



# Course Objectives

Core Concepts

Cluster Architecture

Services & Other Network Primitives

Scheduling

Logging Monitoring

Application Lifecycle Management

Cluster Maintenance

Security

Storage

Networking

Installation, Configuration & Validation

Troubleshooting

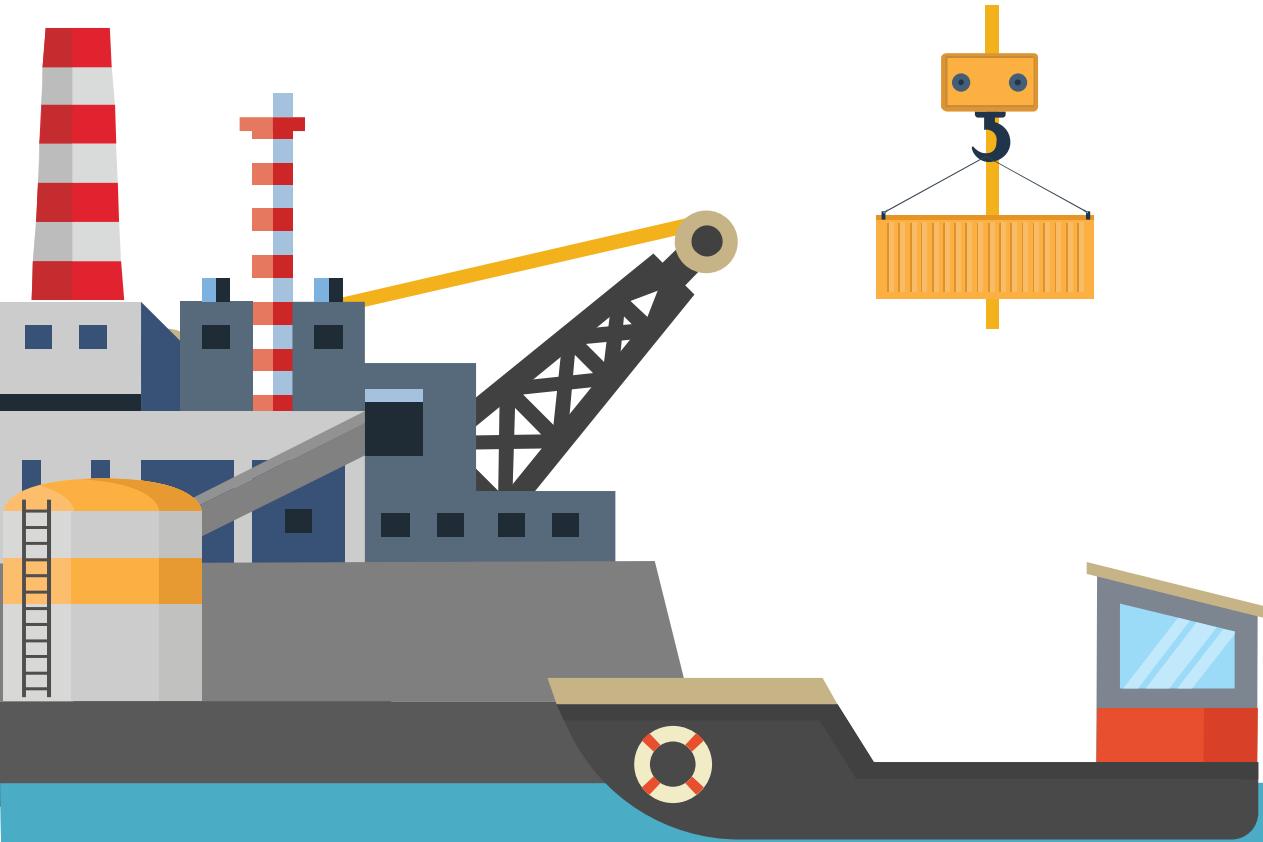
API Primitives

# Cluster Architecture

- Kubernetes Architecture
- ETCD For Beginners
- ETCD in Kubernetes
- Kube-API Server
- Controller Managers
- Kube Scheduler
- Kubelet
- Kube Proxy

# KUBERNETES ARCHITECTURE







## Master

Manage, Plan, Schedule, Monitor  
Nodes



## Worker Nodes

Host Application as Containers

**kube-apiserver**

**Master**

Manage, Plan, Schedule, Monitor  
Nodes

**Worker Nodes**

Host Application as Containers

**kubelet**

**Controller-  
Manager**

**CLUSTER**

**etcd**

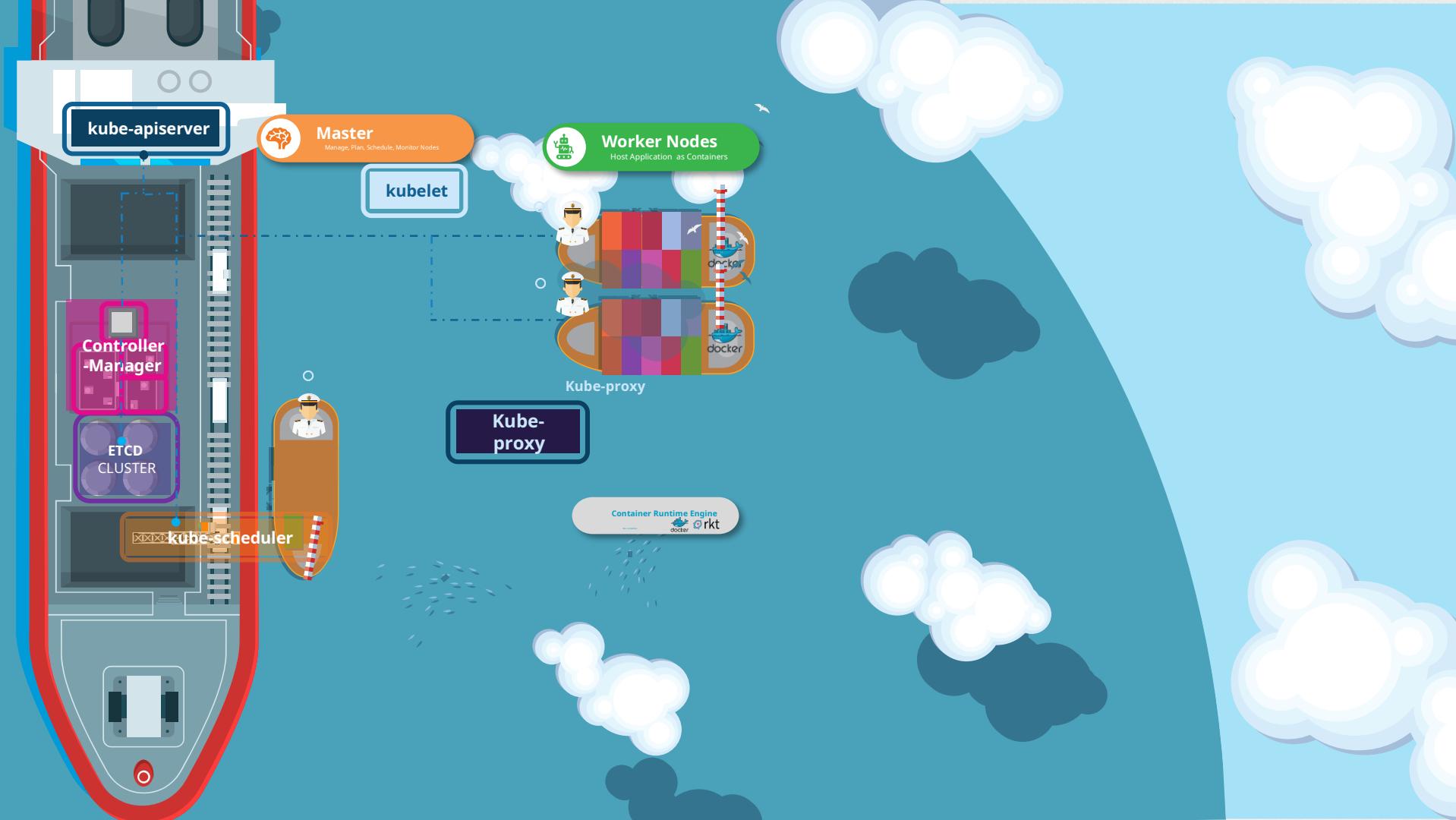
**re-scheduler**

**Kube-proxy**

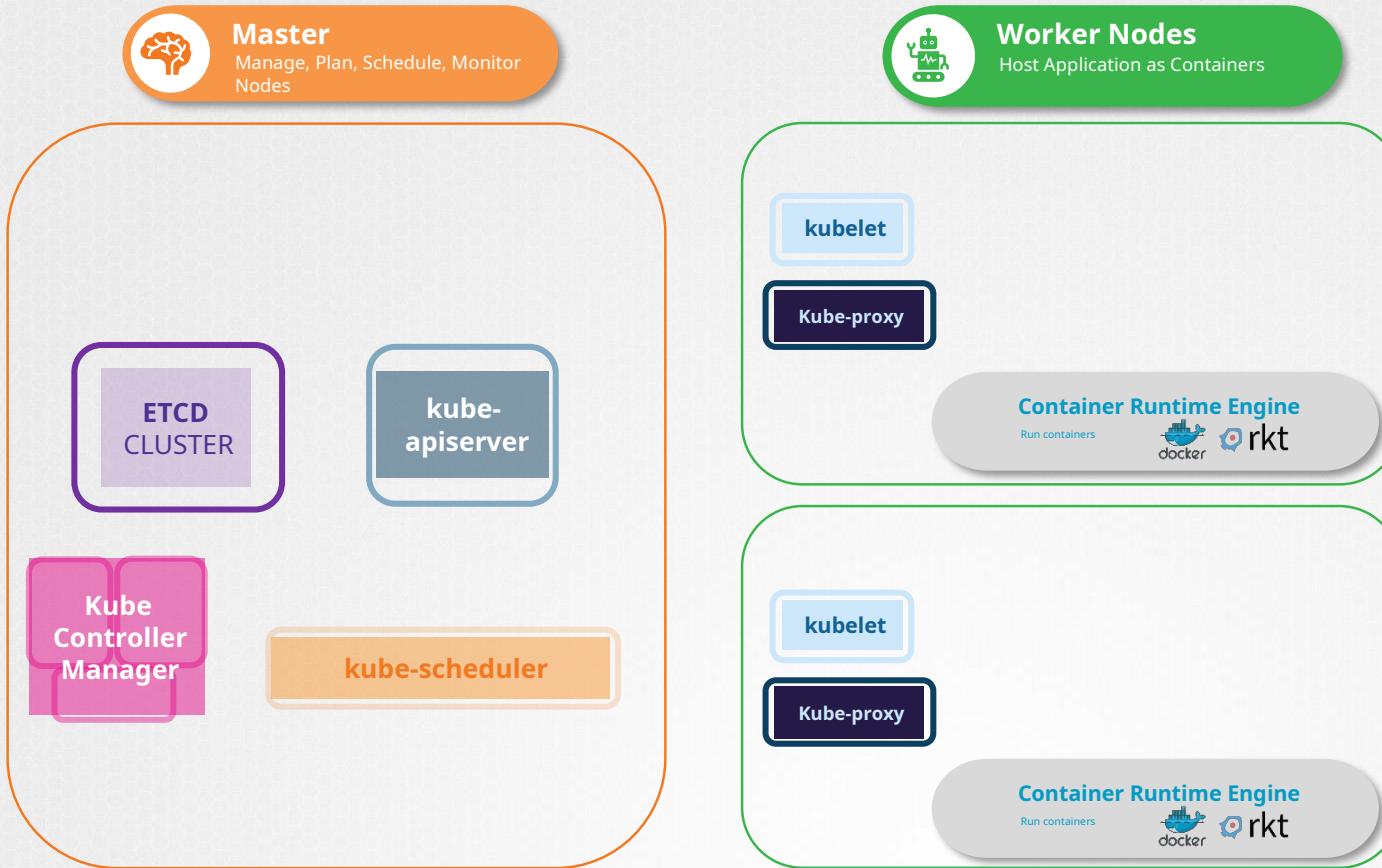
**Container Runtime Engine**

Run containers





# Kubernetes Architecture





# ETCD FOR BEGINNERS

# Objectives

- What is ETCD?
- What is a Key-Value Store?
- How to get started quickly?
- How to operate ETCD?
- What is a distributed system?
- How ETCD Operates
- RAFT Protocol
- Best practices on number of nodes



