TEL4VN

Building the k8s cluster

THAO LUONG
03/2022

Content

❑ Design a kubernetes cluster

❑ Installing k8s master and worker nodes

❑ Building a Highly Available k8s cluster

❑ Configuring secure cluster communication

❑ Running end-to-end tests

Design a Kubernetes cluster

- ❑ Purpose
- ❑ Cloud or On-prem
- ❑ Workload of cluster

Purpose

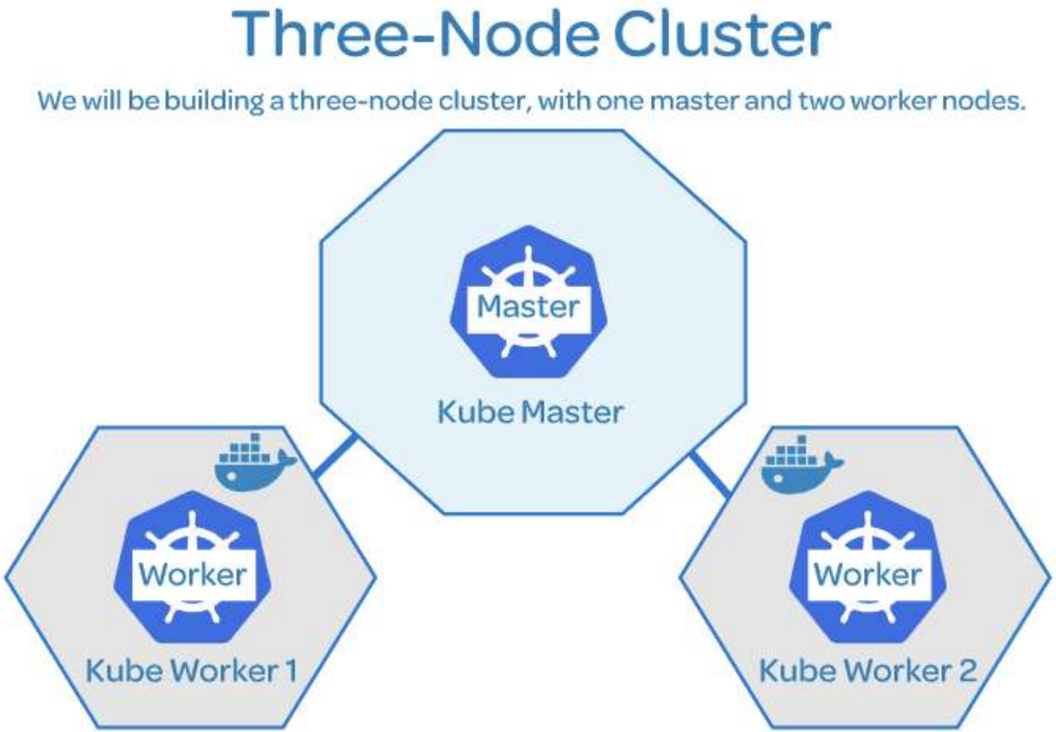
- Education
  - Minikube, kind / single node
- Developing/Testing
  - Cluster with one master/ multi worker.
  - Cloud resource.
- Production application

Cloud or onPrem

- □ Use Kubeadm to install on Prem
- □ Use cloud service

Google Container Engine (GKE)
OpenShift Online
Azure Kubernetes Service
aws
Amazon Elastic Container Service for Kubernetes (EKS)

