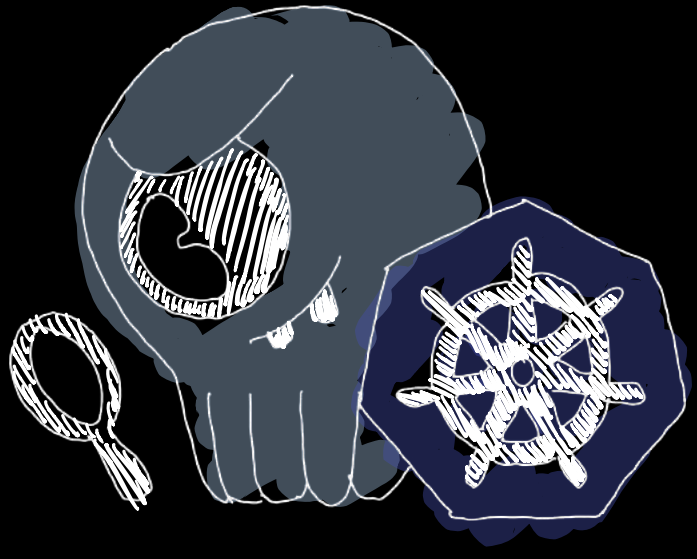
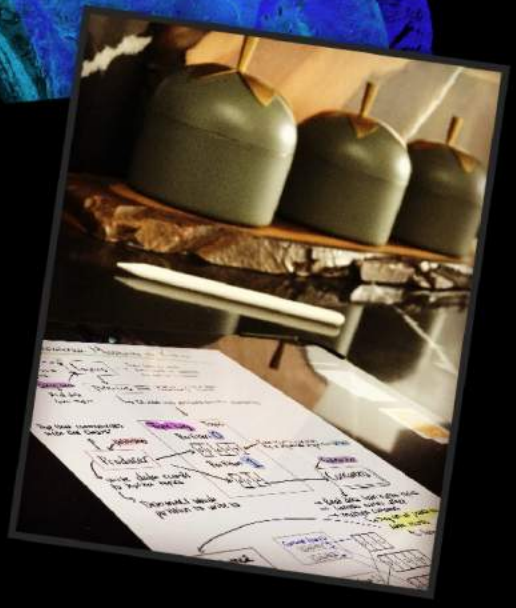
Cloud Native certified



