SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

Unit-4

The Kubernetes

v0.15.1

last update: 2023/01/02

)

- The Practical Kubernetes Training →
- Optional: you need an account on GCP with billing enabled
  - Get started with $300 free credits →
  - Create a project and enable GKE service
  - Install gcloud SDK / CLI: →

Source: https://kubernauts.gitbooks.io/kubernauts-kubernetes-training-courses/content/courses/novice.html

- Other options:
  - Rancher k3d / k3s →
  - Rancher rke →
  - Multipass Rancher →
  - Multipass Kubeadm →
  - Multipass k3s →
  - tk8ctl →
    - TK8 Cattle AWS →
    - TK8 Cattle EKS →

- Checkout the code of Practical Kubernetes Problems

  $ git clone https://github.com/kubernauts/practical-kubernetes-problems.git

- Checkout the code of Kubernetes By Example

  $ git clone https://github.com/openshift-evangelists/kbe

- Visit the Kubernetes By Example Site

  https://kubebyexample.com/

- Checkout the code of Kubernetes By Action

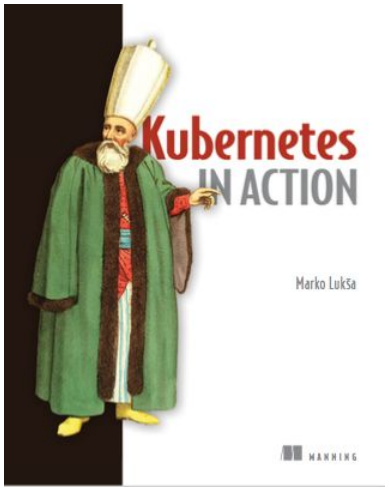
  $ git clone https://github.com/luksa/kubernetes-in-action.git

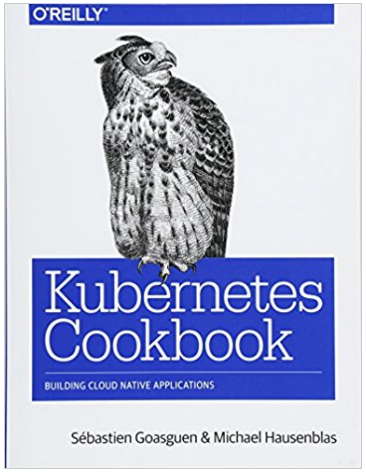
- Checkout the code of K8s intro tutorials

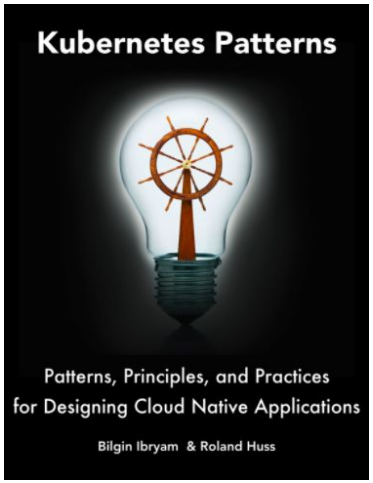
  $ git clone https://github.com/mrbobbytables/k8s-intro-tutorials

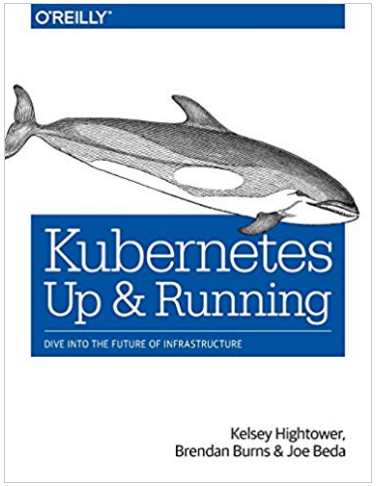
Source: https://kubernauts.gitbooks.io/kubernauts-kubernetes-training-courses/content/courses/novice.html

- Again: almost everything you need to know about Kubernetes & more:
  - https://goo.gl/Rywkpd
- Recommended Books and references:



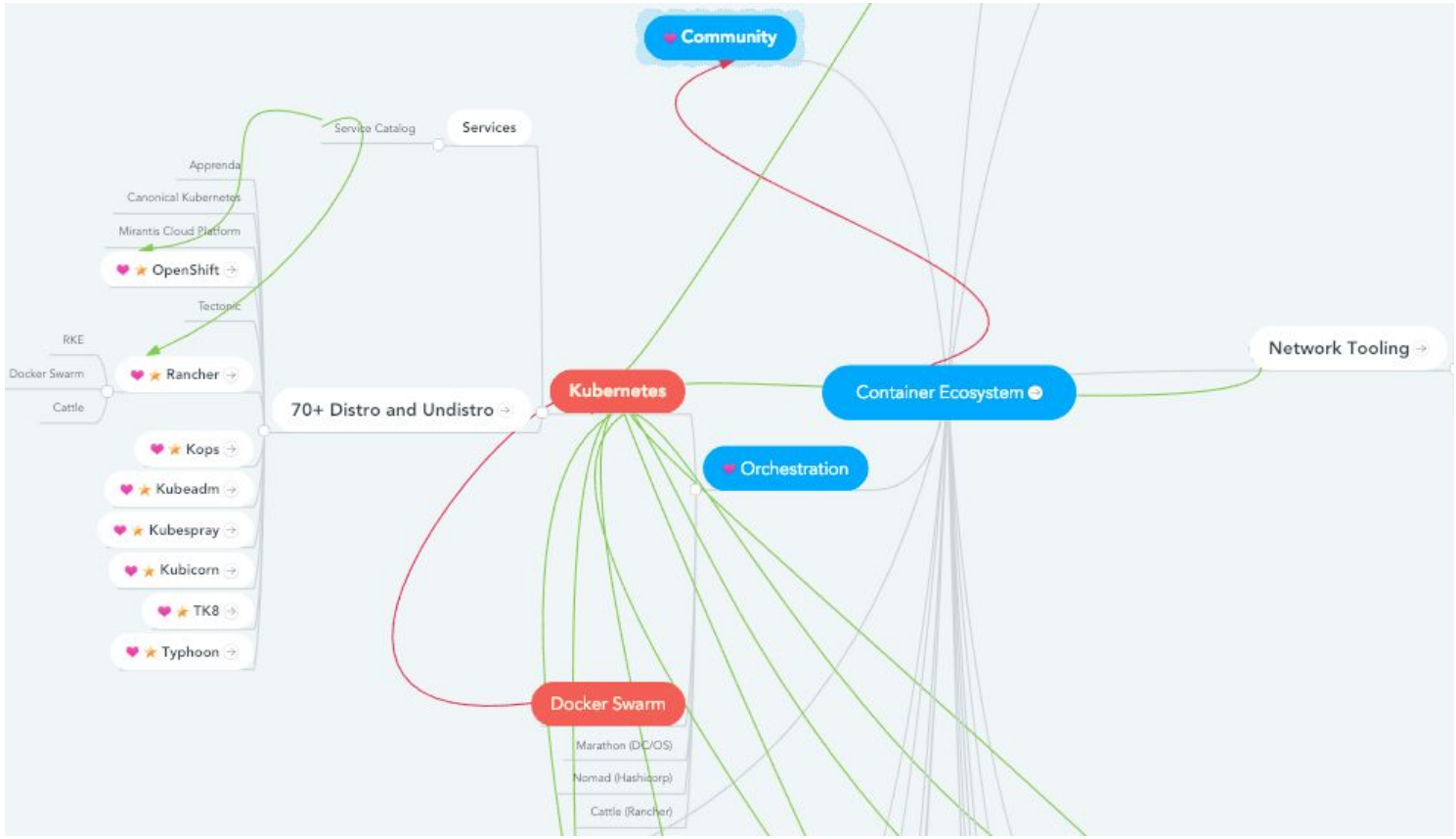






- What is Kubernetes (“k8s” or “kube”)
- Kubernetes Architecture
- Core Concepts of Kubernetes
- Kubernetes resources explained
- Application Optimization on Kubernetes
- Kubernetes effect on the software development life cycle
- Local and Distributed Abstractions and Primitives
- Container Design Patterns and best practices
- Deployment and release strategy with Kubernetes

- Kubernetes v1.8: A Comprehensive Overview →
- Getting started with Kubernetes
  - Deploying and Updating App with Kubernetes
  - Deploy more complex apps and data platforms on k8s



https://www.mindmeister.com/929803117/container-ecosystem?fullscreen=1

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

Agenda

- Agenda
  - What is Kubernetes
  - Deployment and release strategy (in short)
  - Getting started (general)
  - Security
  - Exercises
  - more Exercises

- Agenda
  - HA Installation and Multi-Cluster Management
  - Tips & Tricks, Practice Questions
  - Advanced Exercises
    - Load Testing on K8s with Apache Jmeter
    - Kafka on K8s with Strimzi and Confluent OS
    - TK8 Cattle AWS vs. Cattle EKS
    - TK8 Special with TK8 Web
  - TroubleShooting & Questions

What is Kubernetes?

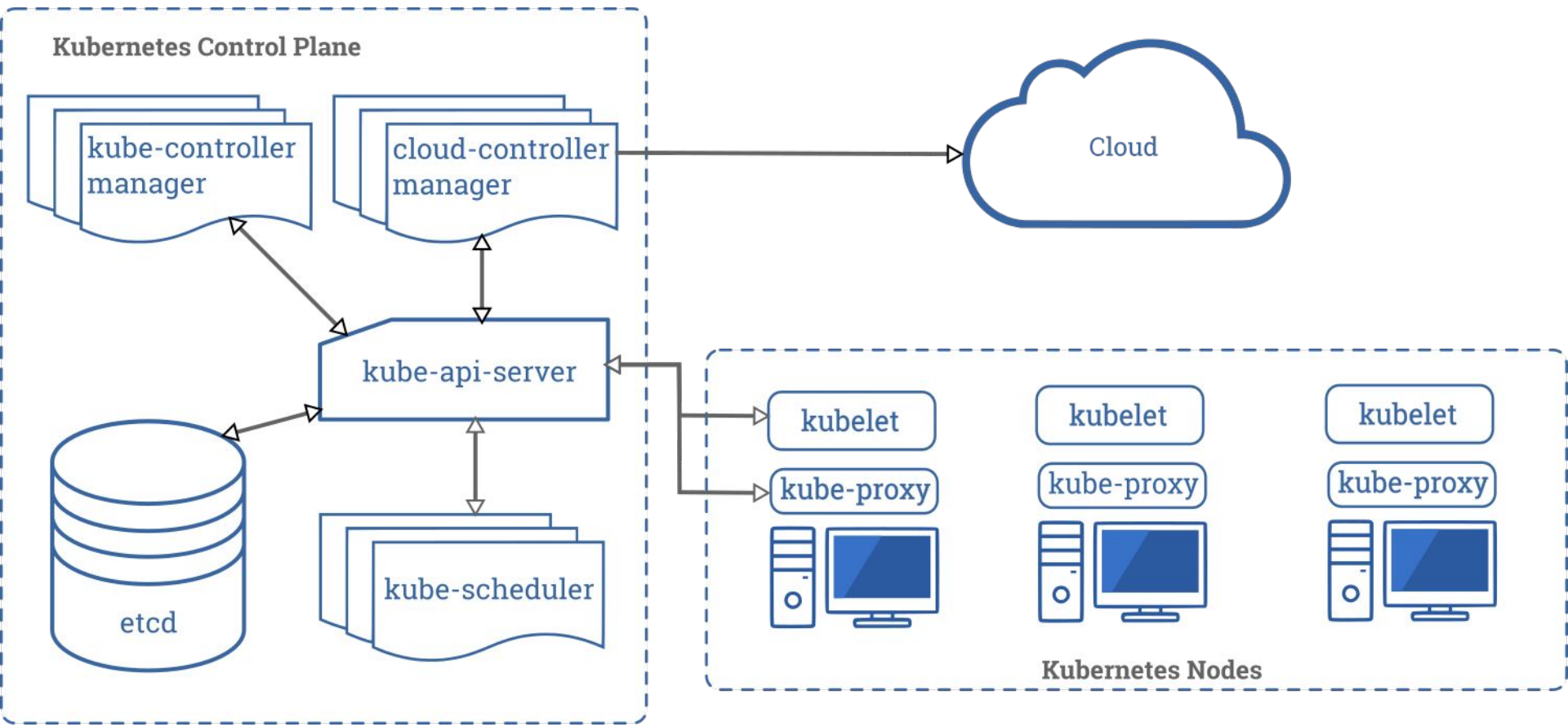
- Kubernetes is Greek for "helmsman", your guide through unknown waters, nice but not true :-)
- Kubernetes is the linux kernel of distributed systems
- Kubernetes is the linux of the cloud!
- Kubernetes is a platform and container orchestration tool for automating deployment, scaling, and operations of application containers.
- Kubernetes supports, Containerd, CRI-O, Kata containers (formerly clear and hyper) and Virtlet

- What is a Container Engine?
- Where are the differences between Docker, CRI-O or Containerd runtimes?
- How does Kubernetes work with container runtimes?
- Which is the best solution?
  - Linux Container Internals by Scott McCarty → →
  - Container Runtimes and Kubernetes by Fahed Dorgaa →
  - Kubernetes Runtime Comparison →

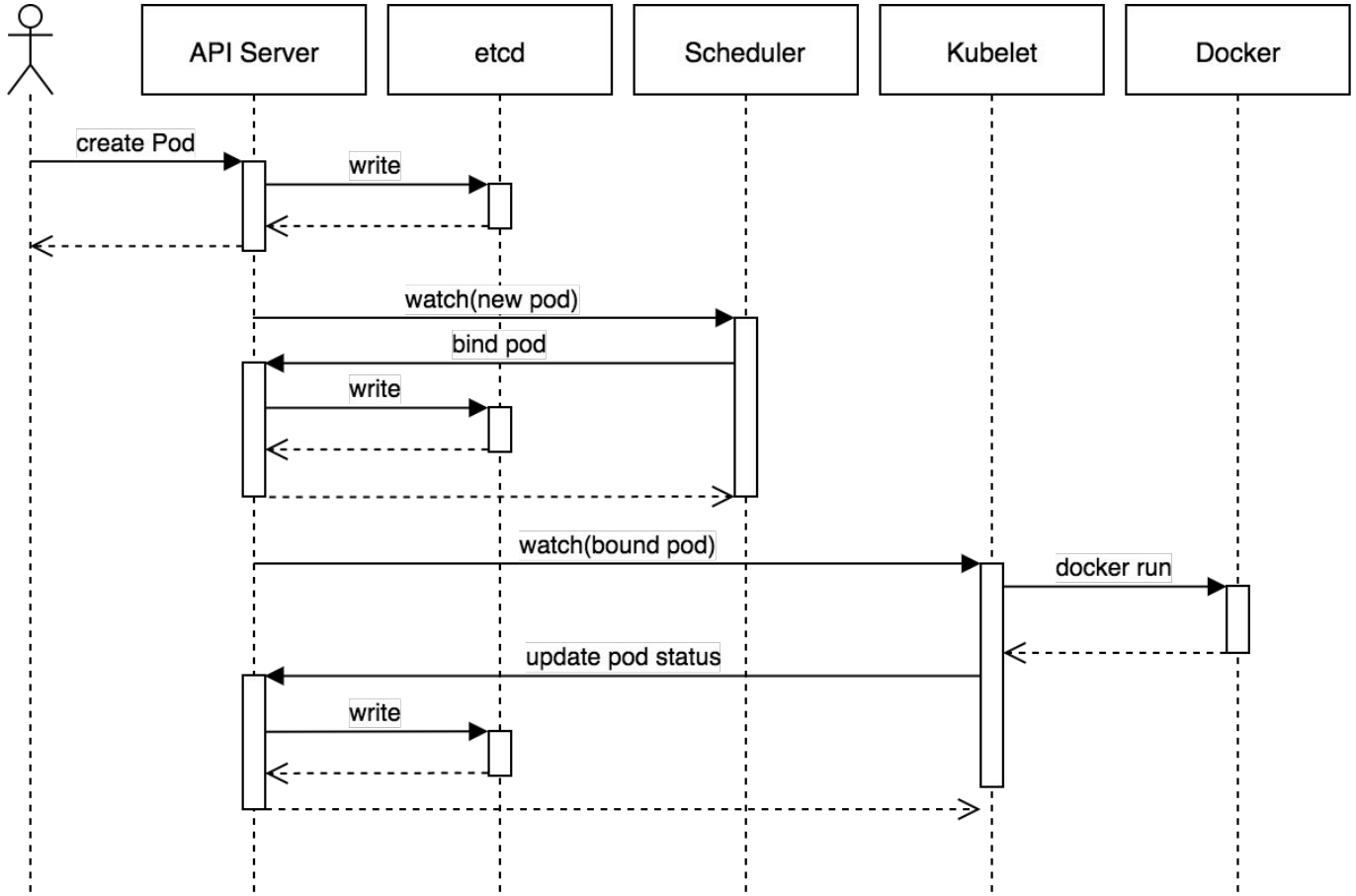
How Kubernetes works?