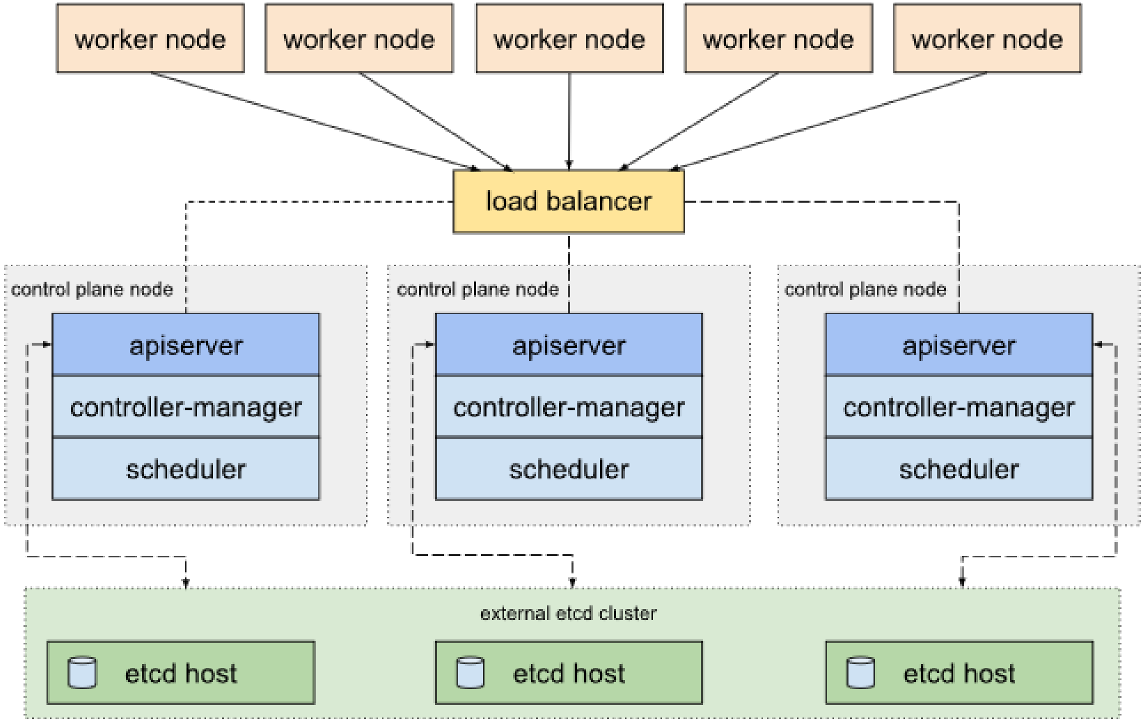
Install the K8s cluster



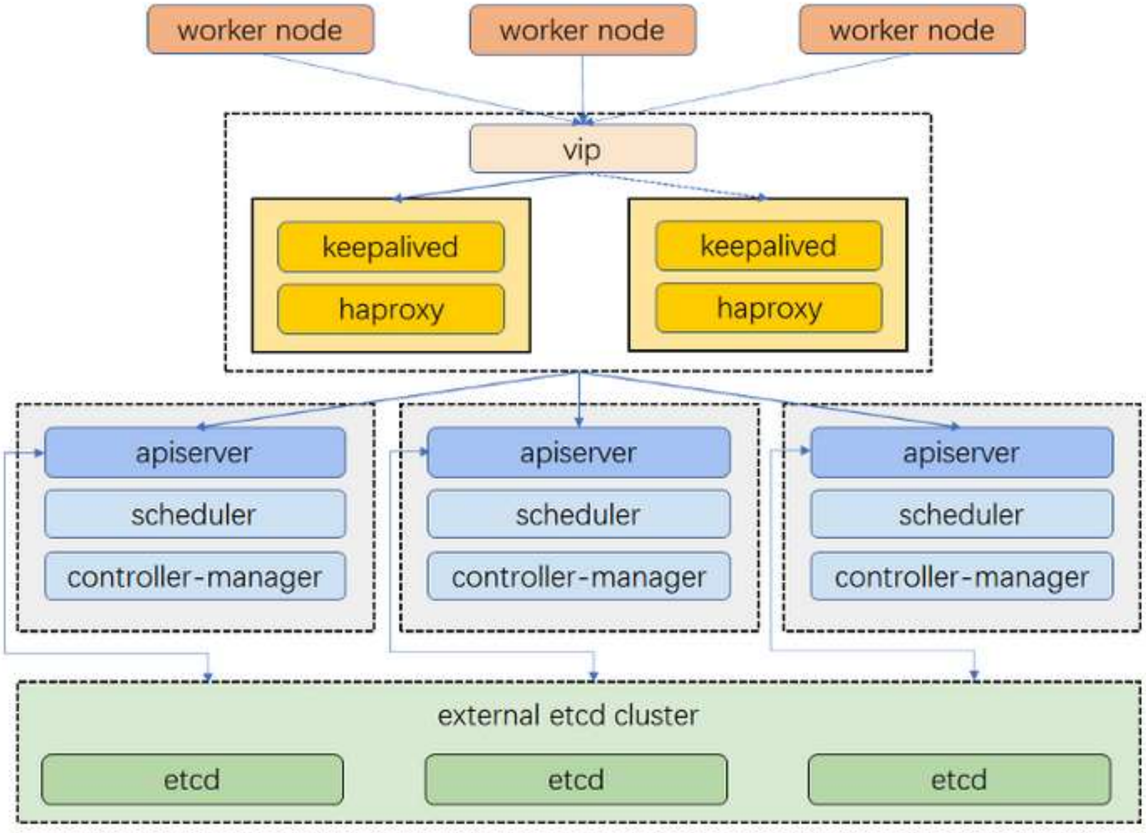
Three-Node Cluster

We will be building a three-node cluster, with one master and two worker nodes.