**ETCD** is a distributed  
reliable key-value store  
that is Simple, Secure &  
Fast

# key-value store

Tabular/Relational Databases

Name	Age	Location	Salary	Grade
John Doe	45	New York	5000	
Dave Smith	34	New York	4000	
Aryan Kumar	10	New York		A
Lauren Rob	13	Bangalore		C
Lily Oliver	15	Bangalore		B

# key-value store

Key	Value
Name	John Doe
Age	45
Location	New York
Salary	5000

Key	Value
Name	Dave Smith
Age	34
Location	New York
Salary	4000
Organization	ACME

Key	Value
Name	Aryan Kumar
Age	10
Location	New York
Grade	A

Key	Value
Name	Lauren Rob
Age	13
Location	Bangalore
Grade	C

Key	Value
Name	Lily Oliver
Age	15
Location	Bangalore
Grade	B

# key-value store

```
{  
  "name": "John Doe",  
  "age": 45,  
  "location": "New York",  
  "salary": 5000  
}
```

```
{  
  "name": "Dave Smith",  
  "age": 34,  
  "location": "New York",  
  "salary": 4000,  
  "organization": "ACME"  
}
```

```
{  
  "name": "Aryan Kumar",  
  "age": 10,  
  "location": "New York",  
  "Grade": "A"  
}
```

```
{  
  "name": "Lily Oliver",  
  "age": 15,  
  "location": "Bangalore",  
  "Grade": "B"  
}
```

```
{  
  "name": "Lauren Rob",  
  "age": 13,  
  "location": "Bangalore",  
  "Grade": "C"  
}
```

# Install ETCD

## 1. Download Binaries

```
curl -L https://github.com/etcd-io/etcd/releases/download/v3.3.11/etcd-  
v3.3.11-linux-amd64.tar.gz -o etcd-v3.3.11-linux-amd64.tar.gz
```

## 2. Extract

```
tar xzvf etcd-v3.3.11-linux-amd64.tar.gz
```

## 3. Run ETCD Service

```
./etcd
```

# Operate ETCD

## 3. Run ETCD Service

```
./etcd
```

```
▶ ./etcdctl set key1 value1
```

```
▶ ./etcdctl get key1
```

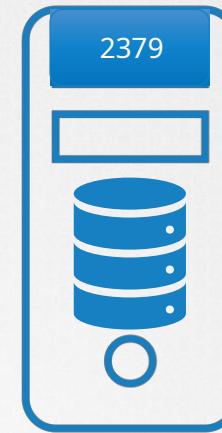
```
value1
```

```
▶ ./etcdctl
```

```
NAME:  
  etcdctl - A simple command line client for etcd.
```

```
COMMANDS:
```

```
  backup      backup an etcd directory  
  cluster-health  check the health of the etcd cluster  
  mk          make a new key with a given value  
  mkdir       make a new directory  
  rm          remove a key or a directory  
  rmdir      removes the key if it is an empty directory or a key-value pa  
  get        retrieve the value of a key
```





# Course Objectives



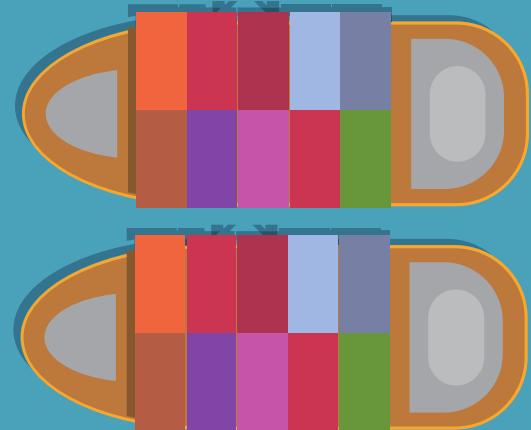
# ETCD In Kubernetes



## Master

Manage, Plan, Schedule, Monitor  
Nodes

- Nodes
- PODs
- Configs
- Secrets
- Accounts
- Roles
- Bindings
- Others



# Setup - Manual

```
▶ wget -q --https-only \
  "https://github.com/coreos/etcd/releases/download/v3.3.9/etcd-v3.3.9-linux-amd64.tar.gz"
```

## etcd.service

```
ExecStart=/usr/local/bin/etcd \
  --name ${ETCD_NAME} \
  --cert-file=/etc/etcd/kubernetes.pem \
  --key-file=/etc/etcd/kubernetes-key.pem \
  --peer-cert-file=/etc/etcd/kubernetes.pem \
  --peer-key-file=/etc/etcd/kubernetes-key.pem \
  --trusted-ca-file=/etc/etcd/ca.pem \
  --peer-trusted-ca-file=/etc/etcd/ca.pem \
  --peer-client-cert-auth \
  --client-cert-auth \
  --initial-advertise-peer-urls https://${INTERNAL_IP}:2380 \
  --listen-peer-urls https://${INTERNAL_IP}:2380 \
  --listen-client-urls https://${INTERNAL_IP}:2379,https://127.0.0.1:2379 \
  --advertise-client-urls https://${INTERNAL_IP}:2379 \
  --initial-cluster-token etcd-cluster-0 \
  --initial-cluster controller-0=https://${CONTROLLER0_IP}:2380,controller-1=https://${CONTROLLER1_IP}:2380 \
  --initial-cluster-state new \
  --data-dir=/var/lib/etcd
```

# Setup - kubeadm

```
▶ kubectl get pods -n kube-system
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	coredns-78fcdf6894-prwvl	1/1	Running	0	1h
kube-system	coredns-78fcdf6894-vqd9w	1/1	Running	0	1h
<b>kube-system</b>	<b>etcd-master</b>	<b>1/1</b>	<b>Running</b>	<b>0</b>	<b>1h</b>
kube-system	kube-apiserver-master	1/1	Running	0	1h
kube-system	kube-controller-manager-master	1/1	Running	0	1h
kube-system	kube-proxy-f6k26	1/1	Running	0	1h
kube-system	kube-proxy-hnzsw	1/1	Running	0	1h
kube-system	kube-scheduler-master	1/1	Running	0	1h
kube-system	weave-net-924k8	2/2	Running	1	1h
kube-system	weave-net-hzfcz	2/2	Running	1	1h

```
▶ kubectl exec etcd-master -n kube-system etcdctl get / --prefix --keys-only
```

```
/registry/apiregistration.k8s.io/apiservices/v1.  
/registry/apiregistration.k8s.io/apiservices/v1.apps  
/registry/apiregistration.k8s.io/apiservices/v1.authentication.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1.authorization.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1.autoscaling  
/registry/apiregistration.k8s.io/apiservices/v1.batch  
/registry/apiregistration.k8s.io/apiservices/v1.networking.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1.rbac.authorization.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1.storage.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1beta1.admissionregistration.k8s.io
```

Run inside the etcd-master POD

# Explore ETCD

```
▶ kubectl exec etcd-master -n kube-system etcdctl get / --prefix --keys-only  
/registry/apiregistration.k8s.io/apiservices/v1.  
/registry/apiregistration.k8s.io/apiservices/v1.apps  
/registry/apiregistration.k8s.io/apiservices/v1.authentication.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1.authorization.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1.autoscaling  
/registry/apiregistration.k8s.io/apiservices/v1.batch  
/registry/apiregistration.k8s.io/apiservices/v1.networking.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1.rbac.authorization.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1.storage.k8s.io  
/registry/apiregistration.k8s.io/apiservices/v1beta1.admissionregistration.k8s.io
```

Run inside the etcd-master POD

Registry

minions

pods

replicasets

deployments

roles

secrets

# ETCD in HA Environment



## etcd.service