Kubernetes Administrator

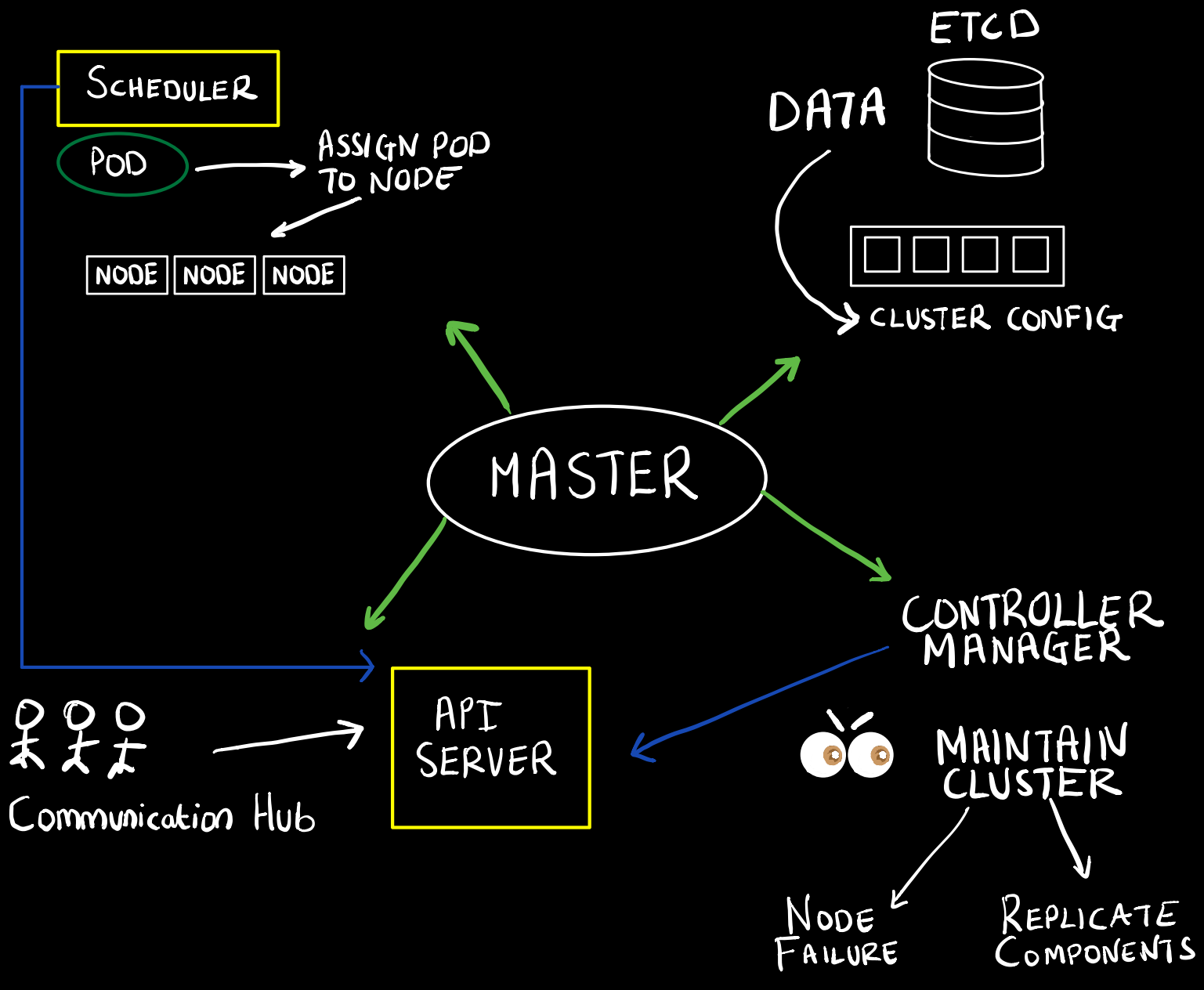
Adnan Rashid

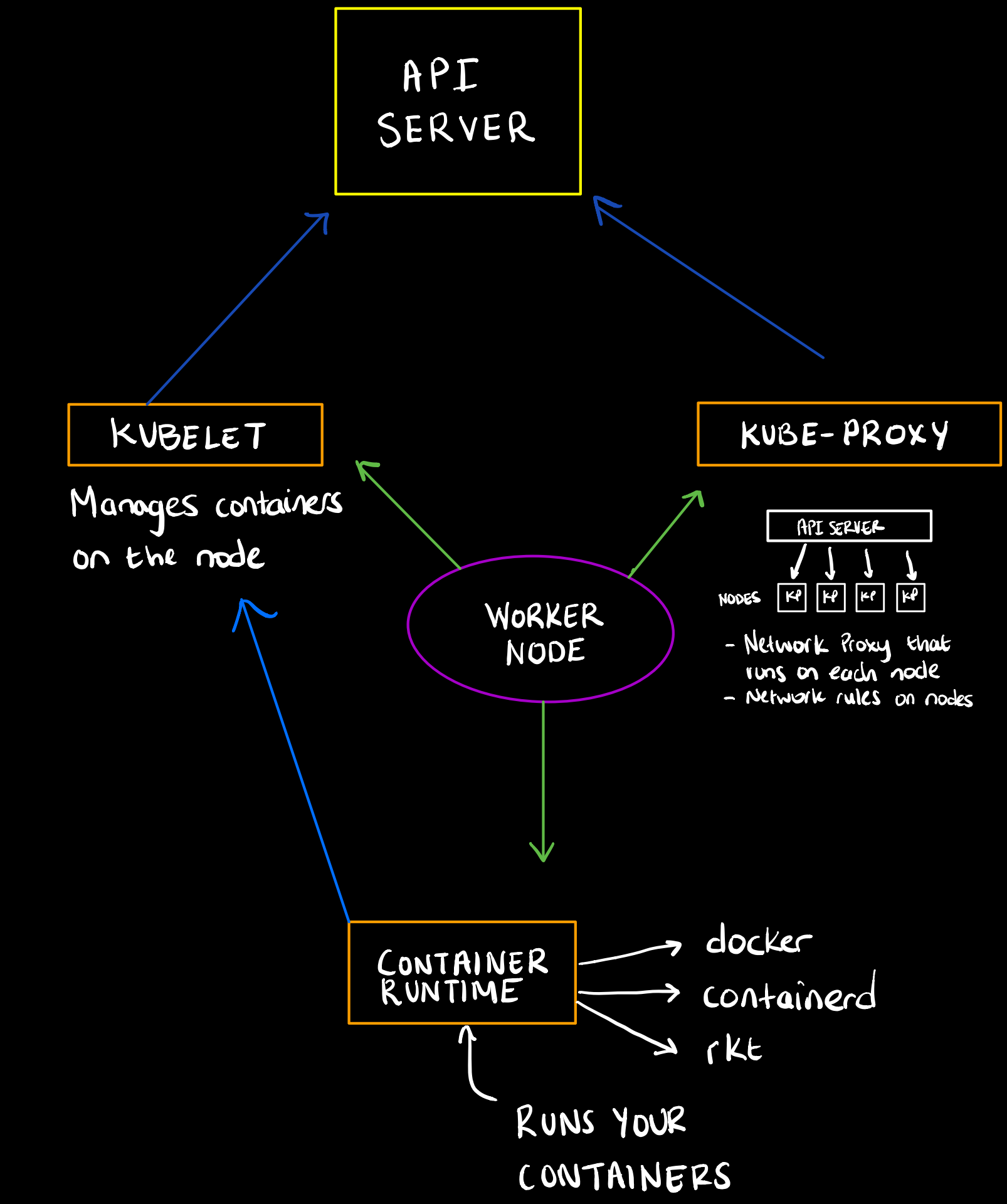
instagram.com/
adnans_techie_studies



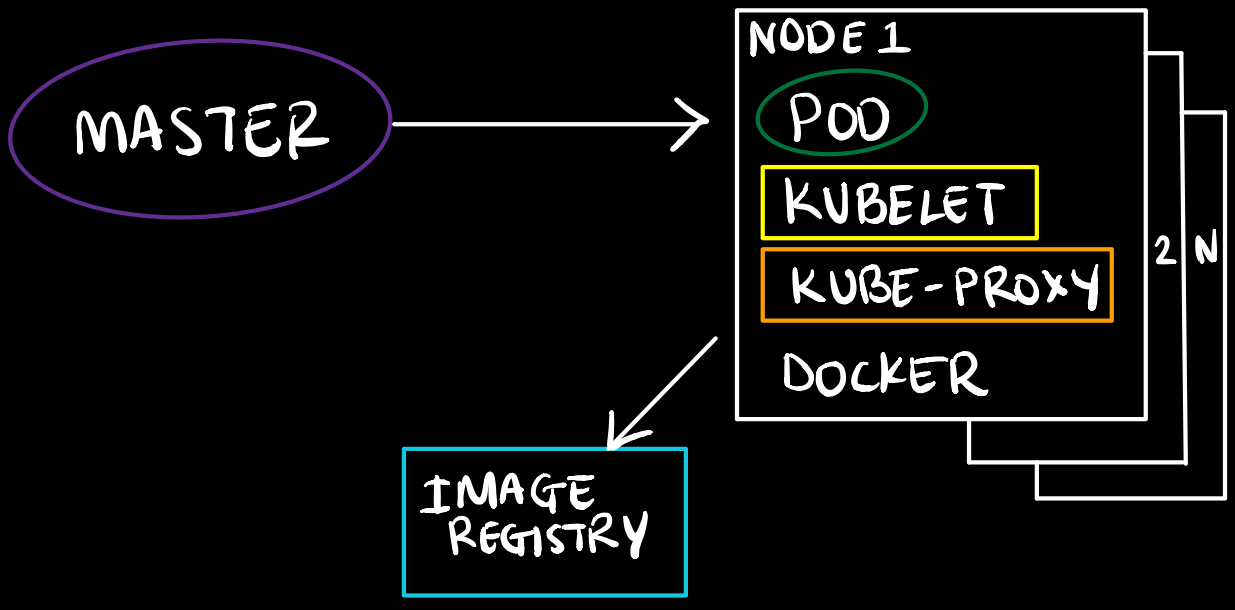
UNDERSTANDING THE KUBERNETES ARCHITECTURE

Cluster Architecture



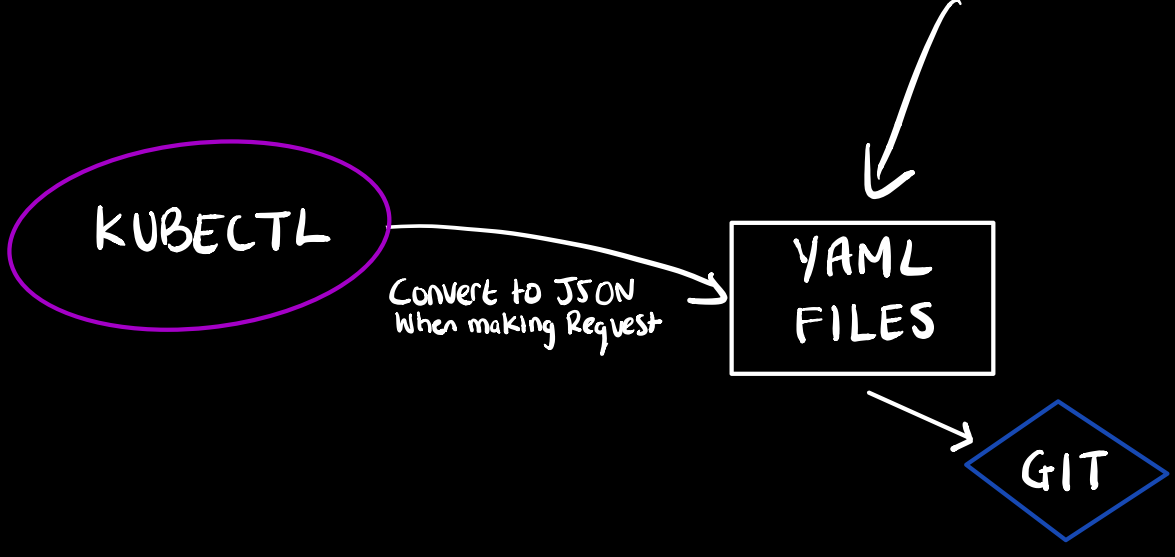

API SERVER
KUBELET
Manages containers on the node
KUBE-PROXY
API SERVER
NODES
KP
KP
KP
KP
- Network Proxy that runs on each node
- Network rules on nodes
WORKER NODE
CONTAINER RUNTIME
docker
containerd
rkt
RUNS YOUR CONTAINERS

Application Running on K8S



API Primitives

API server is the only one that communicates with etcd
Every single component speak with API server only not to each other
Objects like pods and services are declarative intents



YAML File Composition

API VERSION → clear consistent view of resources

KIND → The kind of object you want to create
- → Pod
- → Deployment
- → Job

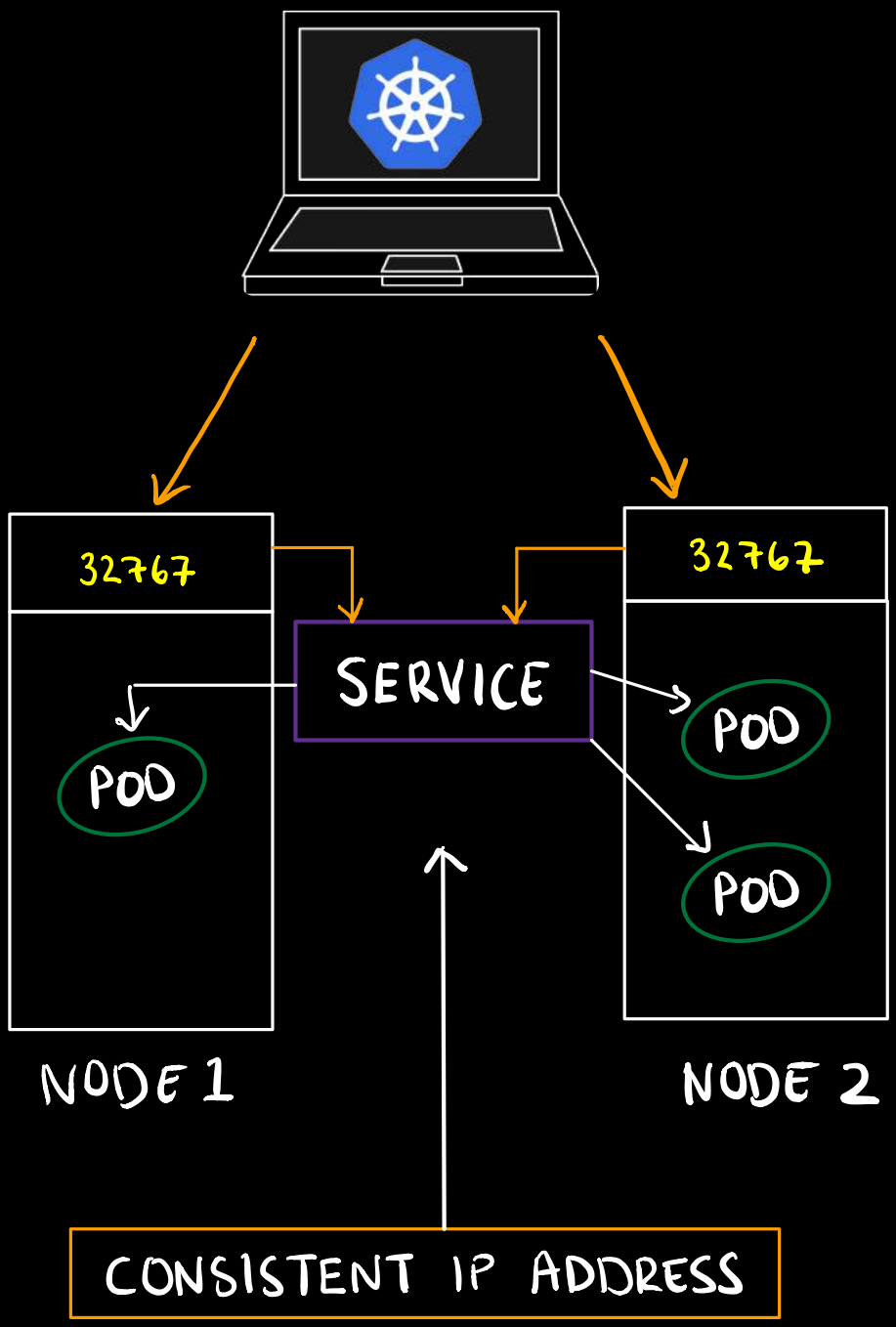
METADATA → uniquely identify the object
- Name String
- UID
- namespace