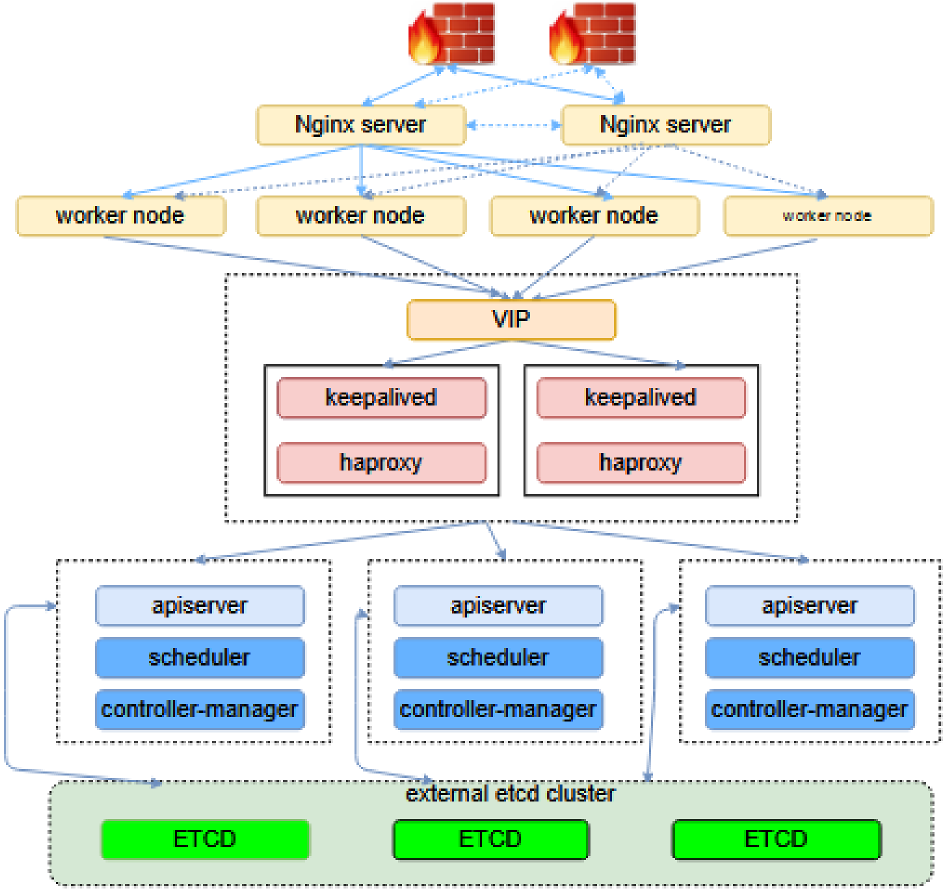
HA cluster


worker node
worker node
worker node
worker node
worker node
load balancer
control plane node
apiserver
controller-manager
scheduler
control plane node
apiserver
controller-manager
scheduler
control plane node
apiserver
controller-manager
scheduler
external etcd cluster
etcd host
etcd host
etcd host