```
ExecStart=/usr/local/bin/etcd \
--name ${ETCD_NAME} \
--cert-file=/etc/etcd/kubernetes.pem \
--key-file=/etc/etcd/kubernetes-key.pem \
--peer-cert-file=/etc/etcd/kubernetes.pem \
--peer-key-file=/etc/etcd/kubernetes-key.pem \
--trusted-ca-file=/etc/etcd/ca.pem \
--peer-trusted-ca-file=/etc/etcd/ca.pem \
--peer-client-cert-auth \
--client-cert-auth \
--initial-advertise-peer-urls https://${INTERNAL_IP}:2380 \
--listen-peer-urls https://${INTERNAL_IP}:2380 \
--listen-client-urls https://${INTERNAL_IP}:2379,https://127.0.0.1:2379 \
--advertise-client-urls https://${INTERNAL_IP}:2379 \
--initial-cluster-token etcd-cluster-0 \
--initial-cluster controller-0=https://${CONTROLLER0_IP}:2380,controller-1=https://${CONTROLLER1_IP}:2380 \
--initial-cluster-state new \
--data-dir=/var/lib/etcd
```



# Course Objectives

Core Concepts

Cluster Architecture

Services & Other Network Primitives

API Primitives

Scheduling

Logging Monitoring

Application Lifecycle Management

Cluster Maintenance

Security

Storage

Networking

Installation, Configuration & Validation

Troubleshooting

# kube-api server



**kube-apiserver**

**Master**

Manage, Plan, Schedule, Monitor  
Nodes



**Worker Nodes**

Host Application as Containers



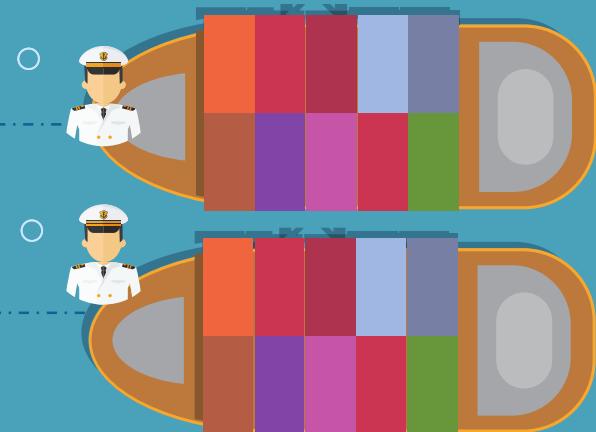
**kubelet**

**Controller-  
Manager**

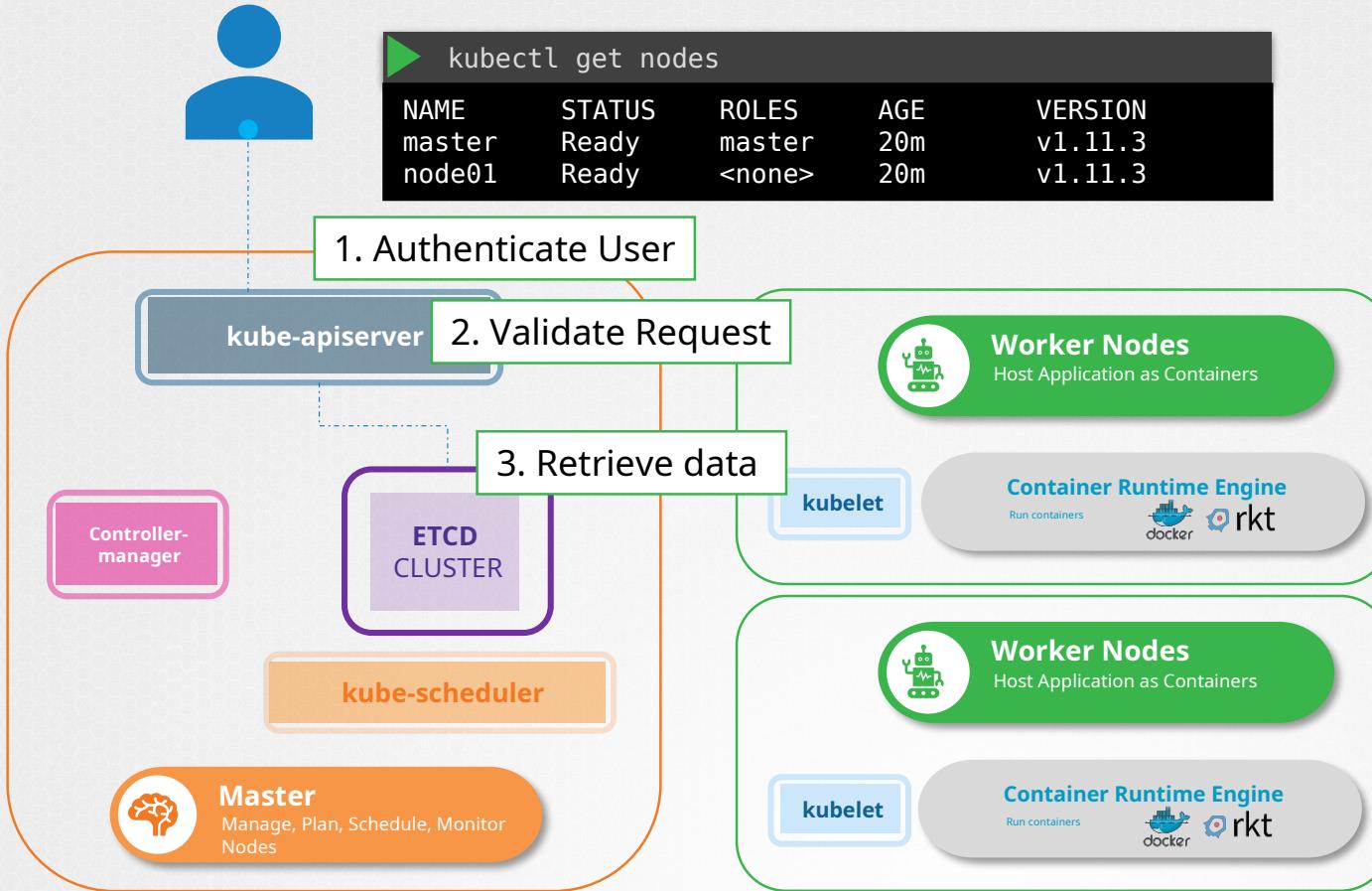
**CLUSTER**

**etcd**

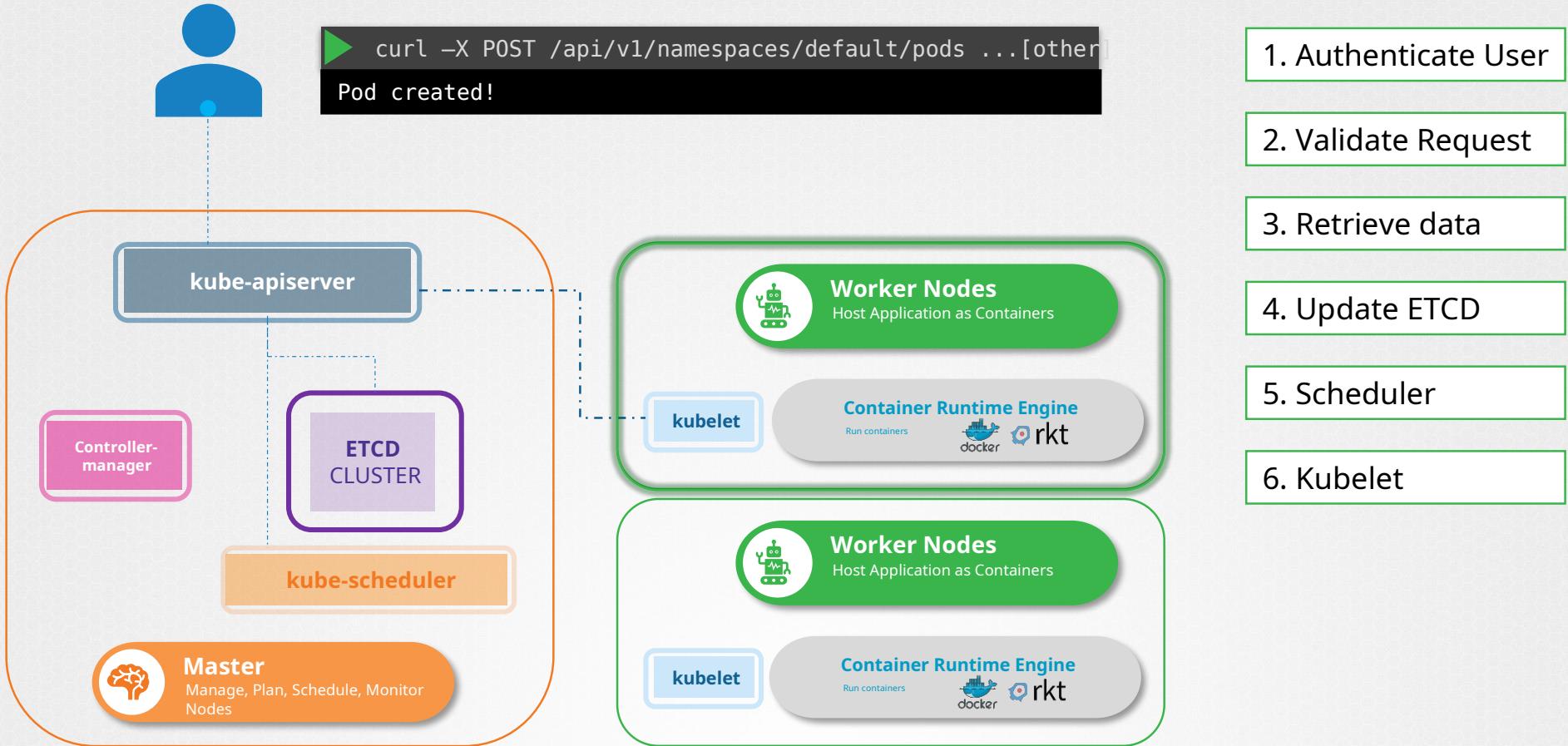
**re-scheduler**



# Kubernetes Architecture



# Kubernetes Architecture



# Kube-api Server

1. Authenticate User

2. Validate Request

3. Retrieve data

4. Update ETCD

5. Scheduler

6. Kubelet

# Installing kube-api server

```
▶ wget https://storage.googleapis.com/kubernetes-release/release/v1.13.0/bin/linux/amd64/kube-apiserver
```

## kube-apiserver.service

```
ExecStart=/usr/local/bin/kube-apiserver \
--advertise-address=${INTERNAL_IP} \
--allow-privileged=true \
--apiserver-count=3 \
--authorization-mode=Node,RBAC \
--bind-address=0.0.0.0 \
--enable-admission-
plugins=Initializers,NamespaceLifecycle,NodeRestriction,LimitRanger,ServiceAccount,DefaultStorageClass,Reso
urceQuota \
--enable-swagger-ui=true \
--etcd-servers=https://127.0.0.1:2379 \
--event-ttl=1h \
--experimental-encryption-provider-config=/var/lib/kubernetes/encryption-config.yaml \
--runtime-config=api/all \
--service-account-key-file=/var/lib/kubernetes/service-account.pem \
--service-cluster-ip-range=10.32.0.0/24 \
--service-node-port-range=30000-32767 \
--v=2
```

# View api-server - kubeadm

```
kubectl get pods -n kube-system
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	coredns-78fcdf6894-hwrq9	1/1	Running	0	16m
kube-system	coredns-78fcdf6894-rzhjr	1/1	Running	0	16m
kube-system	etcd-master	1/1	Running	0	15m
kube-system	kube-apiserver-master	1/1	Running	0	15m
kube-system	kube-controller-manager-master	1/1	Running	0	15m
kube-system	kube-proxy-lzt6f	1/1	Running	0	16m
kube-system	kube-proxy-zm5qd	1/1	Running	0	16m
kube-system	kube-scheduler-master	1/1	Running	0	15m
kube-system	weave-net-29z42	2/2	Running	1	16m
kube-system	weave-net-snmdl	2/2	Running	1	16m