SPEC → container image, volume, exposed ports

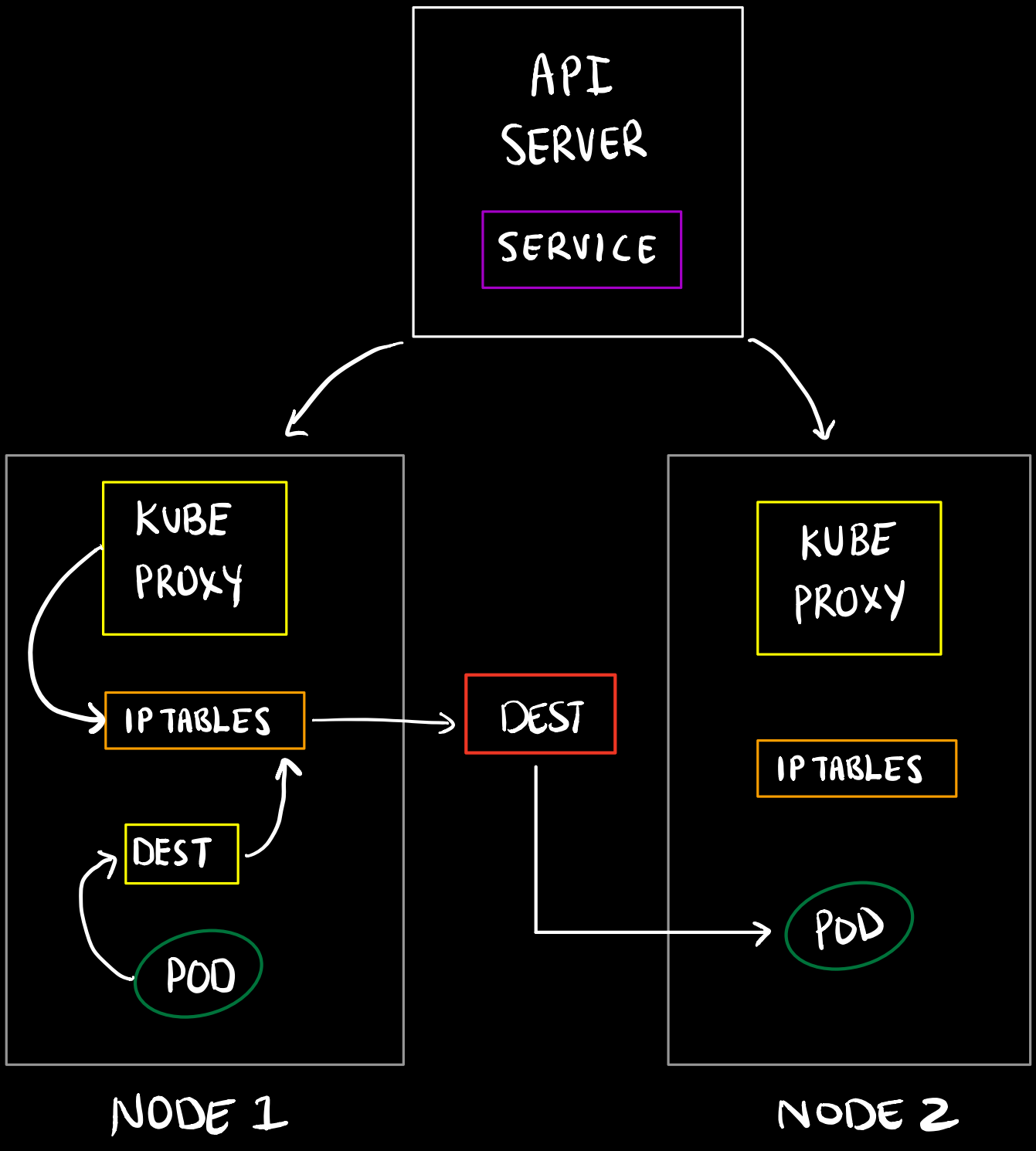
STATUS → State of the object
- ↓ match desired states

Services and Network Primitives

Services allow you to dynamically access a group of replica pods



KUBE-PROXY



KUBE-PROXY HANDLES THE TRAFFIC ASSOCIATED WITH A SERVICE BY CREATING IP TABLE RULES

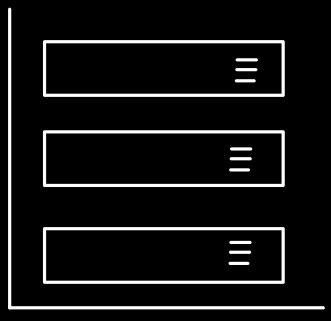
BUILDING THE KUBERNETES CLUSTER

Release Binaries, Provisioning & Types of Clusters

Picking the right Solution

CLOUD

VS



on-prem

ALSO

CUSTOM

- → Install Manually
- → Configure your own network Fabric
- → Locate the release binaries
- → Build your own Images
- → Secure cluster comms