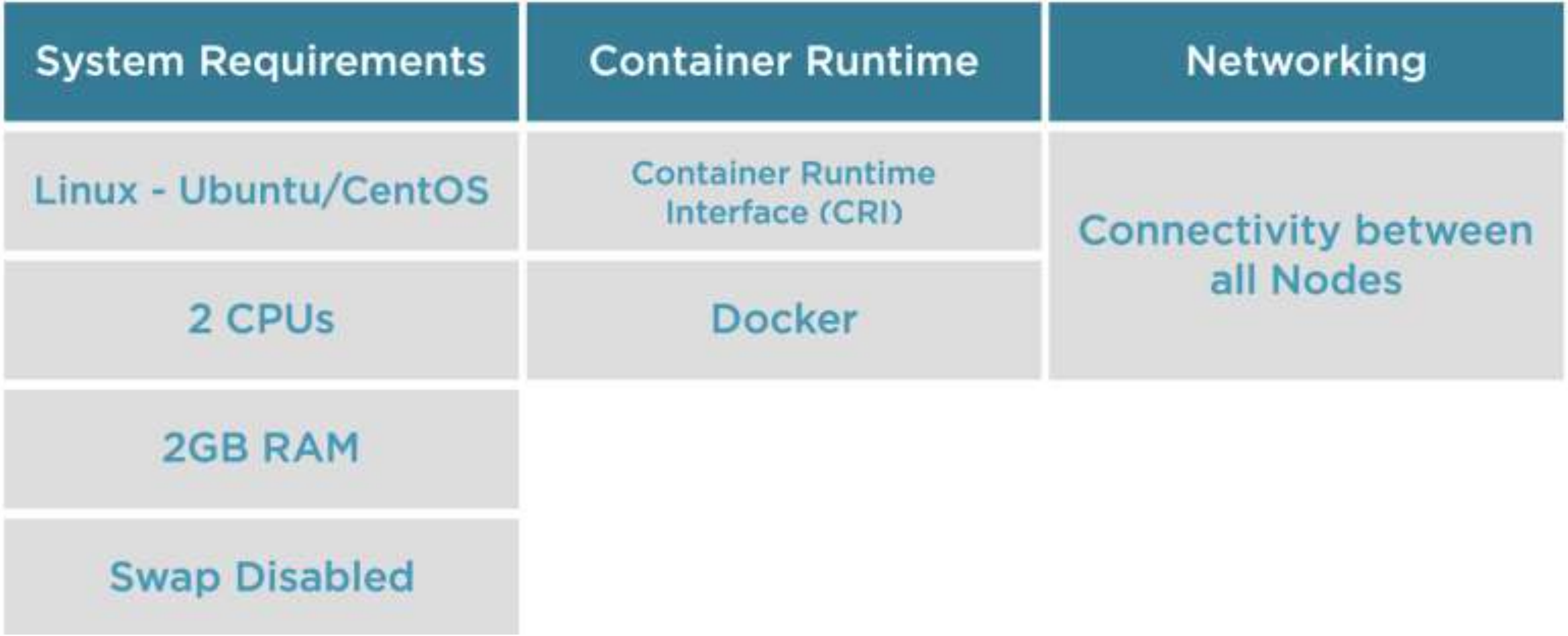
HA cluster


worker node
worker node
worker node
vip
keepalived
haproxy
keepalived
haproxy
apiserver
scheduler
controller-manager
apiserver
scheduler
controller-manager
apiserver
scheduler
controller-manager
external etcd cluster
etcd
etcd
etcd

HA cluster

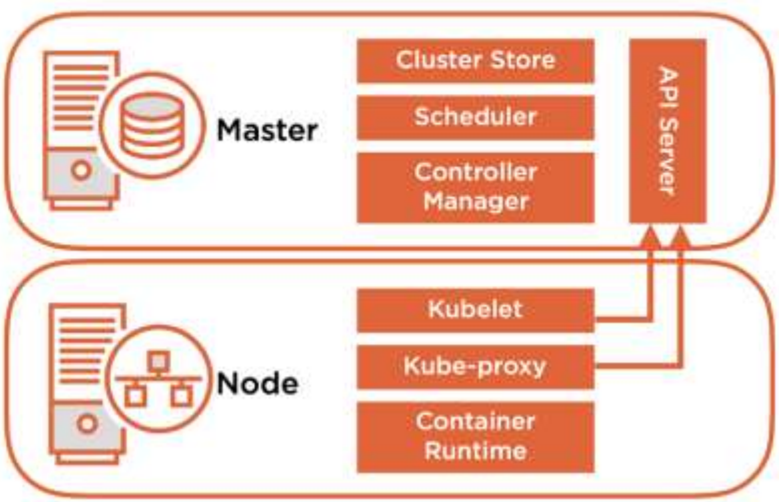

Nginx server
Nginx server
worker node
worker node
worker node
worker node
VIP
keepalived
haproxy
keepalived
haproxy
apiserver
scheduler
controller-manager
apiserver
scheduler
controller-manager
apiserver
scheduler
controller-manager
external etcd cluster
ETCD
ETCD
ETCD

