Kubernetes

what

- orchestration tool

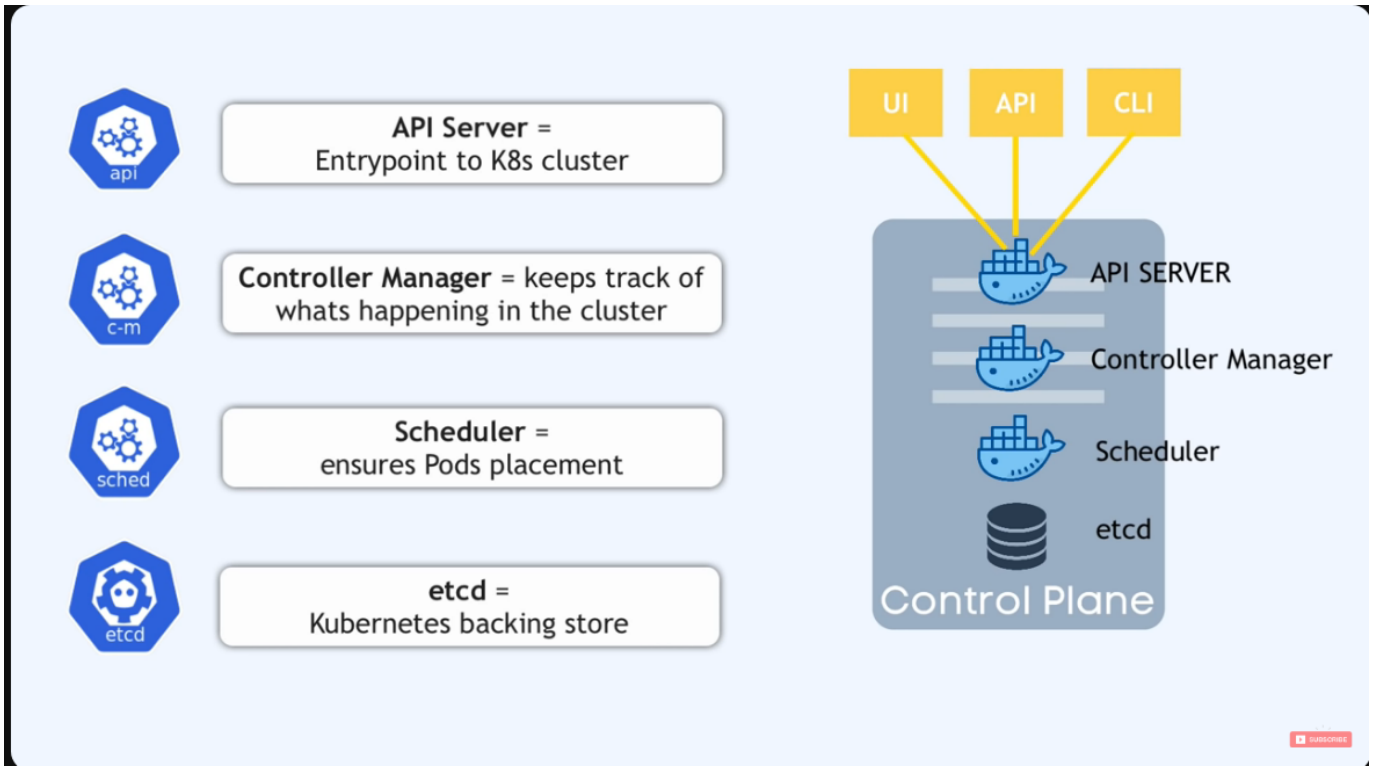
why

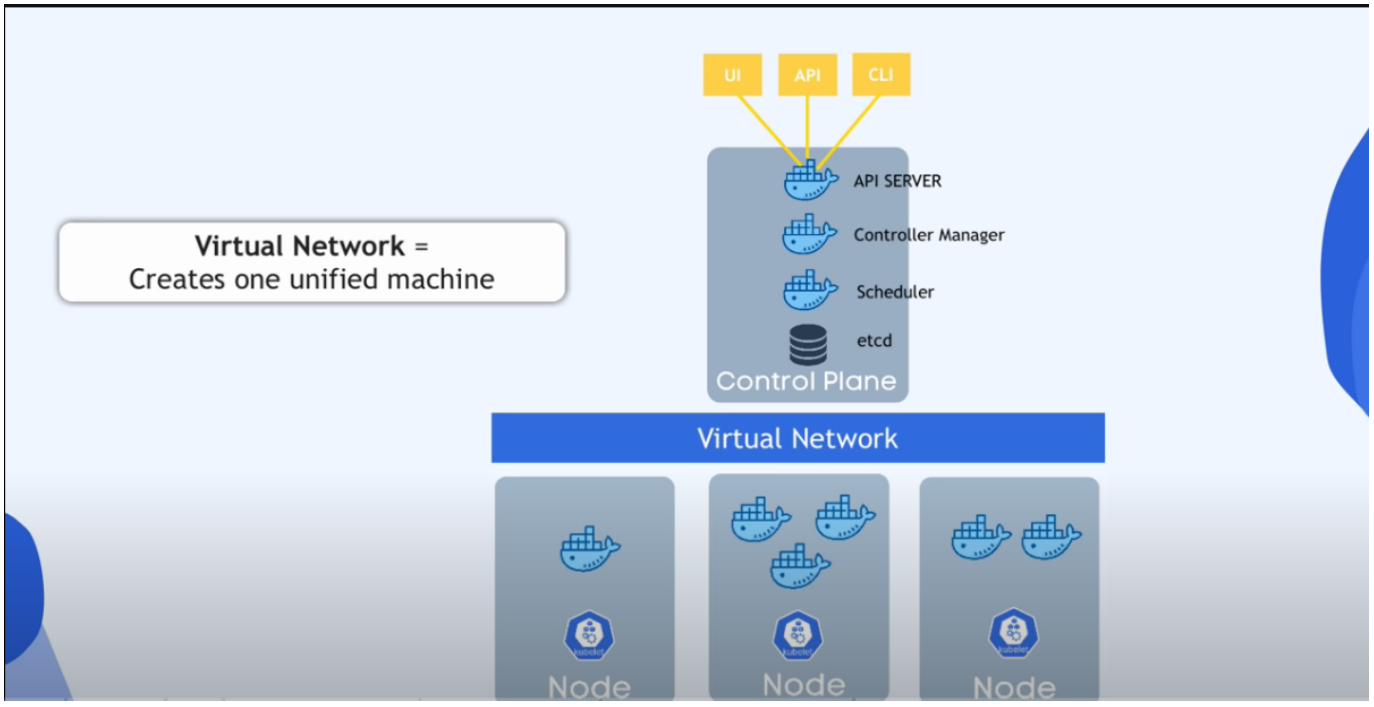
- mirgration to microservices

benefits

- tolerance
- availability
- scalability

architecture





worker nodes:

- higher workload -> much bigger ressources

master node

- handful of jobs
- very important if we lose it, we lose access to all the cluster: buck it up
- duplicated most of the time

main components

Pod

- abstraction over container: k8s abstarcts so can replace and abstracts it technology (docekr or other ..)

- smallest unit
- usually one app per Pod
- one internal ip address per Pod
- ephemeral
- get assigned a new ip address when recreated (they die beacause of crashing for exaplme) : problemtatic for changing the ip address of the pod they connect to
- that's why we need the componenet service

Service

- static ip address can be attached to each Pod
- lifecyle of service and service aren't connected so we do not need to change the endpoint
- Since app should be accessible through the browser: we need an external service : opens communication from external sources
- we do not want db service acccessible: we need an internal service
- internal service is the default
- can be a load balancer
- i we want to change the domain name: we use Ingress

Ingress

- instead of service the request goes to ingress and ingress does the forwarding to service
- so far nothing fancy