VS

Pre-Built

- → Minikube
- → Minishift
- → MicroK8s
- → Ubuntu on LXD
- → AWS, Azure, GCP

✓ using minikube locally on MacOS

Installing Kube Master And Nodes

MASTER + WORKERS

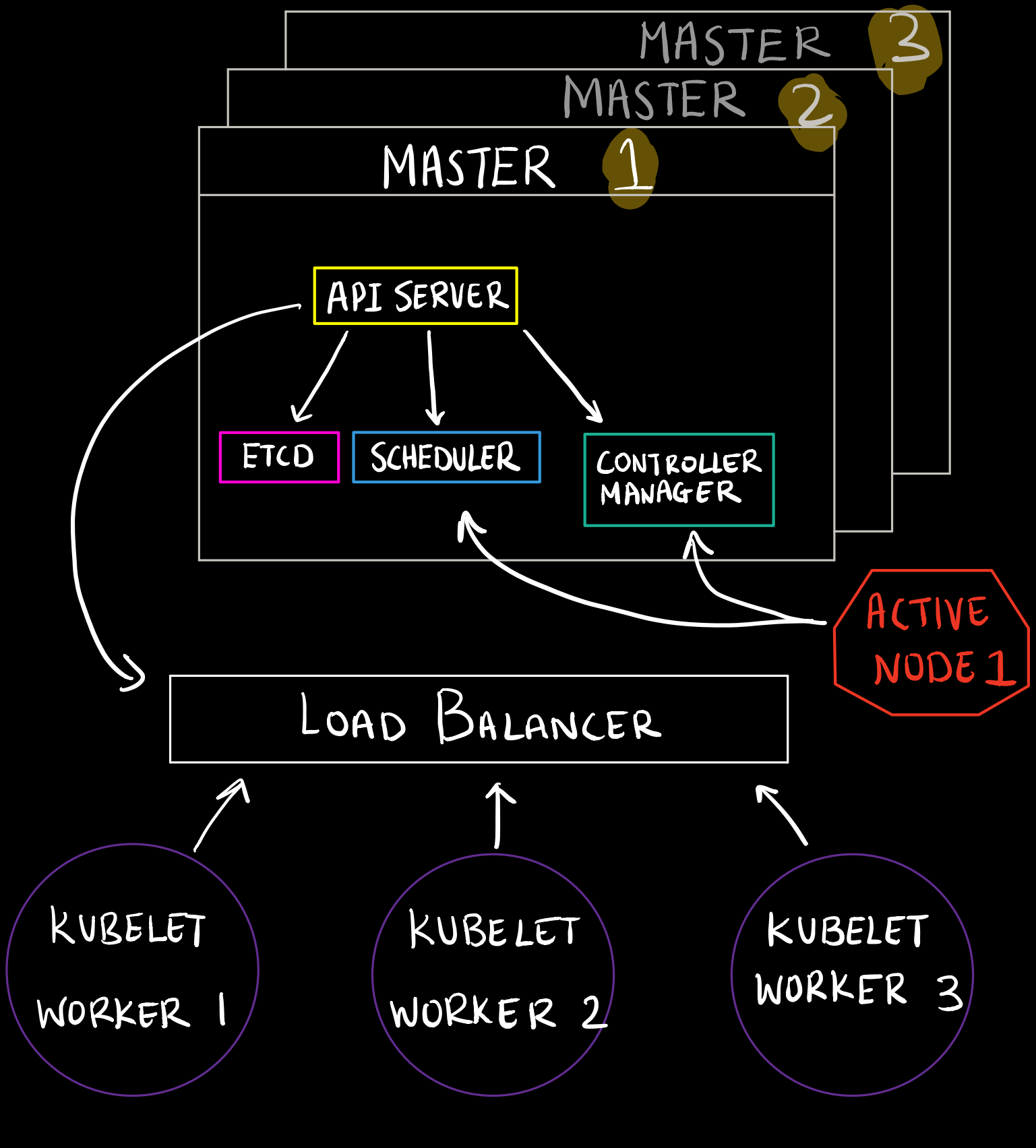
1. Docker + Kube → GPG KEY
   → ADD REPOS
2. UPDATE PACKAGES
3. Install Docker, kubelet, kubeadm, kubectl
4. Modify bridge adapter settings

MASTER ONLY

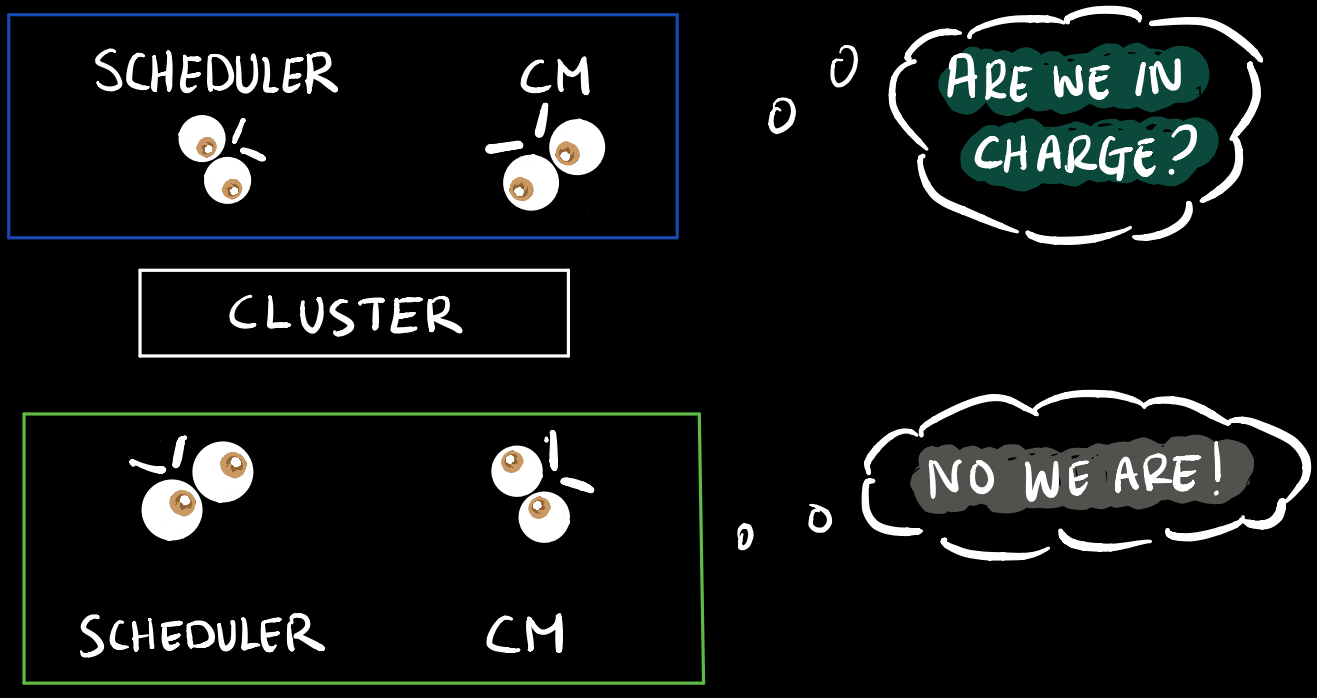
1. Initialise Cluster
2. Make directory for kube
3. Copy kube config
4. Change ownership of config
5. Apply Flannel CNI

Building Highly Available Cluster

All components can be replicated, but only certain can operate simultaneously



The controller manager and the scheduler actively watch the cluster state and take action when it changes



→ Leading to duplicate resources or corruption. → HOW DO WE DECIDE?

LEADER ELECT OPTION → creates endpoint resource
↑ see in scheduler YAML
↑ holderIdentity

Replicating etcd

TOPOLOGY

Stacked

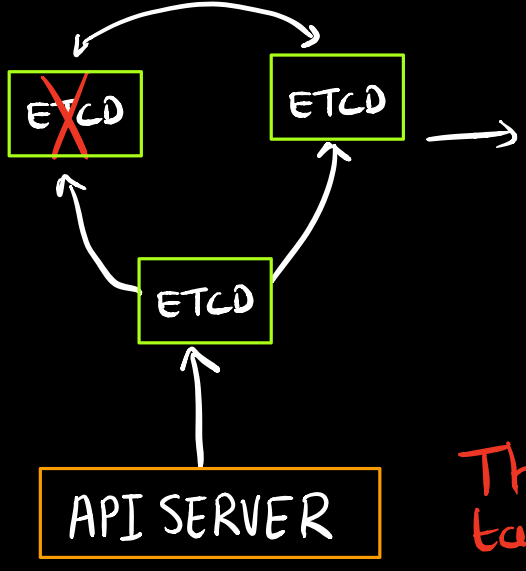
External to Kube Cluster

Each Control Plane node creates local

etcd member

only communicates

API SERVER



RAFT Consensus Algorithm

Requires majority

There must be more than half taking place in the state change

∴ there must be odd number of nodes

Configuring Secure Cluster Communications

→ All communication via Https

→ Kube CTL → Translator → Kubernetes API

Kubernetes API ← Provides CRUD

CRUD → create, read, update, delete

CRUD → Cluster State

Return Response

→ Kube CTL → CREATE A POD

HTTP POST → API SERVER

API SERVER → State Stored in ETCD

API SERVER → MULTIPLE PLUGINS [Authentication, Authorisation, Admission, Validation]

Admission → READ → Skip

Admission → create, Modify, delete

Authorisation → Can this User perform this action?

