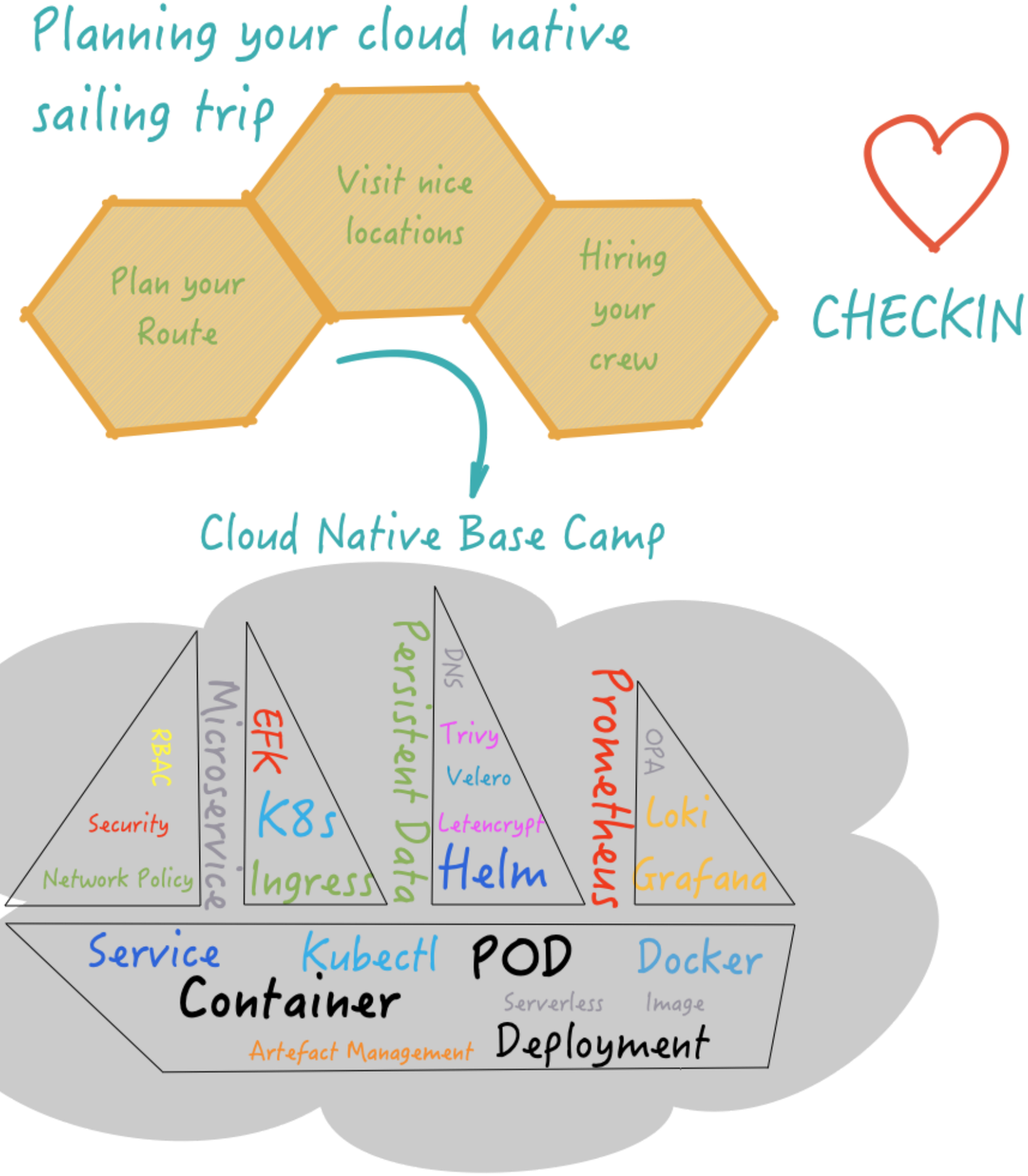


linuxhotel
Schulungen bei den OpenSource'lern

www.linuxhotel.de

Startpage - Linuxhotel

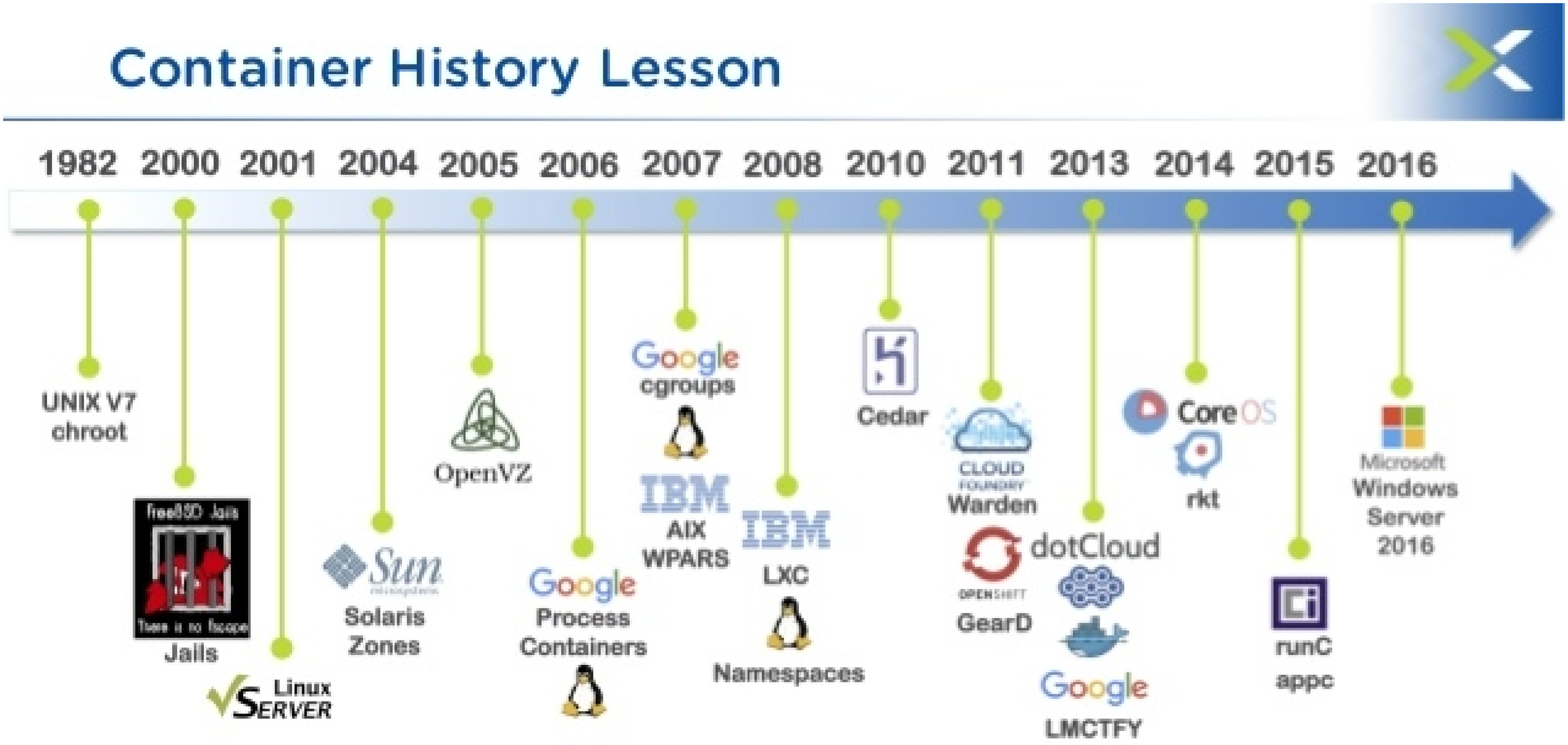
Kurse für Admins, Entwickler und Anwender

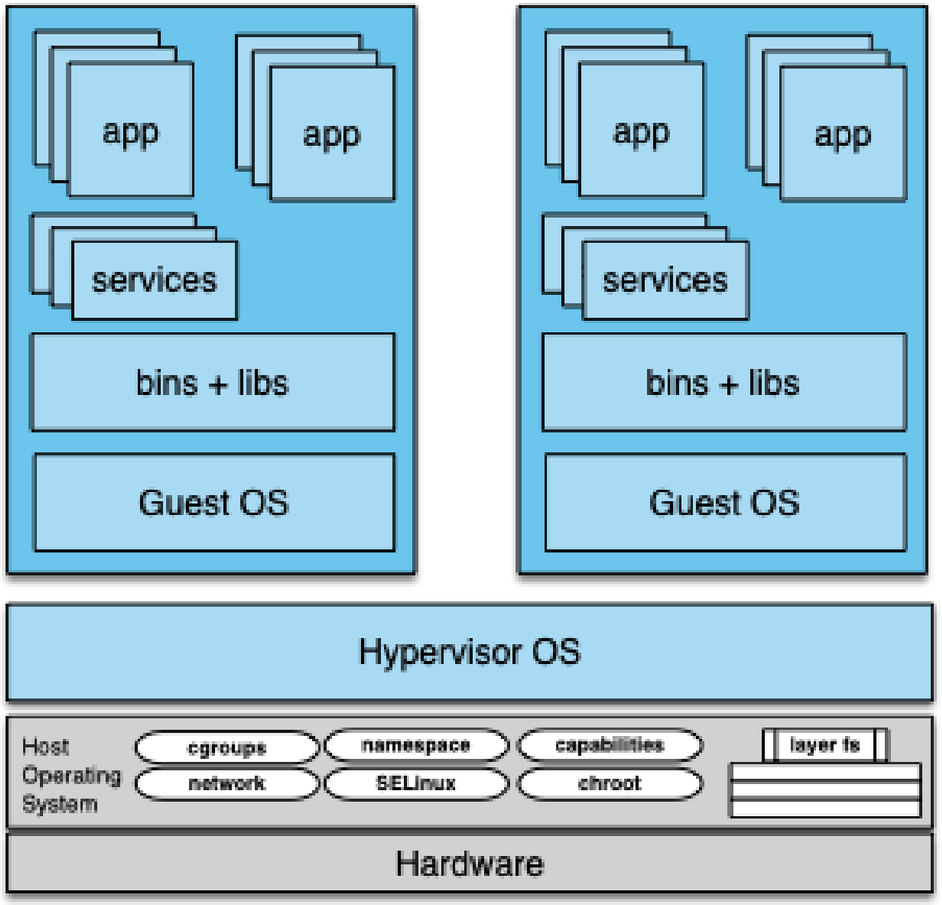
bee42

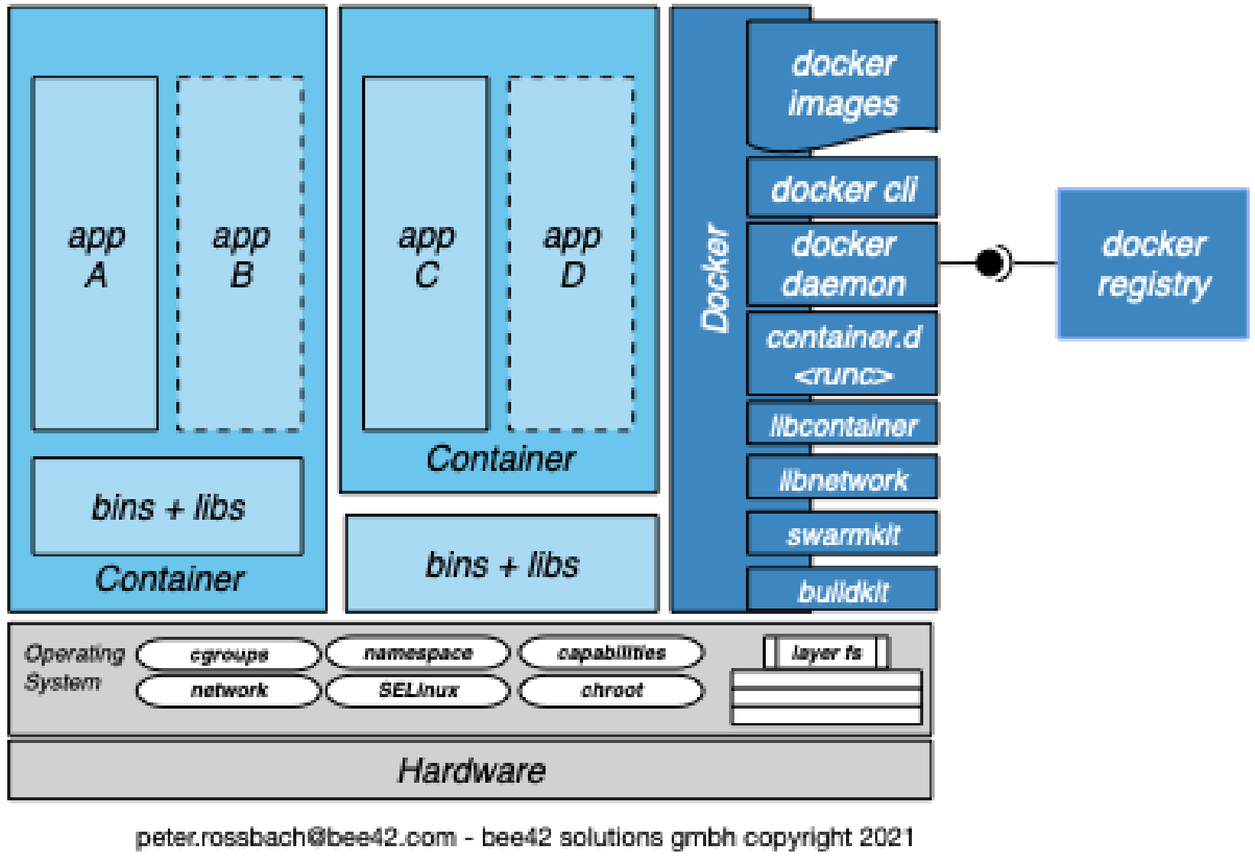
Peter Rossbach
bee42 solutions gmbh
<peter.rossbach@bee42.com>