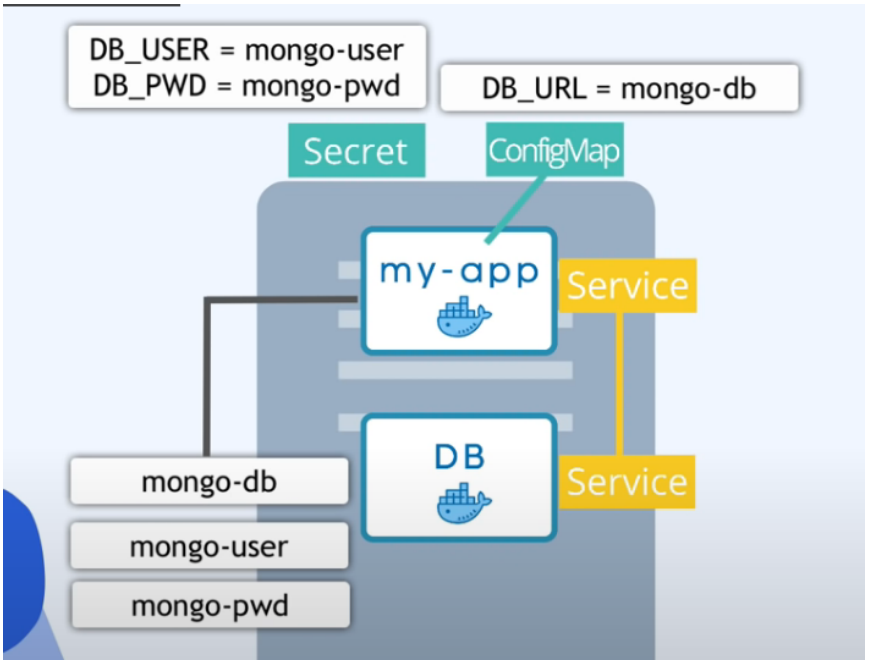
ConfigMap

- pods communicate with Service
- DB has a URL, usually in the built application
- in case it changed: rebuild and push and pull new image to the Pod -> Tedious!
- ConfigMap: configs of the app connected to the pod
- DB user and password can change too
- ConfigMap is not for non-confidential info!

Secret

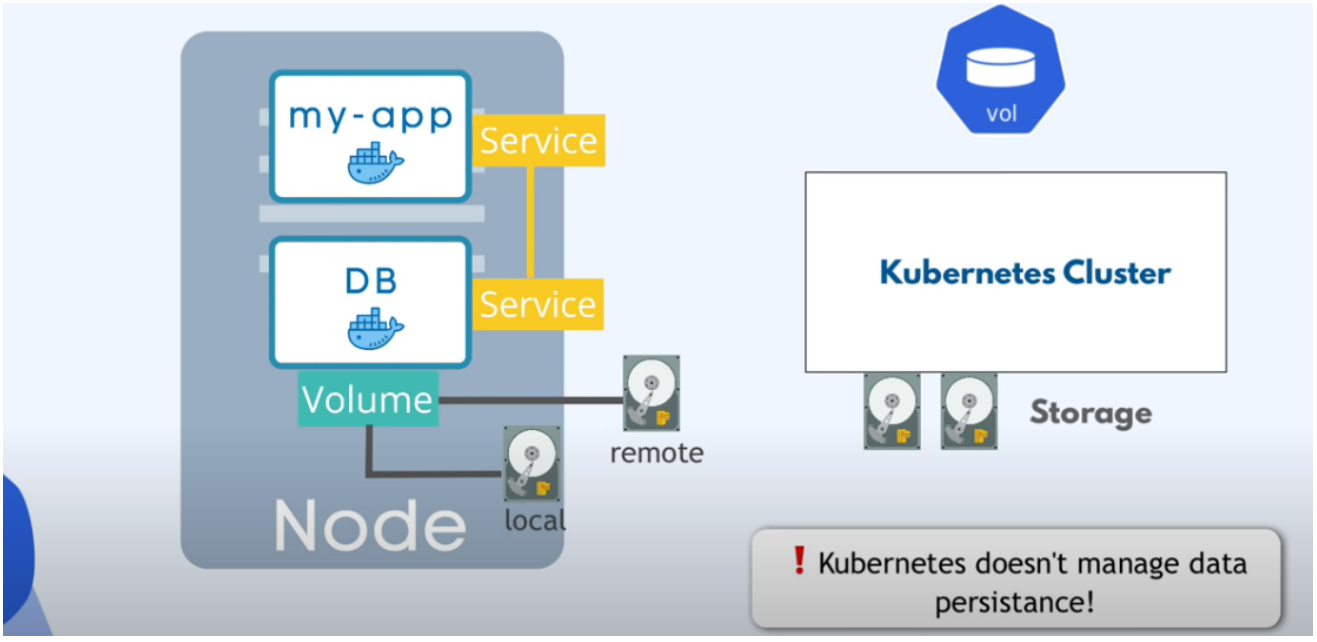
- are in base64 and not encrypted by default
- should use third party encriptor