In Kubernetes, there is a master node and multiple worker nodes, each worker node can handle multiple pods. Pods are just a bunch of containers clustered together as a working unit. You can start designing your applications using pods. Once your pods are ready, you can specify pod definitions to the master node, and how many you want to deploy. From this point, Kubernetes is in control. It takes the pods and deploys them to the worker nods.

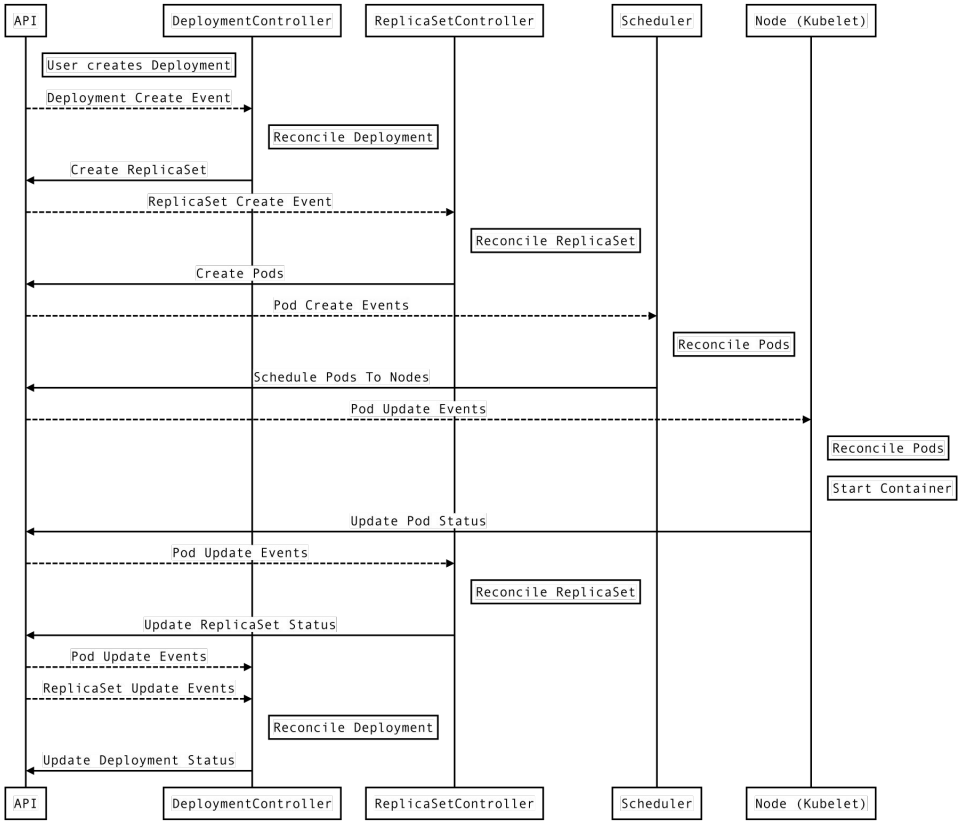
Source: https://itnext.io/successful-short-kubernetes-stories-for-devops-architects-677f8bfed803



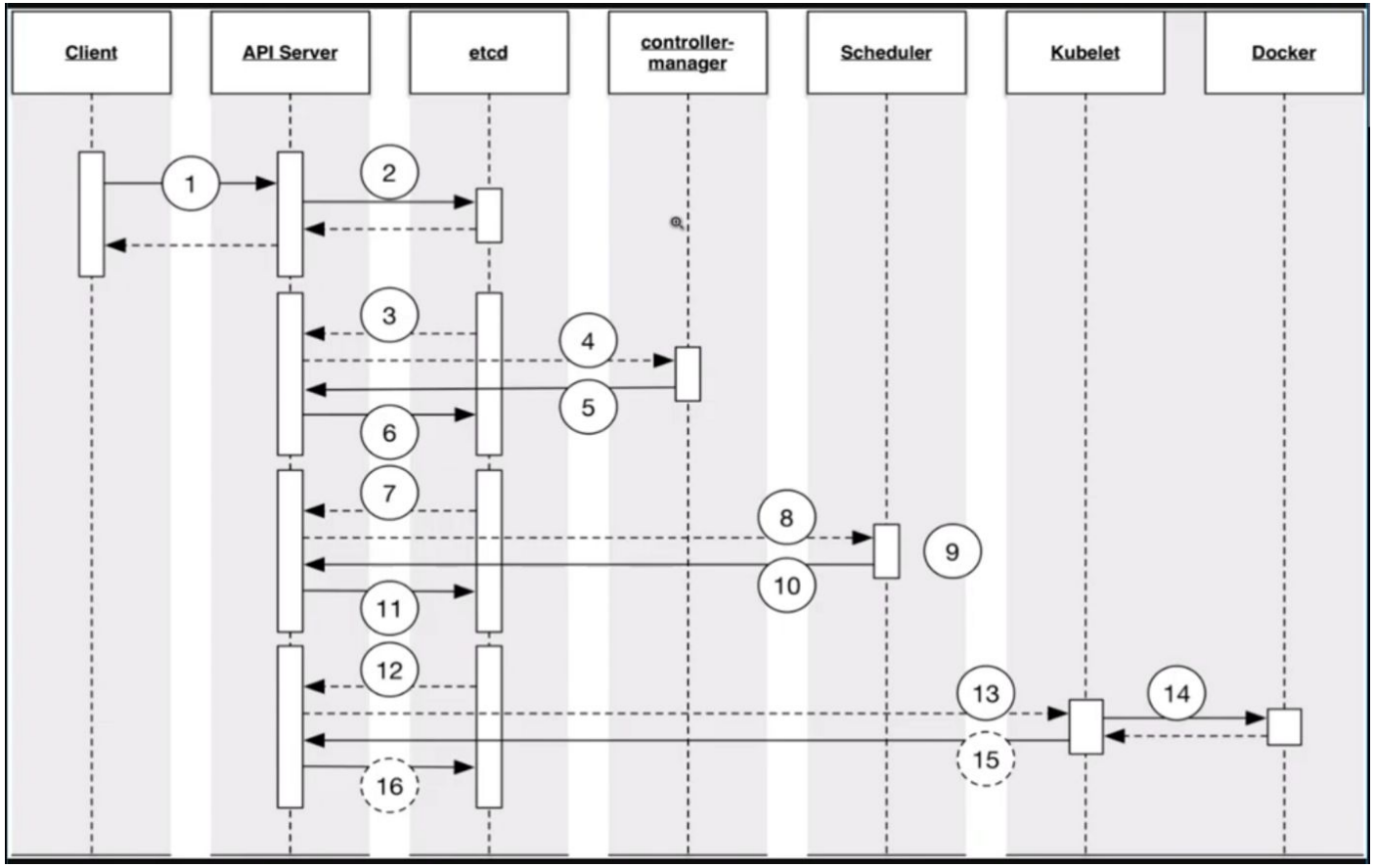
Source: https://kubernetes.io/docs/concepts/overview/components/#master-components



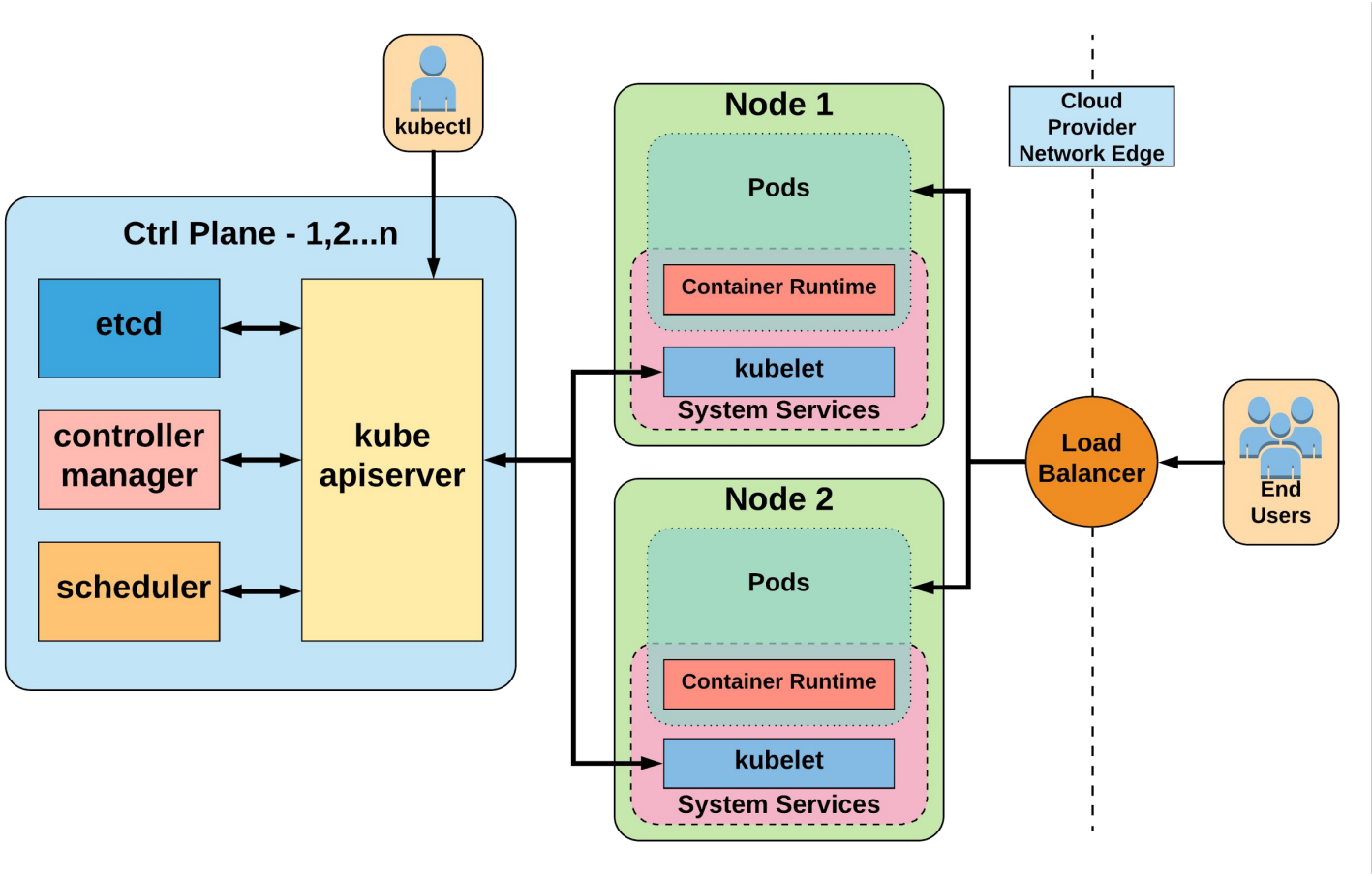
Source: https://blog.heptio.com/core-kubernetes-jazz-improv-over-orchestration-a7903ea92ca



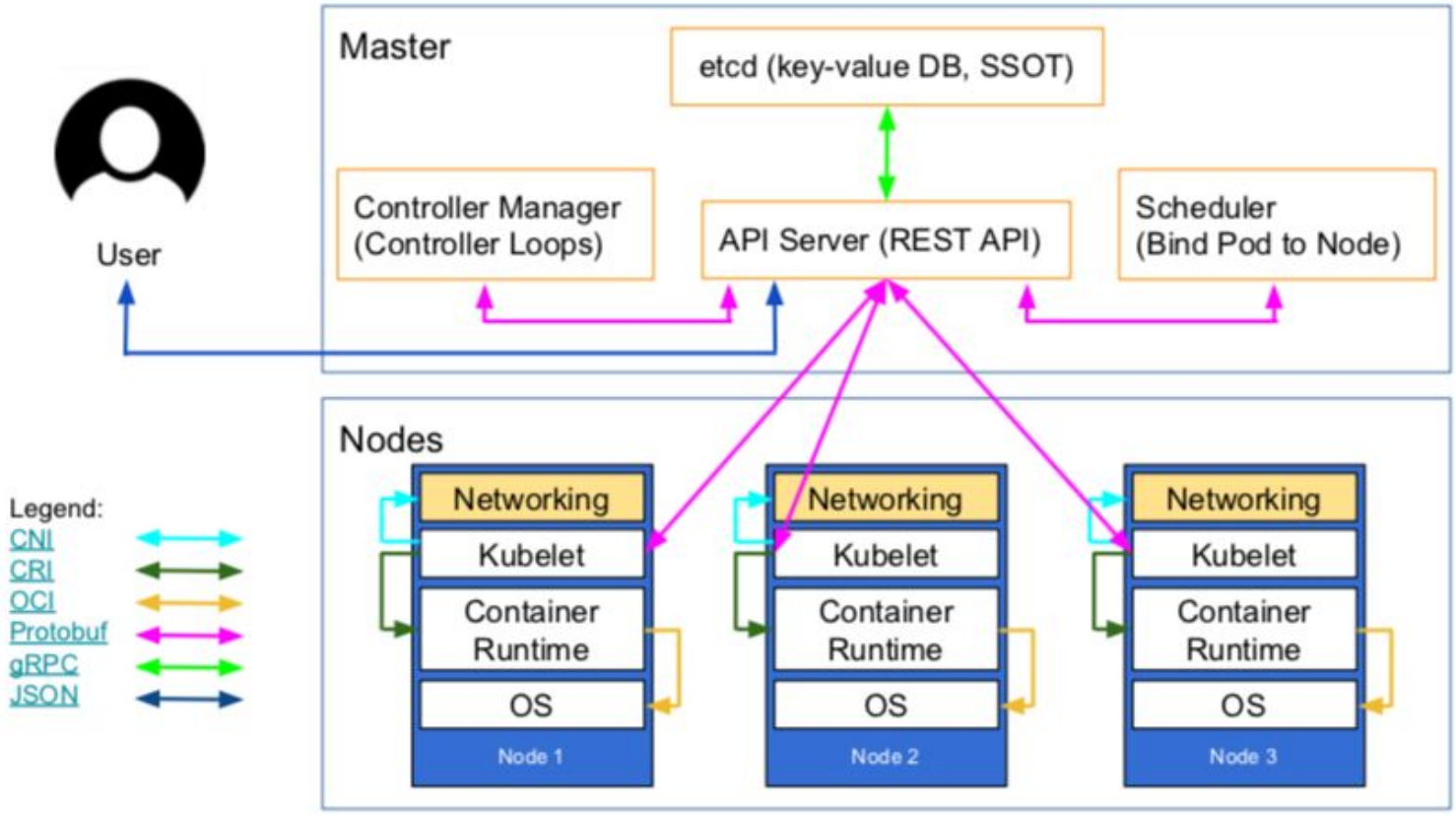
Source: https://medium.com/payscale-tech/imperative-vs-declarative-a-kubernetes-tutorial-4be66c5d8914