42
40FT
20FT

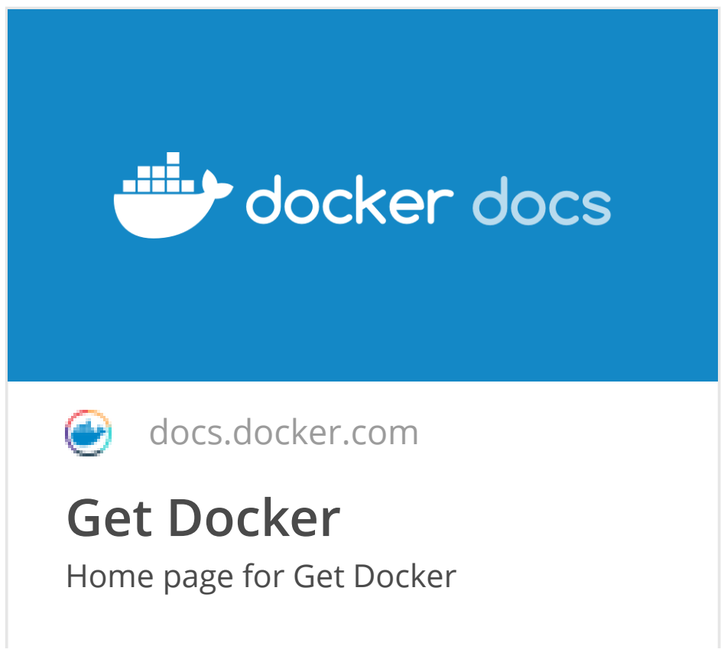
Container History Lesson



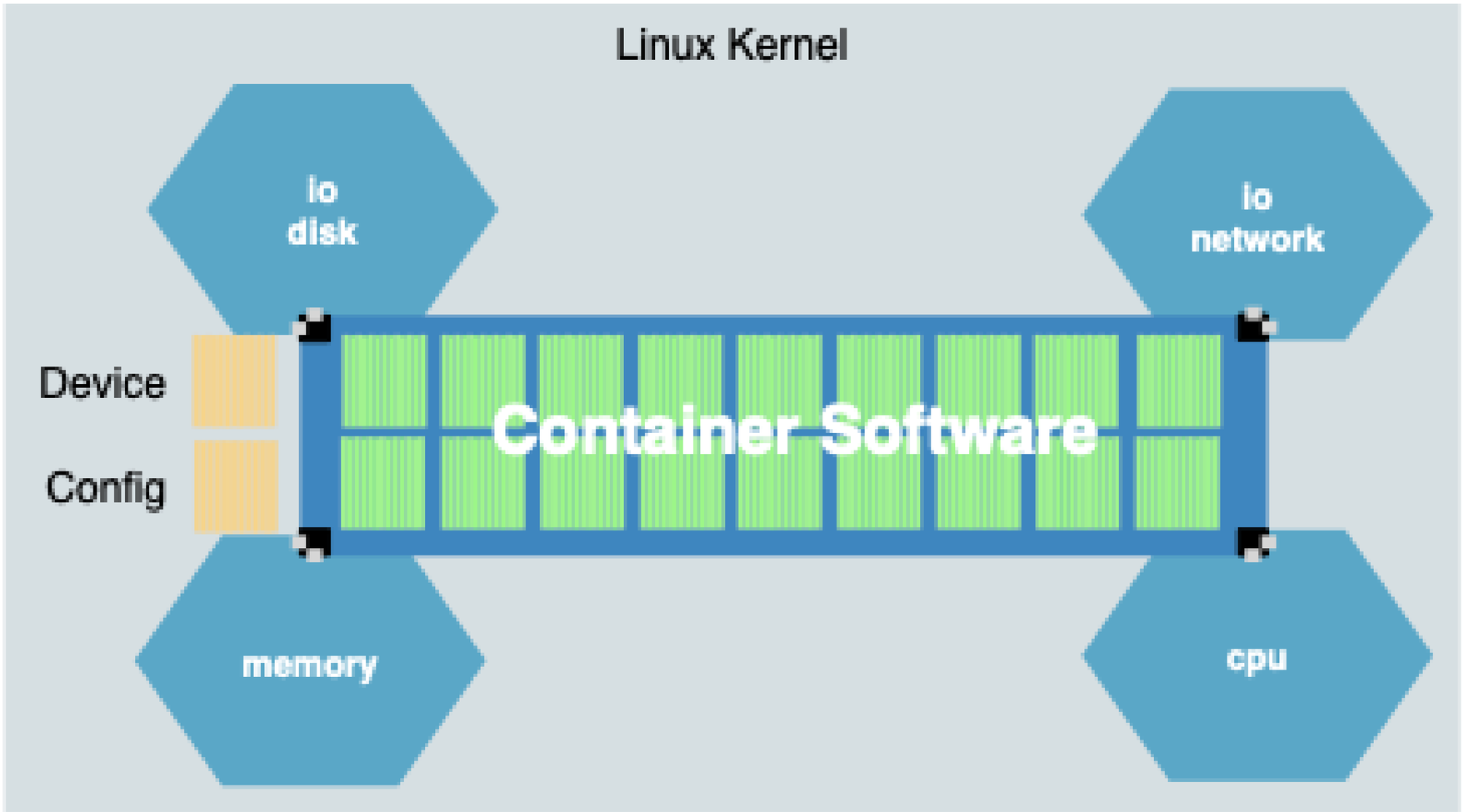




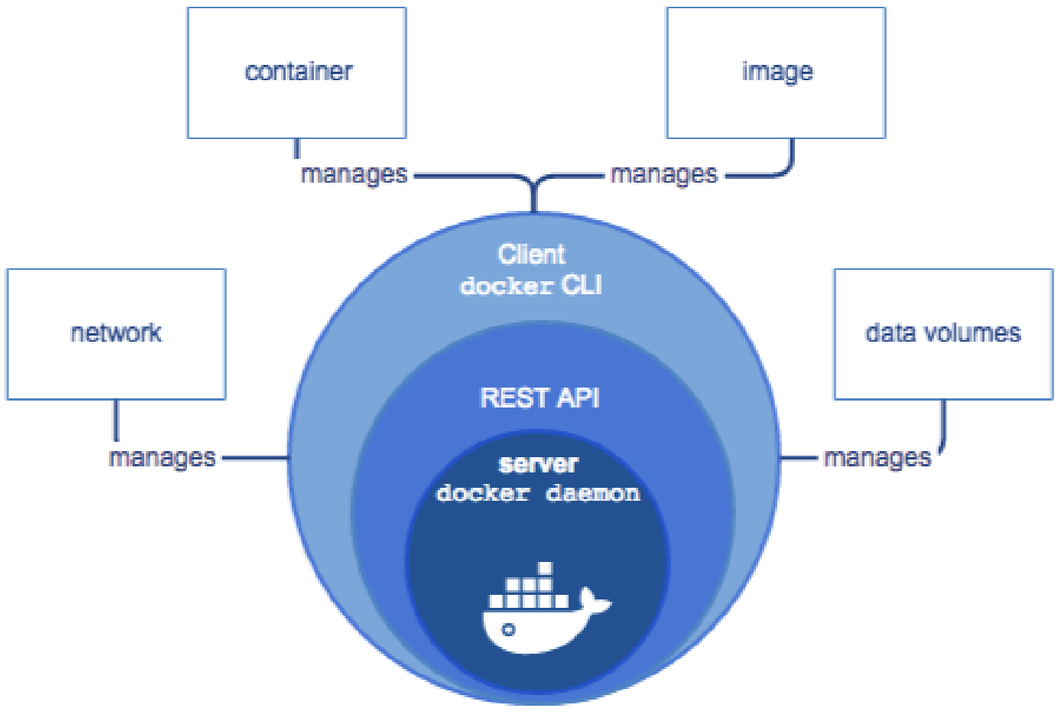
docker run -ti alpine:3.11 /bin/sh




```shell script
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
sudo usermod -aG docker ${USER}
# relogin
```



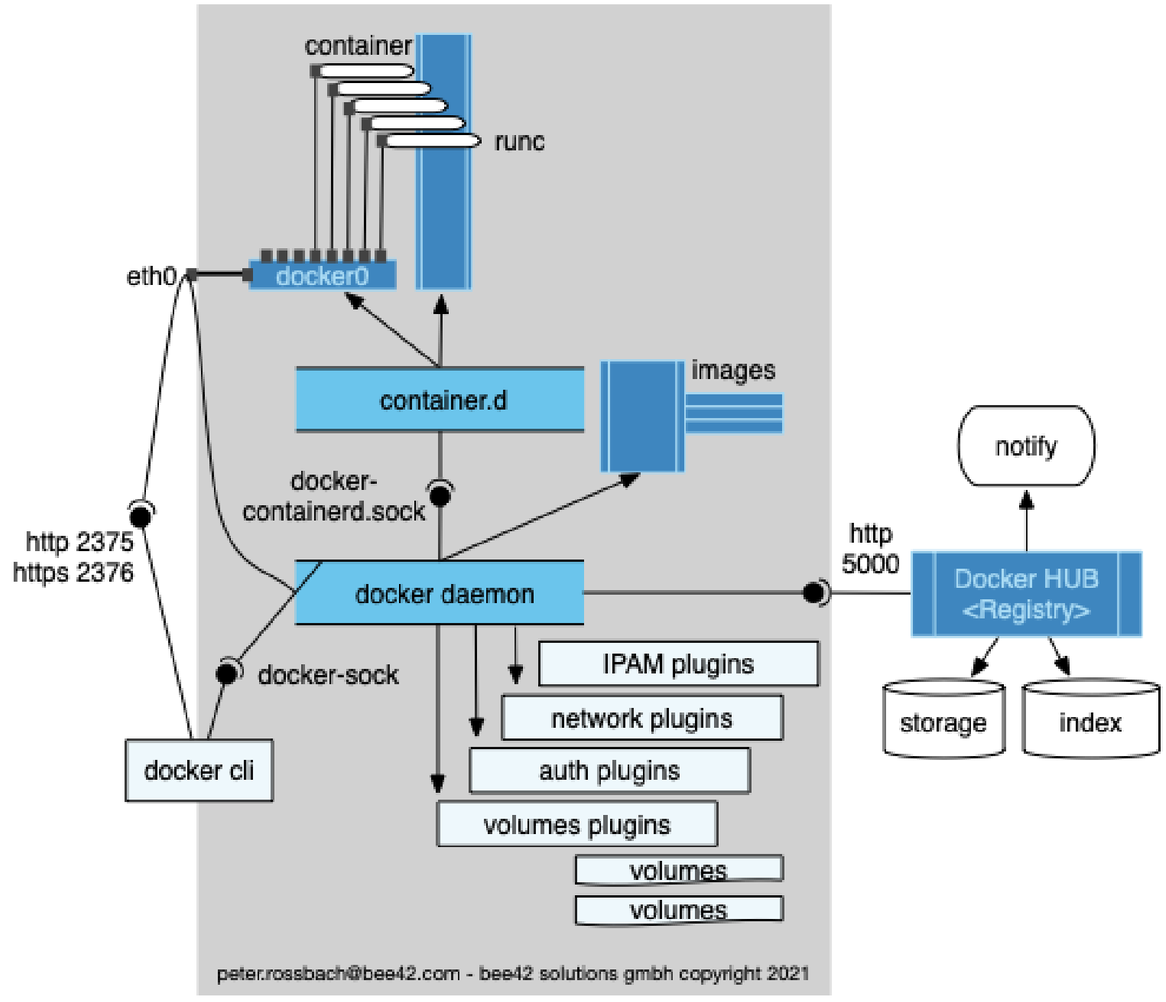
Overview



opencontainers.org

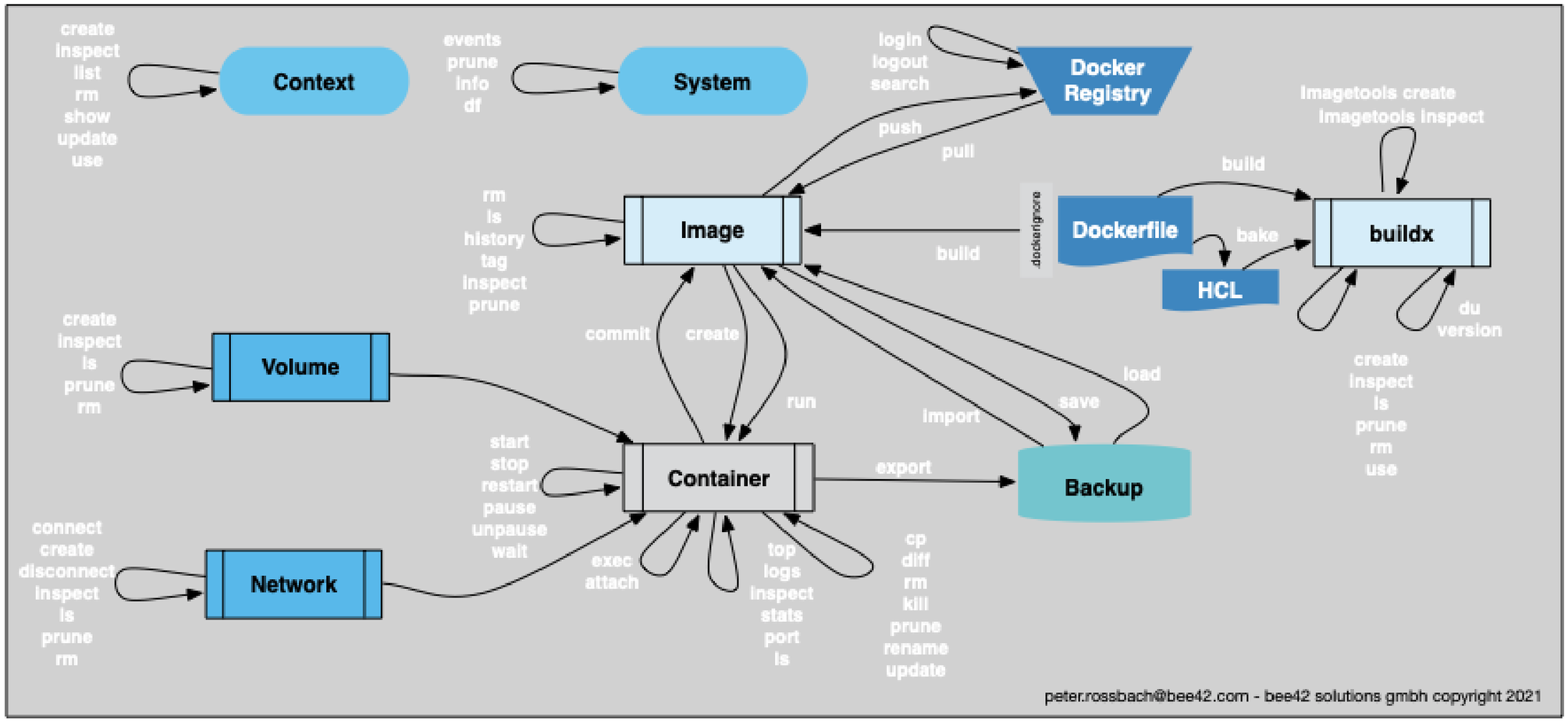
Open Container Initiative - Open Container Initiative

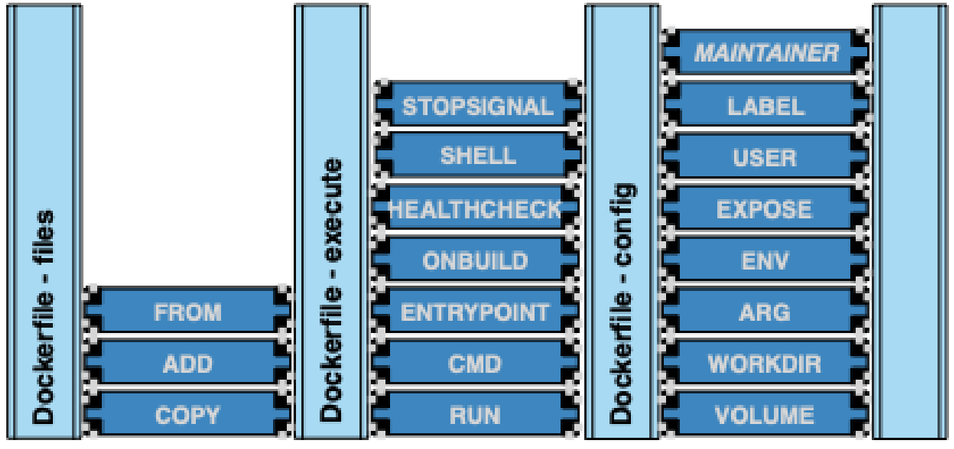
Detail



man7.org

capabilities(7) - Linux manual page





github.com

opencontainers/image-spec

OCI Image Format. Contribute to opencontainers/image-spec development by creating an account on GitHub.

github.com

label-schema/label-schema.org

Static content for Label Schema Spec. Contribute to label-schema/label-schema.org development by creating an account on GitHub.

docker docs

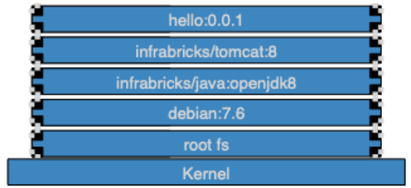
docs.docker.com

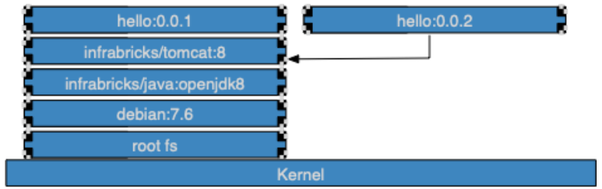
Dockerfile reference

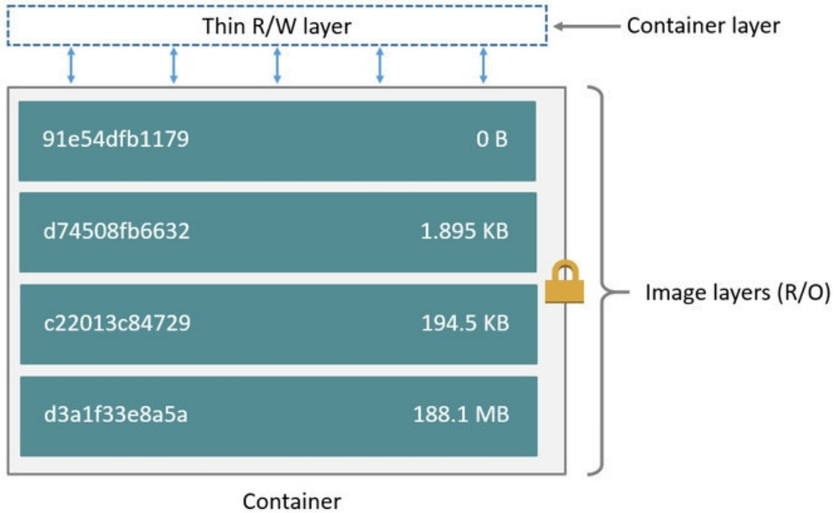
Dockerfiles use a simple DSL which allows you to automate the steps you would normally manually take to create an image.

```
FROM debian

RUN apt update && apt install -y procps iproute2
```

```Dockerfile
cat >Dockerfile <<EOF
FROM debian:10
RUN apt-get update && apt-get install -y apache2
CMD [ "/usr/sbin/apachectl", "-D", "FOREGROUND", "-k", "start"]
EOF
docker build -t myapache .
```