# View api-server options - kubeadm

```
▶ cat /etc/kubernetes/manifests/kube-apiserver.yaml
```

```
spec:
  containers:
    - command:
        - kube-apiserver
        - --authorization-mode=Node,RBAC
        - --advertise-address=172.17.0.32
        - --allow-privileged=true
        - --client-ca-file=/etc/kubernetes/pki/ca.crt
        - --disable-admission-plugins=PersistentVolumeLabel
        - --enable-admission-plugins=NodeRestriction
        - --enable-bootstrap-token-auth=true
        - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
        - --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt
        - --etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key
        - --etcd-servers=https://127.0.0.1:2379
        - --insecure-port=0
        - --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-kubelet-client.crt
        - --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-client.key
        - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
        - --proxy-client-cert-file=/etc/kubernetes/pki/front-proxy-client.crt
        - --proxy-client-key-file=/etc/kubernetes/pki/front-proxy-client.key
        - --requestheader-allowed-names=front-proxy-client
        - --requestheader-client-ca-file=/etc/kubernetes/pki/front-proxy-ca.crt
        - --requestheader-extra-headers-prefix=X-Remote-Extra-
        - --requestheader-group-headers=X-Remote-Group
        - --requestheader-username-headers=X-Remote-User
```

# View api-server options

```
▶ cat /etc/systemd/system/kube-apiserver.service
```

```
[Service]
ExecStart=/usr/local/bin/kube-apiserver \\
--advertise-address=${INTERNAL_IP} \\
--allow-privileged=true \\
--apiserver-count=3 \\
--audit-log-maxage=30 \\
--audit-log-maxbackup=3 \\
--audit-log-maxsize=100 \\
--audit-log-path=/var/log/audit.log \\
--authorization-mode=Node,RBAC \\
--bind-address=0.0.0.0 \\
--client-ca-file=/var/lib/kubernetes/ca.pem \\
--enable-admission-
plugins=Initializers,NamespaceLifecycle,NodeRestriction,LimitRanger,ServiceAccount,Defa
ultStorageClass,ResourceQuota \\
--enable-swagger-ui=true \\
--etcd-cafile=/var/lib/kubernetes/ca.pem \\
--etcd-certfile=/var/lib/kubernetes/kubernetes.pem \\
--etcd-keyfile=/var/lib/kubernetes/kubernetes-key.pem \\
--etcd-
servers=https://10.240.0.10:2379,https://10.240.0.11:2379,https://10.240.0.12:2379 \\
--event-ttl=1h \\
--experimental-encryption-provider-config=/var/lib/kubernetes/encryption-config.yaml
\\
--kubelet-certificate-authority=/var/lib/kubernetes/ca.pem \\
--kubelet-client-certificate=/var/lib/kubernetes/kubernetes.pem \\
```

# View api-server options

```
▶ ps -aux | grep kube-apiserver
```

```
root      2348  3.3 15.4 399040 315604 ?        Sb 1:46   1:22 kube-apiserver --authorization-mode=Node,RBAC --  
advertise-address=172.17.0.32 --allow-privileged=true --client-ca-file=/etc/kubernetes/pki/ca.crt --disable-admission-plugins=PersistentVolumeLabel --enable-admission-plugins=NodeRestriction--enable-bootstrap-token-auth=true --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt --etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key --etcd-servers=https://127.0.0.1:2379 insecure-port=0 --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-kubelet-client.crt --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-client.key --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname --proxy-client-cert-file=/etc/kubernetes/pki/front-proxy-client.crt --proxy-client-key-file=/etc/kubernetes/pki/front-proxy-client.key--requestheader-allowed-names=front-proxy-client --requestheader-client-ca-file=/etc/kubernetes/pki/front-proxy-ca.crt --requestheader-extra-headers-prefix=X-Remote-Extra- --requestheader-group-headers=X-Remote-Group --requestheader-username-headers=X-Remote-User --secure-port=6443 --service-account-key-file=/etc/kubernetes/pki/sa.pub --service-cluster-ip-range=10.96.0.0/12 --tls-cert-file=/etc/kubernetes/pki/apiserver.crt --tls-private-key-file=/etc/kubernetes/pki/apiserver.key
```