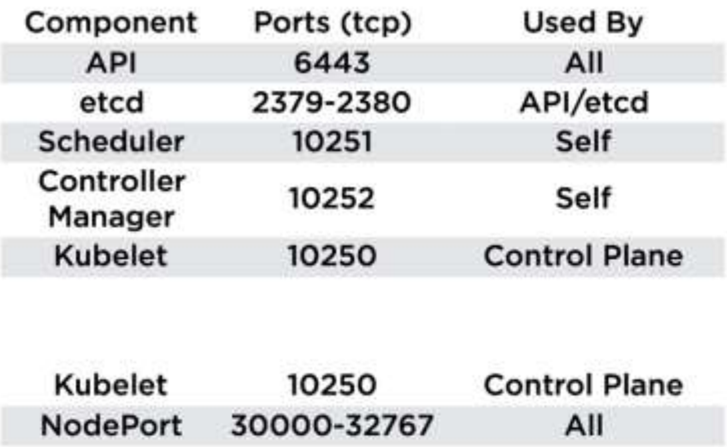
Installation Requirements



System Requirements	Container Runtime	Networking
Linux - Ubuntu/CentOS	Container Runtime Interface (CRI)	Connectivity between all Nodes
2 CPUs	Docker	
2GB RAM		
Swap Disabled		

Cluster Network Port

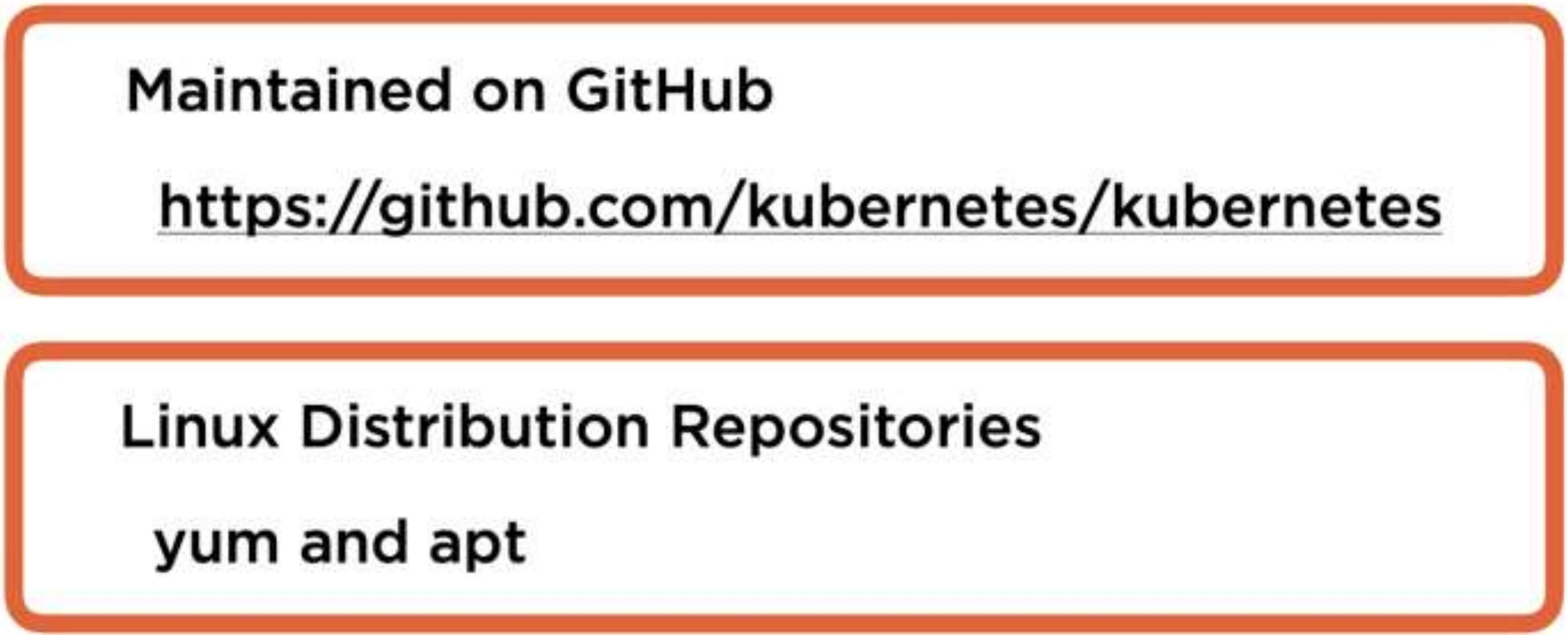




Component	Ports (tcp)	Used By
API	6443	All
etcd	2379-2380	API/etcd
Scheduler	10251	Self
Controller Manager	10252	Self
Kubelet	10250	Control Plane