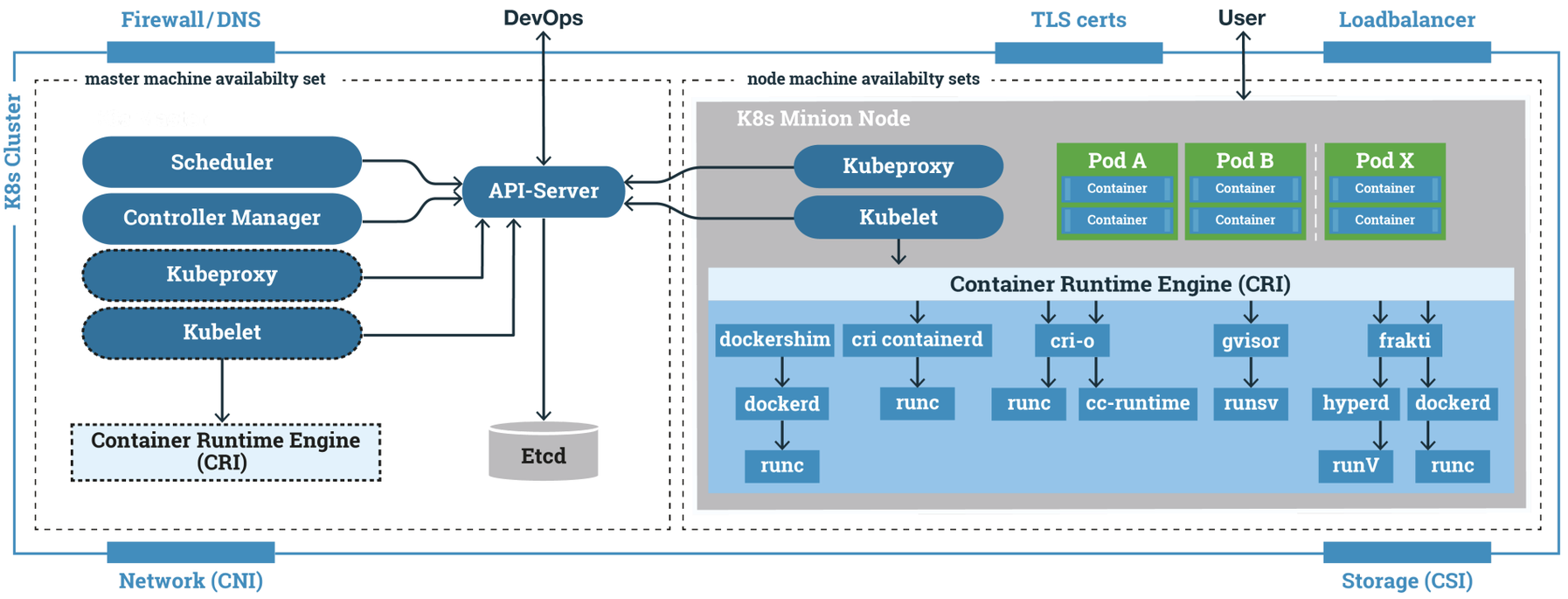


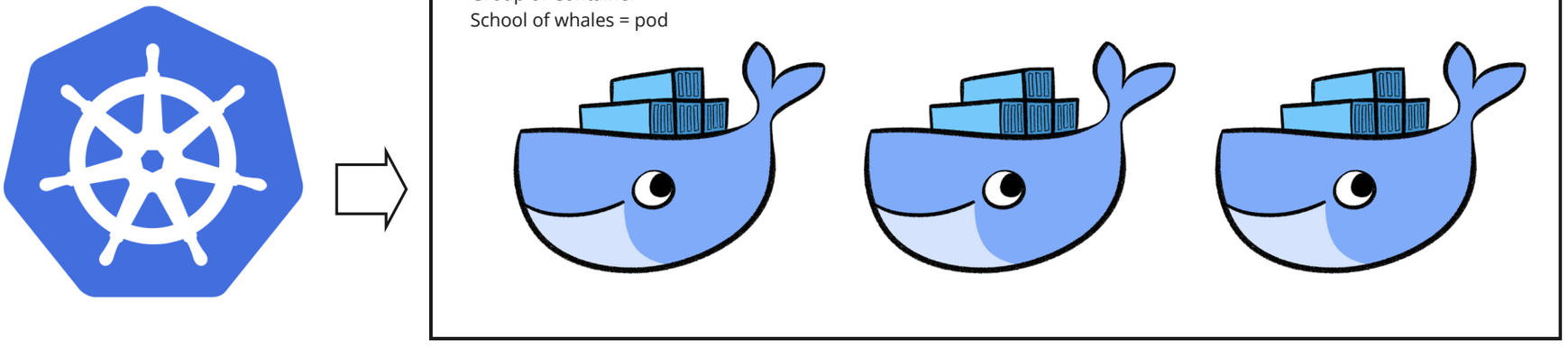




- Container Orchestration
  - Platzierung von jeder Menge Software/Prozess
    - verschiedene Software
    - die gleiche Software skalieren
      - ausfallsicherheit
    - viele Machinen
      - viele Orte
      - Machinen Sets
  - Verwaltung von vielen Maschinen/Computer
  - Management
    - Install
      - Plazierung
      - Wenig Abhängigkeiten beim Start zur Laufzeit
      - Multiple versions & multi stages
    - Update
      - Unterbrechungsfrei
      - verschiedene Versioen
      - Migration
      - Replace
      - Hohe Dynamic
    - Check
      - Monitoring
        - Logging
        - Metrics
        - Tracing
        - Health
      - Failure Detection
        - Automatic Solving
      - Autorecovery
        - Ausfallsicherung
    - Access
      - DNS
      - Routing
        - virtual lan
      - Autoscale
    - remove
      - Rückstandsfrei?
    - Backup/restore
  - as Service
    - Managed Service
      - Database
      - Messaging
      - cache
      - Monitoring Stack
      - apps/api
    - IaaS

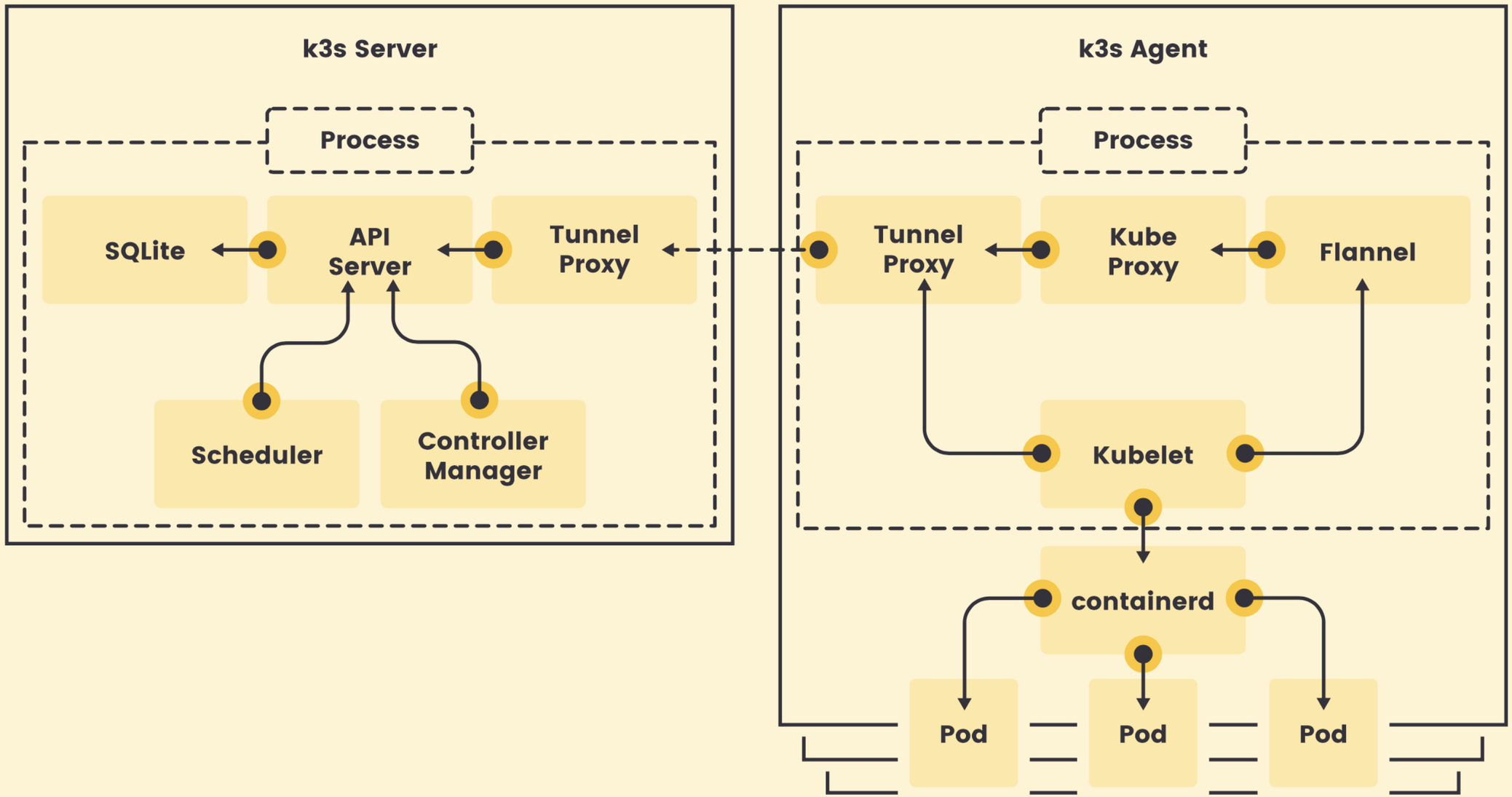
Kubernetes Core Components






```
curl -s https://raw.githubusercontent.com/rancher/k3d/main/install.sh | bash
CLUSTER=cnbc-basic
k3d cluster create $CLUSTER \
 --api-port 8445 \
  -p "8480:80@loadbalancer" \
  -p "8443:443@loadbalancer" \
  --agents=1 --registry-create
```

```
sudo apt-get update && sudo apt-get install -y apt-transport-https
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
echo "deb https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee -a
/etc/apt/sources.list.d/kubernetes.list
sudo apt-get update
sudo apt-get install -y kubectl
```


k3s Server
Process
SQLite
API Server
Tunnel Proxy
Scheduler
Controller Manager
k3s Agent
Process
Tunnel Proxy
Kube Proxy
Flannel
Kubelet
containerd
Pod
Pod
Pod