# Course Objectives

40

Core Concepts

Cluster Architecture

Services & Other Network Primitives

Scheduling

Logging Monitoring

Application Lifecycle Management

Cluster Maintenance

Security

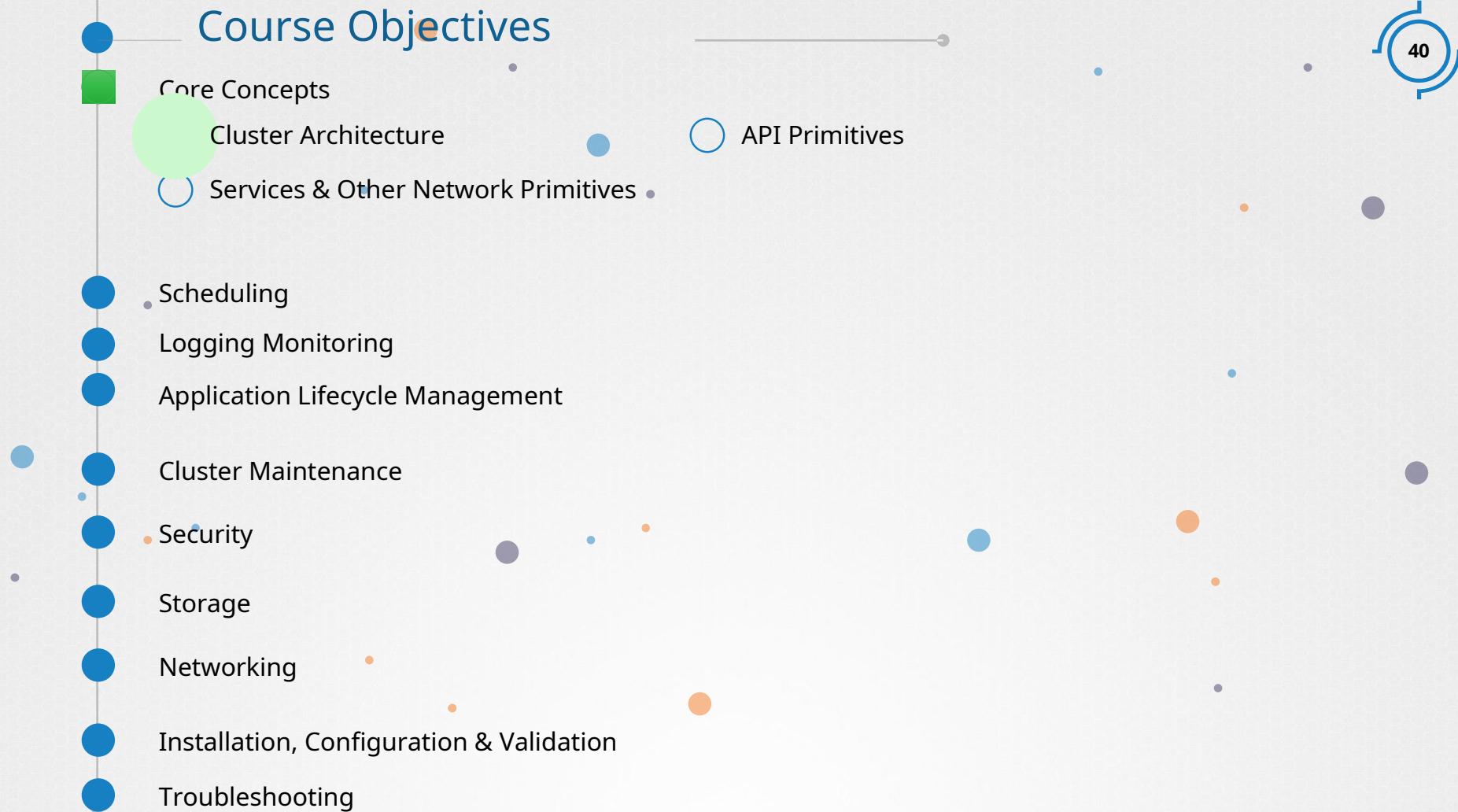
Storage

Networking

Installation, Configuration & Validation

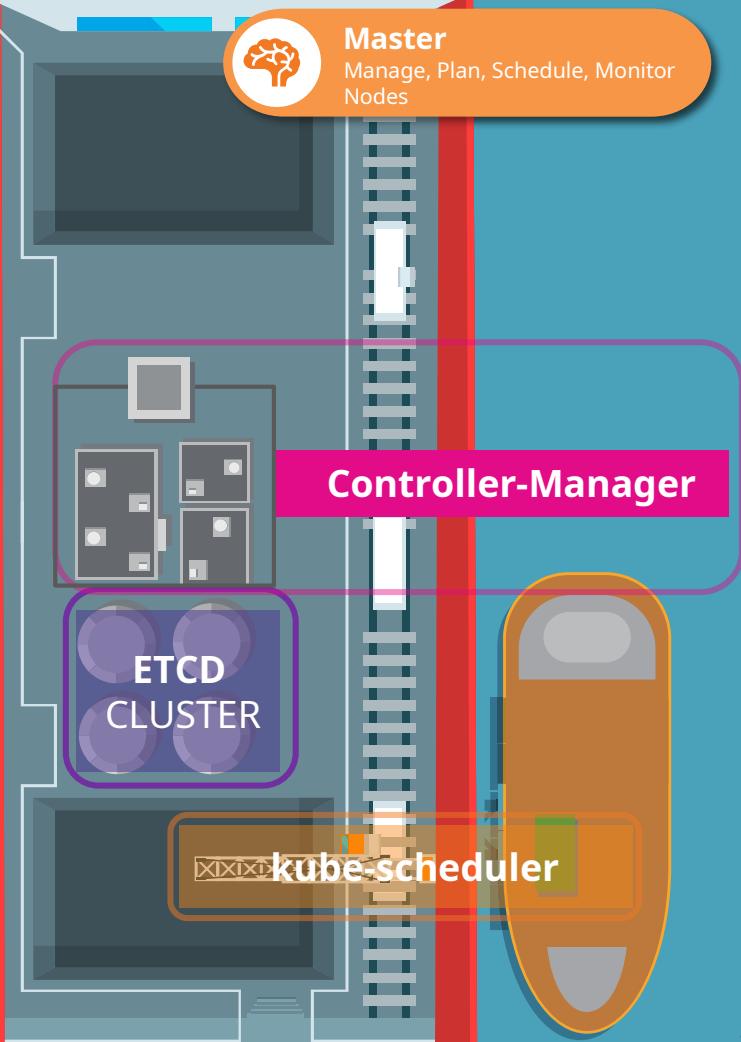
Troubleshooting

API Primitives



# Kube Controller Manager





## Master

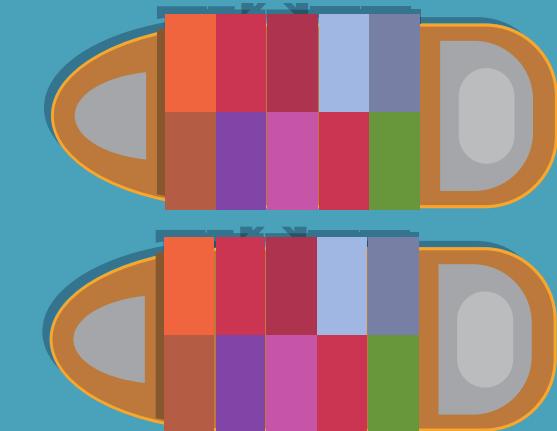
Manage, Plan, Schedule, Monitor  
Nodes

## Worker Nodes

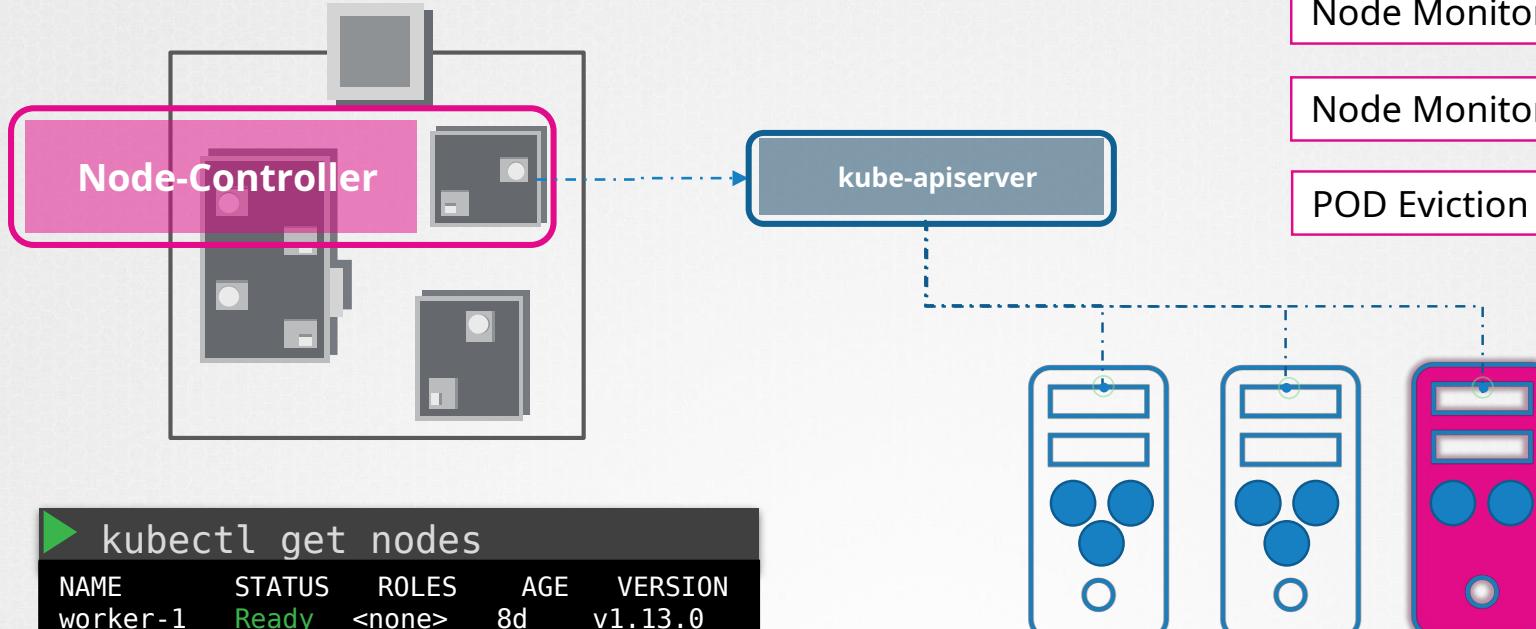
Host Application as Containers

Watch Status

Remediate Situation



# Controller



Watch Status

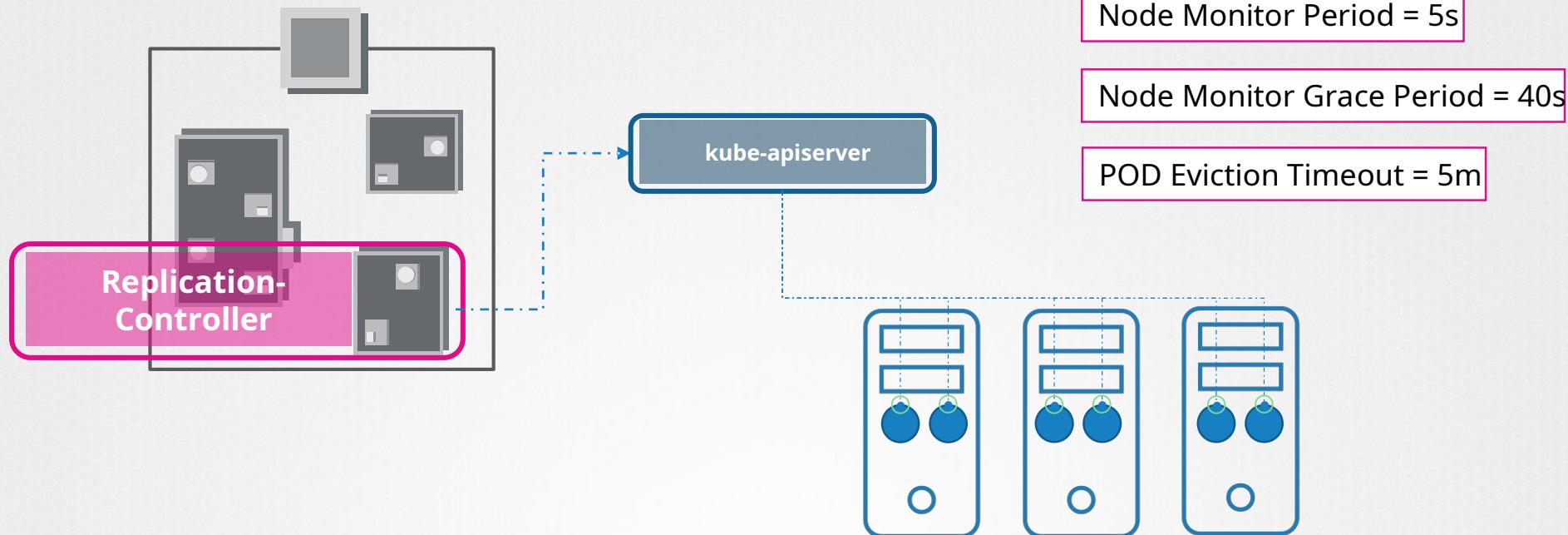
Remediate Situation

Node Monitor Period = 5s

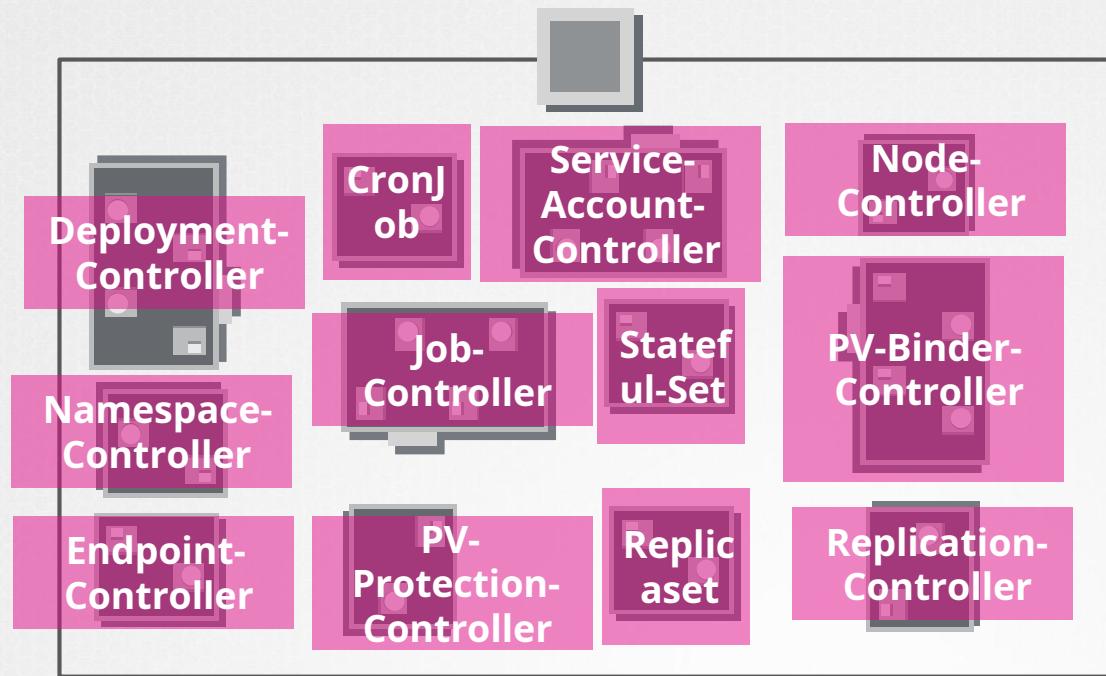
Node Monitor Grace Period = 40s

POD Eviction Timeout = 5m

# Controller



# Controller



Watch Status

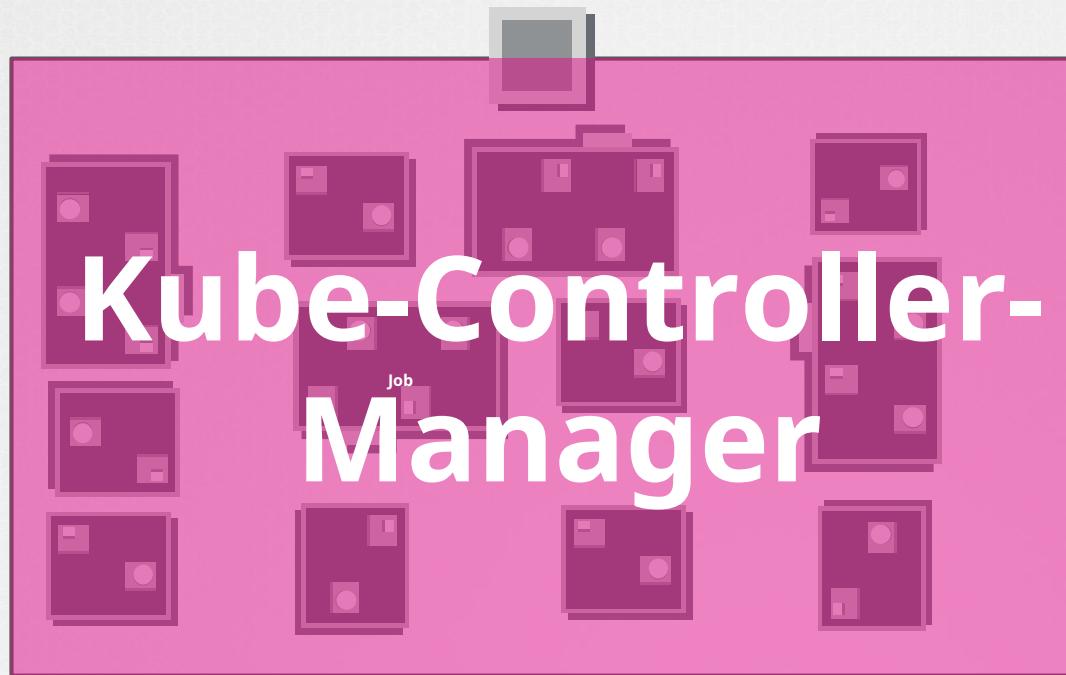
Remediate Situation

Node Monitor Period = 5s

Node Monitor Grace Period = 40s

POD Eviction Timeout = 5m

# Controller



Watch Status

Remediate Situation

Node Monitor Period = 5s

Node Monitor Grace Period = 40s

POD Eviction Timeout = 5m

# Installing kube-controller-manager

```
▶ wget https://storage.googleapis.com/kubernetes-release/release/v1.13.0/bin/linux/amd64/kube-controller-ma
```