Kubelet	10250	Control Plane
NodePort	30000-32767	All

Getting K8s



Maintained on GitHub

https://github.com/kubernetes/kubernetes

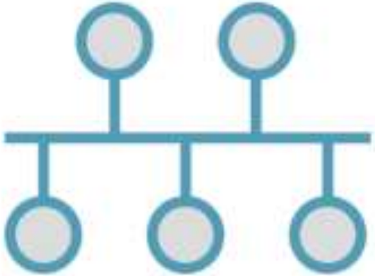
Linux Distribution Repositories

yum and apt

Building your cluster

Install Kubernetes

Create Your Cluster



Configure Pod Networking

Join Nodes to your Cluster

Required Package

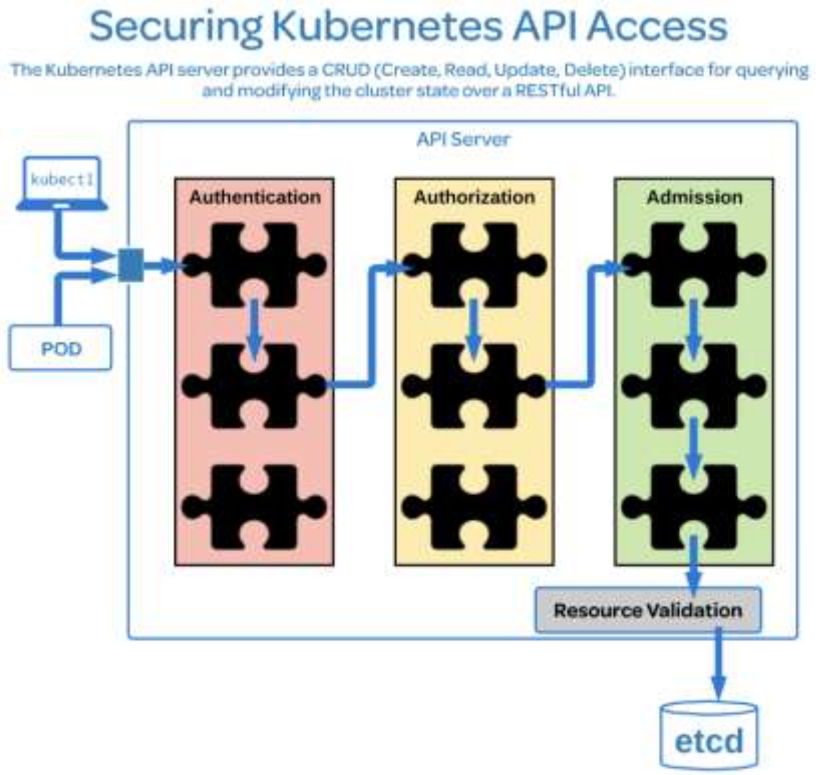
kubelet

kubeadm

kubectl

Container Runtime - Docker

Configuring secure cluster communication



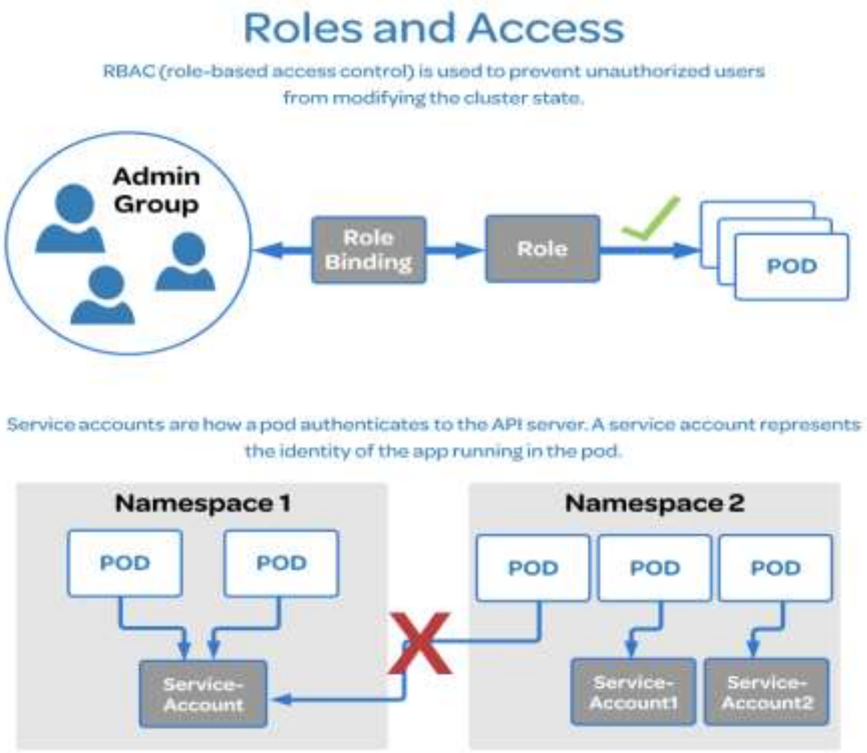
Securing Kubernetes API Access

The Kubernetes API server provides a CRUD (Create, Read, Update, Delete) interface for querying and modifying the cluster state over a RESTful API.