only X_86:64

X_86:64, arm , arm64

kind.sigs.k8s.io

kind

minikube.sigs.k8s.io

minikube start

minikube is local Kubernetes

k3s.io

K3s: Lightweight Kubernetes

k3d.io

k3d

Little helper to run Rancher Lab's k3s in Docker

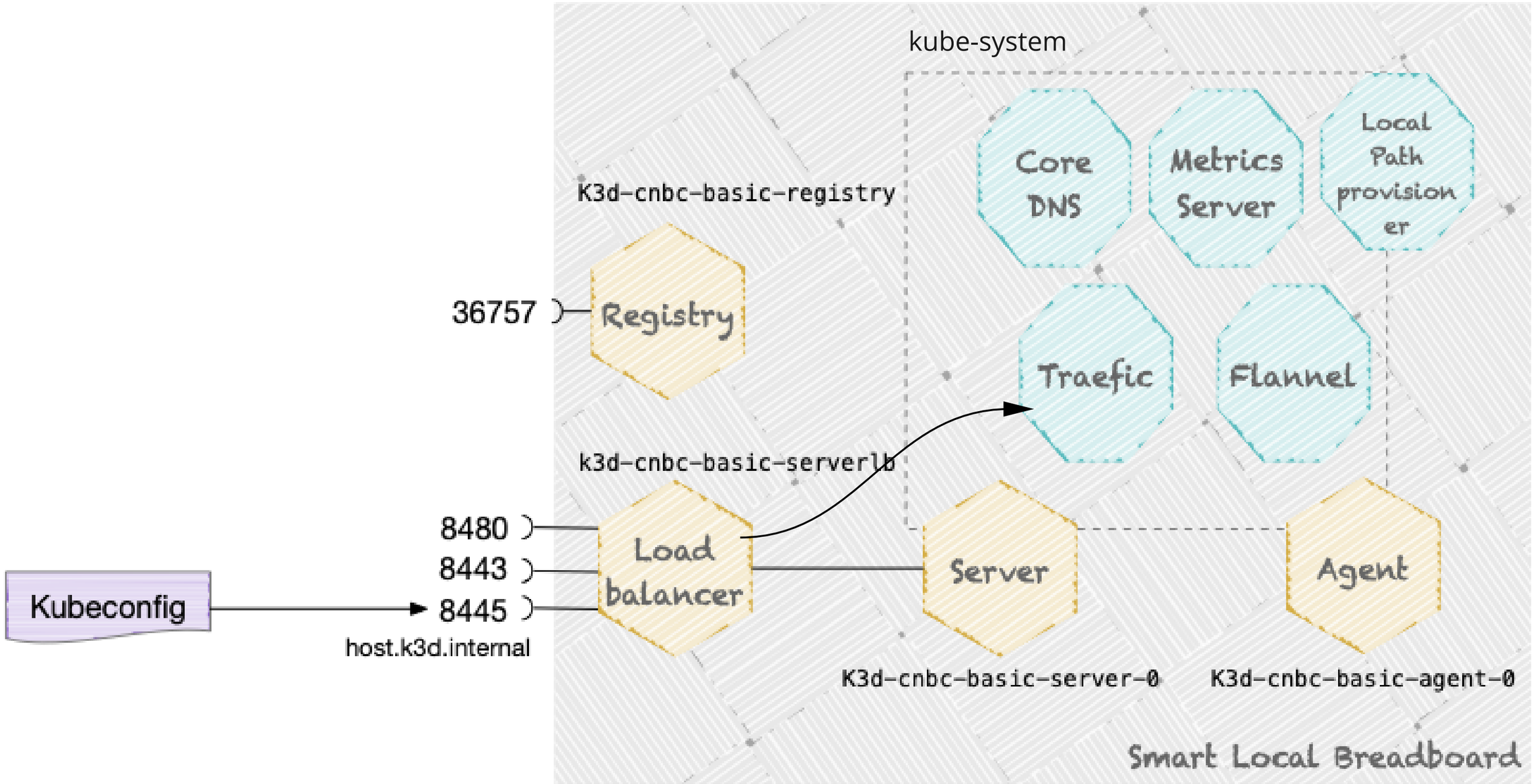
krew.sigs.k8s.io

Krew – kubectl plugin manager

doc.traefik.io

Traefik

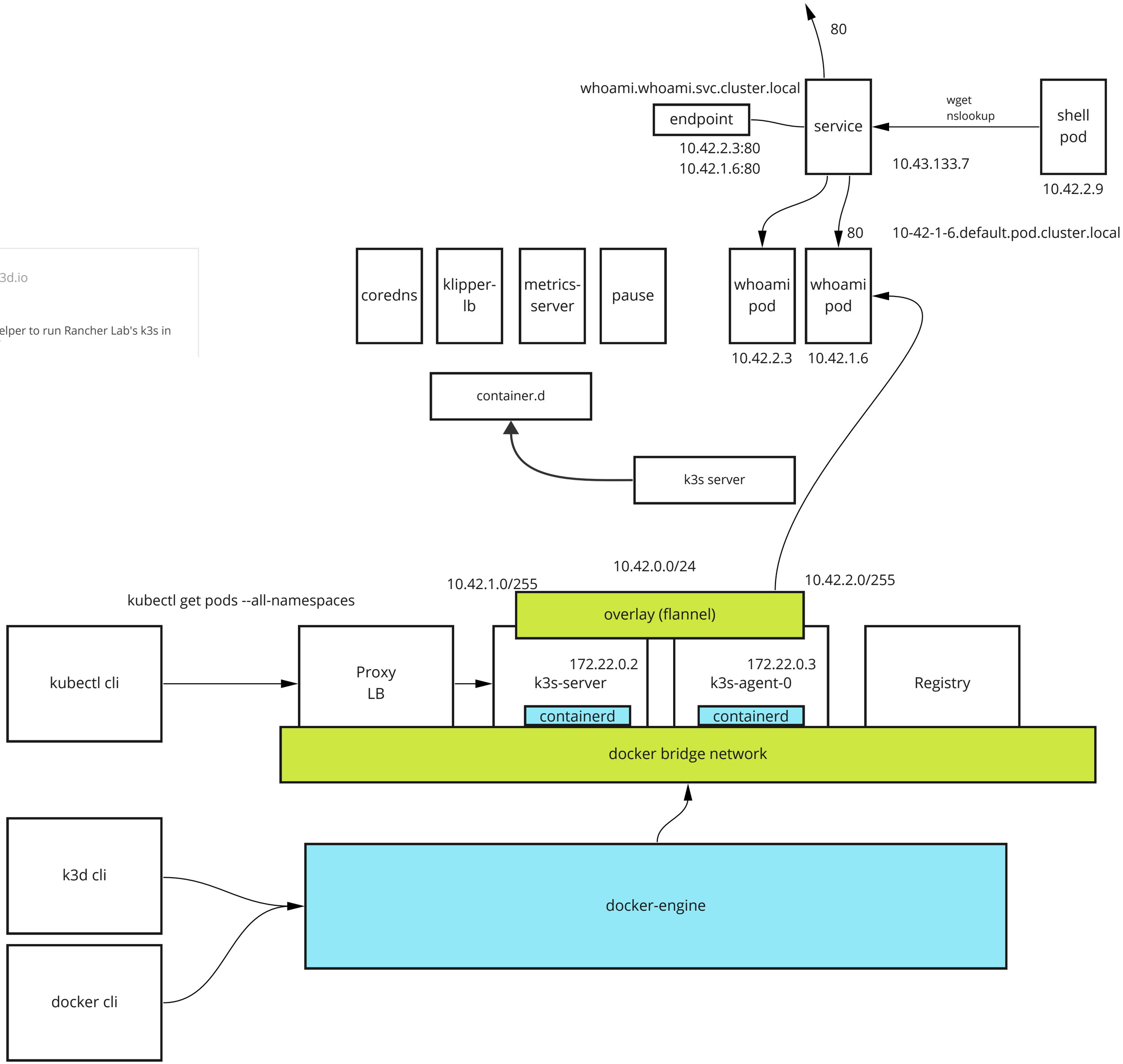
Traefik Documentation

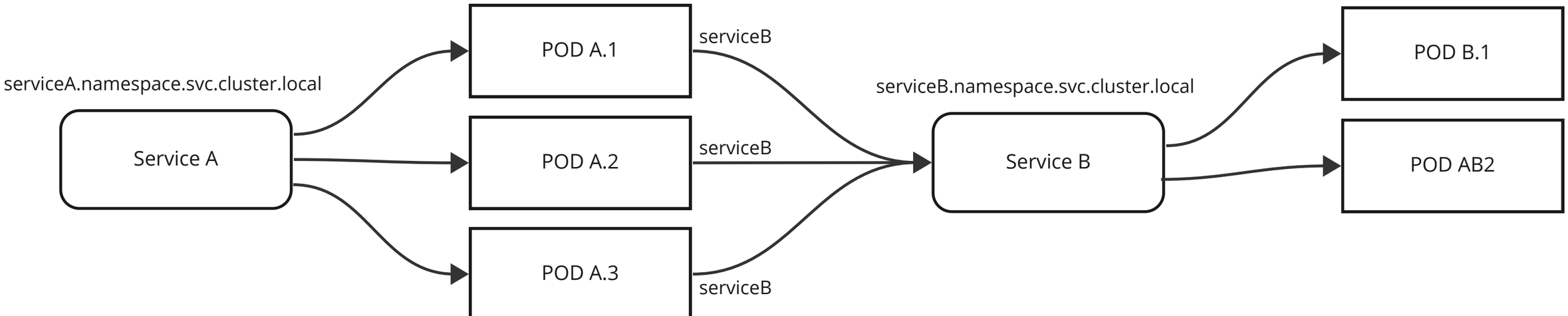

kube-system
K3d-cnbc-basic-registry
Core DNS
Metrics Server
Local Path provisioner
36757
Registry
Traefic
Flannel
k3d-cnbc-basic-serverlb
8480
8443
8445
Load balancer
Server
Agent
Kubeconfig
host.k3d.internal
K3d-cnbc-basic-server-0
K3d-cnbc-basic-agent-0
Smart Local Breadboard

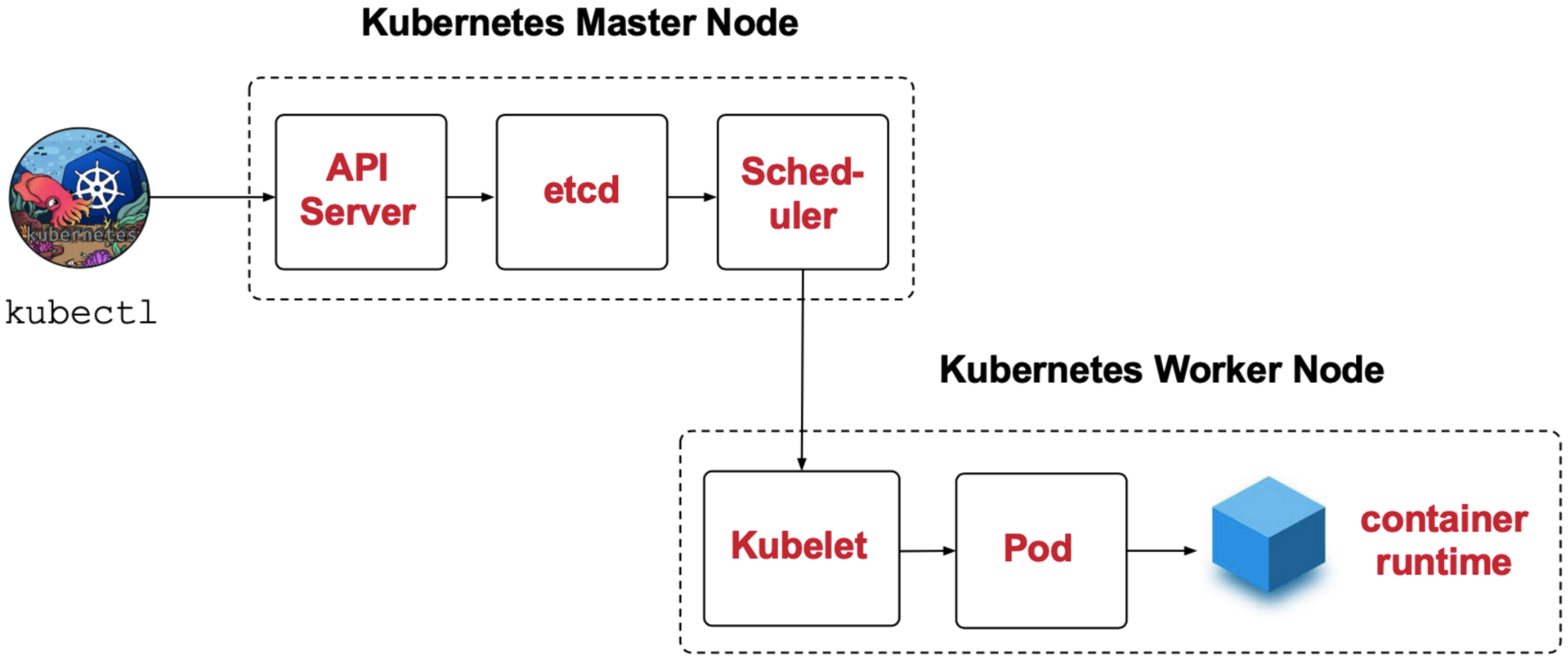
k3d.io

k3d

Little helper to run Rancher Lab's k3s in Docker




serviceA.namespace.svc.cluster.local
Service A
POD A.1
POD A.2
POD A.3
serviceB
serviceB
serviceB
serviceB.namespace.svc.cluster.local
Service B
POD B.1
POD AB2



https://github.com/bmuschko/ckad-crash-course/blob/master/slides.pdf

Kubernetes Object Structure

Kubernetes Object

API Version *v1, apps/v1, ...*

Kind *Pod, Deployment, Quota, ...*

Metadata

Name, Namespace, Labels, ...

Spec

Desired state

Status

Actual state

Object representation in YAML

```
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx
  name: nginx
spec:
  containers:
  - image: nginx
    name: nginx
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Never
status: {}
```

https://github.com/bmuschko/ckad-crash-course/blob/master/slides.pdf

kubernetes

kubernetes.io

Pod Topology Spread Constraints

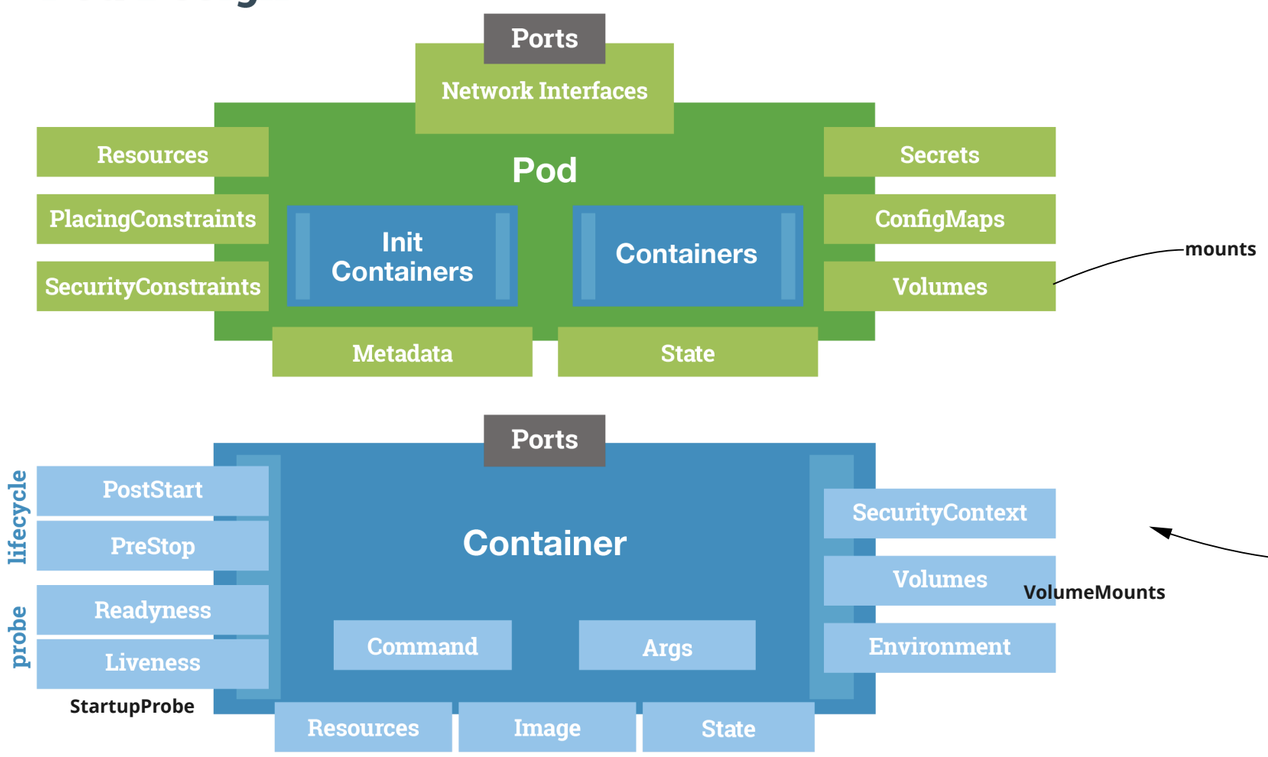
FEATURE STATE: Kubernetes v1.19 [stable] You can use topology spread constraints to control how Pods are spread across your cluster among failure-domains such as regions, zones, nodes, and other user-defined topology domains. This can help to achieve h...

kubernetes

kubernetes.io

Attach Handlers to Container Lifecycle Events

This page shows how to attach handlers to Container lifecycle events. Kubernetes supports the postStart and preStop events. Kubernetes sends the postStart event immediately after a Container is started, and it sends the preStop event immediately before ...



kubernetes

kubernetes.io

Configure a Security Context for a Pod or Container

A security context defines privilege and access control settings for a Pod or Container. Security context settings include, but are not limited to: Discretionary Access Control: Permission to access an object, like a file, is based on user ID (UID) an...

```
apiVersion: v1
kind: Pod
metadata:
 name: lifecycle-demo
spec:
  containers:
  - name: lifecycle-demo-container
    image: nginx
    lifecycle:
      postStart:
        exec:
          command: ["/bin/sh", "-c", "echo Hello from the postStart handler >
/usr/share/message"]
      preStop:
        exec:
          command: ["/bin/sh","-c","nginx -s quit; while killall -0 nginx; do sleep 1; done"]
```

kubernetes

kubernetes.io

Managing Resources for Containers

When you specify a Pod, you can optionally specify how much of each resource a Container needs. The most common resources to specify are CPU and memory (RAM); there are others. When you specify the resource request for Containers in a Pod, the scheduler...

```
containers:
 - name: demo
        image: cnbc-registry:8548/cnbc-teamc/demo/demo:new
        livenessProbe:
          httpGet:
            path: /actuator/health/liveness
            port: 8080
          initialDelaySeconds: 5
          periodSeconds: 3
        readinessProbe:
          httpGet:
            path: /actuator/health/readiness
            port: 8080
          initialDelaySeconds: 5
          periodSeconds: 3
        env:
          - name: SPRING_DATASOURCE_URL
            value: jdbc:postgresql://postgres:5432/demo
```

```
apiVersion: v1
kind: Pod
metadata:
 name: frontend
spec:
  containers:
  - name: app
    image: images.my-company.example/app:v4
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
  - name: log-aggregator
    image: images.my-company.example/log-aggregator:v6
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
```

kubernetes

kubernetes.io

Assign CPU Resources to Containers and Pods

This page shows how to assign a CPU request and a CPU limit to a container. Containers cannot use more CPU than the configured limit. Provided the system has CPU time free, a container is guaranteed to be allocated as much CPU as it requests. Before you...

CPU Limits vs Requests

More details at tinyurl.com/k8s-cpu

All pods have...	CPU limits	No CPU limits
CPU requests	You are guaranteed CPU between the request and limit Excess cpu is unavailable beyond the limit	You are guaranteed your request. Excess CPU is available and not wasted! **
No CPU requests	You are guaranteed the limit, no more, no less* Excess cpu is unavailable	No one is guaranteed any CPU! Wild west.