## kube-controller-manager.service

```
ExecStart=/usr/local/bin/kube-controller-manager \
--address=0.0.0.0 \
--cluster-cidr=10.200.0.0/16 \
--cluster-name=kubernetes \
--cluster-signing-cert-file=/var/lib/kubernetes/ca.pem \
--cluster-signing-key-file=/var/lib/kubernetes/ca-key.pem \
--kubeconfig=/var/lib/kubernetes/kube-controller-manager.kubeconfig \
--leader-elect=true \
--root-ca-file=/var/lib/kubernetes/ca.pem \
--service-account-private-key-file=/var/lib/kubernetes/service-account-key.pem \
--service-cluster-ip-range=10.32.0.0/24 \
--use-service-account-credentials=true \
--v=2
```

--node-monitor-period=5s

```
--controllers stringSlice Default: [*]
A list of controllers to enable. '*' enables all on-by-default controllers, 'foo' enables the controller
named 'foo', '-foo' disables the controller named 'foo'.
All controllers: attachdetach, bootstrapsigner, clusterrole-aggregation, cronjob, csrapproving,
csrcleaner, csrsigning, daemonset, deployment, disruption, endpoint, garbagecollector,
horizontalpodautoscaling, job, namespace, nodeipam, nodelifecycle, persistentvolume-binder,
persistentvolume-expander, podgc, pv-protection, pvc-protection, replicaset, replicationcontroller,
```

# Installing kube-controller-manager

```
--controllers stringSlice  Default: [*]
A list of controllers to enable. '*' enables all on-by-default controllers, 'foo' enables the controller named 'foo', '-foo' disables the controller named 'foo'.
All controllers: attachdetach, bootstrapsigner, clusterrole-aggregation, cronjob, csrapproving, csrcleaner, csrsigning, daemonset, deployment, disruption, endpoint, garbagecollector, horizontalpodautoscaling, job, namespace, nodeipam, nodelifecycle, persistentvolume-binder, persistentvolume-expander, podgc, pv-protection, pvc-protection, replicaset, replicationcontroller, resourcequota, root-ca-cert-publisher, route, service, serviceaccount, serviceaccount-token, statefulset, token-cleaner, ttl, ttl-after-finished
Disabled-by-default controllers: bootstrapsigner, token-cleaner
```

# View kube-controller-manager - kubeadm

```
kubectl get pods -n kube-system
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	coredns-78fcdf6894-hwrq9	1/1	Running	0	16m
kube-system	coredns-78fcdf6894-rzhjr	1/1	Running	0	16m
kube-system	etcd-master	1/1	Running	0	15m
kube-system	kube-apiserver-master	1/1	Running	0	15m
kube-system	<b>kube-controller-manager-master</b>	<b>1/1</b>	<b>Running</b>	<b>0</b>	<b>15m</b>
kube-system	kube-proxy-lzt6f	1/1	Running	0	16m
kube-system	kube-proxy-zm5qd	1/1	Running	0	16m
kube-system	kube-scheduler-master	1/1	Running	0	15m
kube-system	weave-net-29z42	2/2	Running	1	16m
kube-system	weave-net-snmdl	2/2	Running	1	16m

# View kube-controller-manager options - kubeadm

```
▶ cat /etc/kubernetes/manifests/kube-controller-manager.yaml
```

```
spec:  
  containers:  
    - command:  
        - kube-controller-manager  
        - --address=127.0.0.1  
        - --cluster-signing-cert-file=/etc/kubernetes/pki/ca.crt  
        - --cluster-signing-key-file=/etc/kubernetes/pki/ca.key  
        - --controllers=*,bootstrapsigner,tokencleaner  
        - --kubeconfig=/etc/kubernetes/controller-manager.conf  
        - --leader-elect=true  
        - --root-ca-file=/etc/kubernetes/pki/ca.crt  
        - --service-account-private-key-file=/etc/kubernetes/pki/sa.key  
        - --use-service-account-credentials=true
```

# View controller-manager options

```
▶ cat /etc/systemd/system/kube-controller-manager.service
```

```
[Service]
ExecStart=/usr/local/bin/kube-controller-manager \\
--address=0.0.0.0 \\
--cluster-cidr=10.200.0.0/16 \\
--cluster-name=kubernetes \\
--cluster-signing-cert-file=/var/lib/kubernetes/ca.pem \\
--cluster-signing-key-file=/var/lib/kubernetes/ca-key.pem \\
--kubeconfig=/var/lib/kubernetes/kube-controller-manager.kubeconfig \\
--leader-elect=true \\
--root-ca-file=/var/lib/kubernetes/ca.pem \\
--service-account-private-key-file=/var/lib/kubernetes/service-account-key.pem \\
--service-cluster-ip-range=10.32.0.0/24 \\
--use-service-account-credentials=true \\
--v=2
Restart=on-failure
RestartSec=5
```

# View controller-manager options