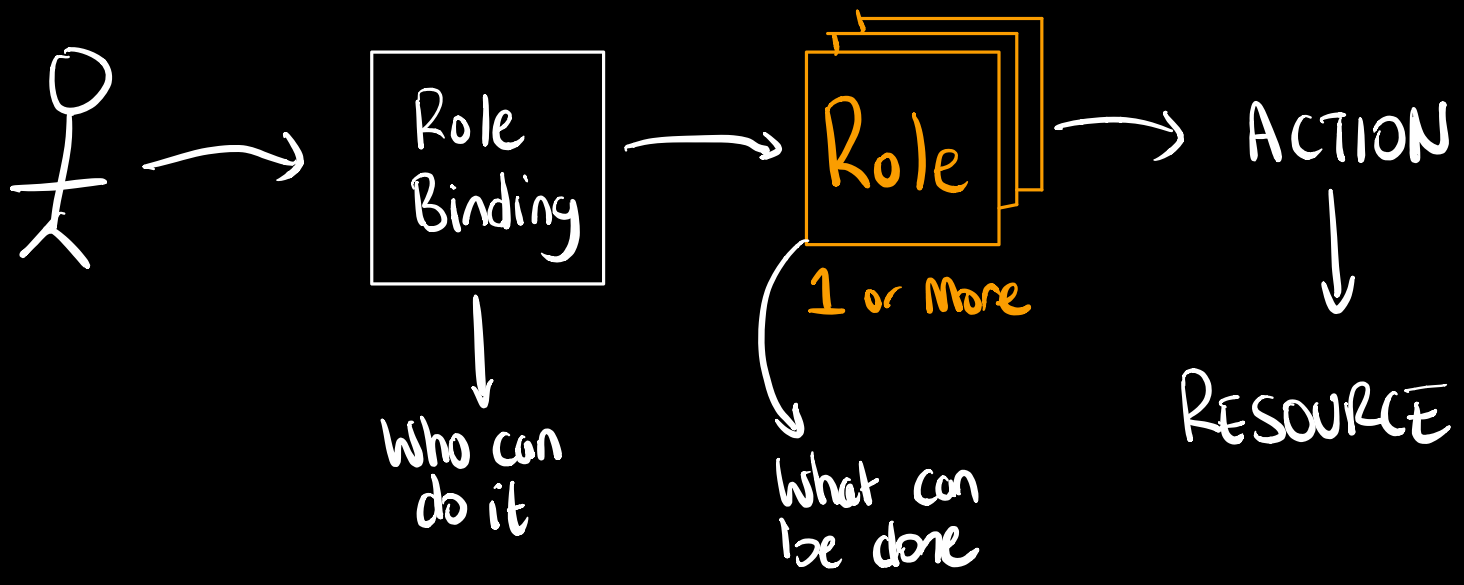
Authentication → Calls to determine request