* This is because Kubernetes automatically sets the request to the limit
** Excess CPU is given to whoever needs it, prioritized by the size of their request

Note: everything here describes only cpu! Memory behaves totally differently because memory is not compressible.

https://home.robusta.dev/blog/stop-using-cpu-limits

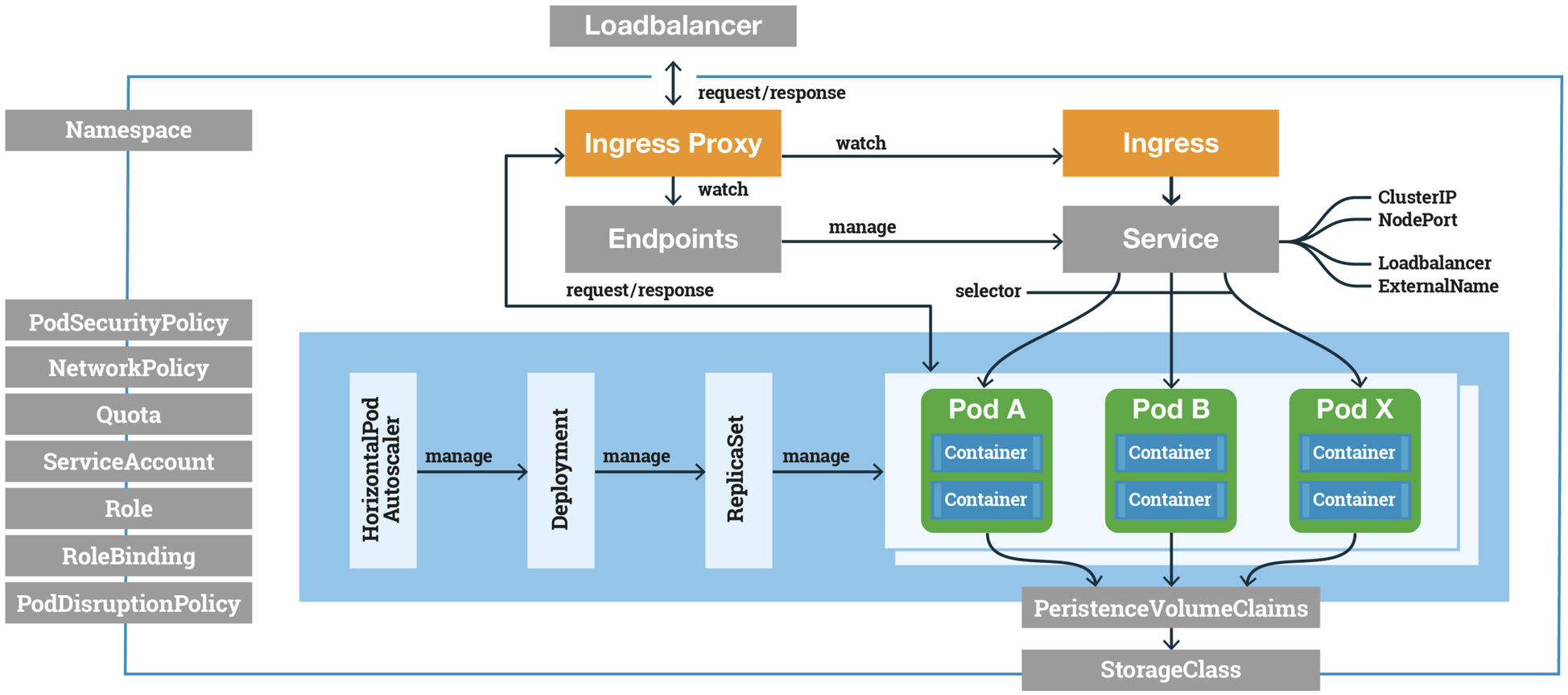
Requests and Limits

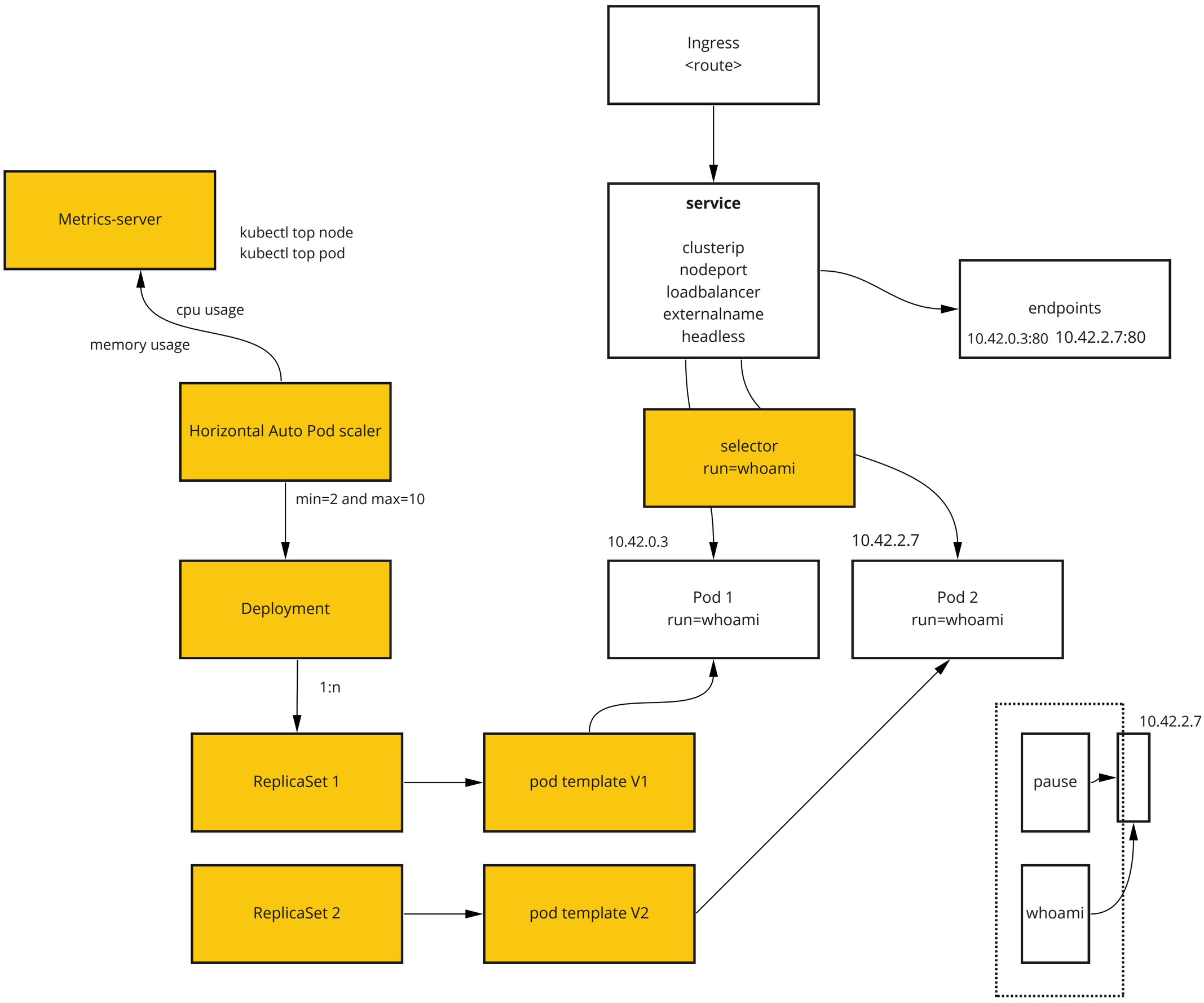
For each resource, containers can specify a resource request and limit, `0 <= request <= Node Allocatable` & `request <= limit <= Infinity`. If a pod is successfully scheduled, the container is guaranteed the amount of resources requested. Scheduling is based on `requests` and not `limits`. The pods and its containers will not be allowed to exceed the specified limit. How the request and limit are enforced depends on whether the resource is compressible or incompressible.

Compressible Resource Guarantees

- For now, we are only supporting CPU.
- Pods are guaranteed to get the amount of CPU they request, they may or may not get additional CPU time (depending on the other jobs running). This isn't fully guaranteed today because cpu isolation is at the container level. Pod level cgroups will be introduced soon to achieve this goal.
- Excess CPU resources will be distributed based on the amount of CPU requested. For example, suppose container A requests for 600 milli CPUs, and container B requests for 300 milli CPUs. Suppose that both containers are trying to use as much CPU as they can. Then the extra 100 milli CPUs will be distributed to A and B in a 2:1 ratio (implementation discussed in later sections).
- Pods will be throttled if they exceed their limit. If limit is unspecified, then the pods can use excess CPU when available.

Kubernetes Service Architecture




Ingress
<route>
service
clusterip
nodeport
loadbalancer
externalname
headless
endpoints
10.42.0.3:80 10.42.2.7:80
selector
run=whoami
10.42.0.3
10.42.2.7
Pod 1
run=whoami
Pod 2
run=whoami
Metrics-server
kubectl top node
kubectl top pod
cpu usage
memory usage
Horizontal Auto Pod scaler
min=2 and max=10
Deployment
1:n
ReplicaSet 1
pod template V1
ReplicaSet 2
pod template V2
10.42.2.7
pause
whoami

apiVersion: v1
kind: Namespace
metadata:
  name: carpool

- Arbeitsschritte
  - carpool review
  - carpool container laufen lassen
  - Anlegen eines ordners
    - anlegen namespace
    - kubectl apply -f namespace.yaml
    - anlegen configmap
    - kubectl apply -f configmap.yaml
    - anlegen pvc
    - kubectl apply -f pvc.yaml
    - kubectl apply -f deployment.yaml
    - kubectl get pods
    - kubectl port-forward service/web 8280:80 &
    - kubectl exec -ti <podname> -c <container name> -- /bin/sh
    - anlegen des services
    - kubectl apply -f service.yaml

kubernetes

kubernetes.io

Use Port Forwarding to Access Applications in a Cluster

This page shows how to use kubectl port-forward to connect to a MongoDB server running in a Kubernetes cluster. This type of connection can be useful for database debugging. Before you begin You need to have a Kubernetes cluster, and the kubectl command...

Self paced Learning

* Create a namespace
* use the standard postgreSQL image to create a pod

* Deployment
 * Start as non root
  * https://hub.docker.com/_/postgres

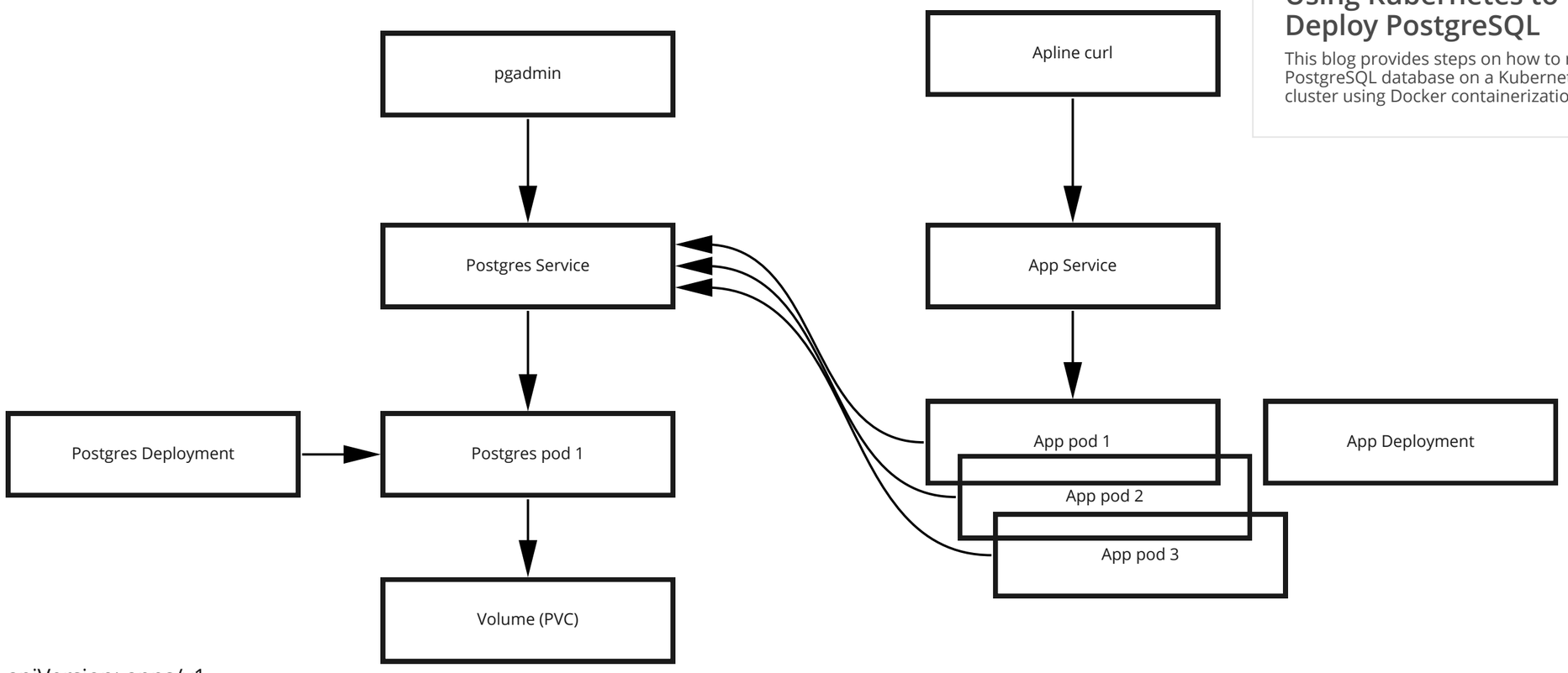
* Add a simple web app to access the database via http!
  * Deployment
  * Scale
  * Use port-forward

Must have

* Push the app image to your company registry!
  * https://docs.docker.com/registry/deploying/
  * Use an ImagePullSecret
    * https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/
  * Nexus?

Sample to push images

```shell script
export REGISTRY_USER=cnbc
export REGISTRY_SHA=xxxx
docker login k3d-$CLUSTER-registry:$PORT
# username
# passwd
docker pull alpine:3.12
docker tag alpine:3.12 k3d-$CLUSTER-registry:$PORT/alpine:3.12
docker push k3d-$CLUSTER-registry:$PORT/alpine:3.12
kubectl run -ti --image k3d-$CLUSTER-registry:$PORT/alpine:3.12 --restart=NEVER -- /bin/sh
```




```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: postgres
spec:
  replicas: 1
  selector:
   matchLabels:
     app: postgres
  template:
   metadata:
     labels:
       app: postgres
   spec:
     containers:
       - name: postgres
         image: postgres:10.4
         imagePullPolicy: "IfNotPresent"
         ports:
           - containerPort: 5432
         envFrom:
           - configMapRef:
               name: postgres-config
         volumeMounts:
           - mountPath: /var/lib/postgresql/data
             name: postgredb
     volumes:
       - name: postgredb
         persistentVolumeClaim:
           claimName: postgres-pv-claim
```

```
kind: Service
apiVersion: v1
metadata:
  name: postgres
  labels:
    app: postgres
spec:
  type: ClusterIP
  ports:
    - port: 5432
      protocol: TCP
      targetPort: 5432
  selector:
    app: postgres
```

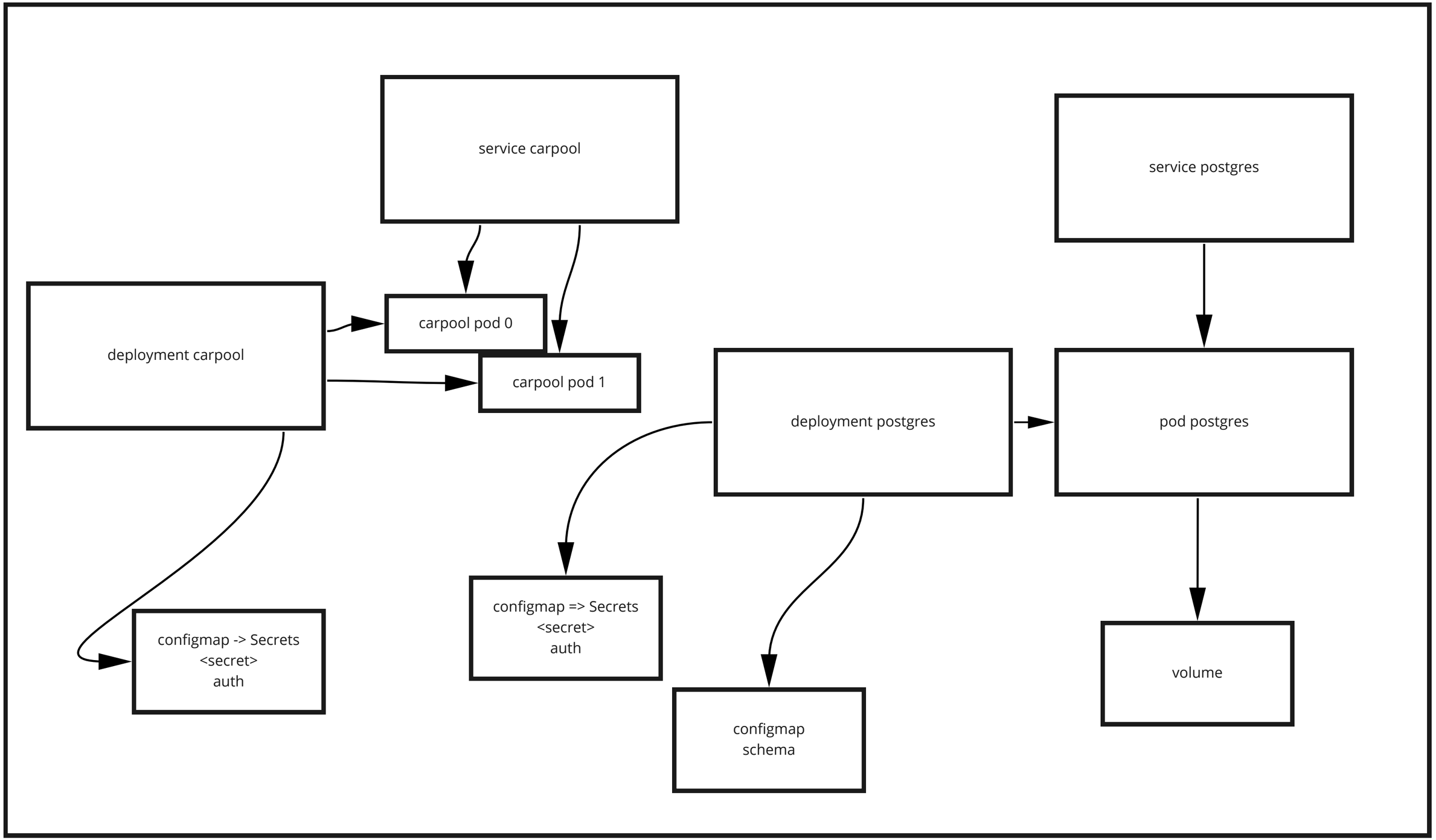
```
apiVersion: v1
kind: ConfigMap
metadata:
  name: postgres-config
  labels:
    app: postgres
data:
  POSTGRES_DB: postgresdb
  POSTGRES_USER: postgres
  POSTGRES_PASSWORD: cnbc
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: postgres-pv-claim
  labels:
    app: postgres
spec:
  storageClassName: local-path
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
  labels:
    app: postgres
data:
  POSTGRES_DB: esentricar
  POSTGRES_PASSWORD: cnbc
  POSTGRES_USER: postgres
  POSTGRES_HOST: postgres
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: carpool
 labels:
   app: carpool
spec:
  replicas: 1
  selector:
    matchLabels:
      app: carpool
  template:
    metadata:
      labels:
        app: carpool
    spec:
      containers:
      - name: carpool
        image: cnbc-
registry:5000/bee42/carpool:0.2.0
        imagePullPolicy: Always
        env:
          - name: FLASK_APP
            value: python_rest
        envFrom:
          - configMapRef:
              name: app-config
```