```
# View the kube config
cat .kube/config | more

# View the service account token
kubectl get secrets
```

Configuring secure cluster communication

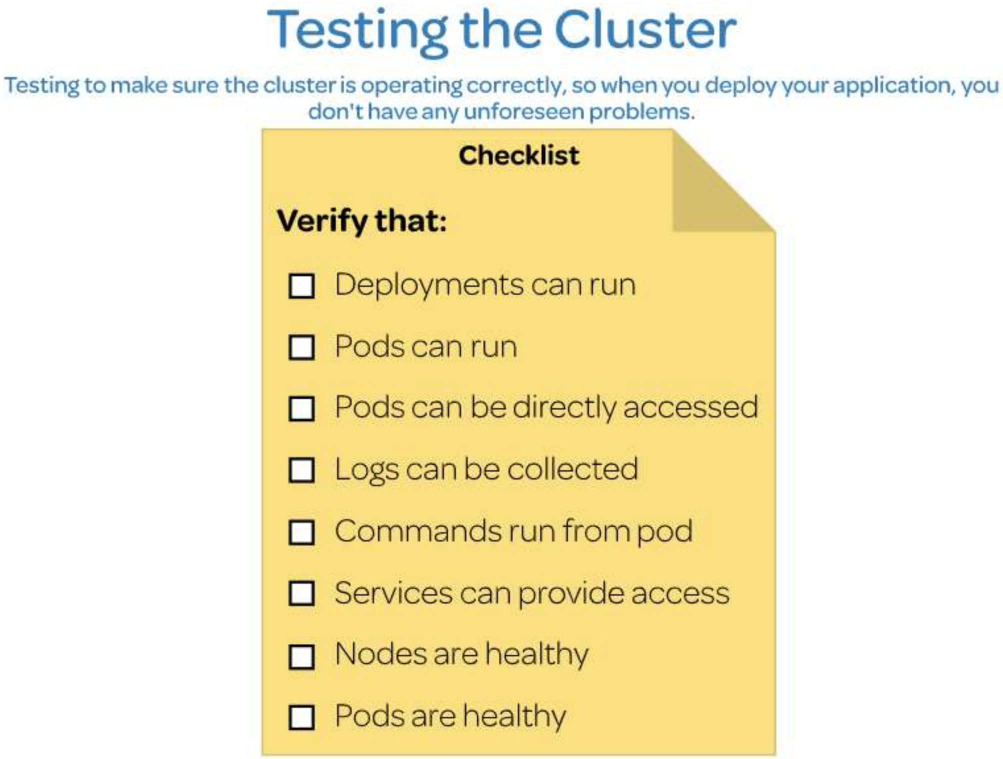


Roles and Access

RBAC (role-based access control) is used to prevent unauthorized users from modifying the cluster state.

Service accounts are how a pod authenticates to the API server. A service account represents the identity of the app running in the pod.

Testing the cluster



Testing the Cluster

Testing to make sure the cluster is operating correctly, so when you deploy your application, you don't have any unforeseen problems.