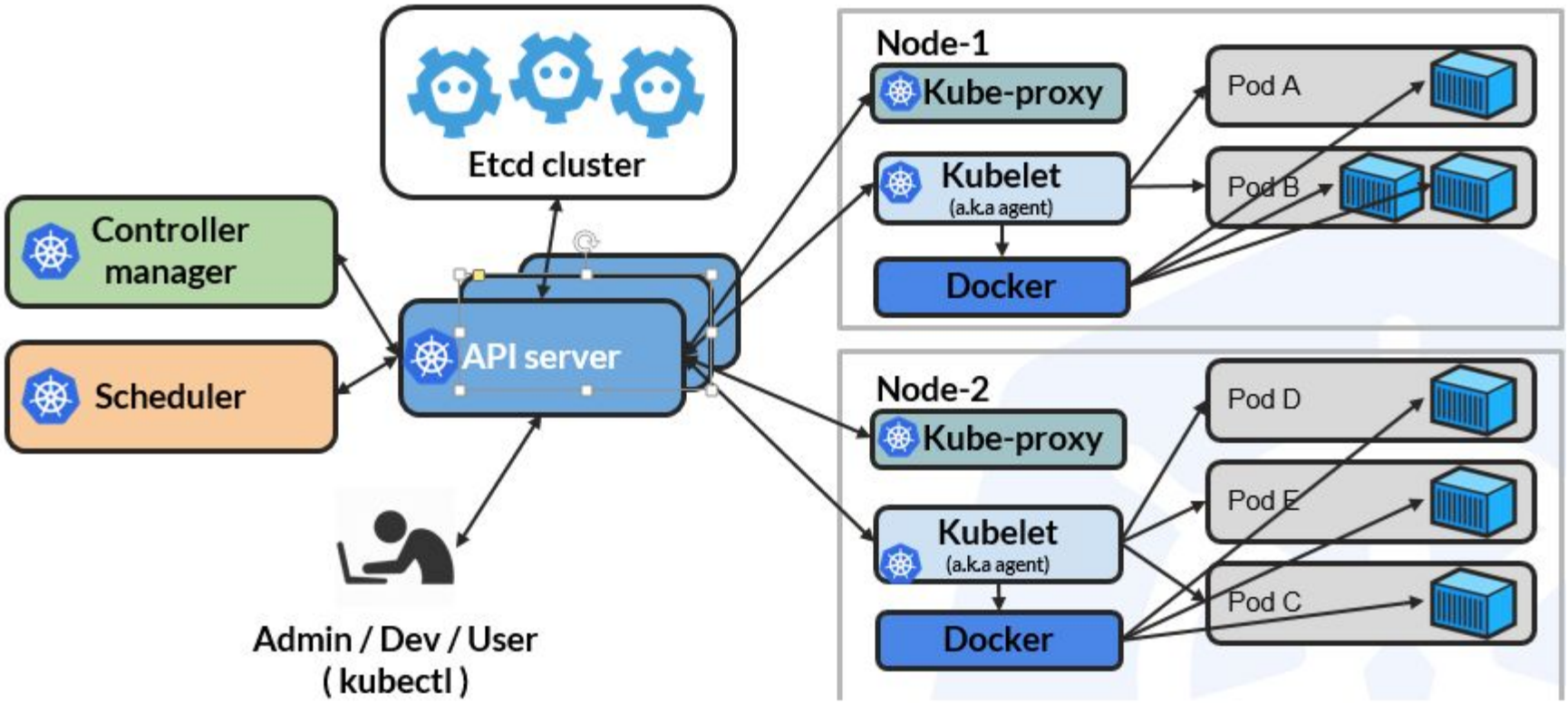
Source: https://medium.com/cloud-heroes/exploring-the-flexibility-of-kubernetes-9f65db2360a0



Source: Introduction to Kubernetes

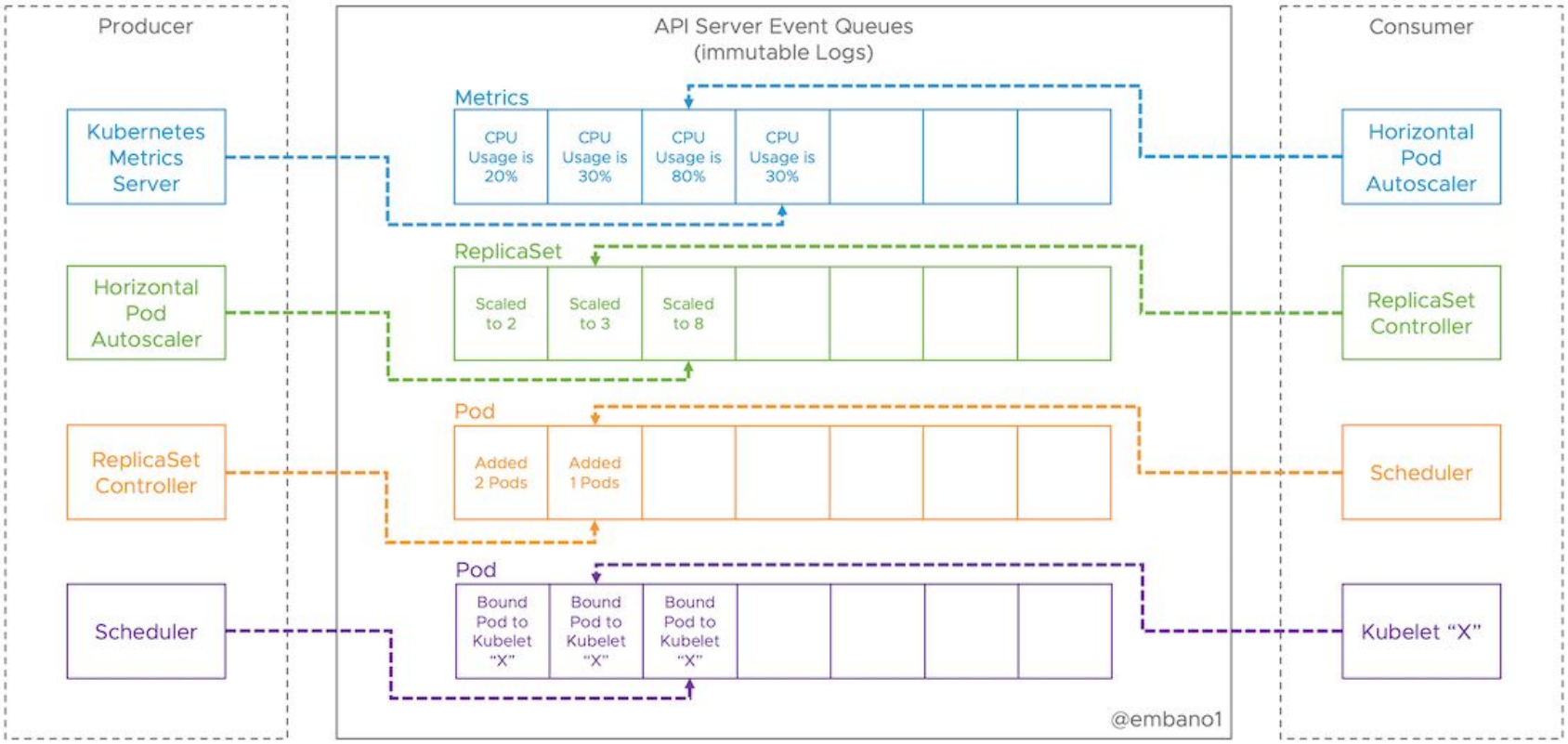


Source: https://www.weave.works/blog/what-does-production-ready-really-mean-for-a-kubernetes-cluster

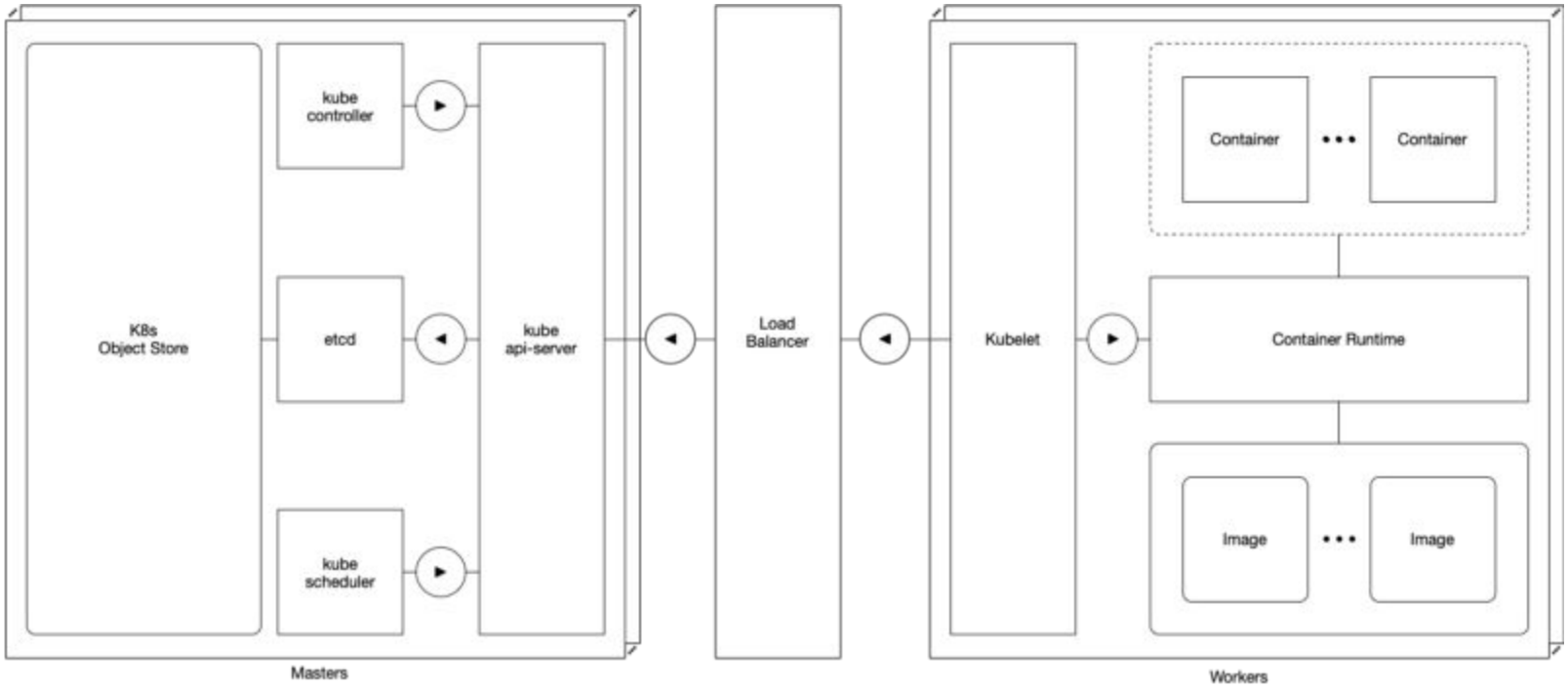

Etcd cluster
Controller manager
Scheduler
API server
Admin / Dev / User
(kubectl)
Node-1
Kube-proxy
Kubelet
(a.k.a agent)
Docker
Pod A
Pod B
Node-2
Kube-proxy
Kubelet
(a.k.a agent)
Docker
Pod D
Pod E
Pod C

Source: https://www.aquasec.com/wiki/display/containers/Kubernetes+Architecture+101

Kubernetes: “Autonomous processes reacting to events from the API server”.



Source: [Events, the DNA of Kubernetes]



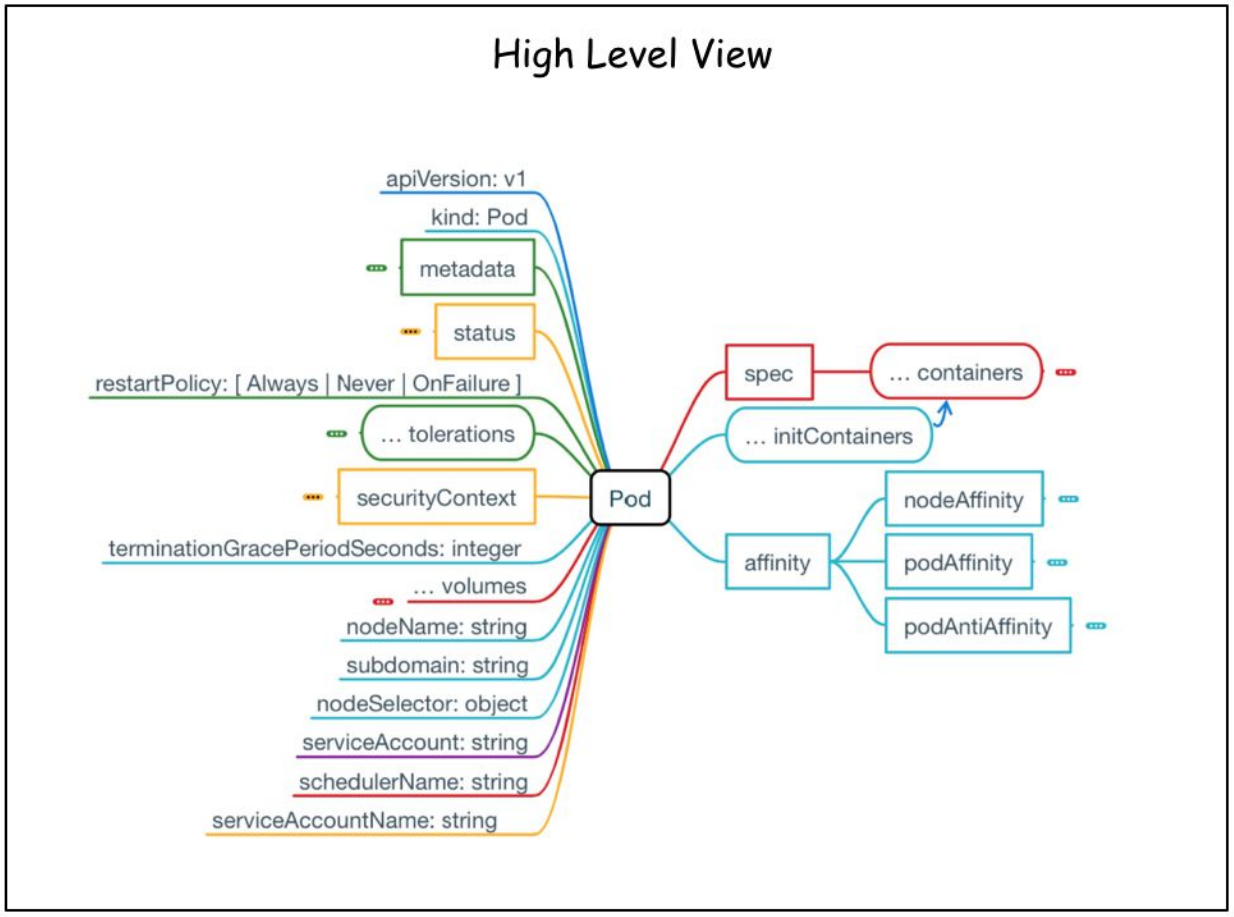
Don’t miss: https://medium.com/@dominik.tornow/kubernetes-high-availability-d2c9cbbdd864

- Pod →
- Label and selectors →
- Controllers
  - Deployments →
  - ReplicaSet →
  - ReplicationController →
  - DaemonSet →
- Service →

- StatefulSets →
- ConfigMaps →
- Secrets →
- Persistent Volumes (attaching storage to containers) →
- Life Cycle of Applications in Kubernetes →
  - Updating Pods
  - Rolling updates
  - Rollback

	Resource (abbr.) [API version]	Description
	Namespace* (ns) [v1]	Enables organizing resources into non-overlapping groups (for example, per tenant)
Deploying Workloads	Pod (po) [v1]	The basic deployable unit containing one or more processes in co-located containers
	ReplicaSet	Keeps one or more pod replicas running
	ReplicationController	The older, less-powerful equivalent of a ReplicaSet
	Job	Runs pods that perform a completable task
	CronJob	Runs a scheduled job once or periodically
	DaemonSet	Runs one pod replica per node (on all nodes or only on those matching a node selector)
	StatefulSet	

Source: Kubernetes in Action book by Marko Lukša



Source: [The Pod Cheat Sheet by the awesome Jimmy Song]