namespace carpool



ingress
<http proxy>

Service

selector:
app: web

replicaset v1

replicaset v2

pod v1.1

pod v1.2

pod v1.3

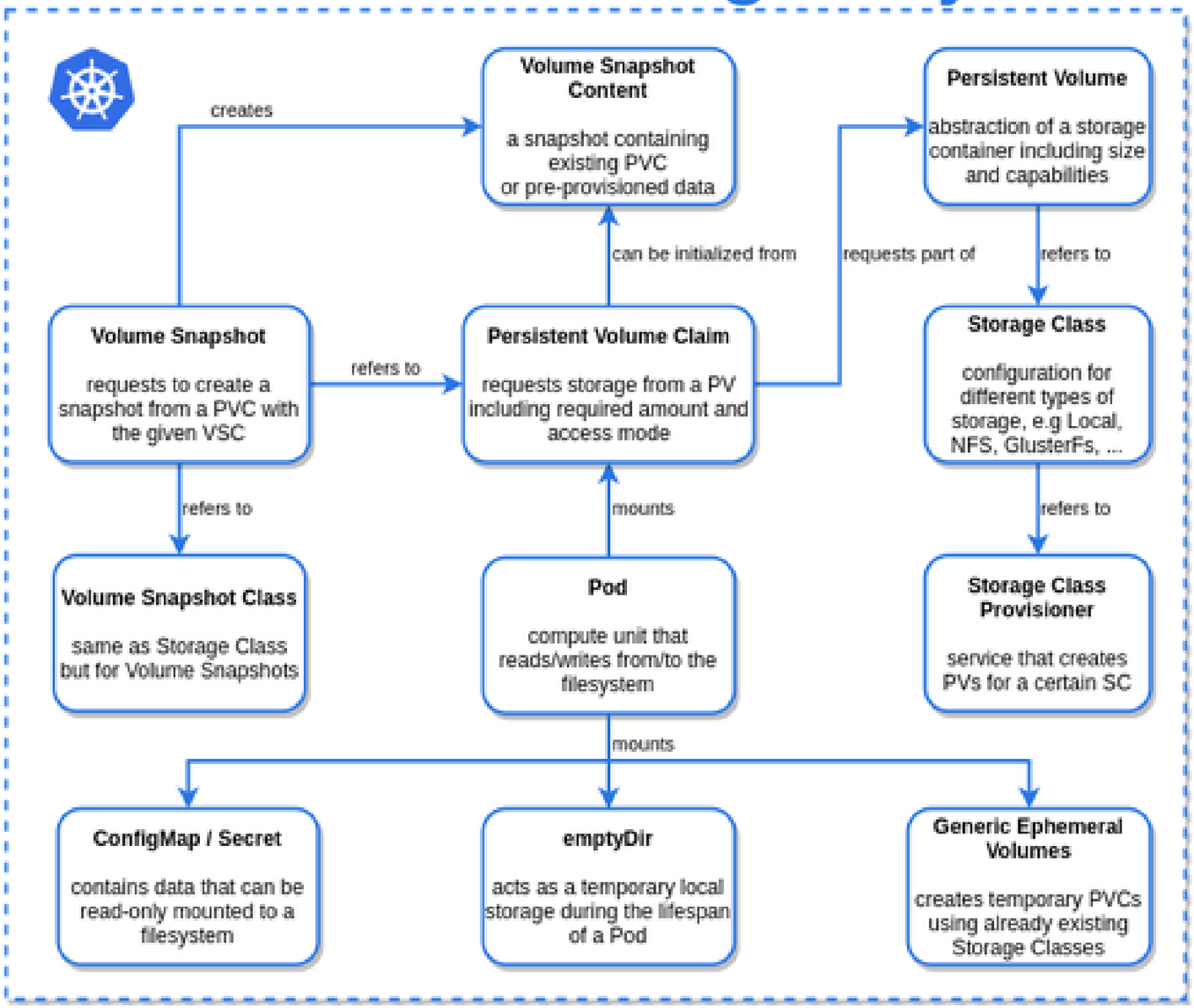
pod v2.1

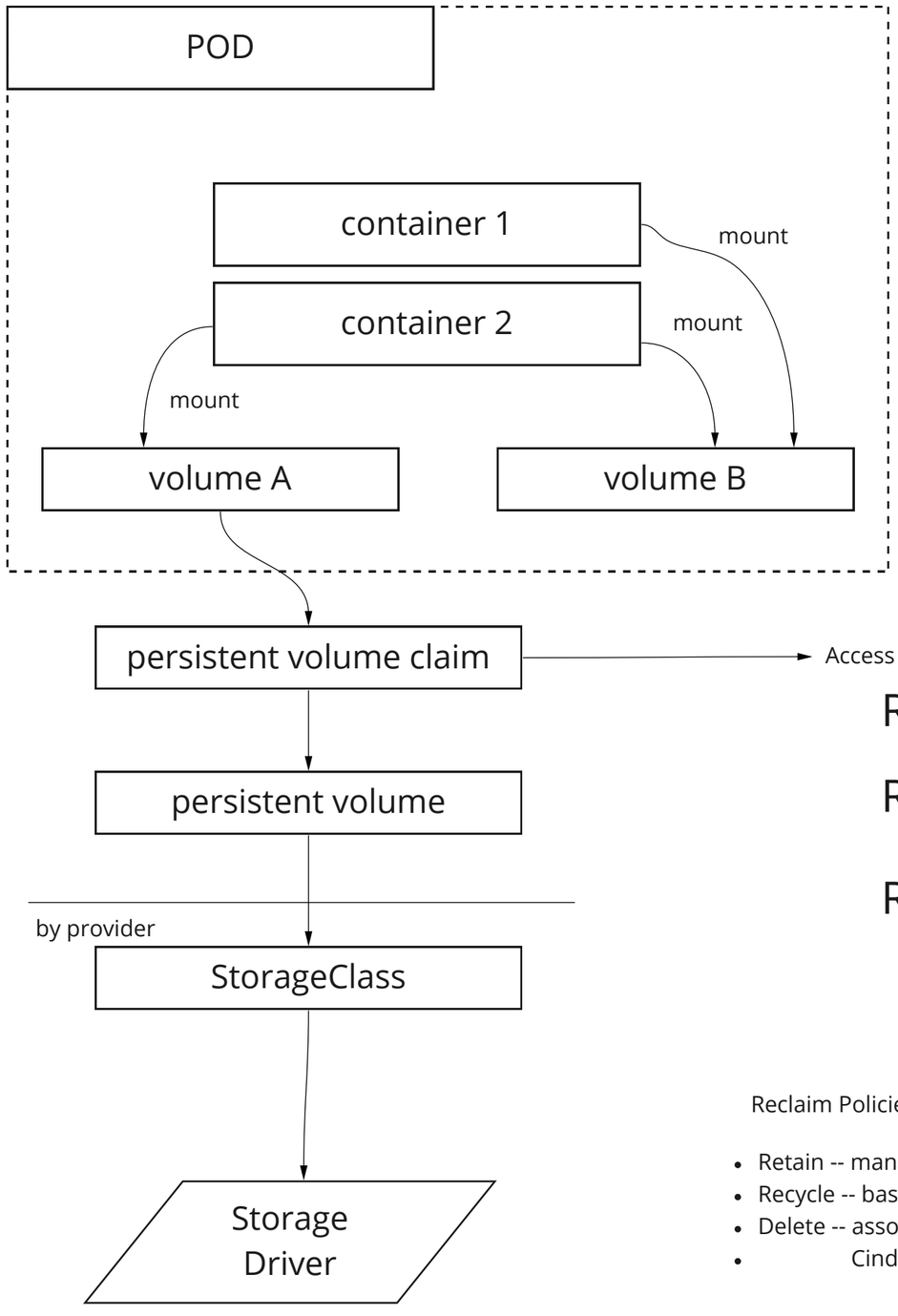
pod v2.2

pod v2.3

container V2

Kubernetes Storage Objects





Access Modes

READWRITEONCE

READWRITEMANY

READONLYMANY

kubernetes.io

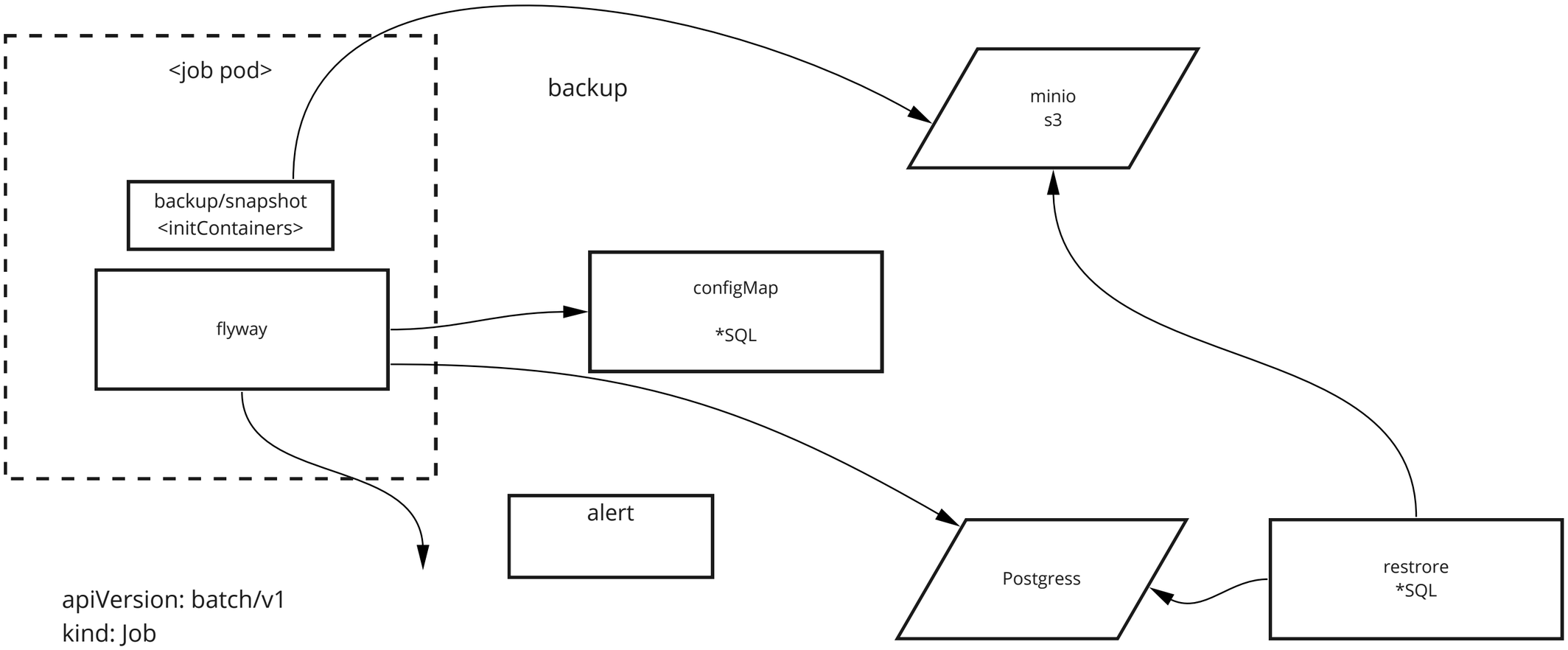
Persistent Volumes

This document describes the current state of persistent volumes in Kubernetes. Familiarity with volumes is suggested. Introduction Managing storage is a distinct problem from managing compute instances. The PersistentVolume subsystem provides an API for...

- GCEPersistentDisk
- AWSElasticBlockStore
- AzureFile
- AzureDisk
- CSI
- FC (Fibre Channel)
- FlexVolume
- Flocker
- NFS
- iSCSI
- RBD (Ceph Block Device)
- CephFS
- Cinder (OpenStack block storage)
- Glusterfs
- VsphereVolume
- Quobyte Volumes
- HostPath (Single node testing only -- local storage is not supported in any way and WILL NOT WORK in a multi-node cluster)
- Portworx Volumes
- ScaleIO Volumes
- StorageOS

Reclaim Policies

- Retain -- manual reclamation
- Recycle -- basic scrub (rm -rf /thevolume/*)
- Delete -- associated storage asset such as AWS EBS, GCE PD, Azure Disk, or OpenStack
- Cinder volume is deleted




```
apiVersion: batch/v1
kind: Job
metadata:
 name: pi
spec:
  template:
    spec:
      containers:
      - name: pi
        image: perl
        command: ["perl",  "-Mbignum=bpi", "-wle", "print bpi(2000)"]
      restartPolicy: Never
backoffLimit: 4
```

Beispiel eines initContainers

```
apiVersion: v1
kind: Pod
metadata:
 name: myapp-pod
  labels:
    app: myapp
spec:
  containers:
  - name: myapp-container
    image: busybox:1.28
    command: ['sh', '-c', 'echo The app is running! && sleep 3600']
  initContainers:
  - name: init-myservice
    image: busybox:1.28
    command: ['sh', '-c', "until nslookup myservice.$(cat
/var/run/secrets/kubernetes.io/serviceaccount/namespace).svc.cluster.local; do echo
waiting for myservice; sleep 2; done"]
  - name: init-mydb
    image: busybox:1.28
    command: ['sh', '-c', "until nslookup mydb.$(cat
/var/run/secrets/kubernetes.io/serviceaccount/namespace).svc.cluster.local; do echo
waiting for mydb; sleep 2; done"]
```