Checklist

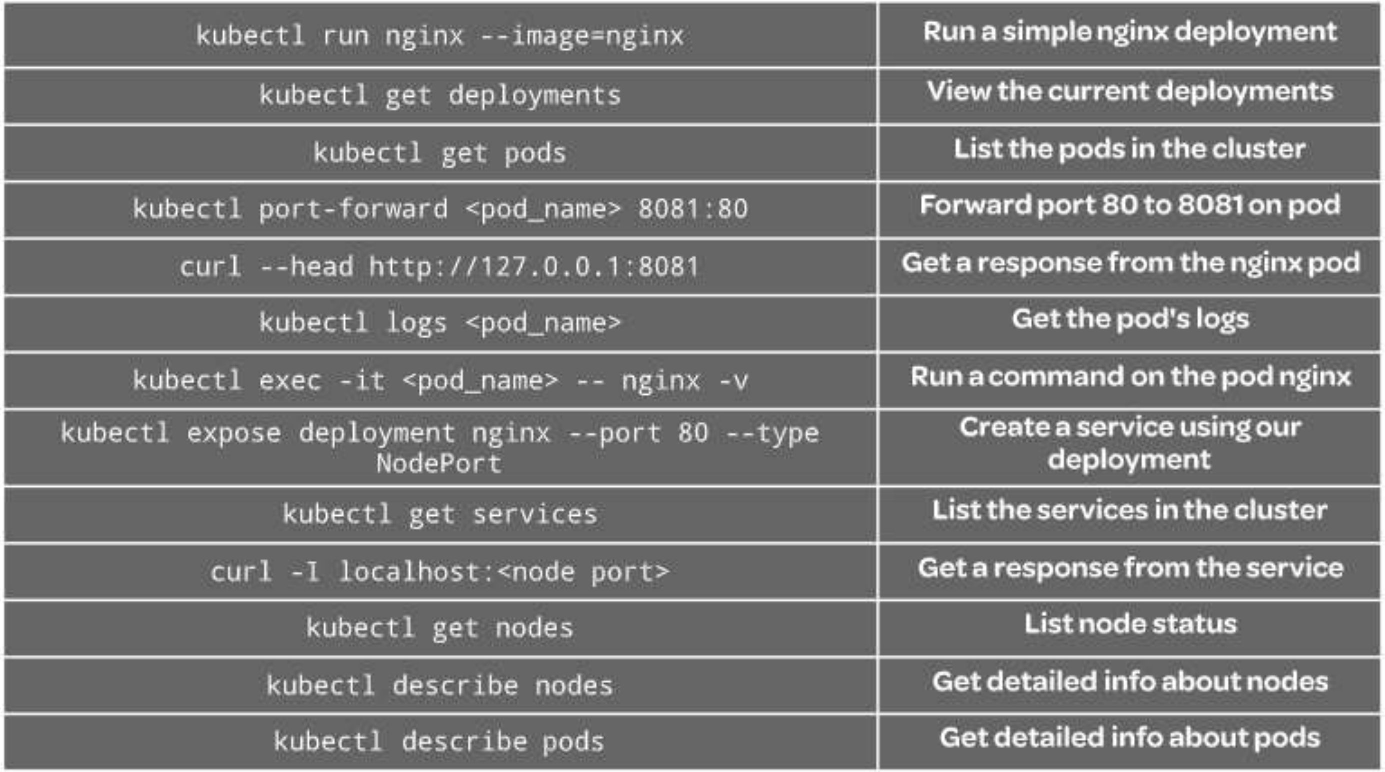
Verify that:

- □ 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

Testing the cluster

- ❑ Conformance tests
  - ▪ Kubetest.
  - ▪ Sonobouy test.
- ❑ End to end test

End to end test



`kubectl run nginx --image=nginx`	Run a simple nginx deployment
`kubectl get deployments`	View the current deployments
`kubectl get pods`	List the pods in the cluster
`kubectl port-forward <pod_name> 8081:80`	Forward port 80 to 8081 on pod
`curl --head http://127.0.0.1:8081`	Get a response from the nginx pod
`kubectl logs <pod_name>`	Get the pod's logs
`kubectl exec -it <pod_name> -- nginx -v`	Run a command on the pod nginx
`kubectl expose deployment nginx --port 80 --type NodePort`	Create a service using our deployment
`kubectl get services`	List the services in the cluster
`curl -I localhost:<node port>`	Get a response from the service
`kubectl get nodes`	List node status
`kubectl describe nodes`	Get detailed info about nodes
`kubectl describe pods`	Get detailed info about pods

Q
A