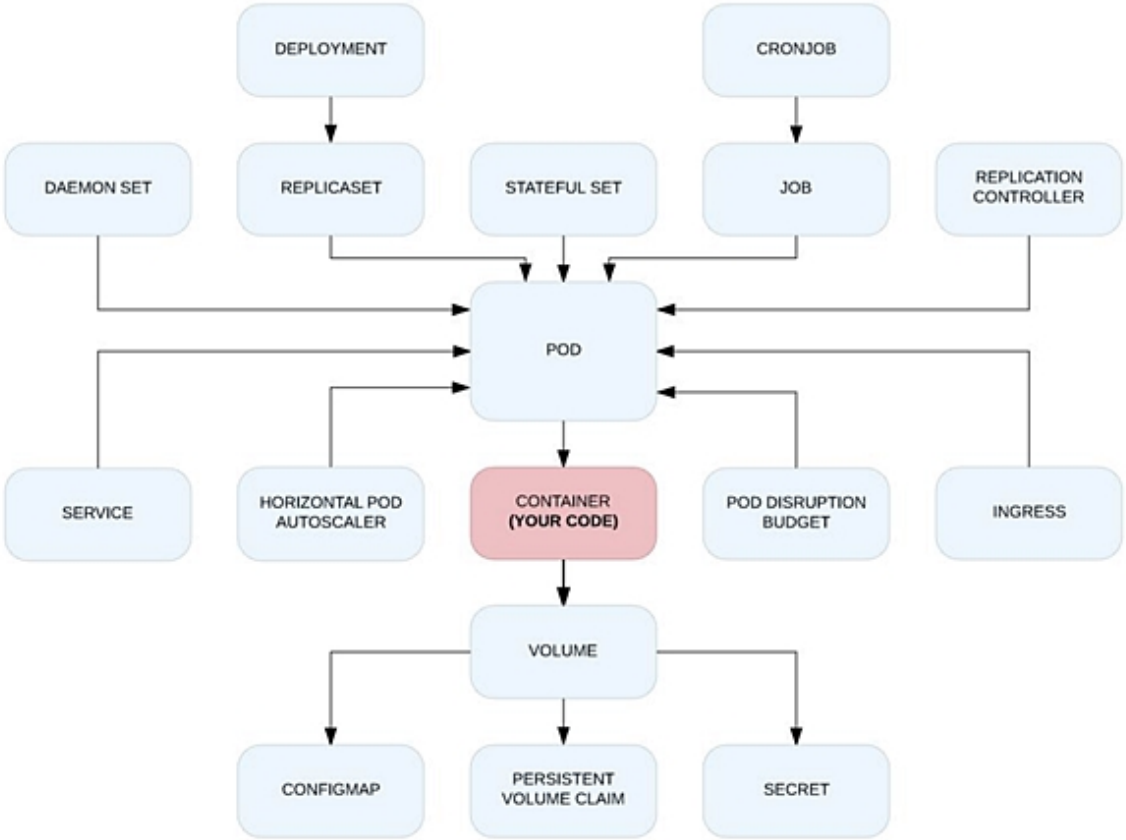
	Resource (abbr.) [API version]	Description
Services	**Service** (svc) [v1]	Exposes one or more pods at a single and stable IP address and port pair
	Endpoints (ep) [v1]	Defines which pods (or other servers) are exposed through a service
	Ingress (ing) [extensions/v1beta1]	Exposes one or more services to external clients through a single externally reachable IP address
Config	ConfigMap (cm) [v1]	A key-value map for storing non-sensitive config options for apps and exposing it to them
	Secret [v1]	Like a ConfigMap, but for sensitive data
Storage	PersistentVolume* (pv) [v1]	Points to persistent storage that can be mounted into a pod through a PersistentVolumeClaim

Source: Kubernetes in Action book by Marko Luksa

	Resource (abbr.) [API version]	Description
Scaling	HorizontalPodAutoscaler (hpa) [autoscaling/v2beta1**]	Automatically scales number of pod replicas based on CPU usage or another metric
	PodDisruptionBudget (pdb) [policy/v1beta1]	Defines the minimum number of pods that must remain running when evacuating nodes
Resources	LimitRange (limits) [v1]	Defines the min, max, default limits, and default requests for pods in a namespace
	ResourceQuota (quota) [v1]	Defines the amount of computational resources available to pods in the namespace
Cluster state	Node* (no) [v1]	Represents a Kubernetes worker node
	Cluster* [federation/v1beta1]	A Kubernetes cluster (used in cluster federation)
	ComponentStatus* (cs) [v1]	Status of a Control Plane component
	Event (ev) [v1]	A report of something that occurred in the cluster

Source: Kubernetes in Action book by Marko Lukša

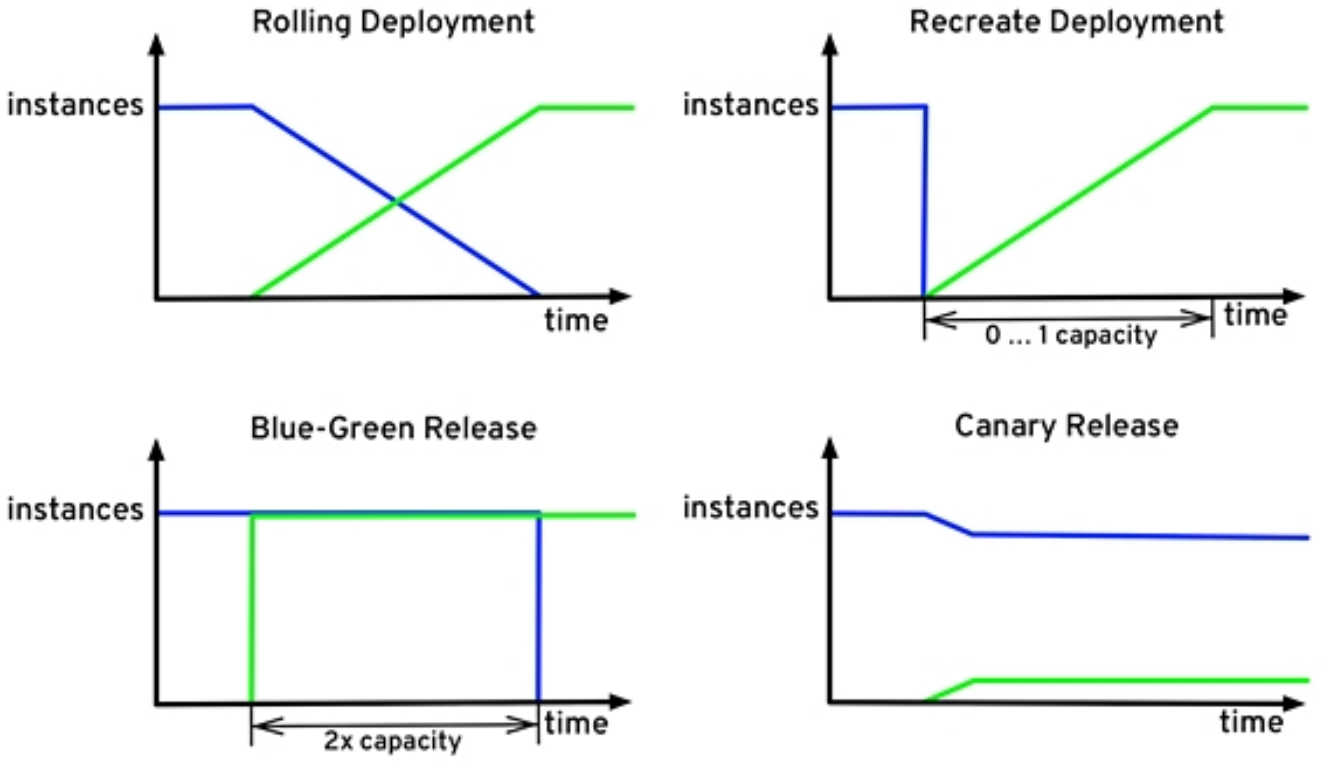
	Resource (abbr.) [API version]	Description
Security	ServiceAccount (sa) [v1]	An account used by apps running in pods
	Role [rbac.authorization.k8s.io/v1]	Defines which actions a subject may perform on which resources (per namespace)
	ClusterRole* [rbac.authorization.k8s.io/v1]	Like Role, but for cluster-level resources or to grant access to resources across all namespaces
	RoleBinding [rbac.authorization.k8s.io/v1]	Defines who can perform the actions defined in a Role or ClusterRole (within a namespace)
	ClusterRoleBinding* [rbac.authorization.k8s.io/v1]	Like RoleBinding, but across all namespaces
	PodSecurityPolicy* (psp) [extensions/v1beta1]	A cluster-level resource that defines which security- sensitive features pods can use



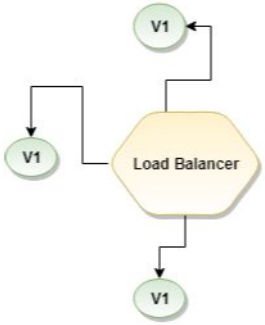
Source: Kubernetes effect by Bilgin Ibryam

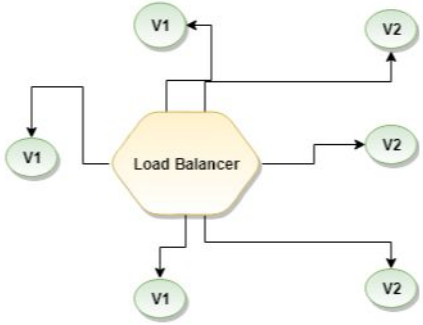
SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

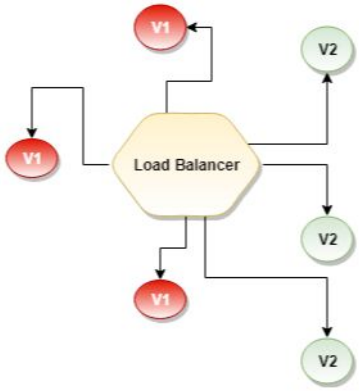
Deployment and Release Strategy



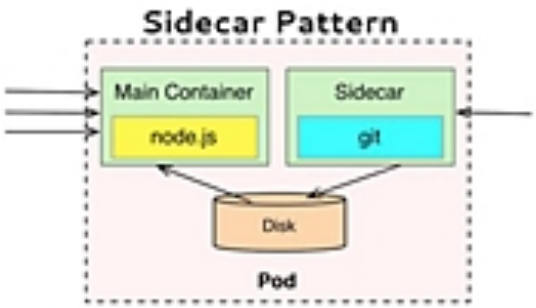
Source: Kubernetes effect by Bilgin Ibryam

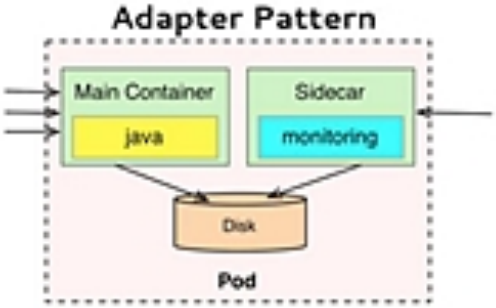


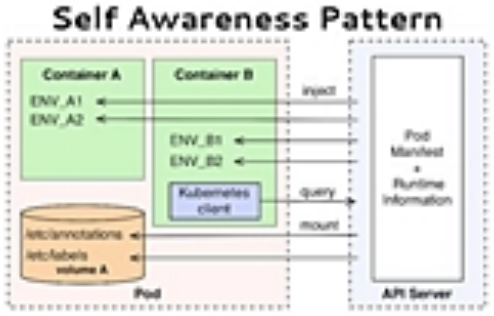


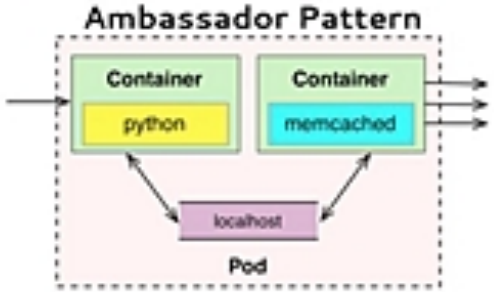


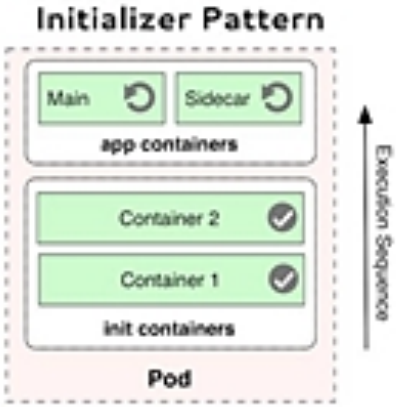
Source: [Kubernetes Deployment Strategy Types]

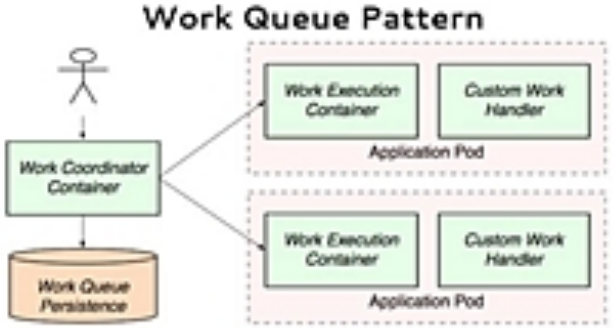












Source: [Kubernetes effect by Bilgin Ibryam]

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

Getting Started

- Kubernetes.IO documentation →
- Kubernetes Bootcamp →
- Install Kubernetes CLI kubectl →
- Create a local cluster with
  - Docker For Desktop →
  - Minikube →
  - MiniShift →
  - DinD → or Kind →

- Follow this Minikube tutorial by the awesome Abhishek Tiwari
  - https://abhishek-tiwari.com/local-development-environment-for-kubernetes-using-minikube/

- Create a Kubernetes cluster on AWS
  - Kubeadm →
  - TK8 & TK8EKS →

- On macOS: brew install kubectl
- On linux and windows follow the official documentation: https://kubernetes.io/docs/tasks/tools/install-kubectl/

- “kubectl version” gives the client and server version
- “which kubectl”
- alias k=’kubectl’
- Enable shell autocompletion (e.g. on linux):
  - echo "source <(kubectl completion bash)" >> ~/.bashrc

- Great kubectl helpers by Ahmet Alp Balkan
  - kubectx and kubens →
- Kubernetes prompt for bash and zsh
  - kube-ps1 →
- Kubed-sh (kube-dash) →
- Kubelogs →
- kns and ktx →
- K9s →
- **The golden Kubernetes Tooling and helpers list →**

- alias k="kubectl"
- alias g="gcloud"
- alias kx="kubectx"
- alias kn="kubens"
- alias kon="kubeon"
- alias koff="kubeoff"
- alias kcvm="kubectl config view --minify"
- alias kgn="kubectl get nodes"
- alias kgp="kubectl get pods"

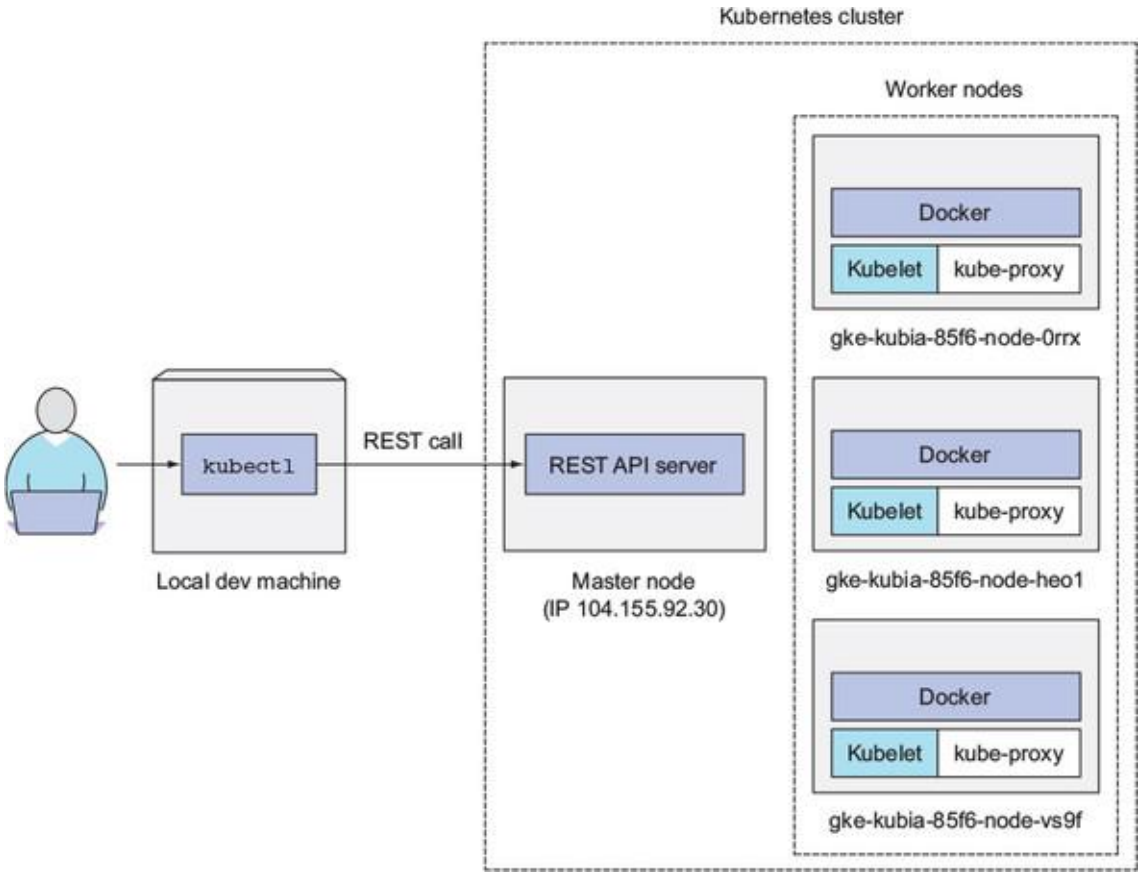
- Switch to another namespace on the current context (cluster):
  - kubectl config set-context <cluster-name> --namespace=efk
- Switch to another cluster
  - kubectl config use-context <cluster-name>
- Merge kube configs
  - cp ~/.kube/config ~/.kube/config.bak
  - KUBECONFIG=./kubeconfig.yaml:~/.kube/config.bak kubectl config view --flatten > ~/.kube/config
- Again: use kubectx and kubens, it makes the life easier :-)
- A great Cheat Sheet by Denny Zhang →
- Kubectl: most useful commands by Matthew Davis →