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod-map
spec:
  initContainers:
  - name: init-myservice
    image: busybox:1.32.0
    command: ['sh', '-c', "echo \"hello world\" >/usr/share/nginx/html/index.html"]
    volumeMounts:
      - name: data
        mountPath: /usr/share/nginx/html
  containers:
  - name: nginx
    image: nginx:latest
    ports:
    - name: http
      containerPort: 8080
    securityContext:
      allowPrivilegeEscalation: true
      runAsUser: 1000
    volumeMounts:
    - name: nginx-config
      mountPath: /etc/nginx/nginx.conf
      subPath: nginx.conf
    - name: data
      mountPath: /usr/share/nginx/html
    - name: cache-volume
      mountPath: /var/cache/nginx
    - name: run-volume
      mountPath: /var/run
    - name: logs
      mountPath: /var/logs/nginx
  volumes:
  - name: nginx-config
    configMap:
      name: nginx-configmap
  - name: data
    persistentVolumeClaim:
      claimName: persistent-data
  - name: cache-volume
    emptyDir: {}
  - name: run-volume
    emptyDir: {}
  - name: logs
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: persistent-data
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: local-path
```

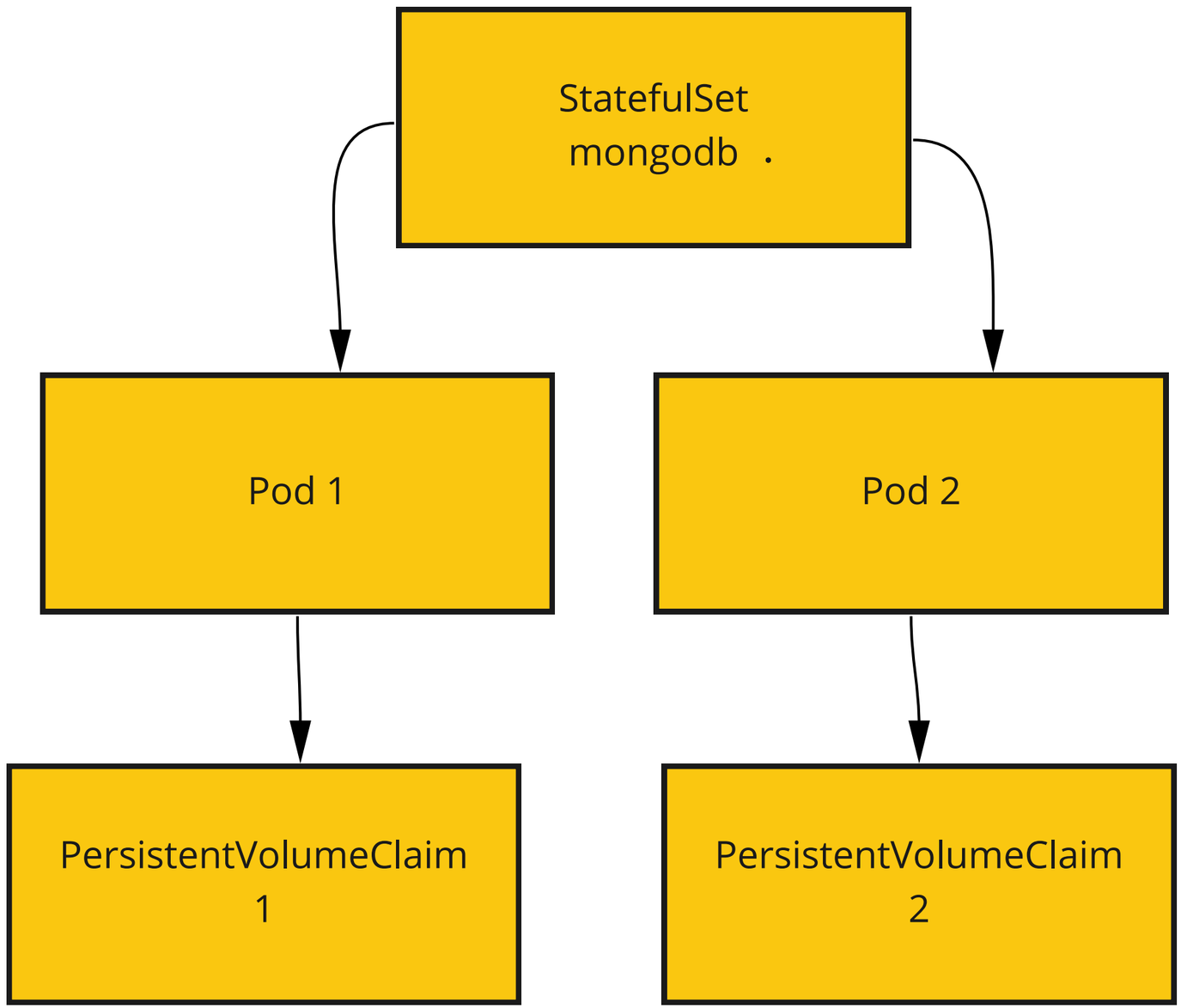
```
apiVersion: v1
data:
  nginx.conf: |2

    worker_processes  auto;

    error_log  /var/log/nginx/error.log notice;
    pid        /var/run/nginx.pid;


    events {
        worker_connections  1024;
    }


    http {
        include       /etc/nginx/mime.types;
        default_type  application/octet-stream;

        log_format  main  '$remote_addr - $remote_user [$time_local] "$requ
                          '$status $body_bytes_sent "$http_referer" '
                          '"$http_user_agent" "$http_x_forwarded_for"';

        access_log  /var/log/nginx/access.log  main;

        sendfile        on;
        #tcp_nopush     on;

        keepalive_timeout  65;

        #gzip  on;

        include /etc/nginx/conf.d/*.conf;
    }
kind: ConfigMap
metadata:
  creationTimestamp: null
  name: nginx-configmap
```

MongoDB StatefulSet
in
Kubernetes

deeptiman.medium.com

MongoDB StatefulSet in Kubernetes

In this article, We will see how we can deploy a MongoDB replica set into the Kubernetes as a StatefulSet. There is a brief understanding...

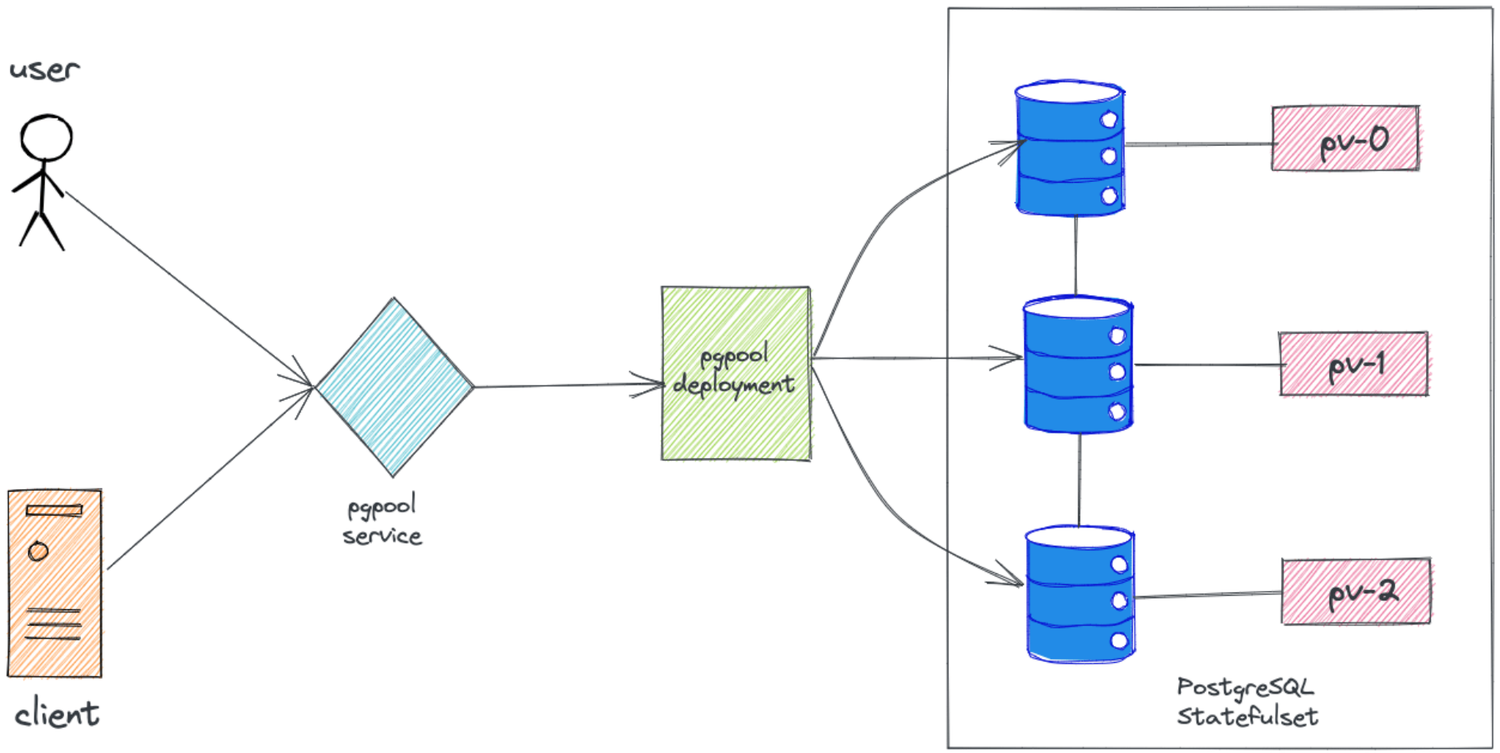


velero.io

Velero

krew.sigs.k8s.io

Krew – kubectl plugin manager



https://devopscube.com/deploy-postgresql-statefulset/

https://github.com/bitnami/charts/tree/master/bitnami/postgresql-ha

https://github.com/bitnami/bitnami-docker-postgresql-repmgr

The Twelve Factors

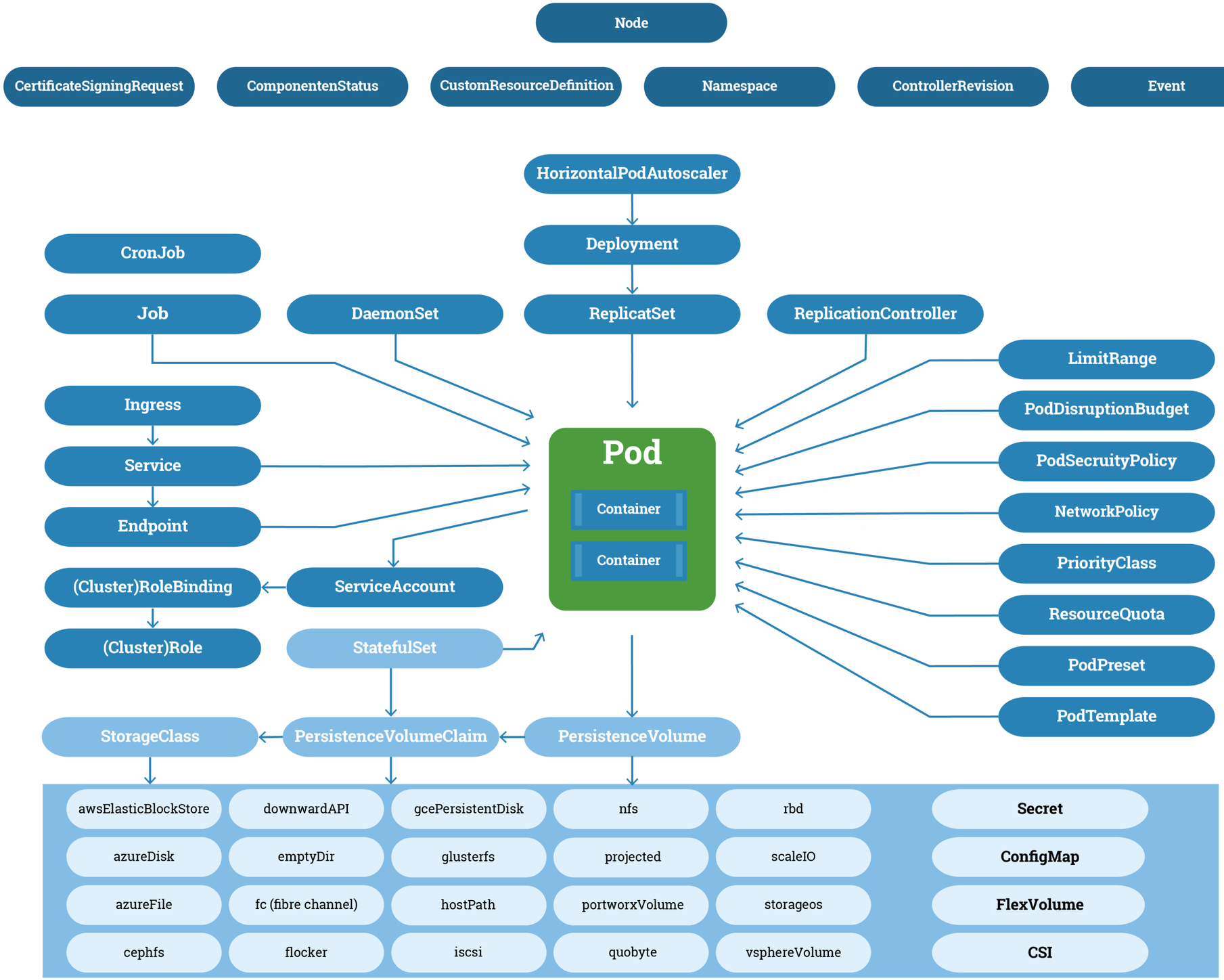
* I. Codebase
  * One codebase tracked in revision control, many deploys
* II. Dependencies
  + Explicitly declare and isolate dependencies
* III. Config
  * Store config in the environment
* IV. Backing services
  * Treat backing services as attached resources
* V. Build, release, run
  * Strictly separate build and run stages
* VI. Processes
  * Execute the app as one or more stateless processes
* VII. Port binding
  * Export services via port binding
* VIII. Concurrency
  * Scale out via the process model
* IX. Disposability
  * Maximize robustness with fast startup and graceful shutdown
* X. Dev/prod parity
  * Keep development, staging, and production as similar as possible
* XI. Logs
  * Treat logs as event streams
* XII. Admin processes
  * Run admin/management tasks as one-off processes

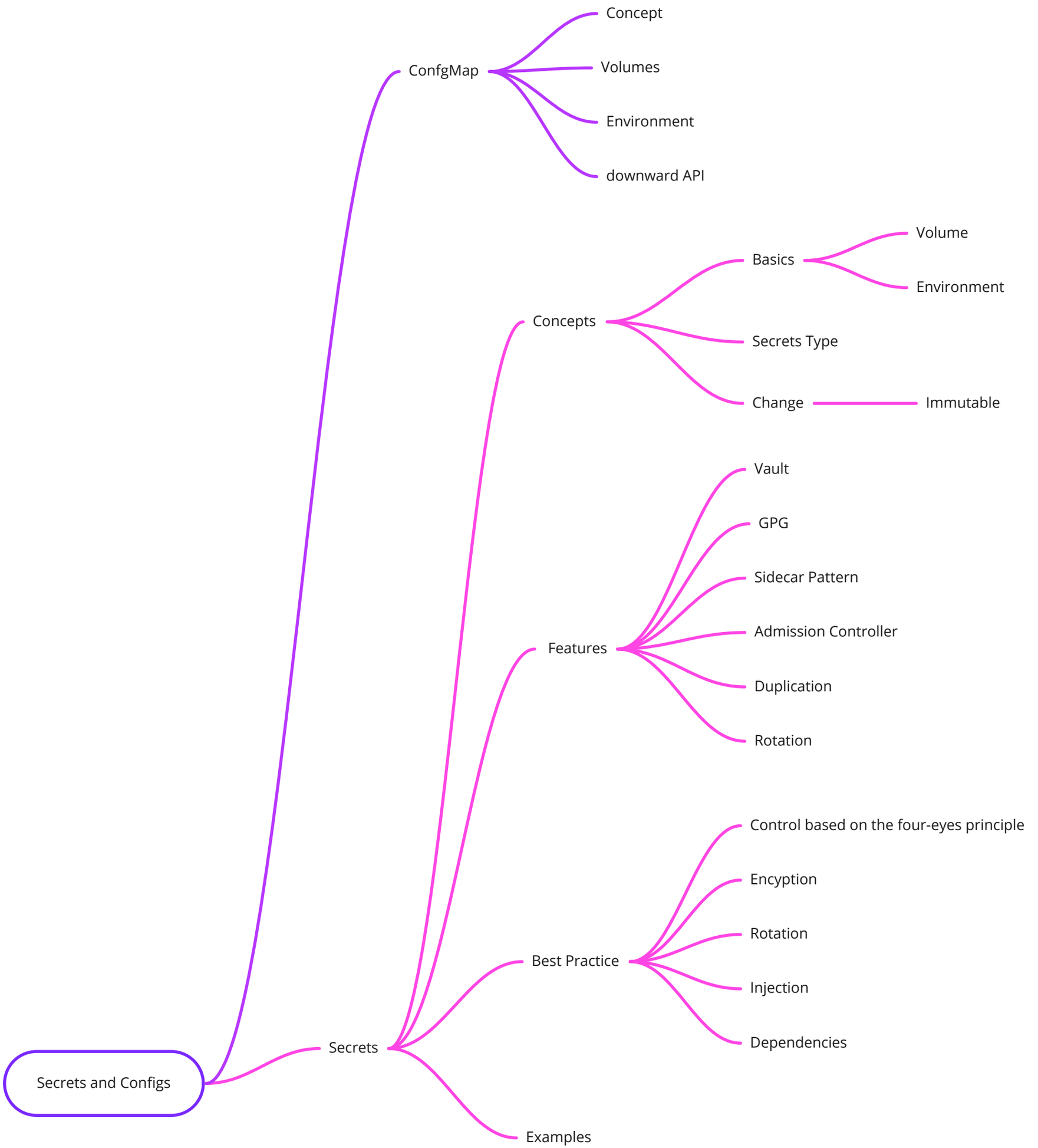
12factor.net

The Twelve-Factor App

A methodology for building modern, scalable, maintainable software-as-a-service apps.

Kubernetes Resources




Secrets and Configs
ConfgMap
Concept
Volumes
Environment
downward API
Secrets
Concepts
Basics
Volume
Environment
Secrets Type
Change
Immutable
Features
Vault
GPG
Sidecar Pattern
Admission Controller
Duplication
Rotation
Best Practice
Control based on the four-eyes principle
Encyption
Rotation
Injection
Dependencies
Examples

ConfigMap