Calls to determine request → HTTP HEADER

Calls to determine request → CERTIFICATE

Roles And Access

RBAC is used to prevent unauthorised users from modifying the cluster state

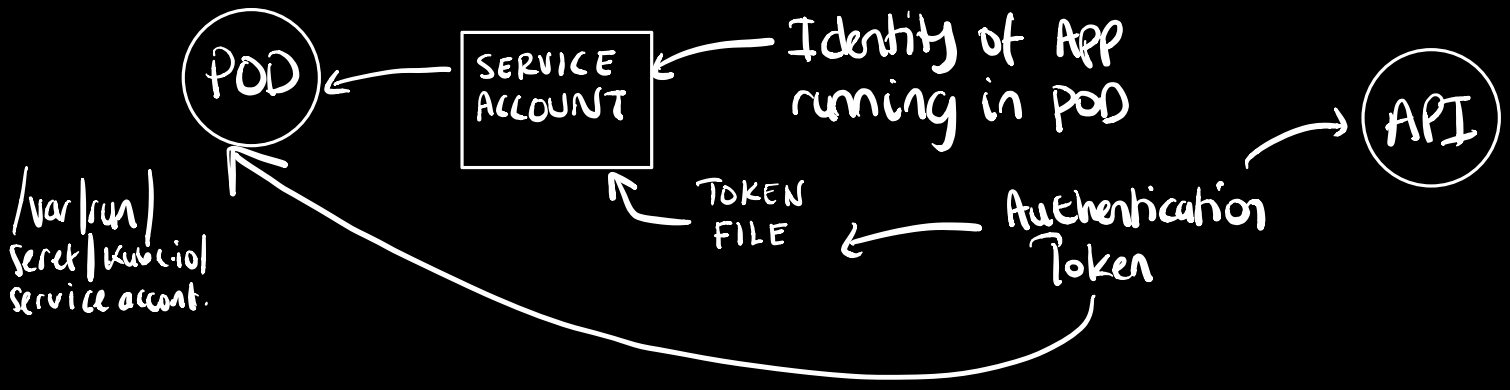


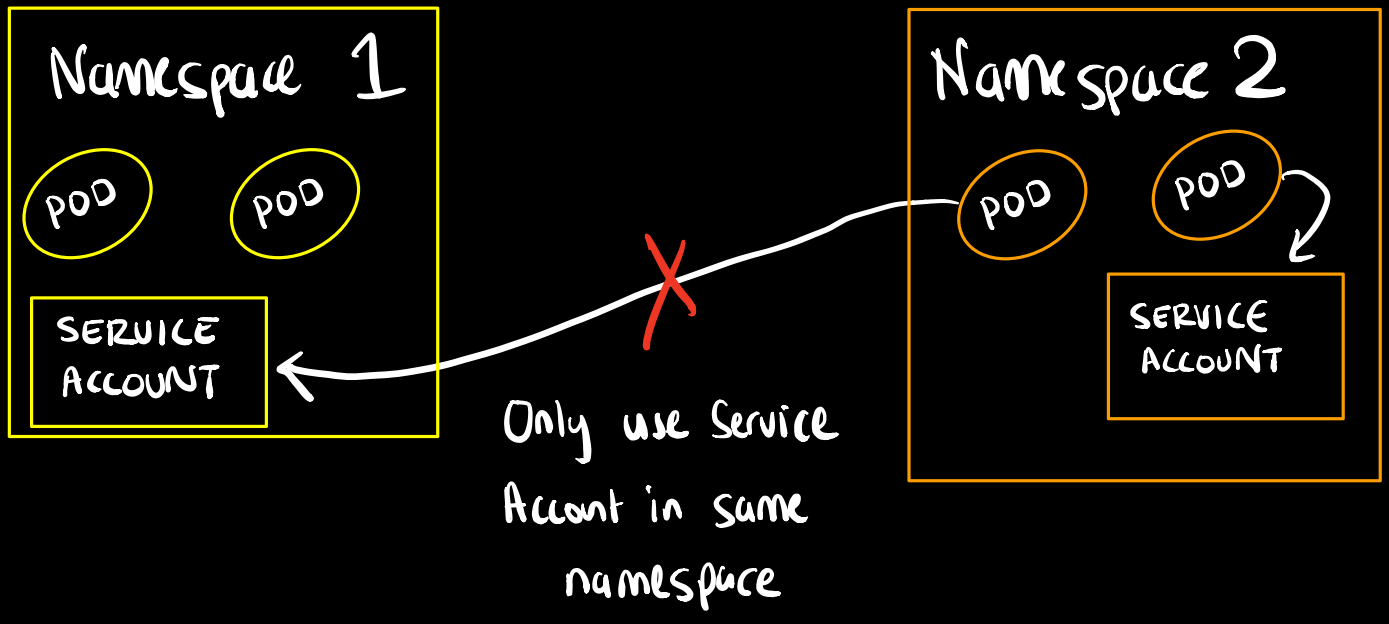
Role	Role Binding	→ Namespace Resource

ClusterRole	ClusterRoleBinding

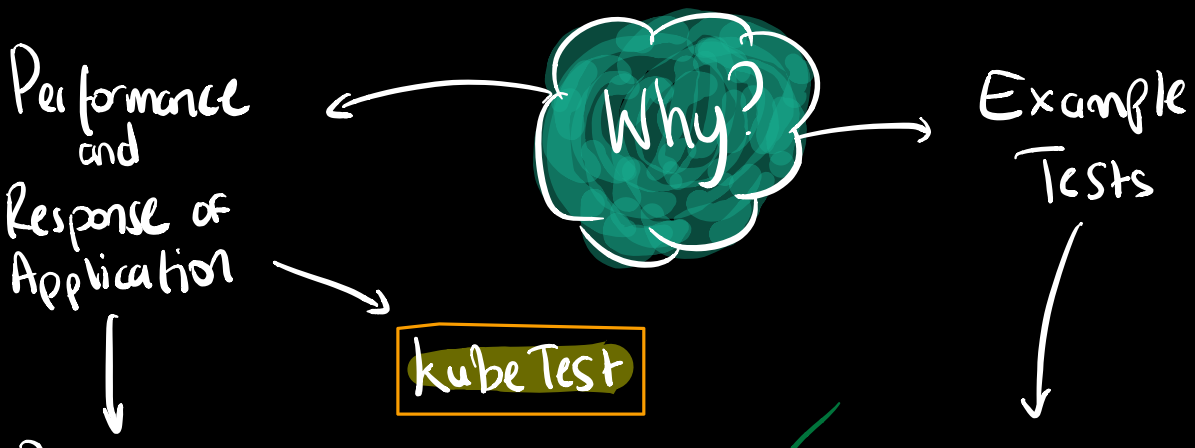
↳ Cluster level resources

Service Account





Running End to End Tests On Cluster



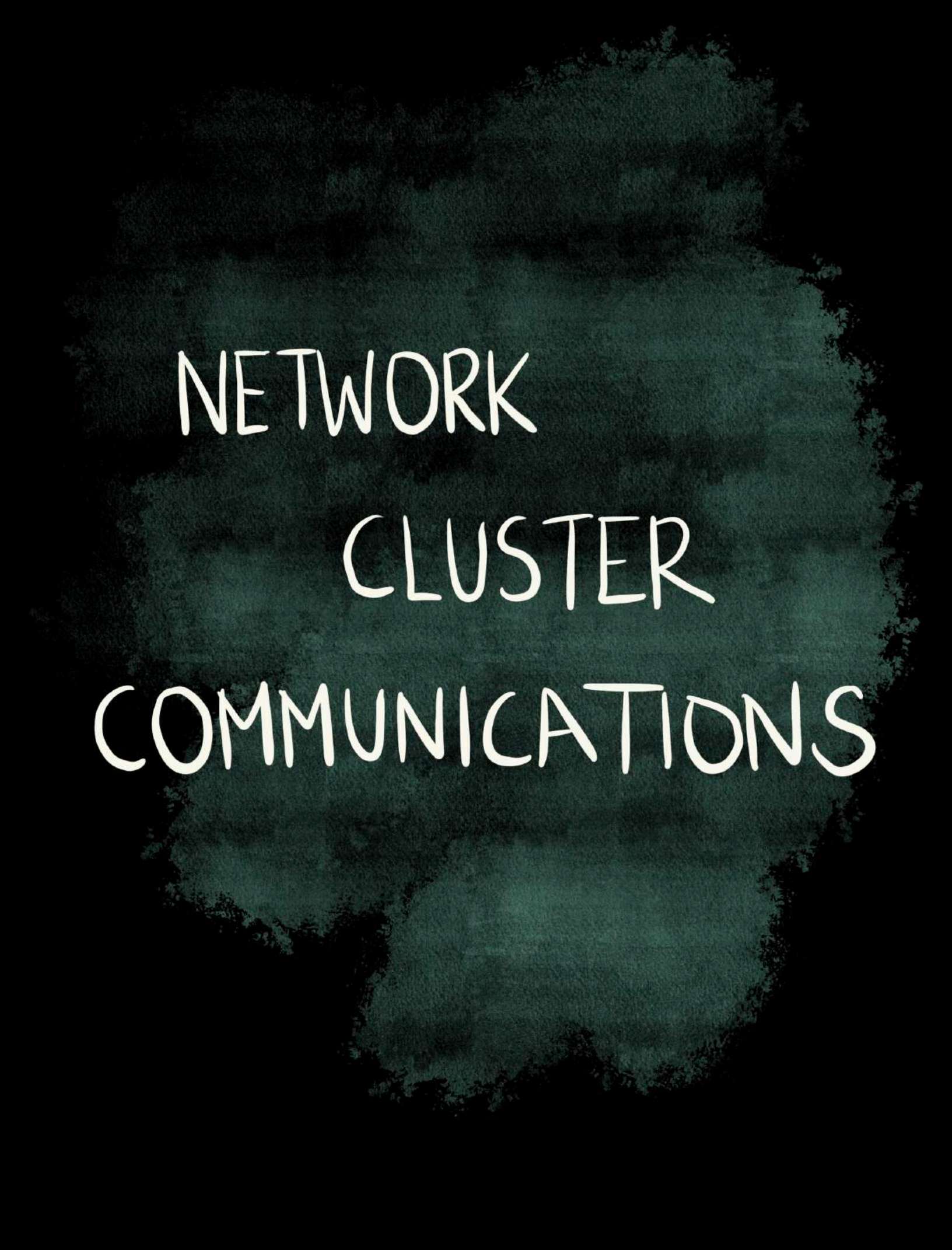
- ✓ Deployments can run
- ✓ Pods can run
- ✓ Pods can be directly accessed
- ✓ Logs can be collected
- ✓ Commands run from pod
- ✓ Services can provide access
- ✓ Nodes are healthy
- ✓ Pods are healthy

Managing Cluster

Upgrading Cluster
- Kubeadm
- Kubectl

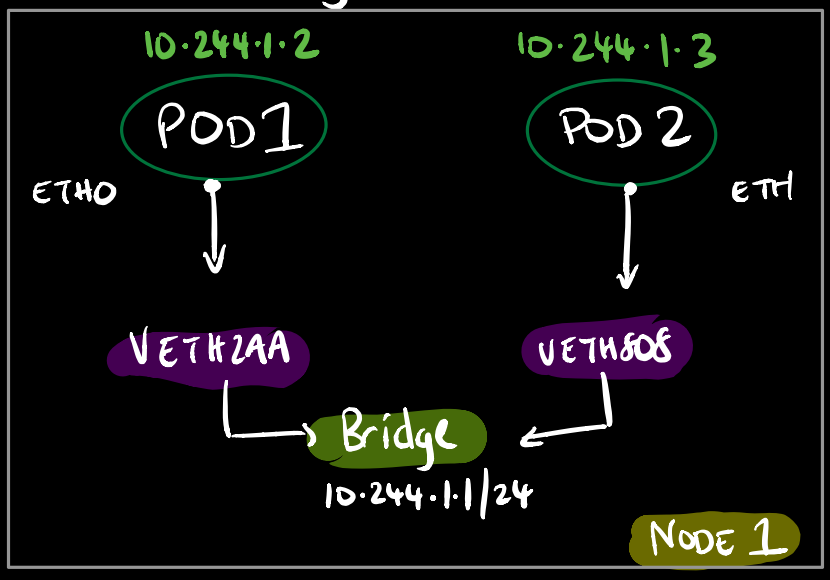
Operating System Upgrades
- Use Deployments or replica sets
- Get pods
  Drain Node
  Uncordon
  [Put back to Service