- You need an account on GCP with billing enabled
- Create a project and enable GKE service
- Install gcloud SDK / CLI:
  - https://cloud.google.com/sdk/

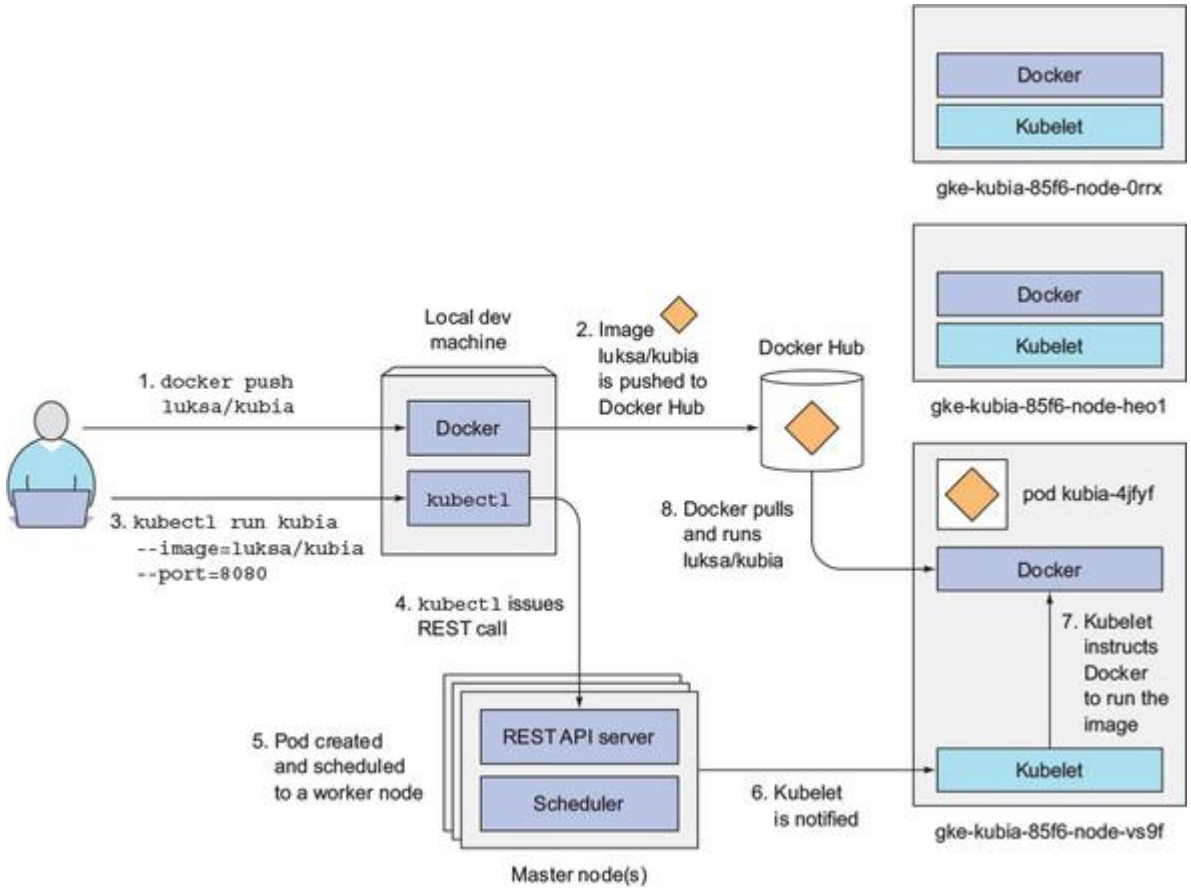
Source:

- gcloud projects create kubernauts-trainings
- gcloud config set project kubernauts-trainings
- gcloud container clusters create my-training-cluster --zone=us-central1-a
  - Note: message=The Kubernetes Engine API is not enabled for project training-220218. Please ensure …
- Kubectl get nodes

```
NAME                                          STATUS    ROLES     AGE       VERSION
gke-my-gke-k8s-cluster-default-pool-3a43c197-9kb6   Ready     <none>    1m        v1.8.7-gke.1
gke-my-gke-k8s-cluster-default-pool-3a43c197-g8hg   Ready     <none>    1m        v1.8.7-gke.1
gke-my-gke-k8s-cluster-default-pool-3a43c197-xjwx   Ready     <none>    1m        v1.8.7-gke.1
```



Source: Kubernetes in Action book by Marko Lukša



Source: [Kubernetes in Action book by Marko Lukša]

- List your clusters
  - gcloud container clusters list
- Set a default Compute Engine zone
  - gcloud config set compute/zone us-central1-a
- Define a standard project with your ProjectID
  - gcloud config set project kubernauts-trainings
- Access the Kubernetes dashboard
  - kubectl proxy →

Source:

- Login to one of the nodes
  - gcloud compute ssh <node-name>
- Get more information about a node
  - kubectl describe node <node name>
- Delete / clean up your training cluster
  - gcloud container clusters delete my-training-cluster --zone=europe-west3-a

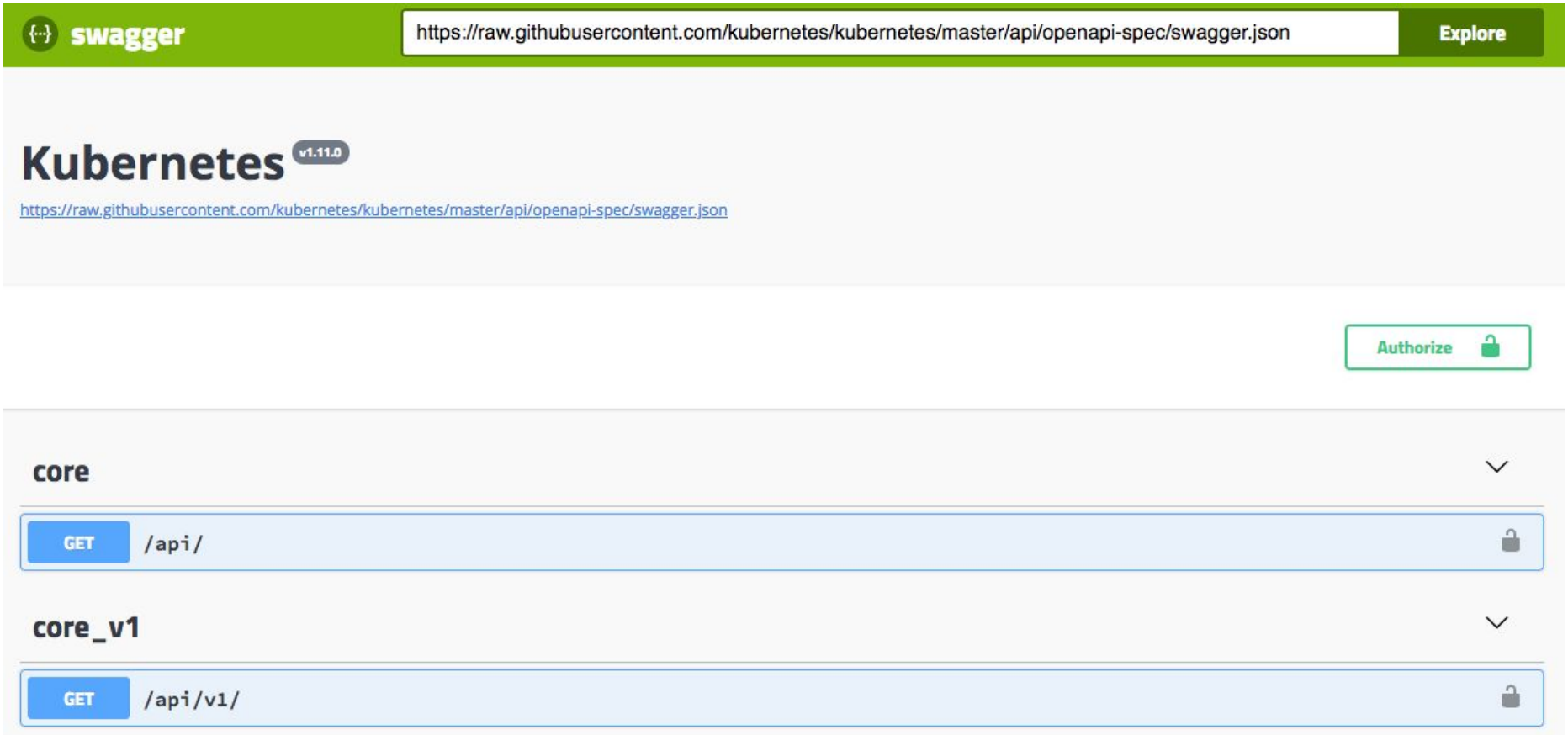
Note: deleting a cluster doesn’t delete your storage / disks on GKE, you’ve to delete them manually

- Create a Kubernetes cluster on AWS
  - Typhoon →
  - Kubeadm →
  - Kops FastStart →
  - Kubicorn →
  - TK8 →
  - Kubernetes Cluster API →

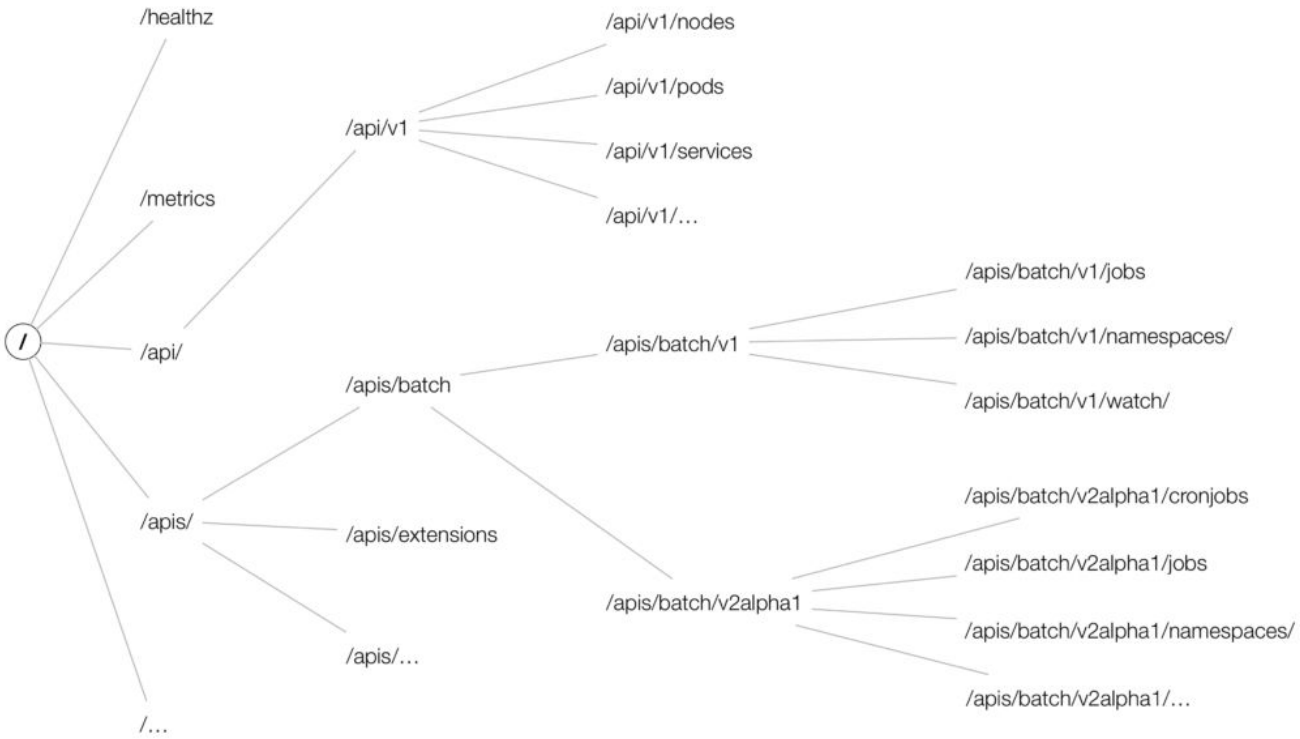
- Create a Kubernetes cluster on ACS
  - Please refer to Kubernetes CookBook

Source:

- Install Swagger UI on Minikube / Minishift / Tectonic
  - k run swagger-ui --image=swaggerapi/swagger-ui:latest
  - On Tectonic →
    - k expose deployment swagger-ui --port=8080 --external-ip=172.17.4.101 --type=NodePort
  - On Minikube →
    - k expose deployment swagger-ui --port=8080 --external-ip=$(minikube ip) --type=NodePort
  - Use swagger.json to explore the API


swagger
https://raw.githubusercontent.com/kubernetes/kubernetes/master/api/openapi-spec/swagger.json
Explore
Kubernetes
v1.11.0
https://raw.githubusercontent.com/kubernetes/kubernetes/master/api/openapi-spec/swagger.json
Authorize
core
GET
/api/
core_v1
GET
/api/v1/

Enjoy the Kubernetes API deep dive →



Get all API resources supported by your K8s cluster:

$ kubectl api-resources -o wide

Get API resources for a particular API group:

$ kubectl api-resources --api-group apps -o wide

Get more info about the particular resource:

$ kubectl explain configmap

Source: https://akomljen.com/kubernetes-api-resources-which-group-and-version-to-use/

Get all API versions supported by your K8s cluster:

$ kubectl api-versions

Check if a particular group/version is available for some resource:

$ kubectl get deployments.v1.apps -n kube-system

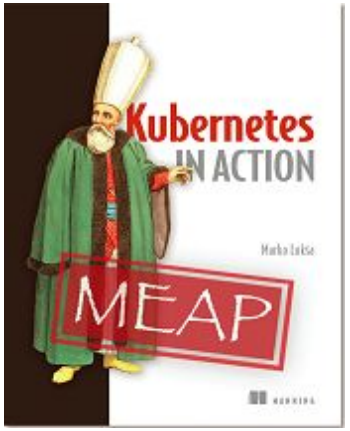
Source: https://akomljen.com/kubernetes-api-resources-which-group-and-version-to-use/