- connect it to Pod: env variables or properties file



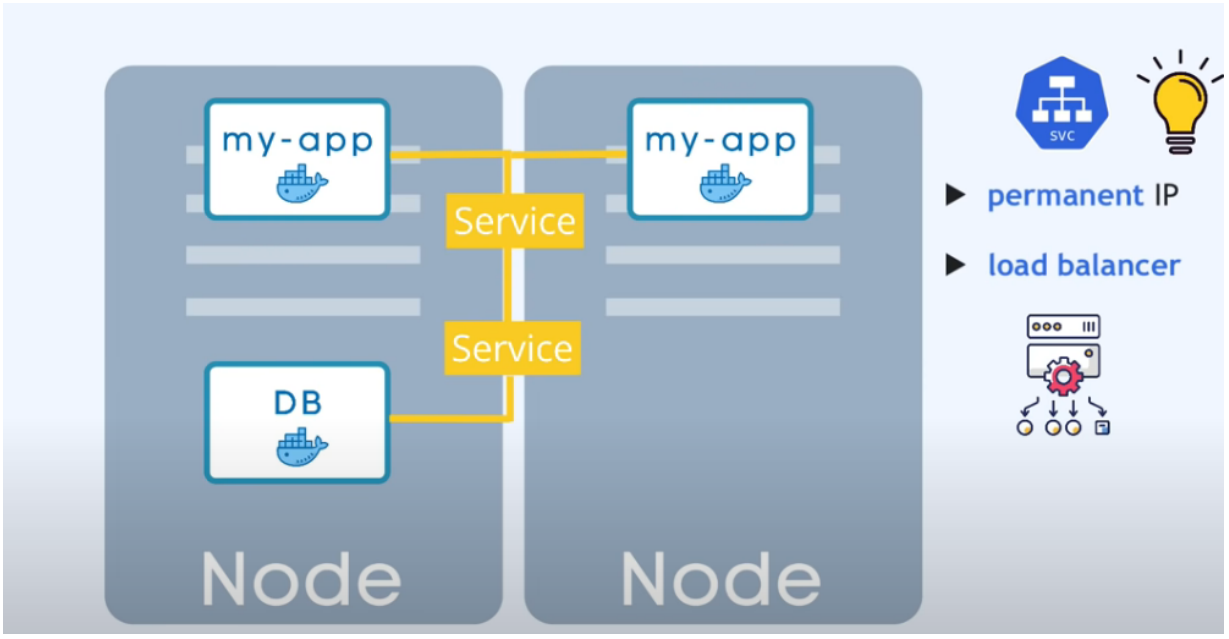
Volume

- Pods d not persist data
- attach physical hard drive to Pod: in the local machine or outside the cluster(cloud or on premise)
- storage:external hard drive plugged into the k8s cluster
- Kubernetes doesn't manage data persistence!
- developer should take care of bucking up



Deployment

- if an application crashes, the service will forwrd the requests to another one



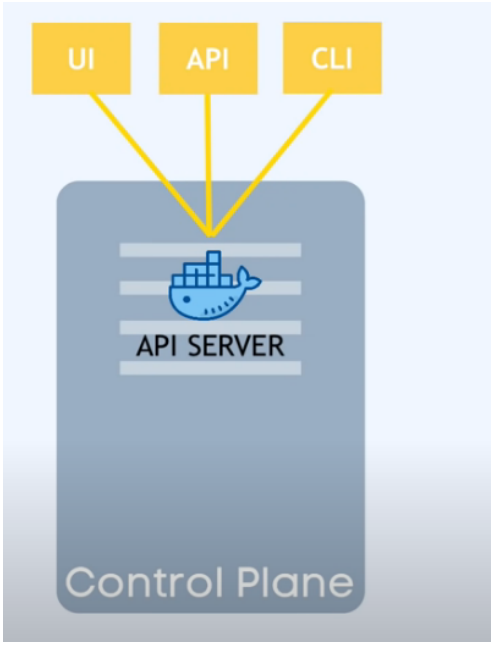
- to define blueprint for Pod and how many replicas(scale up or down)
- we mostly create deployments not pods
- deployment an abstarction over Pods
- DB ca't be replciated using Deployment, beacause it has state
- if we want replicas, they need to access the same shared data storage to avoid data inconsistencies