Back-up and Restore Cluster
- cluster state → etcd
  - Save externally

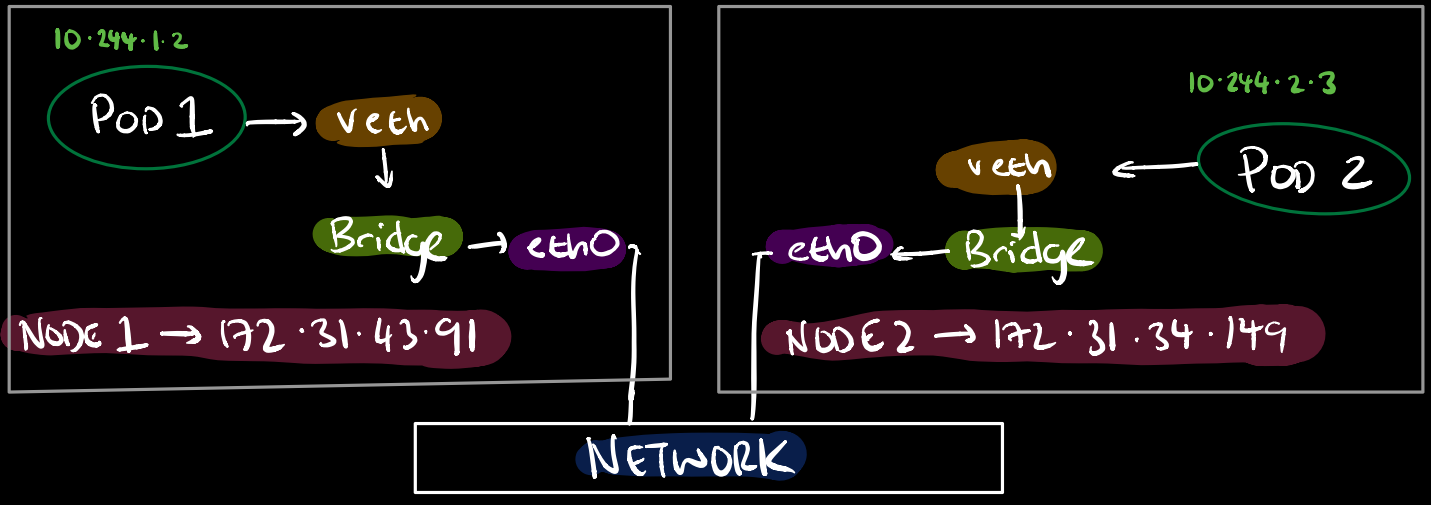

NETWORK
CLUSTER
COMMUNICATIONS

Pod and Node Networking

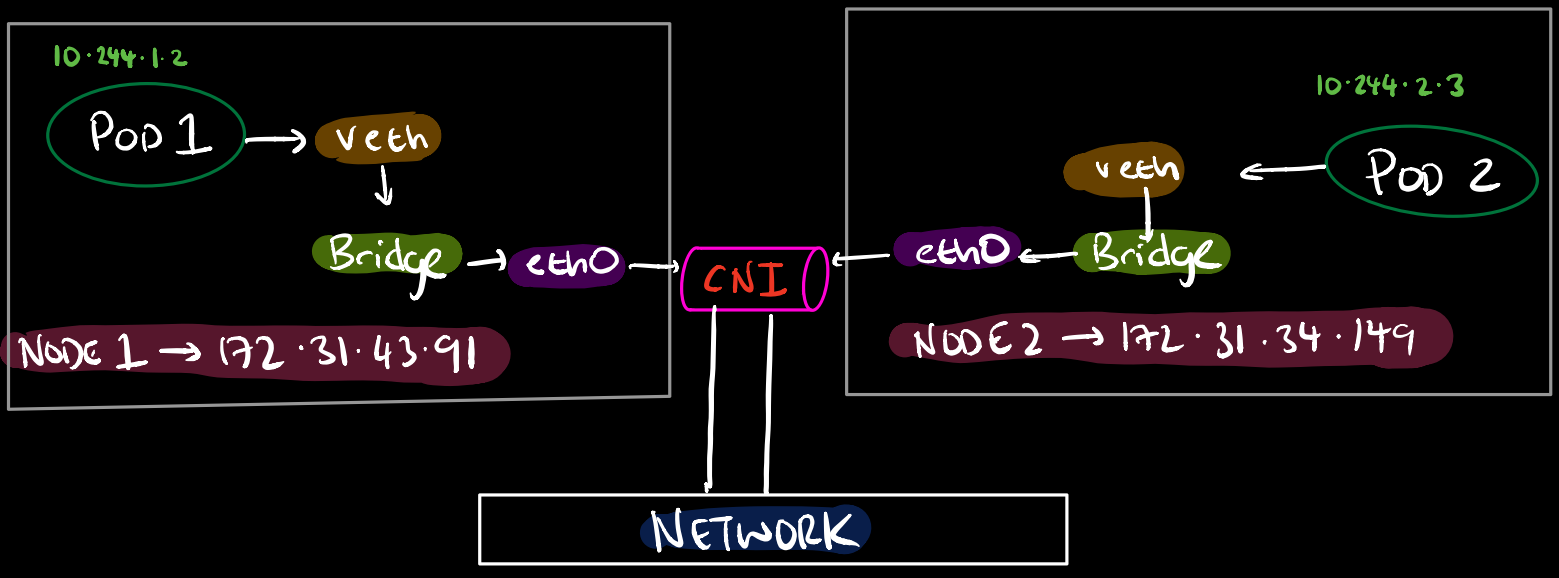
Networking within a node



Networking Outside of the Node



Container Network Interface



CNI is a network Overlay

↳ Allows building tunnel between nodes

↳ Sits on top of existing networks

↳ Encapsulates Packet

↳ changes SRC & DST

How CNI do this?

→ There is a mapping associated in user space

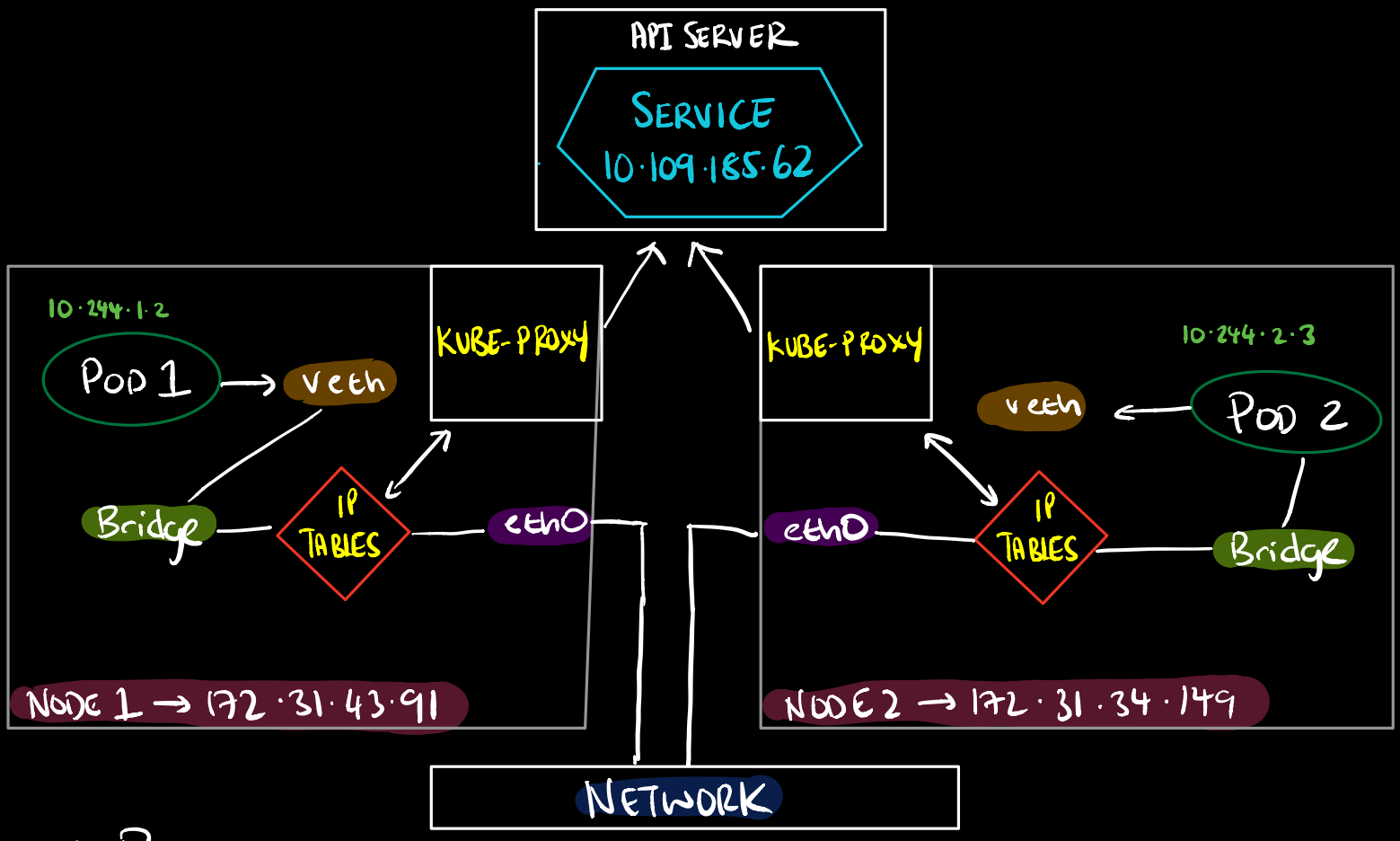
↳ Program all pod ip addresses to node IP, when reach other node, de-encapsulate packet and give to bridge

Example CNI

- Calico
- Flannel

Appears Local to Node!

Service Networking



→ Pods come and go.

↳ How does cluster keep track?

↳ Services!

↳ Provides virtual interface → Auto assigned to pods behind interface.

Example — Cluster IP Services → Auto created on cluster creation

↳ Takes care of internal routing → No matter where moves other pods know how to communicate to it