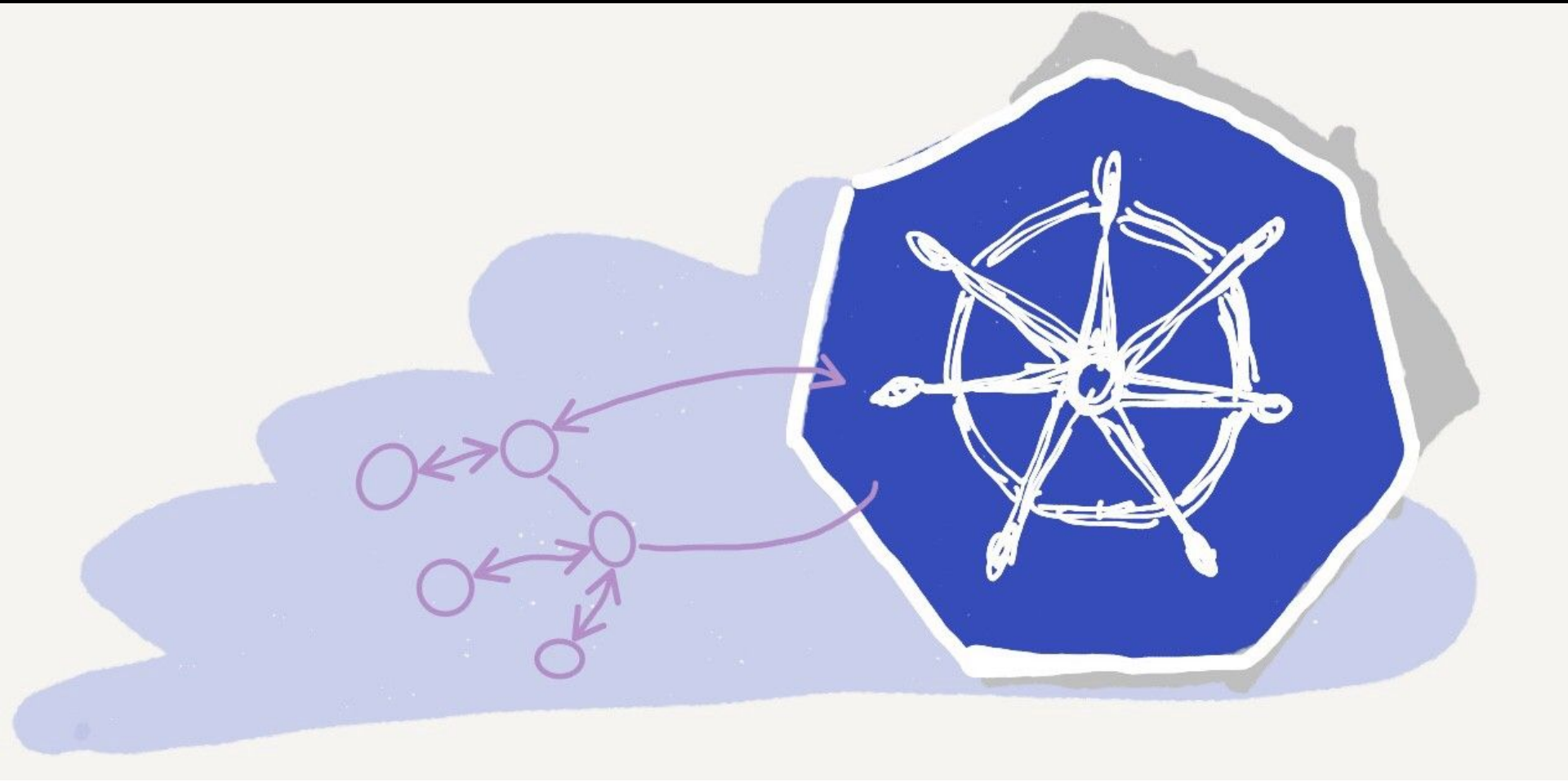
- Start the Ghost micro-blogging platform
  - kubectl run ghost --image=ghost:0.9
  - kubectl expose deployments ghost --port=2368 --type=LoadBalancer
  - k expose deployment ghost --port=2368 --external-ip=$(minikube ip) --type=NodePort
  - kubectl get svc
  - kubectl get deploy
  - kubectl edit deploy ghost

```
NAME                     READY   STATUS              RESTARTS   AGE
ghost-7cbd79df7d-6shhh   0/1     ContainerCreating   0          2s
ghost-7cbd79df7d-7vtjw   1/1     Running             0          1m
ghost-7cbd79df7d-fd7b9   1/1     Running             0          11m
```

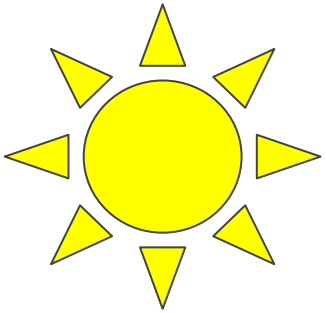
- Log into the pod
  - kubectl exec -it ghost-xxx bash
- Get the logs from the pod
  - kubectl logs ghost-xxx
- Delete the Ghost micro-bloging platform
  - kubectl delete deploy ghost
- Get the cluster state
  - kubectl cluster-info dump --all-namespaces --output-directory=$PWD/cluster-state

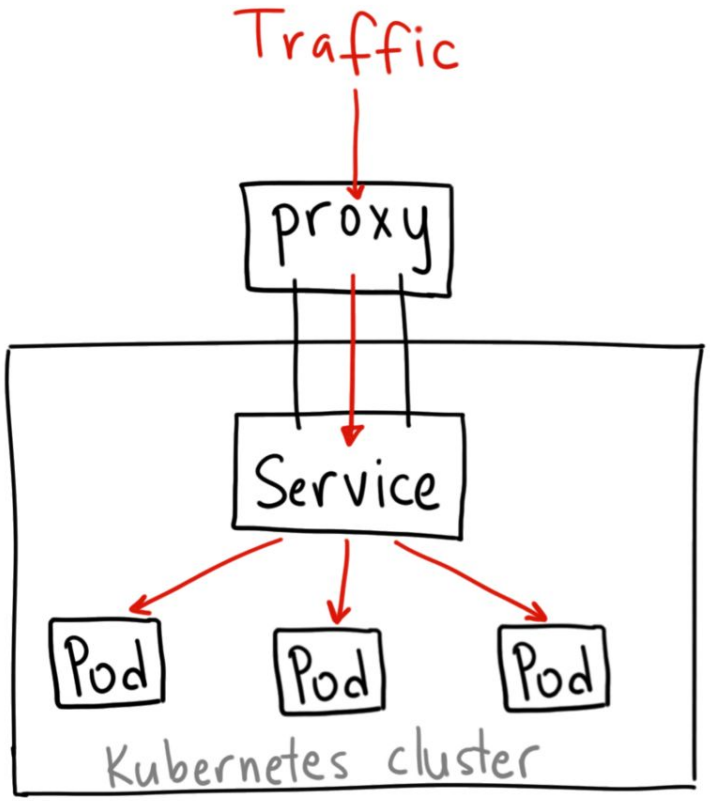
- Please read and understand this great free chapter from Kubernetes in Action book by Marko Lukša.





Source: https://medium.com/@metaphorical/internal-and-external-connectivity-in-kubernetes-space-a25cba822089



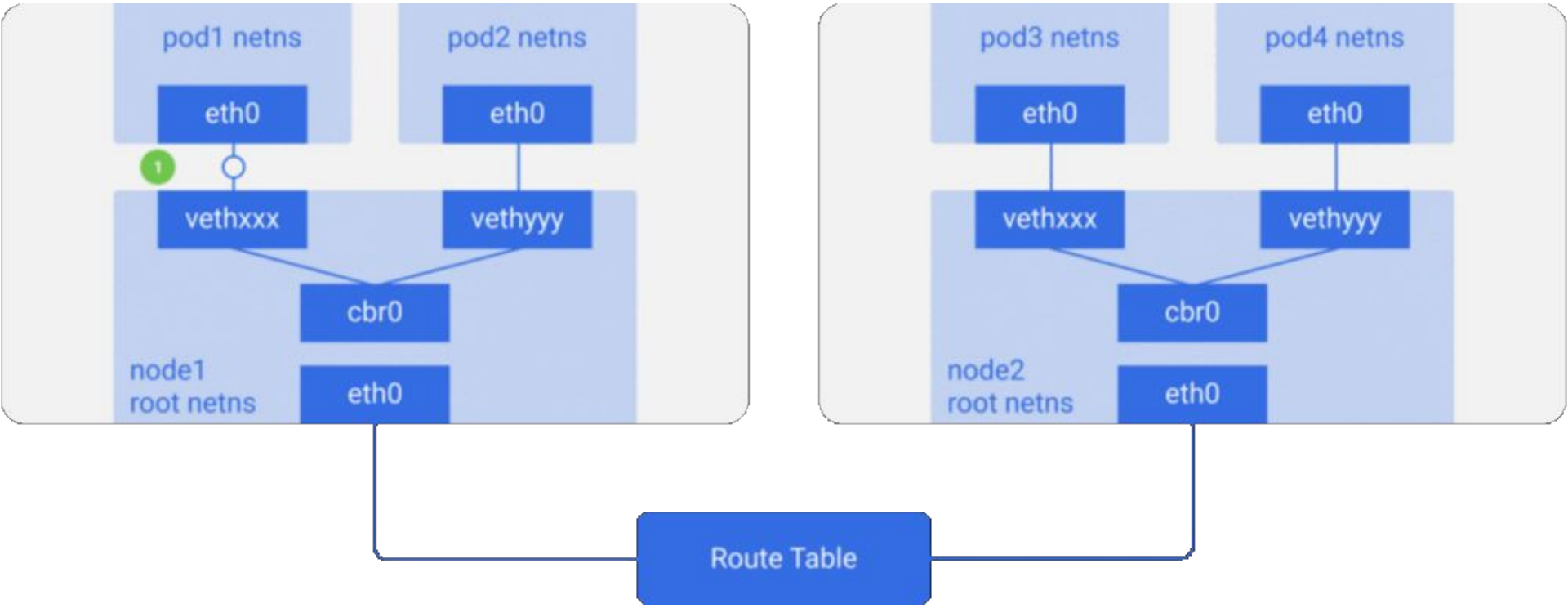


Source: https://medium.com/google-cloud/kubernetes-nodeport-vs-loadbalancer-vs-ingress-when-should-i-use-what-922f010849e0

- 3 Ways to expose your services in Kubernetes
  - NodePort
  - External LoadBalancer
    - MetalLB consideration
  - Ingress
    - Ingress Controller
    - Ingress resource
  - More + →
  - More ++ →

- Ambassador is an open source, Kubernetes-native microservices API gateway built on the Envoy Proxy.
- Ambassador is awesome and powerful, eliminates the shortcomings of Kubernetes ingress capabilities
- Ambassador is easily configured via Kubernetes annotations
- Ambassador is in active development by datawire.io
- Needles to say Ambassador is open source →

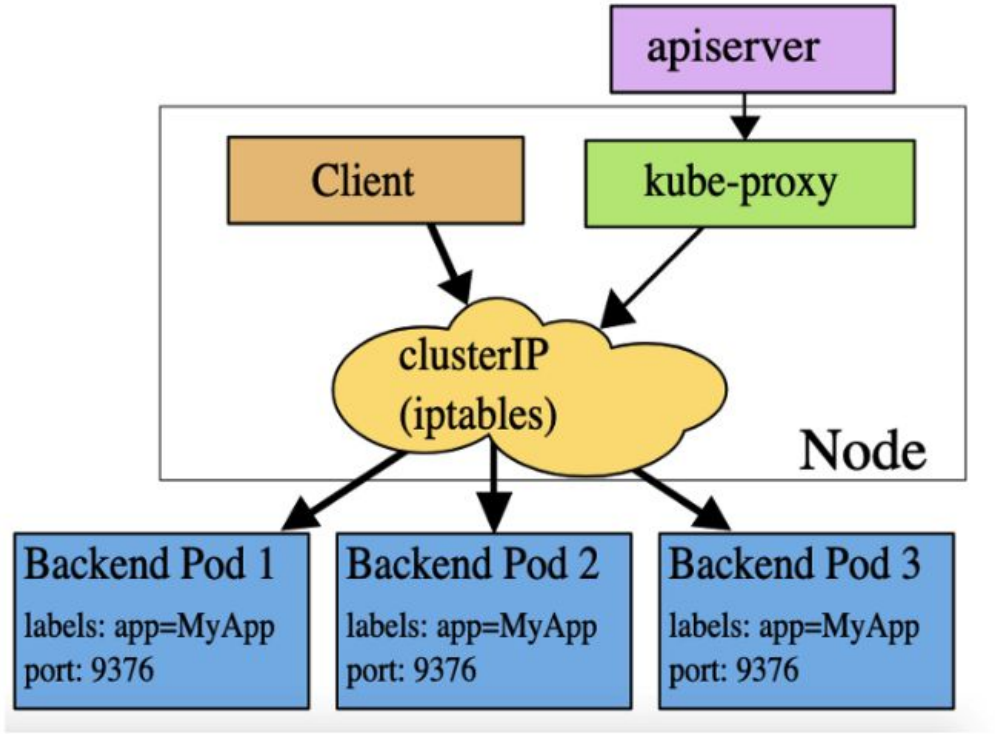
Source: https://blog.getambassador.io/kubernetes-ingress-nodeport-load-balancers-and-ingress-controllers-6e29f1c44f2d



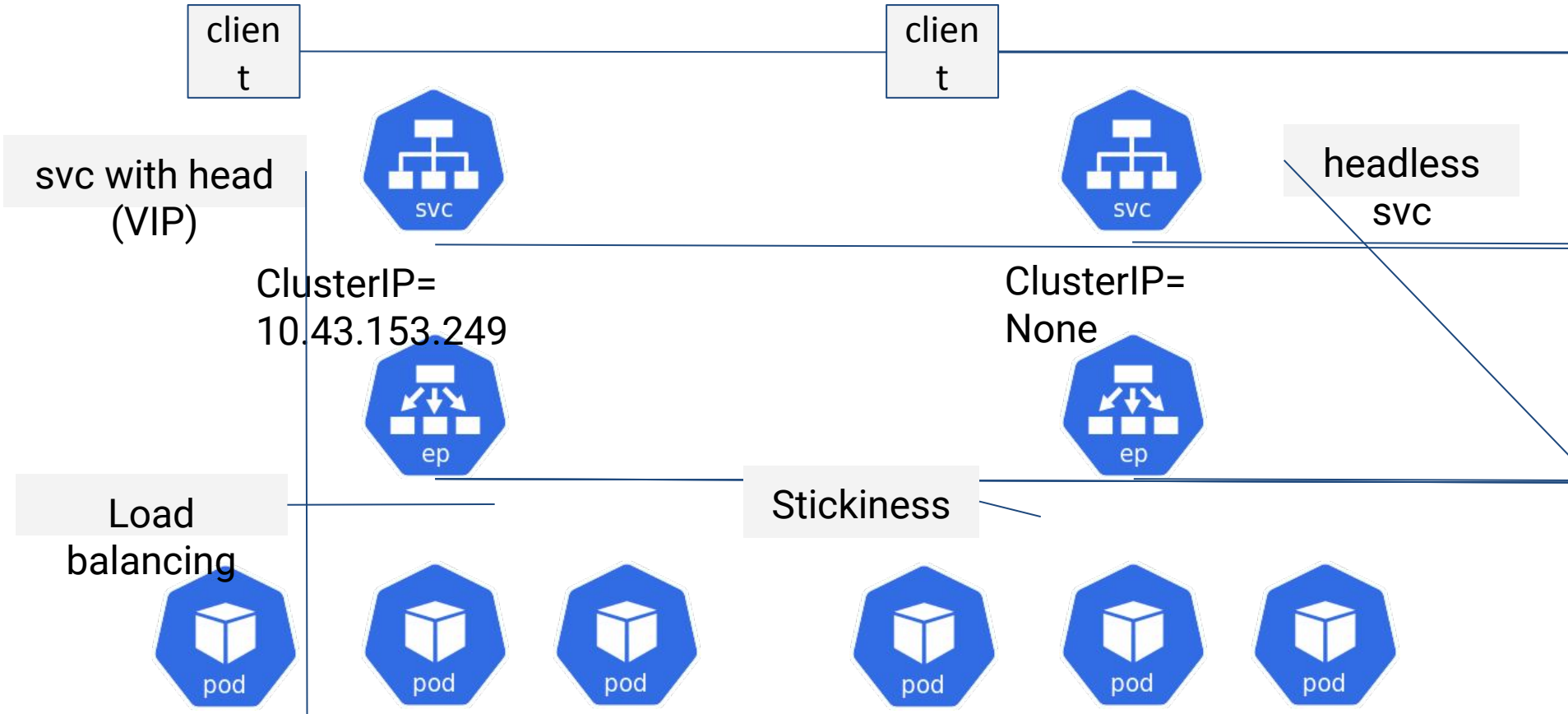
Source: https://itnext.io/an-illustrated-guide-to-kubernetes-networking-part-1-d1ede3322727

- Every Pod has a unique IP
- Pod IP is shared by all the containers in this Pod, and it's routable from all the other Pods.
- All containers within a pod can communicate with each other.
- All Pods can communicate with all other Pods without NAT.
- All nodes can communicate with all Pods (and vice-versa) without NAT.
- The IP that a Pod sees itself as, is the same IP that others see it as.