StatefulSet

- for stateful apps
- takes care of the consistency over the replcias
- using this service is tedious -> DBs are often hosted on external serivces

K8s config

- goes through the master with the proces API server: the only entrypoint to the cluster
- requests in yaml or json
- config requests are declarative
- controller manager checks: desired state == actual state ?
- config files are with the code or in their own git repo



Config files parts

metadata

```
.:
kind
metadata:
        name
```

specification

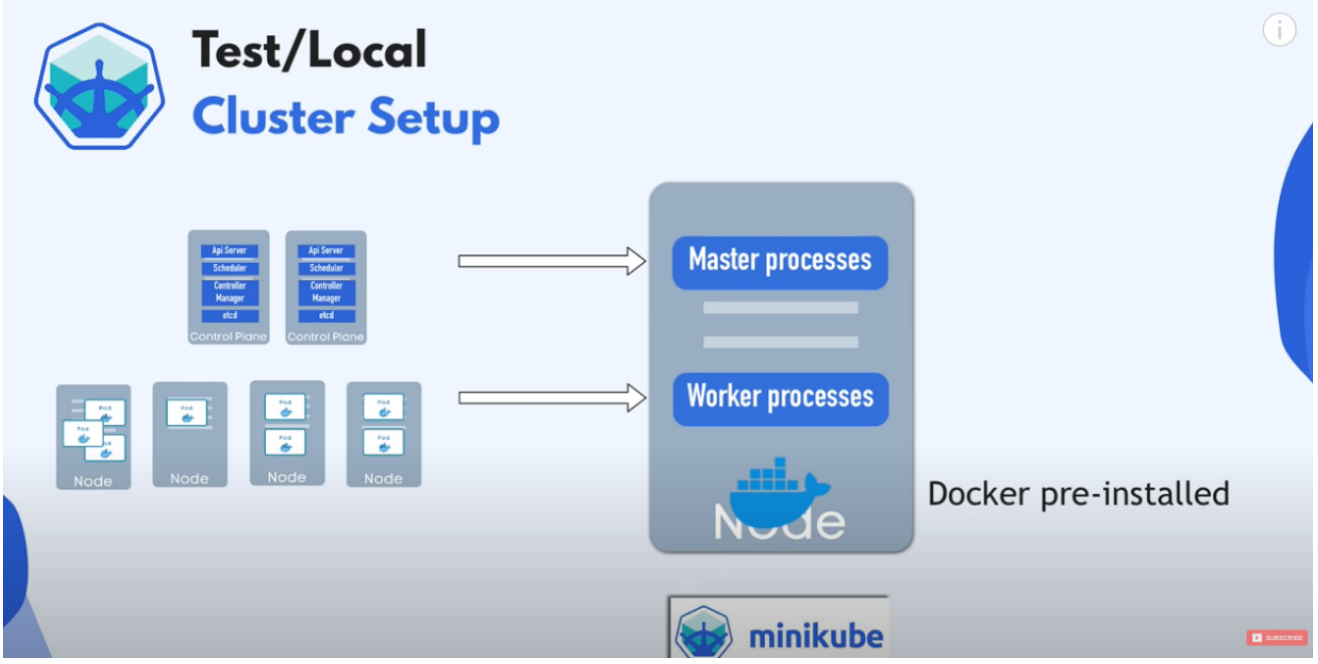
- specific to the kind

status

- automatically generated
- checking desired == actual
- k8s updates status contunuously
- the status info comes from the brain of the cluster: etcd

Minikube

- if we want to test cluster set ups locally
- 1 node with master processes, worker processes and docker preinstalled to run pods



Cubectl

- CLI to for k8s cluster (cloud or Minikube)