Ingress Rules and Load Balancers

Load Balancer

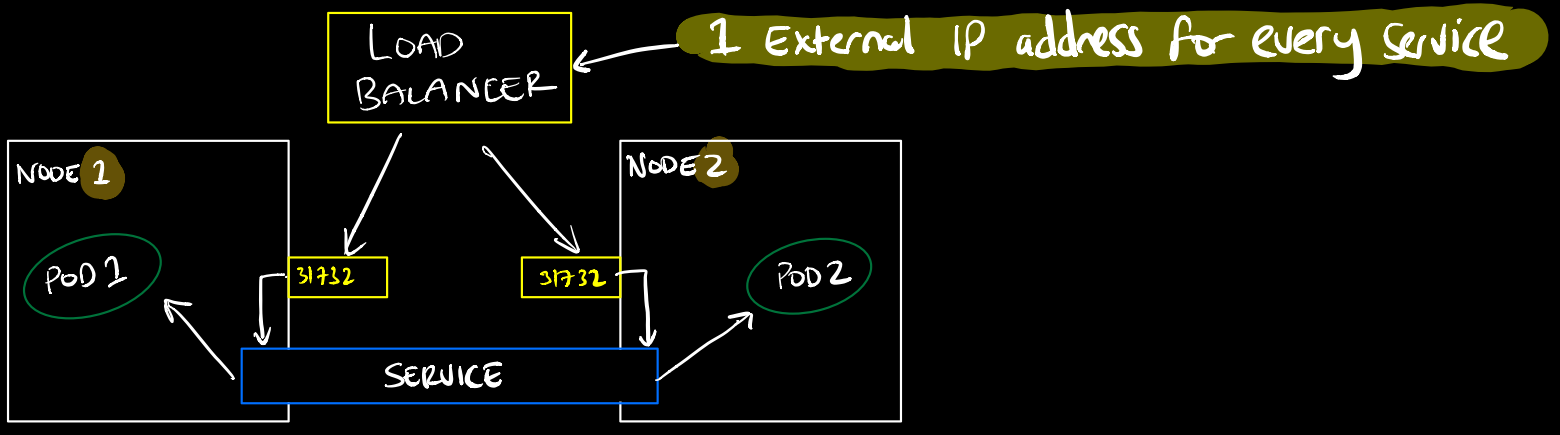
Redirect traffic to all nodes + ports

Clients talk to LB to access application

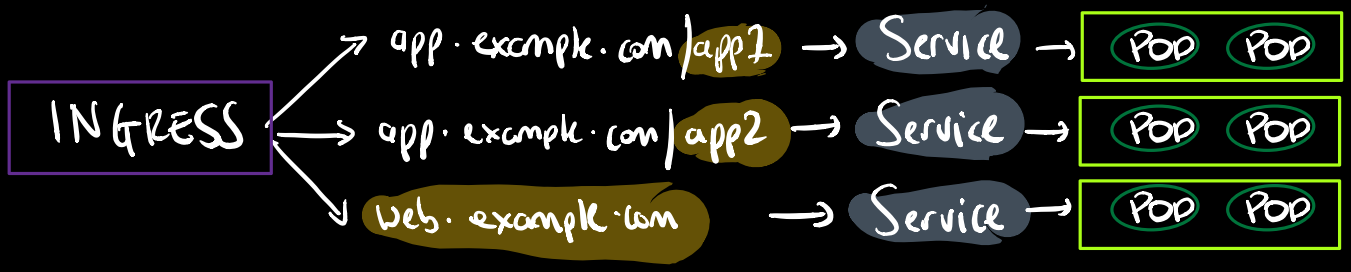
Service → Talks to 1 node at a time

↳ cannot split traffic like a load balancer

Only accessible internally → does not have external IP



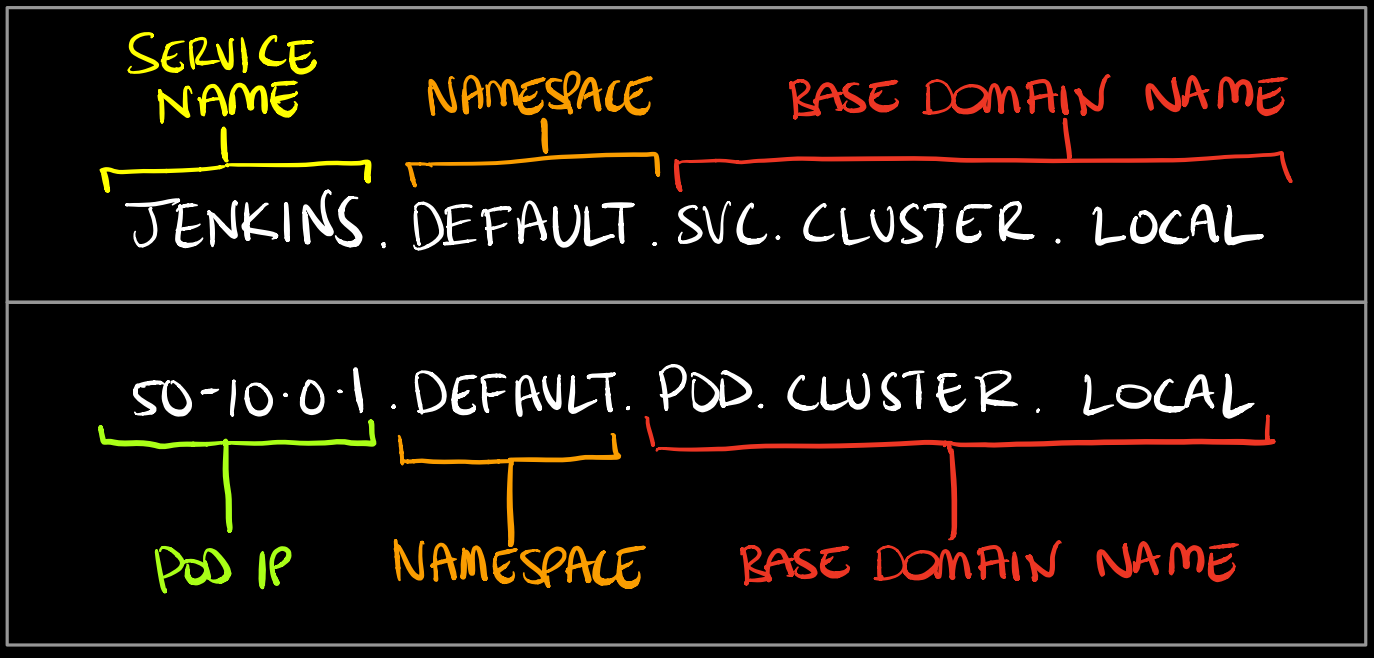
Ingress



Access multiple Services with Single IP Address

Cluster Dns

Every service defined in the cluster is assigned a DNS name



A pods DNS search will include the pods own namespace and the clusters default domain

POD SCHEDULING WITHIN THE KUBERNETES CLUSTER

Configuring the Kubernetes Scheduler

Scheduler responsible for assigning pod to node based on resource requirements of the pod

RULES ARE PLACED BY DEFAULT → HOWEVER, CAN CREATE OWN → WHY??

SAME NODE TO SAVE MONEY

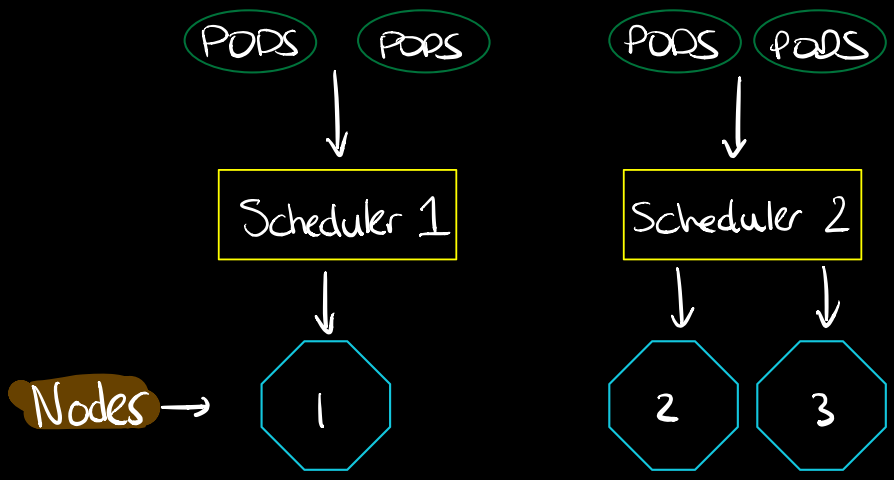
WORKER NODES HAVE DIFFERENT DISKS

SCHEDULER →

1. DOES THE NODE HAVE ADEQUATE HARDWARE RESOURCES?
2. IS THE NODE RUNNING OUT OF RESOURCES
3. DOES THE POD REQUEST A SPECIFIC NODE?
4. DOES THE NODE HAVE A MATCHING LABEL?
5. IF POD REQUESTS A PORT, IS IT AVAILABLE?
6. IF POD REQUESTS A VOLUME, CAN IT BE MOUNTED?
7. DOES THE POD TOLERATE THE TAINTS OF THE NODE?
8. DOES THE POD SPECIFY NODE OR POD AFFINITY?

Running Multiple Schedulers for Multiple Pods

It is possible to have 2 schedulers working alongside each other.



Sheduling Pods with Limits and Label Selectors

Taints → Repel Work → Example Master Node No Schedual

Tolerations → Allow you to tolerate a taint → Example kube-proxy ← Daemon set pod must run on all nodes

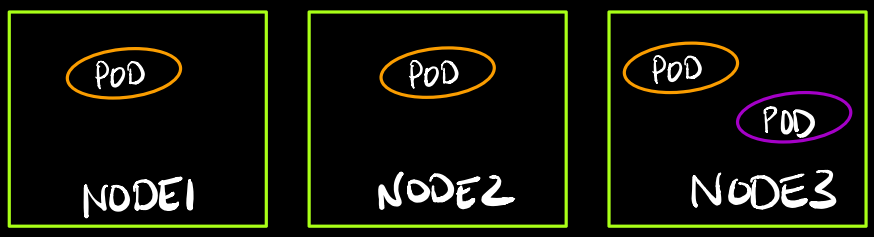
CPU/memory
- Sheduler looks at the sum of resources requested by existing pods
- Pod may not be using all requested resource at a given time.

Scheduler
Post 8 steps
- Most requested priority
- Least requested priority

Why? ← Cloud environments you are paying for all resources

DaemonSets

DaemonSets ensure that a single replica of a pod is running on each node at all times



POD DAEMONSET POD

POD REPLICASET POD

If you try delete a daemonset pod, it will simply recreate it.

Display Scheduler Events