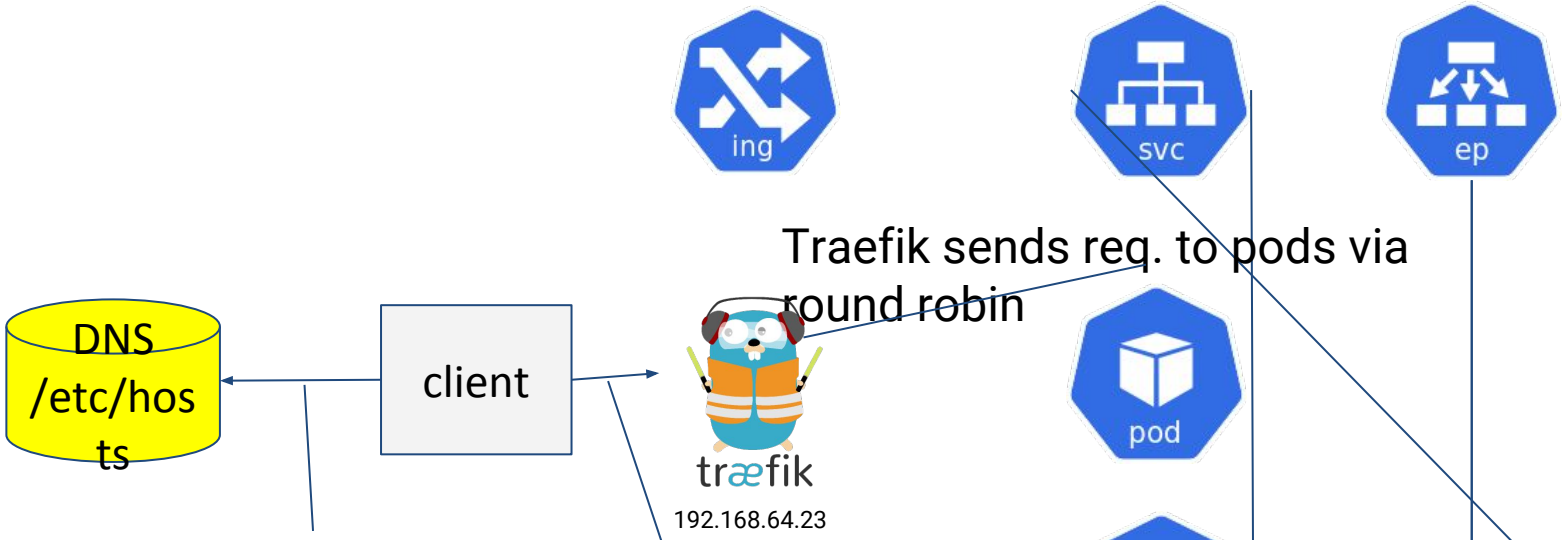
Source: https://itnext.io/an-illustrated-guide-to-kubernetes-networking-part-1-d1ede3322727

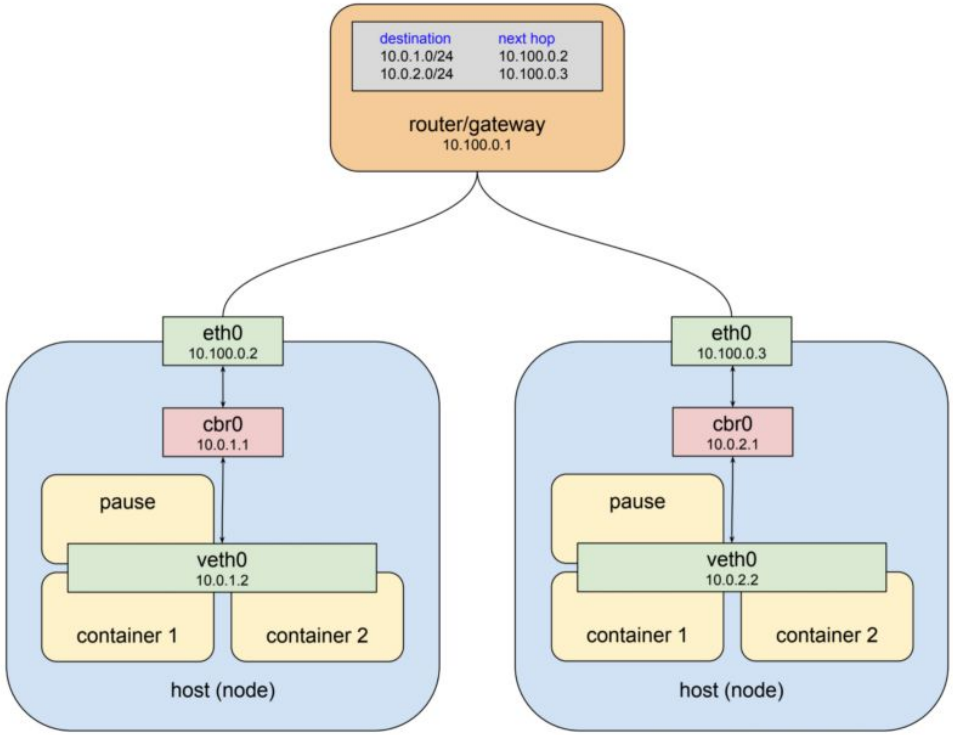


Source: https://medium.com/devopslinks/kubernetes-headless-service-vs-clusterip-and-traffic-distribution-904b058f0dfd

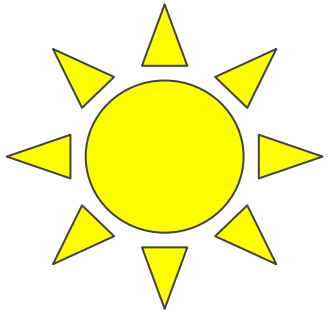


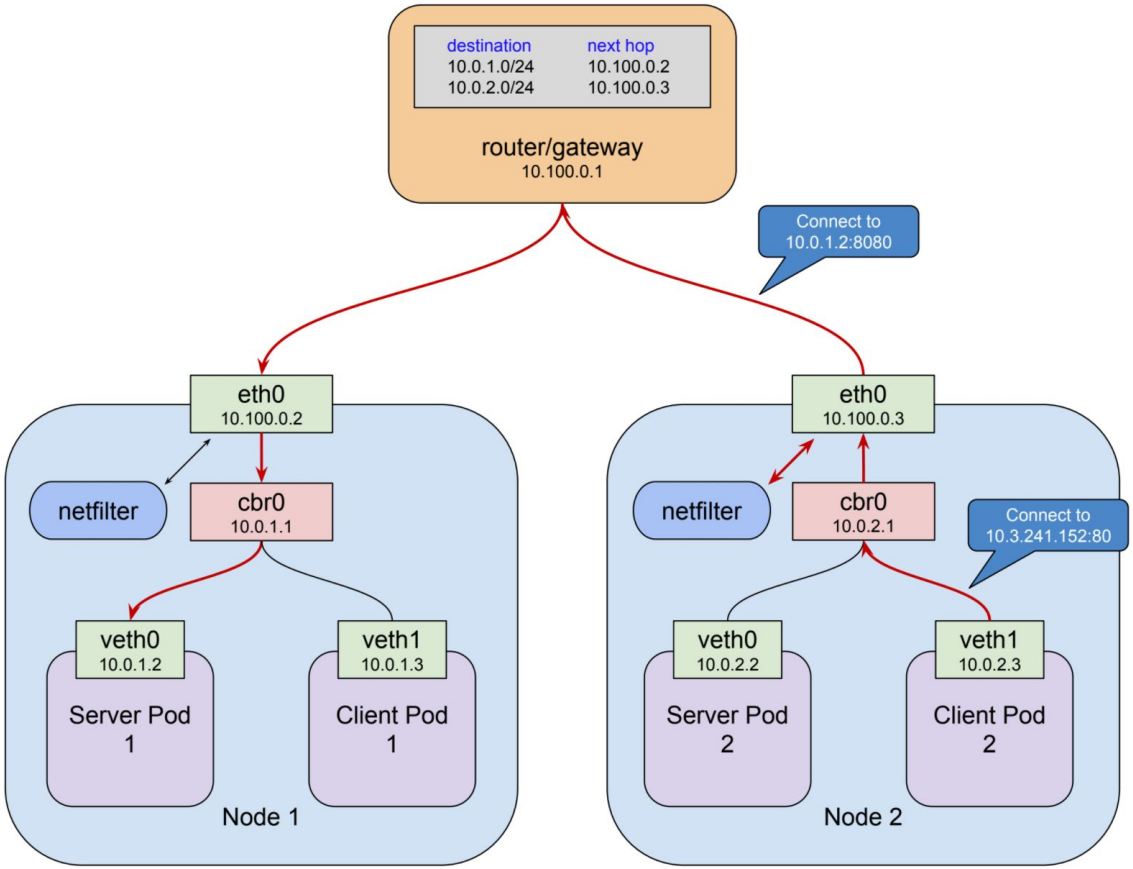
https://github.com/arashkaffamanesh/practical-kubernetes-problems#headless-services-for-stickiness


ing
svc
ep
Traefik sends req. to pods via round robin
DNS /etc/hosts
client
træfik
192.168.64.23
pod
pod
1- Client looks up my.ghost.svc
2- Client sends HTTP GET req. to controller with my.ghost.svc In host header



Source: https://medium.com/@tao_66792/how-does-the-kubernetes-networking-work-part-1-5e2da2696701





Source: https://medium.com/google-cloud/understanding-kubernetes-networking-pods-7117dd28727

- How can you have same experience of using a load balancer service type on your bare metal cluster just like public clouds?
- This is what Metallb aims to solve.
- Layer 2/ARP mode: Only one worker node can respond to the Load Balancer IP address
- BGP mode: This is more scalable, all the worker nodes will respond to the Load Balancer IP address, this means that even of one of the worker nodes is unavailable, other worker nodes will take up the traffic. This is one of the advantages over Layer 2 mode but you need a BGP router on your network (open source routers Free Range Router, Vyos)

Source: https://metallb.universe.tf/

- Work around for the Layer 2 disadvantage is to use a CNI plugin that supports BGP like Kuberouter
- Kuberouter will then advertise the LB IP via BGP as ECMP route which will be available via all the worker nodes.

Source: https://github.com/cloudnativelabs/kube-router/blob/master/docs/user-guide.md#advertising-ips

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: config
data:
  config: |
    address-pools:
    - name: my-ip-space
      protocol: layer2
      addresses:
      - 84.200.xxx.xxx-84.200.xxx.xxx
```

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    peers:
    - my-asn: 64500
      peer-asn: 64500
      peer-address: 10.96.0.100
    - my-asn: 64500
      peer-asn: 64500
      peer-address: 10.96.0.101
    address-pools:
    - name: my-ip-space
      protocol: bgp
      addresses:
      - 198.51.100.0/24
```

Security

- Make sure to always scan all your Docker Images and Containers for potential threats
- Never use any random Docker Image(s) and always use authorised images in your environment
- Categorise and accordingly split up your cluster through Namespace
- Use Network Policies to implement proper network segmentation and Role Based Access Control(RBAC) to create administrative boundaries between resources for proper segregation and control

Source: https://blog.kubernauts.io/kubernetes-best-practices-d5cbef02fe1b

- Limit SSH access to Kubernetes nodes, and Ask users to use kubectl exec instead.
- Never use Passwords, or API tokens in plain text or as environment variables, use secrets instead
- Use non-root user inside container with proper host to container, UID and GID mapping

Source: https://blog.kubernauts.io/kubernetes-best-practices-d5cbef02fe1b

- If you’re serious about security in Kubernetes, you need a secret management tool that provides a single source of secrets, credentials, attaching security policies, etc.
- In other words, you need Hashicorp Vault.

Source: https://blog.kubernauts.io/managing-secrets-in-kubernetes-with-vault-by-hashicorp-f0db45cc208a

Exercises

kubectl cheat sheet

→ https://github.com/dennyzhang/cheatsheet-kubernetes-A4

$ kubectl get events --sort-by=.metadata.creationTimestamp # List Events sorted by timestamp
$ kubectl get services --sort-by=.metadata.name # List Services Sorted by Name
$ kubectl get pods --sort-by=.metadata.name
$ kubectl get endpoints
$ kubectl explain pods,svc
$ kubectl get pods -A # --all-namespaces
$ kubectl get nodes -o jsonpath='{.items[*].spec.podCIDR}'
$ kubectl get pods -o wide
$ kubectl get pod my-pod -o yaml --export > my-pod.yaml
$ kubectl get pods --show-labels # Show labels for all pods (or other objects)
$ kubectl get pods --sort-by='.status.containerStatuses[0].restartCount'
$ kubectl cluster-info
$ kubectl api-resources
$ kubectl get apiservice

- By the awesome Kubernaut Michael Hausenblas
- Hands-On introduction to Kubernetes →

Note: you can run the examples on minikube, OpenShift, GKE or any other Kubernetes Installations.

- By the awesome Bob Killen
- Introduction to Kubernetes →

  (The best introduction which I know about!)
- Kubernetes Tutorials →

More Exercises

- Create a deployment running nginx version 1.12.2 that will run in 2 pods
  - Scale this to 4 pods
  - Scale it back to 2 pods
  - Upgrade the nginx image version to 1.13.8
  - Check the status of the upgrade
  - Check the history
  - Undo the upgrade
  - Delete the deployment

Source:

- Create nginx version 1.12.2 with 2 pods
  - kubectl run nginx --image=nginx:1.12.2 --replicas=2 --record
- Scale to 5 pods
  - kubectl scale --replicas=5 deployment nginx
- Scale back to 2 pods
  - kubectl scale --replicas=2 deployment nginx
- Upgrade the nginx image to 1.13.8 version
  - kubectl set image deployment nginx nginx=nginx:1.13.8

Source:

- Check the status of the upgrade
  - kubectl **rollout status** deployment nginx
- Get the history of the actions
  - kubectl **rollout history** deployment nginx
- Undo / rollback the upgrade
  - kubectl **rollout undo** deployment nginx
- Delete the deployment
  - k delete deploy/nginx

Source:

- Create the deployment with a manifest:
  - kubectl create -f nginx.yaml

Note: Pods, services, configmaps, secrets in our examples are all part of the /api/v1 API group, while deployments are part of the **/apis/extensions/v1beta1** API group.

The group an object is part of is what is referred to as apiVersion in the object specification, available via the API reference.

```
$ cat nginx.yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.12.2
        ports:
        - containerPort: 80
```

- Edit the deployment: change the replicas to 5 and image version to 1.13.8
  - kubectl **edit deployment** nginx
- Get some info about the deployment and ReplicaSet
  - kubectl get deploy
  - kubectl get rs
  - k get pods -o wide (set alias k=’kubectl’)
  - k describe pod nginx-xyz

- kubectl expose deployments nginx --port=80 --type=**LoadBalancer**

- k get svc

```
NAME         TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
kubernetes   ClusterIP      10.35.240.1     <none>           443/TCP        2h
nginx        LoadBalancer   10.35.254.180   35.198.104.213   80:31846/TCP   3m
```

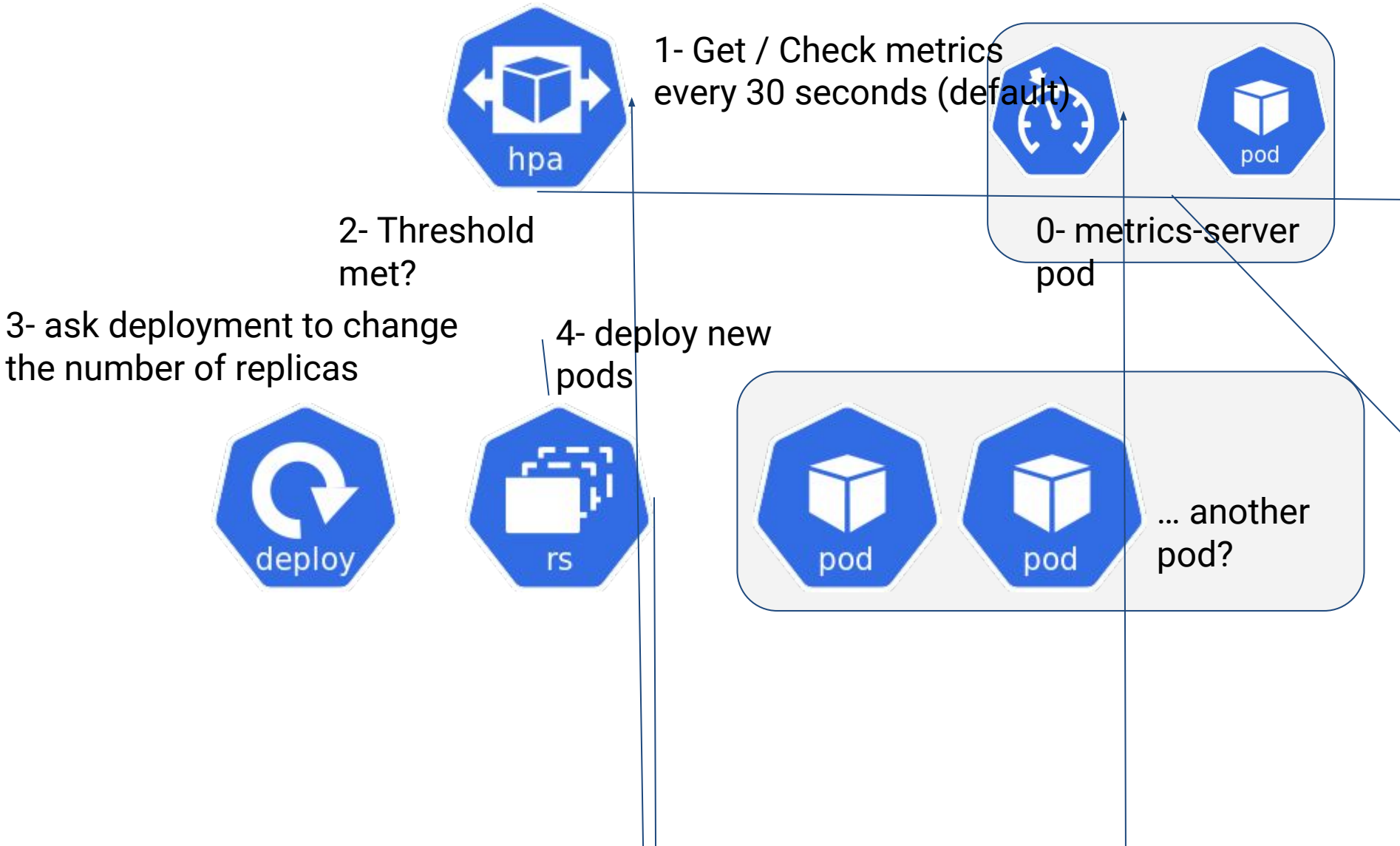
- Write an ingress rule that redirects calls to /foo to one service and to /bar to another
  - k create -f ingress.yaml

```
$ cat ingress.yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: test
  annotations:
    ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - host: kubernauts.io
    http:
      paths:
      - path: /foo
        backend:
          serviceName: s1
          servicePort: 80
      - path: /bar
        backend:
          serviceName: s2
          servicePort: 80
```

kubectl run kubia --image=luksa/kubia --port=8080 --generator=run/v1
kubectl run kubia --image=luksa/kubia --port=8080
k get svc
k get pods
k get rc
k get rs
kubectl describe rs kubia-57478bf476
k get svc
k expose rc kubia --type=LoadBalancer --name kubia-http
k expose rs kubia --type=LoadBalancer --name kubia-http2
k expose rs kubia-57478bf476 --type=LoadBalancer --name kubia-http2
k get pods
k scale rc kubia --replicas=3
k get pods
k scale rs kubia-57478bf476 --replicas=3 —> can't work, you should scale the deployment
k scale deployment kubia --replicas=3
k port-forward kubia-xxxxx 8888:8080