```
▶ ps -aux | grep kube-controller-manager
```

```
root      1994  2.7  5.1 154360 105024 ?          00:45   1:25 kube-controller-manager --  
address=127.0.0.1 --cluster-signing-cert-file=/etc/kubernetes/pki/ca.crt --cluster-signing-  
key-file=/etc/kubernetes/pki/ca.key --controllers=*,bootstrapsigner,tokencleaner --  
kubeconfig=/etc/kubernetes/controller-manager.conf --leader-elect=true --root-ca-  
file=/etc/kubernetes/pki/ca.crt --service-account-private-key-file=/etc/kubernetes/pki/sa.k  
--use-service-account-credentials=true
```

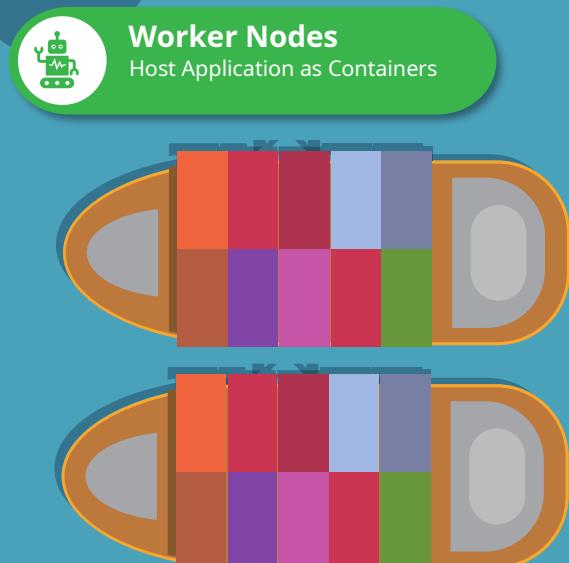
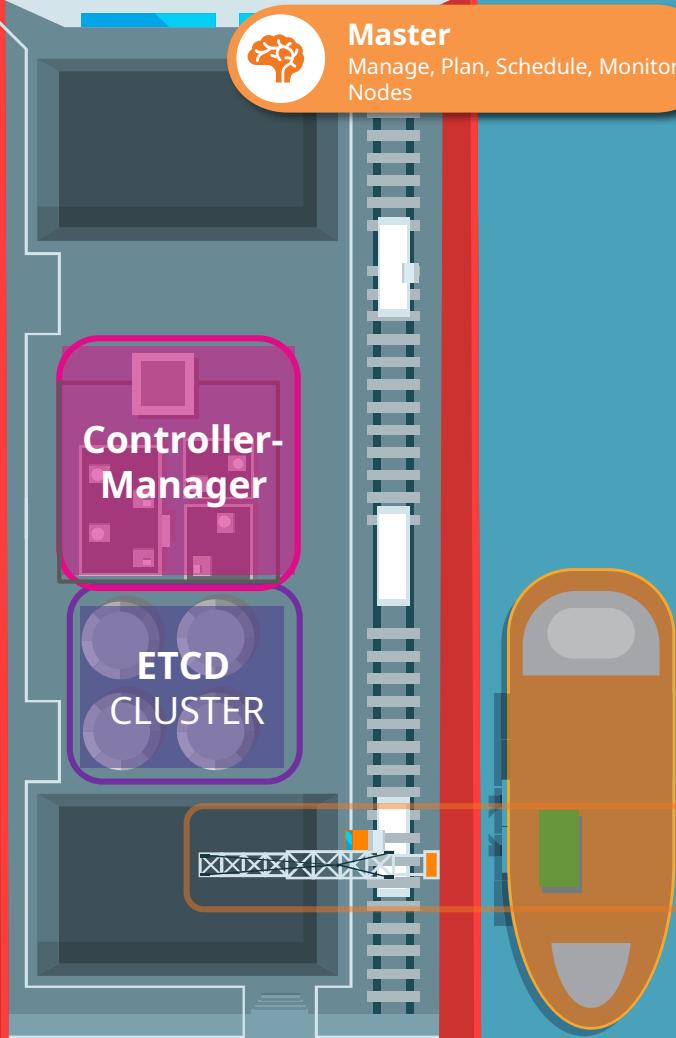


# Course Objectives

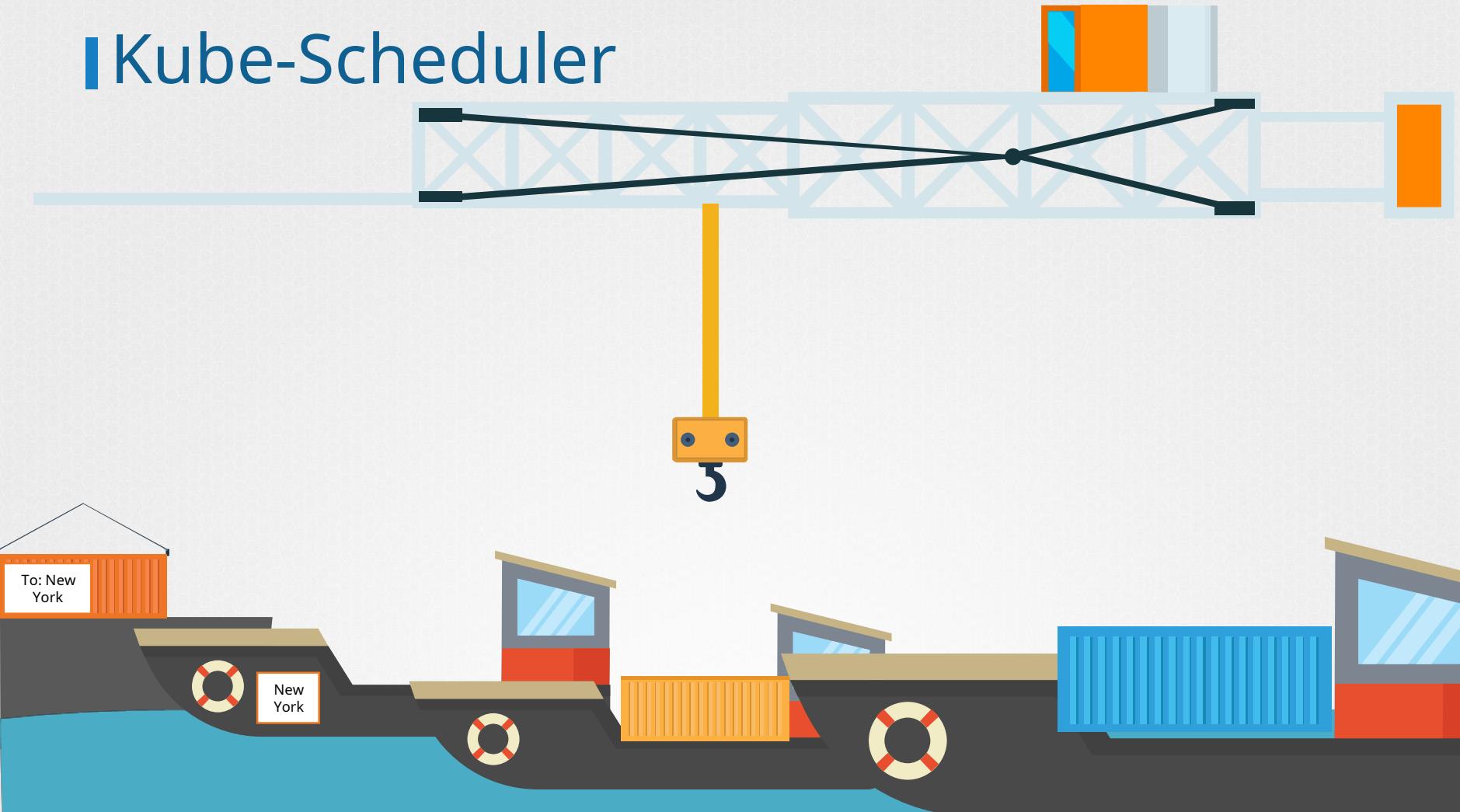


# Kube Scheduler

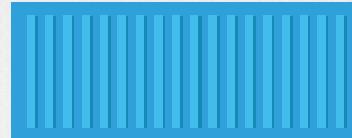
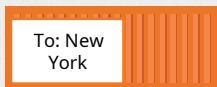




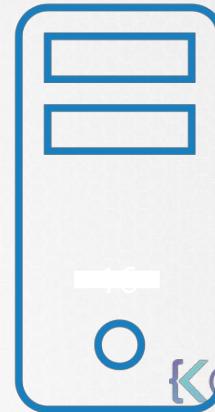
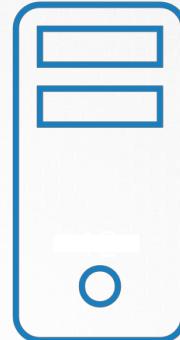
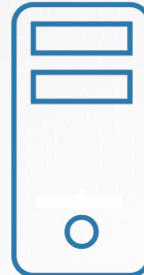
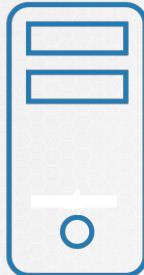
# Kube-Scheduler



# Kube-Scheduler



The scheduler decides which Pod should be put onto which Node.  
The actual deployment of the Pod is performed by the kubelet.

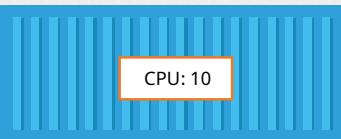
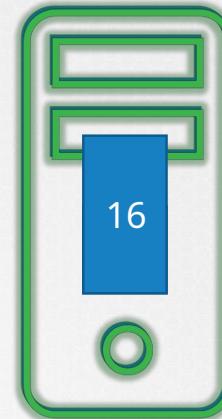
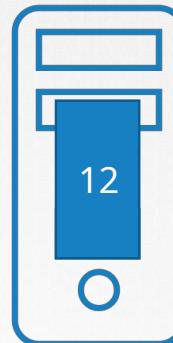


# Kube-Scheduler

1. Filter Nodes



2. Rank Nodes



# | More Later...

- Resource Requirements and Limits
- Taints and Tolerations
- Node Selectors/Affinity

# Course Objectives

## Scheduling

- Labels & Selectors
- Resource Limits
- Manual Scheduling
- Daemon Sets
- Multiple Schedulers
- Scheduler Events
- Configure Kubernetes Scheduler

## Logging Monitoring

## Application Lifecycle Management

## Cluster Maintenance

## Security

## Storage

## Troubleshooting

# Installing kube-scheduler

```
▶ wget https://storage.googleapis.com/kubernetes-release/release/v1.13.0/bin/linux/amd64/kube-scheduler
```

```
kube-scheduler.service
```

```
ExecStart=/usr/local/bin/kube-scheduler \
--config=/etc/kubernetes/config/kube-scheduler.yaml \
--v=2
```

# I View kube-scheduler options - kubeadm

```
▶ cat /etc/kubernetes/manifests/kube-scheduler.yaml
```

```
spec:  
  containers:  
    - command:  
        - kube-scheduler  
        - --address=127.0.0.1  
        - --kubeconfig=/etc/kubernetes/scheduler.conf  
        - --leader-elect=true
```

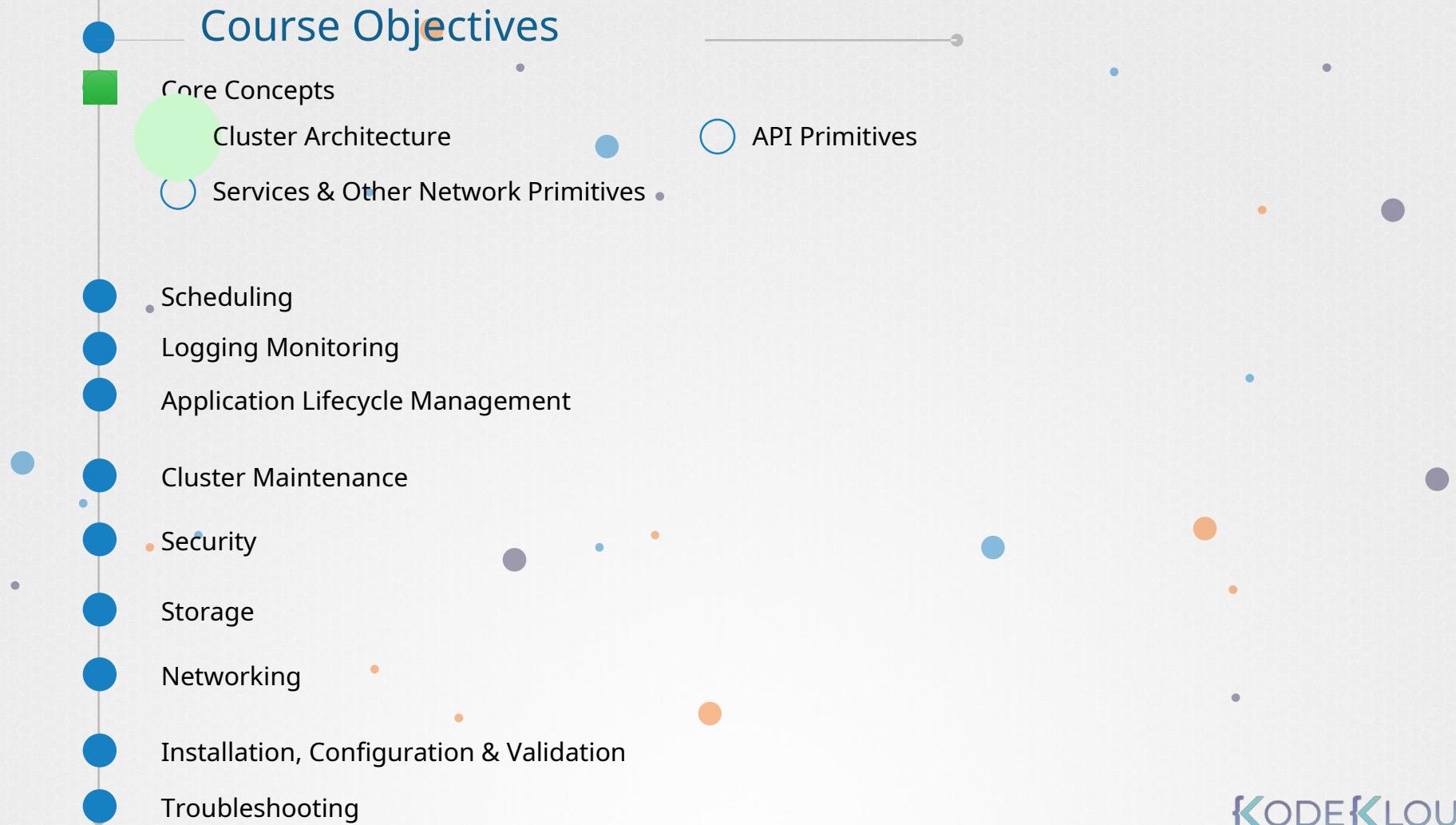
# View kube-scheduler options

