoles - Available Globally to cluster

Role - Available to a namespace only

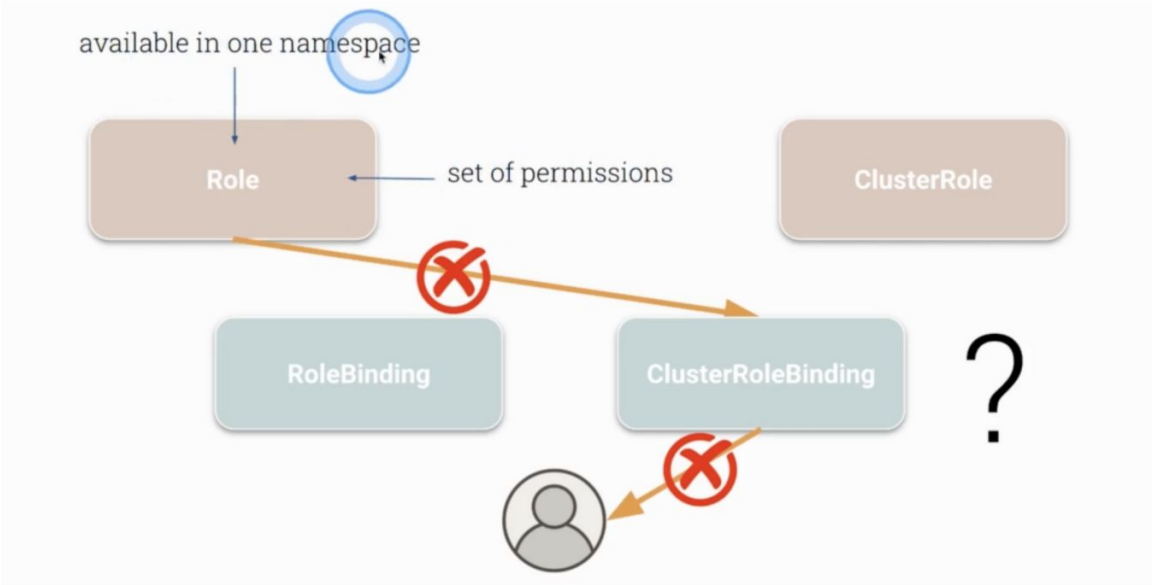
# Role Bindings

Cluster role and a role can both be binded by a role binding to a namespace



# Cluster Role binding

Can only bind cluster roles and binding would give cluster wide permissions



# ResourceQuota

# Resource Quota

Ref: https://kubernetes.io/docs/concepts/policy/resource-quotas/

- Administration tool for a namespace
- Specify or provide a resource Quota to be used in a given namespace.
  - Namespace training cannot exceed the request quota of 4GB memory
- Limits and Requests can be used in quota

# Best Practices

# Pod/Deployment Best Practices

- Always have readiness checks and liveness checks
- Have requests and limits defined
  - Make sure limits don't cause a resource overcommit
- Never run containers as root or privileged
- Have security contexts properly defined
- If you are using deployments with RWO, use recreate strategy instead of rolling update
- Set PodAntiAffinity to make sure replicas having highest availability
- Use PodDisruption Budgets, Resource Quotas and LimitRanges where possible

# CronJobs/Jobs

- Make sure scheduled jobs don’t eat up cluster resources
  - Badly configured CronJob or Job can destroy a cluster in a matter of minutes!
- Make sure to run a Job at most once, avoid parallel runs unless absolutely necessary
- Retry Jobs only if necessary
- Job Cleanup should be enabled

# Cluster

- Upgrade/Run maintenance atleast every 3 months
- Run KubeBench https://github.com/aquasecurity/kube-bench and fix vulnerabilities
- Don’t expose k8s API publicly
- Use a CNI that support Network Policies
- Configure Network Policies to the cluster
- Use RBAC and Workload Identity

# Tips for the Exam

# Bookmarks

- https://kubernetes.io/docs/reference/kubectl/cheatsheet/
- https://kubernetes.io/docs/tasks/administer-cluster/kubeadm/kubeadm-upgrade/
- https://kubernetes.io/docs/tasks/administer-cluster/configure-upgrade-etcd/#backing-up-an-etcd-cluster
- https://kubernetes.io/docs/concepts/storage/volumes/
- https://kubernetes.io/docs/tasks/configure-pod-container/configure-persistent-volume-storage/#create-a-persistentvolumeclaim
- https://kubernetes.io/docs/concepts/services-networking/network-policies/
- https://kubernetes.io/docs/reference/command-line-tools-reference/kube-apiserver/
- https://kubernetes.io/docs/reference/config-api/kubelet-config.v1beta1/
- https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale-walkthrough/#create-horizontal-pod-autoscaler
- https://kubernetes.io/docs/reference/access-authn-authz/rbac/
- https://kubernetes.io/docs/concepts/scheduling-eviction/taint-and-toleration/

# All tips available here

https://medium.com/platformer-blog/how-i-passed-the-cka-certified-kubernetes-administrator-exam-8943aa24d71d

# Thank You

Reach out to me on
Email: nilesh93.j@gmail.com
Linkedin: https://www.linkedin.com/in/nilesh93/