Note: the kubia image is from the Kubernetes in Action book by Marko Lukša

http://127.0.0.1:8888/


hpa
1- Get / Check metrics every 30 seconds (default)
pod
0- metrics-server pod
2- Threshold met?
3- ask deployment to change the number of replicas
4- deploy new pods
deploy
rs
pod
pod
… another pod?

- On GKE:

kubectl run ghost --image=ghost:0.9 **--requests="cpu=100m"**
k expose deployment ghost --port=2368 --type=LoadBalancer
k autoscale deployment ghost --min=1 --max=4 --cpu-percent=10
export loadbalancer_ip=$(k get svc -o wide | grep ghost | awk '{print $4}')
while true; do curl http://$loadbalancer_ip:2368/ ; done
k get hpa -w
k describe hpa

- On Minikube (hpa doesn't work for now on minikube → bug??)

minikube addons enable heapster
kubectl run ghost --image=ghost:0.9 **--requests="cpu=100m"**
k expose deployment ghost --port=2368 --type=NodePort --external-ip=$(minikube ip)
k autoscale deployment ghost --min=1 --max=4 --cpu-percent=10
while true; do curl http://$(minikube ip):2368/ ; done
k get hpa -w
k describe hpa
→ unable to get metrics for resource cpu

```
gcloud compute disks create --size=1GiB --zone=us-central1-a pv-a
gcloud compute disks create --size=1GiB --zone=us-central1-a  pv-b
gcloud compute disks create --size=1GiB --zone=us-central1-a  pv-c
k create -f persistent-volumes-gcepd.yaml
k create -f kubia-service-headless.yaml
k create -f kubia-statefulset.yaml
k get po
k get po kubia-0 -o yaml
k get pvc
k proxy
k create -f kubia-service-public.yaml
k proxy
```

Note: This example is from the Chapter 10 of the Kubernetes in Action book by Marko Lukša

```
minikube stop
minikube start --extra-config=apiserver.Authorization.Mode=RBAC
k create ns foo
k create ns bar
k run test --image=luksa/kubectl-proxy -n foo
k run test --image=luksa/kubectl-proxy -n bar
k get po -n foo
k get po -n bar
k exec -it test-xxxxxxxxx-yyyyy -n foo sh
k exec -it test-yyyyyyyyy-xxxxx -n bar sh
curl localhost:8001/api/v1/namespaces/foo/services
curl localhost:8001/api/v1/namespaces/bar/services
cd Chapter12/
cat service-reader.yaml
k create -f service-reader.yaml -n foo
k create role service-reader --verb=get --verb=list --resource=services -n bar
k create rolebinding test --role=service-reader --serviceaccount=foo:default -n foo
k create rolebinding test --role=service-reader --serviceaccount=bar:default -n bar
k edit rolebinding test -n foo
k edit rolebinding test -n bar
```

Practical K8s Problems

https://github.com/arashkaffamanesh/practical-kubernetes-problems

Tips & Tricks

- List all Persistent Volumes sorted by their name
  - kubectl get pv | grep -v NAME | sort -k 2 -rh
- Find which pod is taking max CPU
  - kubectl top pod
- Find which node is taking max CPU
  - kubectl top node
- Getting a Detailed Snapshot of the Cluster State
  - kubectl cluster-info dump --all-namespaces > cluster-state
- Save the manifest of a running pod
  - kubectl get pod name -o yaml --export > pod.yml
- Save the manifest of a running deployment
  - kubectl get deploy name -o yaml --export > deploy.yml
- Use dry-run to create a manifest for a deployment
  - kubectl run ghost --image=ghost --restart=Always --expose --port=80 --output=yaml --dry-run > ghost.yaml
  - k apply -f ghost.yaml
  - k get all
- Delete evicted pods
  - `kubectl get po -A -o json | jq  '.items[] | select(.status.reason!=null) | select(.status.reason | contains("Evicted")) | "kubectl delete po \(.metadata.name) -n \(.metadata.namespace)"' | xargs -n 1 bash -c`

- Find all deployments which have no resource limits set
  - kubectl get deploy -o json |

    jq ".items[] | select(.spec.template.spec.containers[].resources.limits==null) | {DeploymentName:.metadata.name}"
- Create a yaml for a job
  - kubectl run --generator=job/v1 test --image=nginx --dry-run -o yaml
- Find all pods in the cluster which are not running
  - kubectl get pod --all-namespaces -o json | jq '.items[] | select(.status.phase!="Running") | [.metadata.namespace,.metadata.name,.status.phase] | join(":")'
- List the top 3 nodes with the highest CPU usage
  - kubectl top nodes | sort --reverse --numeric -k 3 | head -n3
- List the top 3 nodes with the highest MEM usage
  - kubectl top nodes | sort --reverse --numeric -k 5 | head -n3
- Get rolling Update details for deployments
  - kubectl get deploy -o json |

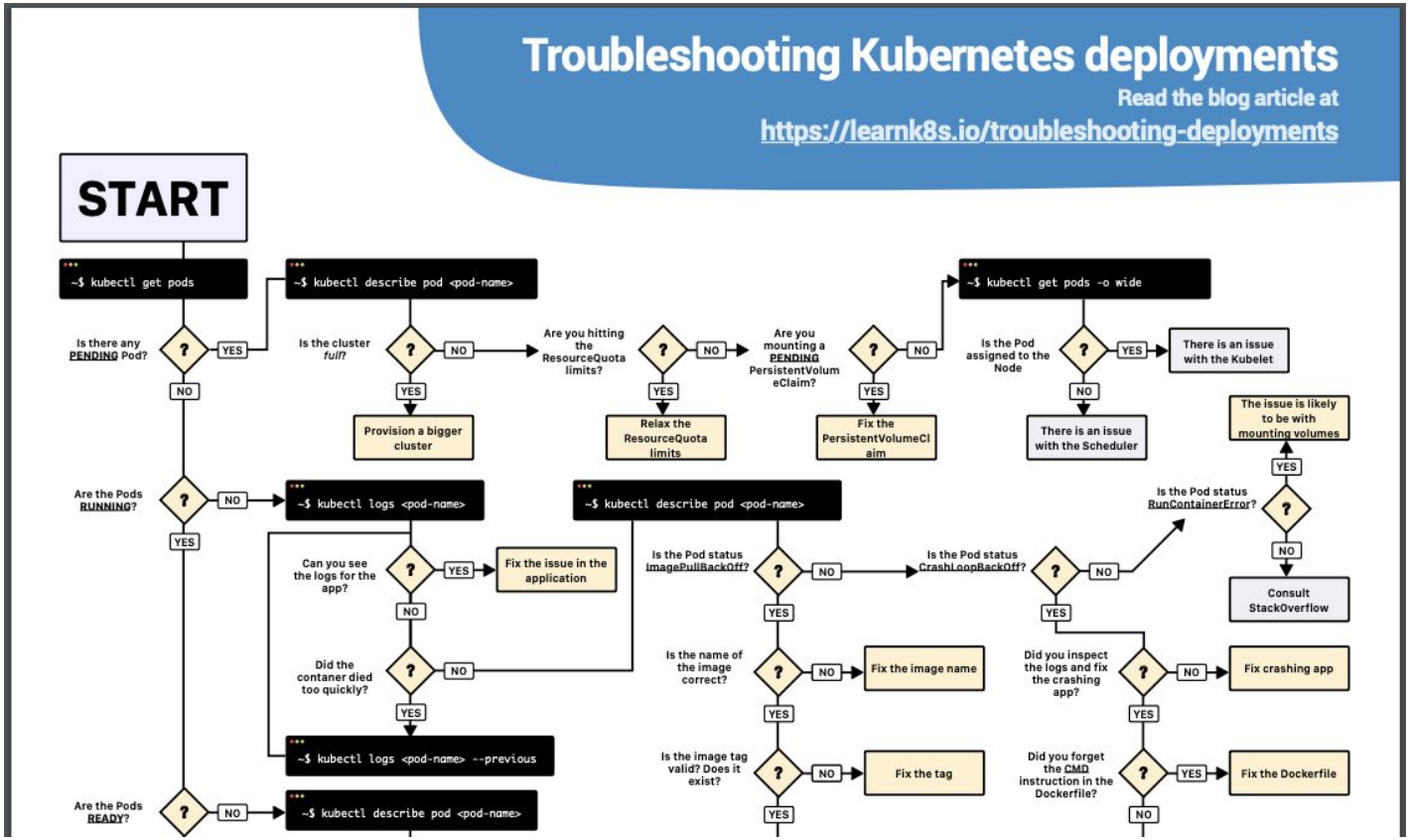
    jq ".items[] | {name:.metadata.name} + .spec.strategy.rollingUpdate"
- List pods and its corresponding containers
  - kubectl get pods

- Troubleshoot a faulty node
  - Check the status of kubelet
    - systemctl status kubelet
  - If it's running, check the logs locally with
    - journalctl -u kubelet
  - If it's not running, you probably need to start it:
    - systemctl restart kubelet
  - If a node is not getting pods schedule to it, describe the node
    - kubectl describe node <nodename>
  - If your pods are stuck in pending, check your scheduler services:
    - systemctl status kube-scheduler
  - Or by scheduler pods in a kubeadm / rancher cluster
    - kubectl get pods -n kube-system
    - kubectl logs kube-scheduler-master -n kube-system
- Get quota for each node:

  kubectl get nodes --no-headers | awk '{print $1}' | xargs -I {} sh -c 'echo {}; kubectl describe node {} | grep Allocated -A 5 | grep -ve Event -ve Allocated -ve percent -ve -- ; echo'
- Get nodes which have no taints

  ```
  kubectl get nodes -o json | jq  -r '.items[] | select(.spec.taints == null) | "\(.metadata.name)"'
  ```
- Find out the unused/unupdated deployments in your clusters

  ```
  kubectl get deploy --all-namespaces -ojson | jq '.items[] | "\(.metadata.namespace) \(.metadata.name) \(.spec.replicas) \(.status.conditions[0].lastUpdateTime)"'
  ```



https://learnk8s.io/troubleshooting-deployments

K8s Practice Questions

- Create a yaml for a job that calculates the value of pi
- Create an Nginx Pod and attach an EmptyDir volume to it.
- Create an Nginx deployment in the namespace “kube-cologne” and corresponding service of type NodePort . Service should be accessible on HTTP (80) and HTTPS (443)
- Add label to a node as "arch=gpu"
- Create a Role in the “conference” namespace to grant read access to pods.
- Create a RoleBinding to grant the "pod-reader" role to a user "john" within the “conference” namespace.
- Create an Horizontal Pod Autoscaler to automatically scale the Deployment if the CPU usage is above 50%.

- Deploy a default Network Policy for each resources in the default namespace to deny all ingress and egress traffic.
- Create a pod that contain multiple containers : nginx, redis, postgres with a single YAML file.
- Deploy nginx application but with extra security using PodSecurityPolicy
- Create a Config map from file.
- Create a Pod using the busybox image to display the entire content of the above ConfigMap mounted as Volumes.
- Create configmap from literal values
- Create a Pod using the busybox image to display the entire ConfigMap in environment variables automatically.
- Create a ResourceQuota in a namespace "kube-cologne" that allows maximum of

- Create ResourceQuota for a namespace "quota-namespace"
- Create Pod quota for a namespace "pod-quota"
- Deployment Exercise
  - Create nginx deployment and scale to 3
  - Check the history of the previous Nginx deployment
  - Update the Nginx version to the 1.9.1 in the previous deployment
  - Check the history of the deployment to note the new entry
- Add liveness and readiness probe to kuard container

And the solutions:

https://github.com/ipochi/k8s-practice-questions/blob/master/practice-questions-with-solutions.md

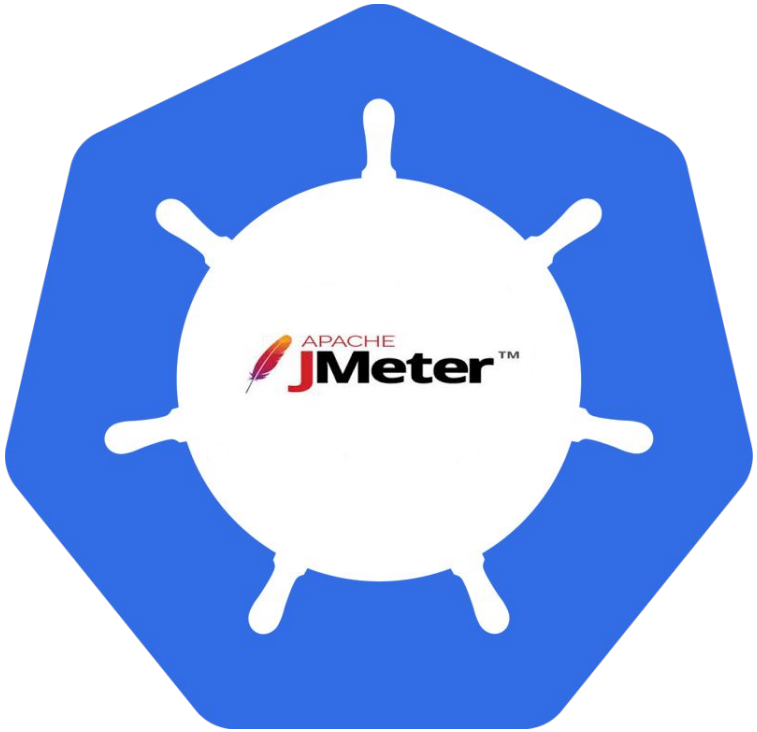
General Questions

- What happens to services, when the control plane goes down?
  - The services are not affected as far they don’t change. e.g. if you expose a service to the world via LB it should still work.
- If it is not exposed via LB, how can pods can communicate with a service internally, if control pane is down. How does the pod know about which end points this service is connected to?
  - Those endpoint are defined by kube-proxy (iptable) in the node , when you add a new service the iptable of kube-proxy is updated, no matter the plane control falls or not. You need to know that the nodes can work without api-server thanks to the kubelet with static manifests

Source: https://kubernauts.slack.com/archives/G6CCNMVKM/p1562305149191600

Advanced Exercises

- A more complete example: https://goo.gl/k5rFpb



- TK8 on Github:

  https://github.com/kubernauts/tk8

- Github link:
  - https://github.com/kubernauts/kafka-confluent-platform