```
▶ ps -aux | grep kube-scheduler
```

```
root      2477  0.8  1.6  48524 34044 ?          Ssl:31   0:08 kube-scheduler --  
address=127.0.0.1 --kubeconfig=/etc/kubernetes/scheduler.conf --leader-elect=true
```



# Course Objectives



# Kubelet



# kube-apiserver

## Master

Manage, Plan, Schedule, Monitor  
Nodes



Controller-  
Manager



CLOUD  
CLUSTER



e-scheduler

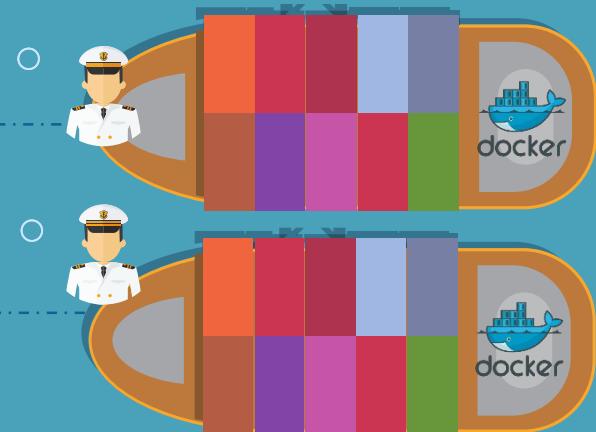


# kubelet



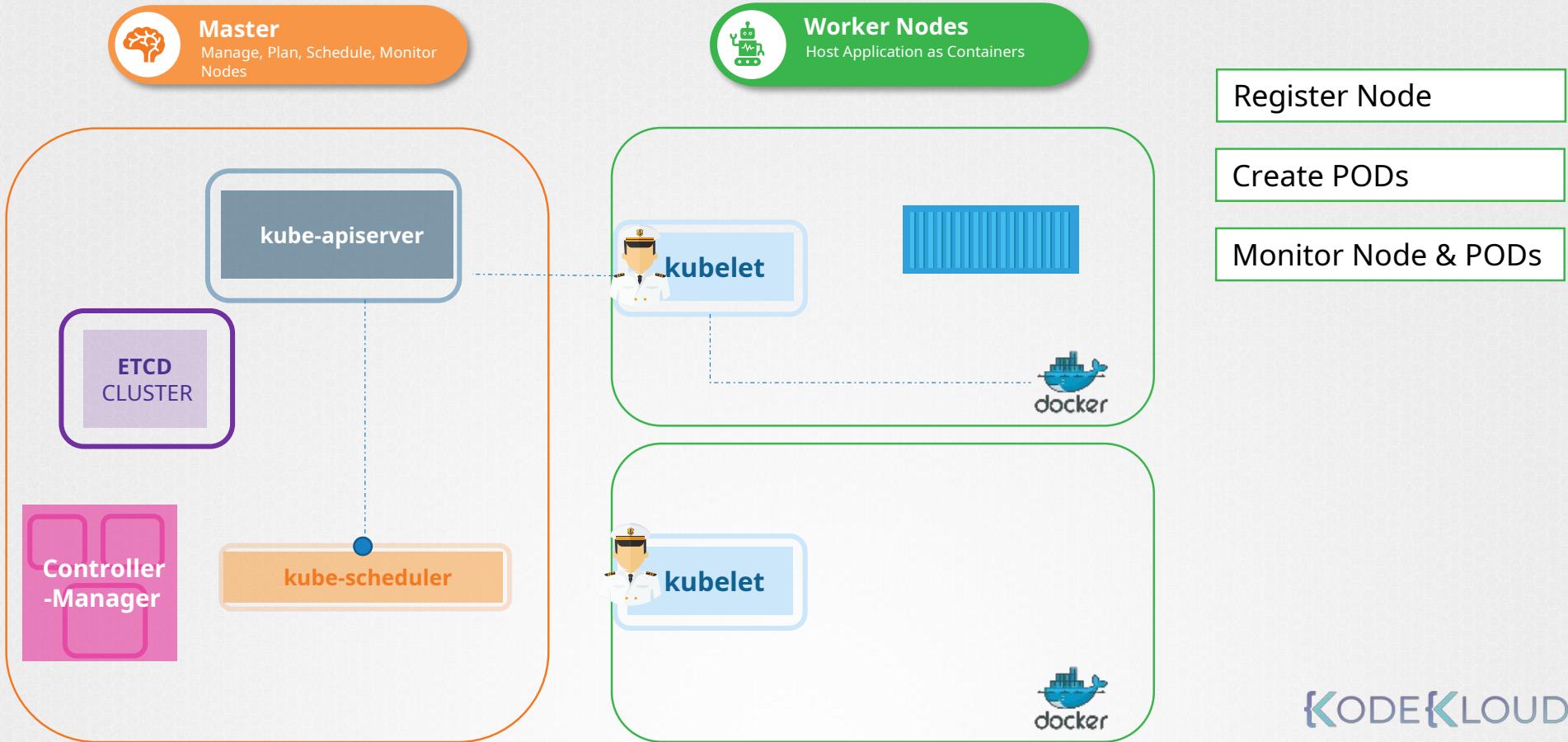
## Worker Nodes

Host Application as Containers



- kubelet load and unload containers as scheduled by master.
- registers node with kubernetes cluster
- instructs the container runtime to pull image and run container
- needs to be installed manually always

# Kubernetes Architecture



# Installing kubelet

```
▶ wget https://storage.googleapis.com/kubernetes-release/release/v1.13.0/bin/linux/amd64/kubelet
```

## kubelet.service

```
ExecStart=/usr/local/bin/kubelet \
--config=/var/lib/kubelet/kubelet-config.yaml \
--container-runtime=remote \
--container-runtime-endpoint=unix:///var/run/containerd/containerd.sock \
--image-pull-progress-deadline=2m \
--kubeconfig=/var/lib/kubelet/kubeconfig \
--network-plugin=cni \
--register-node=true \
--v=2
```



Kubeadm does not deploy  
Kubelets

# View kubelet options

```
▶ ps -aux | grep kubelet
```

```
root      2095  1.8  2.4 960676 98788 ?          Ss2:32   0:36 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf config=/var/lib/kubelet/config.yaml --cgroup-driver=cgroupfs --cni-bin-dir=/opt/cni/bin --conf-dir=/etc/cni/net.d --network-plugin=cni
```



# Course Objectives

Core Concepts

Cluster Architecture

Services & Other Network Primitives

API Primitives

Scheduling

Logging Monitoring

Application Lifecycle Management

Cluster Maintenance

Security

Storage

Networking

Installation, Configuration & Validation

Troubleshooting

# Kube-proxy



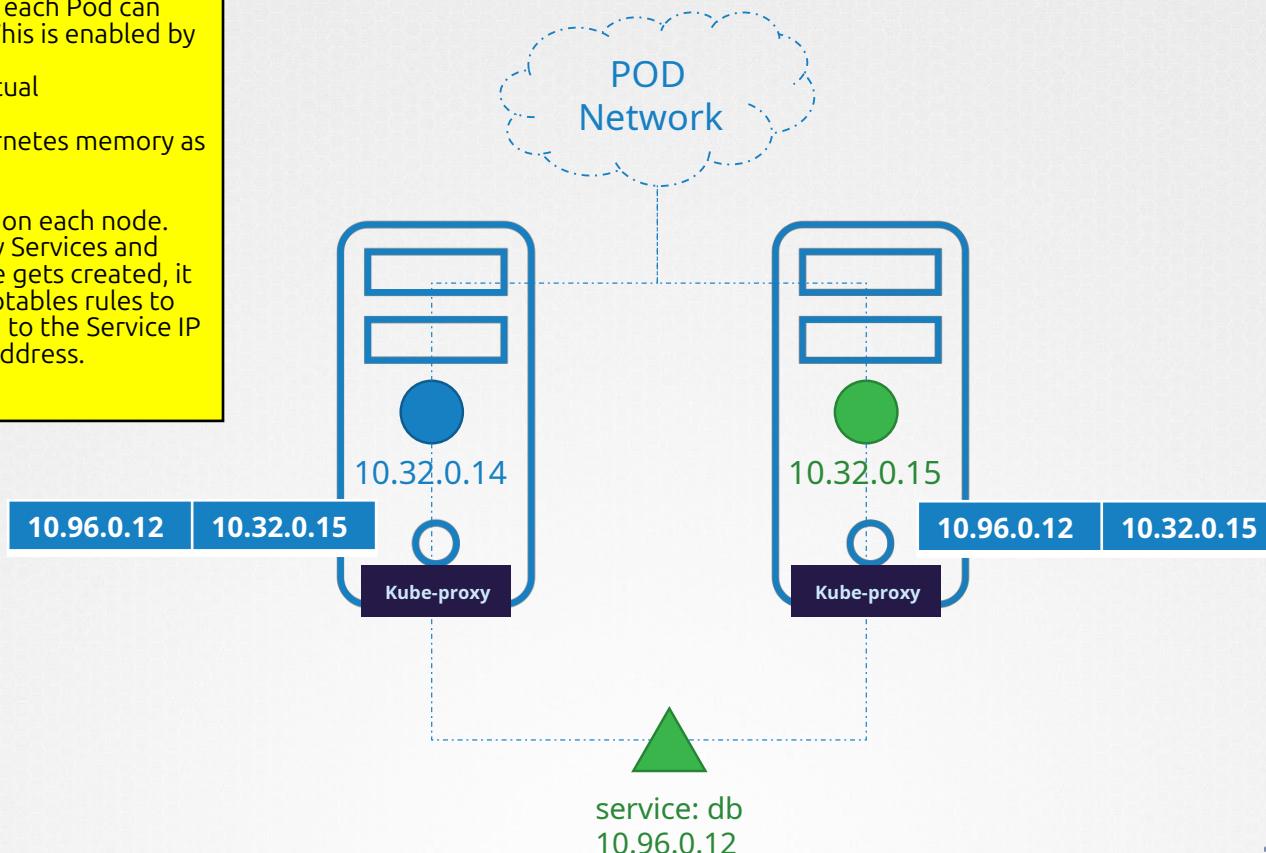
# Kube-proxy

In a kubernetes cluster, each Pod can reach each other Pod. This is enabled by Services.

The service is not an actual thing/container/Pod.

It only lives in the kubernetes memory as a virtual process.

kube-proxy is a process on each node. Its job is to look for new Services and whenever a new Service gets created, it creates the necessary iptables rules to forward traffic directed to the Service IP to the relevant Pod IP address.



# Installing kube-proxy

```
▶ wget https://storage.googleapis.com/kubernetes-release/release/v1.13.0/bin/linux/amd64/kube-proxy
```

## kube-proxy.service

```
ExecStart=/usr/local/bin/kube-proxy \\
--config=/var/lib/kube-proxy/kube-proxy-config.yaml
Restart=on-failure
RestartSec=5
```

# View kube-proxy - kubeadm

```
▶ kubectl get pods -n kube-system
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	coredns-78fcdf6894-hwrq9	1/1	Running	0	16m
kube-system	coredns-78fcdf6894-rzhjr	1/1	Running	0	16m
kube-system	etcd-master	1/1	Running	0	15m
kube-system	kube-apiserver-master	1/1	Running	0	15m
kube-system	kube-controller-manager-master	1/1	Running	0	15m
kube-system	kube-proxy-lzt6f	1/1	Running	0	16m
kube-system	kube-proxy-zm5qd	1/1	Running	0	16m
kube-system	kube-scheduler-master	1/1	Running	0	15m
kube-system	weave-net-29z42	2/2	Running	1	16m
kube-system	weave-net-snmdl	2/2	Running	1	16m

```
▶ kubectl get daemonset -n kube-system
```

NAME	DESIRED	CURRENT	READY	UP-TO-DATE	AVAILABLE	NODE SELECTOR	AGE
kube-proxy	2	2	2	2	2	beta.kubernetes.io/arch=amd64	1h