```
apiVersion: v1
kind: ConfigMap
metadata:
 name: game-demo
data:
  # property-like keys; each key maps to a simple value
  player_initial_lives: "3"
  ui_properties_file_name: "user-interface.properties"

  # file-like keys
  game.properties: |
    enemy.types=aliens,monsters
    player.maximum-lives=5
  user-interface.properties: |
    color.good=purple
    color.bad=yellow
    allow.textmode=true
```

Env

```
apiVersion: v1
kind: Pod
metadata:
 name: configmap-demo-pod
spec:
  containers:
    - name: demo
      image: alpine
      command: ["sleep", "3600"]
      env:
        # Define the environment variable
        - name: PLAYER_INITIAL_LIVES # Notice that the case is different here
                                     # from the key name in the ConfigMap.
          valueFrom:
            configMapKeyRef:
              name: game-demo           # The ConfigMap this value comes from.
              key: player_initial_lives # The key to fetch.
        - name: UI_PROPERTIES_FILE_NAME
          valueFrom:
            configMapKeyRef:
              name: game-demo
              key: ui_properties_file_name
      volumeMounts:
      - name: config
        mountPath: "/config"
        readOnly: true
  volumes:
    # You set volumes at the Pod level, then mount them into containers inside that Pod
    - name: config
      configMap:
        # Provide the name of the ConfigMap you want to mount.
        name: game-demo
        # An array of keys from the ConfigMap to create as files
        items:
        - key: "game.properties"
          path: "game.properties"
        - key: "user-interface.properties"
          path: "user-interface.properties"
```

mount

Volume

```
apiVersion: v1
kind: ConfigMap
metadata:
 ...
data:
  ...
immutable: true
```

```yaml
apiVersion: v1
kind: Secret
metadata:
 name: secret-sa-sample
  annotations:
    kubernetes.io/service-account.name: "sa-name"
type: kubernetes.io/service-account-token
data:
  # You can include additional key value pairs as you do with Opaque Secrets
  extra: YmFyCg==
```

Builtin Type	Usage
Opaque	arbitrary user-defined data
kubernetes.io/service-account-token	service account token
kubernetes.io/dockercfg serialized	~/.dockercfg file
kubernetes.io/dockerconfigjson	serialized ~/.docker/config.json file
kubernetes.io/basic-auth	credentials for basic authentication
kubernetes.io/ssh-auth	credentials for SSH authentication
kubernetes.io/tls	data for a TLS client or server
bootstrap.kubernetes.io/token	bootstrap token data

```yaml
apiVersion: v1
kind: Pod
metadata:
 name: mypod
spec:
  containers:
  - name: mypod
    image: redis
    volumeMounts:
    - name: foo
      mountPath: "/etc/foo"
      readOnly: true
  volumes:
  - name: foo
    secret:
      secretName: mysecret
```

```yaml
apiVersion: v1
kind: Secret
metadata:
 name: secret-basic-auth
type: kubernetes.io/basic-auth
stringData:
  username: admin
  password: t0p-Secret
```

```
apiVersion: v1
kind: Pod
metadata:
 name: kubernetes-downwardapi-volume-example
  labels:
    zone: us-est-coast
    cluster: test-cluster1
    rack: rack-22
  annotations:
    build: two
    builder: john-doe
spec:
  containers:
    - name: client-container
      image: k8s.gcr.io/busybox
      command: ["sh", "-c"]
      args:
      - while true; do
        if [[ -e /etc/podinfo/labels ]]; then
          echo -en '\n\n'; cat /etc/podinfo/labels; fi;
        if [[ -e /etc/podinfo/annotations ]]; then
          echo -en '\n\n'; cat /etc/podinfo/annotations; fi;
        sleep 5;
      done;
    volumeMounts:
      - name: podinfo
        mountPath: /etc/podinfo
volumes:
  - name: podinfo
    downwardAPI:
      items:
        - path: "labels"
          fieldRef:
            fieldPath: metadata.labels
        - path: "annotations"
          fieldRef:
            fieldPath: metadata.annotations
```

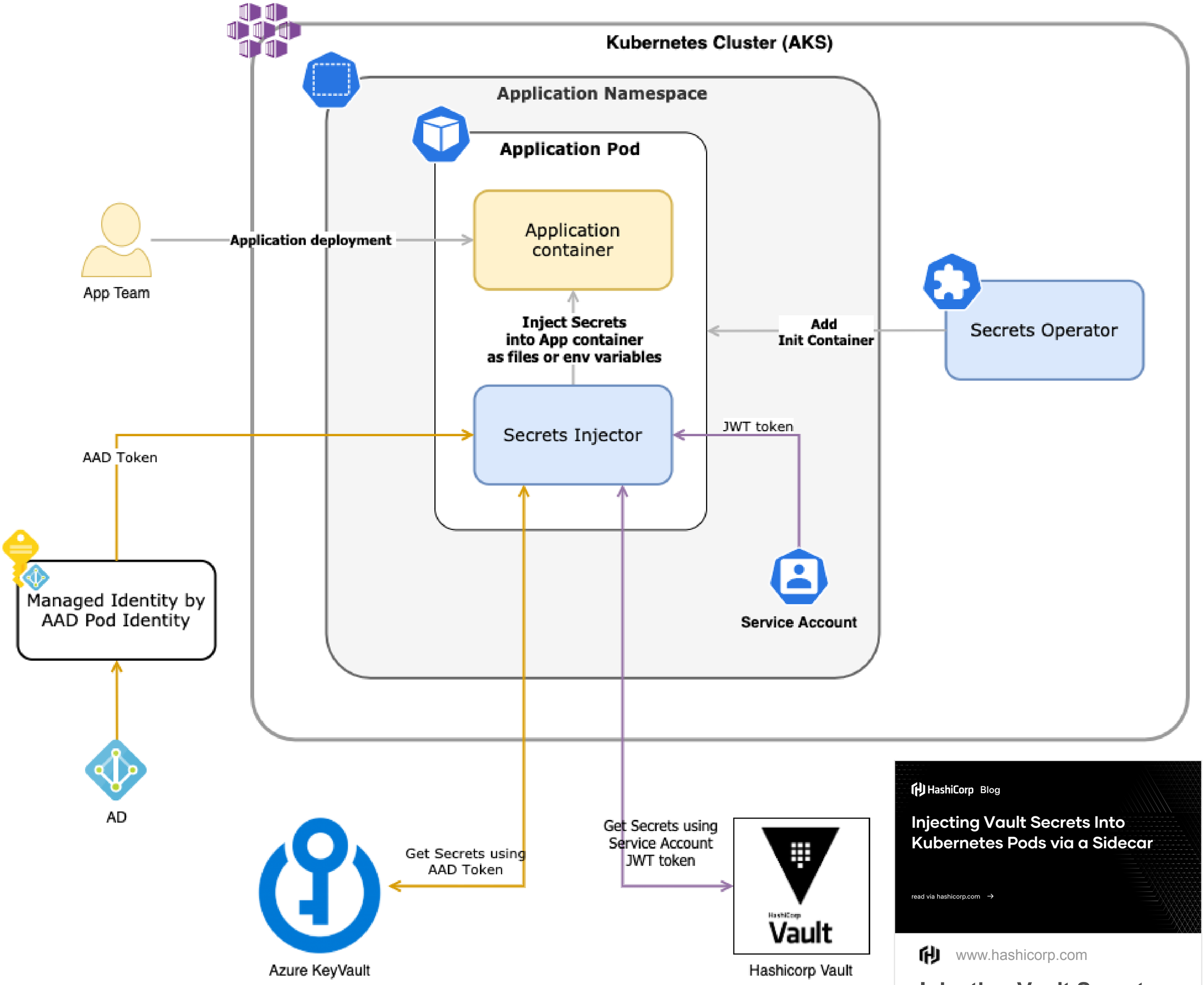
```
apiVersion: v1
kind: Pod
metadata:
 name: dapi-envars-fieldref
spec:
  containers:
    - name: test-container
      image: k8s.gcr.io/busybox
      command: [ "sh", "-c"]
      args:
      - while true; do
          echo -en '\n';
          printenv MY_NODE_NAME MY_POD_NAME MY_POD_NAMESPACE;
          printenv MY_POD_IP MY_POD_SERVICE_ACCOUNT;
          sleep 10;
        done;
      env:
        - name: MY_NODE_NAME
          valueFrom:
            fieldRef:
              fieldPath: spec.nodeName
        - name: MY_POD_NAME
          valueFrom:
            fieldRef:
              fieldPath: metadata.name
        - name: MY_POD_NAMESPACE
          valueFrom:
            fieldRef:
              fieldPath: metadata.namespace
        - name: MY_POD_IP
          valueFrom:
            fieldRef:
              fieldPath: status.podIP
        - name: MY_POD_SERVICE_ACCOUNT
          valueFrom:
            fieldRef:
              fieldPath: spec.serviceAccountName
  restartPolicy: Never
```

kubernetes

kubernetes.io

Expose Pod Information to Containers Through Environment Variables

This page shows how a Pod can use environment variables to expose information about itself to Containers running in the Pod. Environment variables can expose Pod fields and Container fields. Before you begin You need to have a Kubernetes cluster, and th...



www.hashicorp.com

Injecting Vault Secrets Into Kubernetes Pods via a Sidecar

We are excited to announce a new Kubernetes integration that allows applications with no native HashiCorp Vault logic built-in to leverage static and dynamic secrets sourced from Vault.

github.com

mozilla/sops

Simple and flexible tool for managing secrets. Contribute to mozilla/sops development by creating an account on GitHub.

github.com

mittwald/kubernetes-replicator

Kubernetes controller for synchronizing secrets & config maps across namespaces - mittwald/kubernetes-replicator

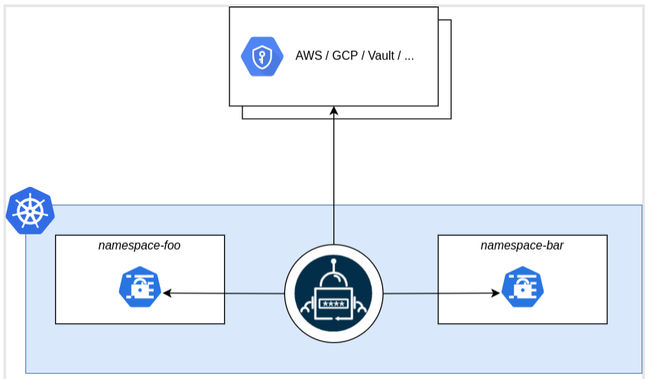
manpages.ubuntu.com

Ubuntu Manpage: git-crypt - transparent file encryption in Git

github.com

isindir/sops-secrets-operator

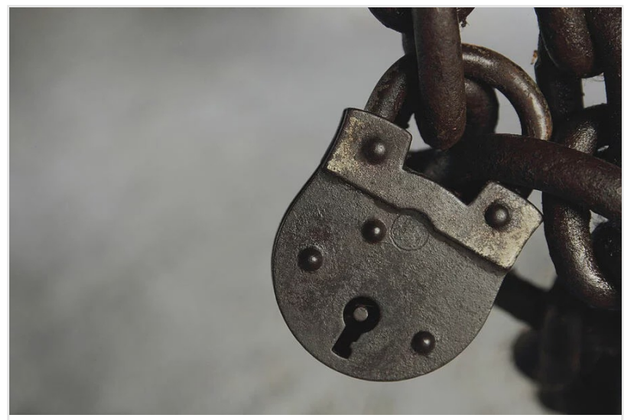
Kubernetes SOPS secrets operator. Contribute to isindir/sops-secrets-operator development by creating an account on GitHub.



external-secrets.io

External Secrets Operator

External Secrets Operator is a Kubernetes operator that integrates external secret management systems like AWS Secrets Manager, HashiCorp Vault, Google Secrets Manager, Azure Key Vault and many more. The operator reads information from external APIs and...



medium.com

Encrypting Helm Secrets

Secrets management has always been a weakness for Helm. Further investigation provided us with couple of solutions and one of the possible...

github.com

bitnami-labs/sealed-secrets

A Kubernetes controller and tool for one-way encrypted Secrets - bitnami-labs/sealed-secrets

helm plugin install https://github.com/futuresimple/helm-secrets

GIT REPO

```
apiVersion: bitnami.com/v1alpha1
kind: SealedSecret
metadata:
 name: mysecret
  namespace: mynamespace
spec:
  encryptedData:
    foo: AgBy3i4OJSWK+PiTySYZZA9rO43cGDEq.....
```

generated

Kubernetes API

```
apiVersion: v1
kind: Secret
metadata:
 name: mysecret
  namespace: mynamespace
data:
  foo: YmFy
```

helm.sh

Chart Template Guide

Helm - The Kubernetes Package Manager.

helm.sh

Helm

Helm - The Kubernetes Package Manager.

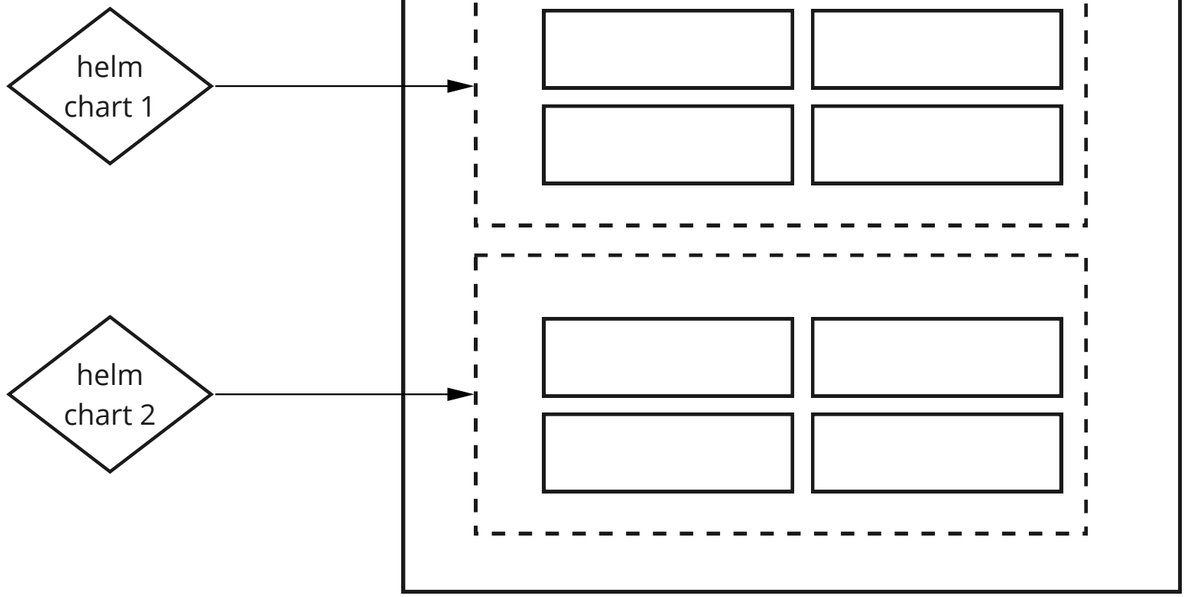
ArtifactHUB

Find, install and publish Kubernetes packages

artifacthub.io

Artifact Hub

Find, install and publish Kubernetes packages



kustomize.io

Kustomize - Kubernetes native configuration management

jsonnet.org

Jsonnet - The Data Templating Language

A powerful DSL for elegant description of JSON data.

github.com

roboll/helmfile

Deploy Kubernetes Helm Charts. Contribute to roboll/helmfile development by creating an account on GitHub.

github.com

ksonnet/ksonnet

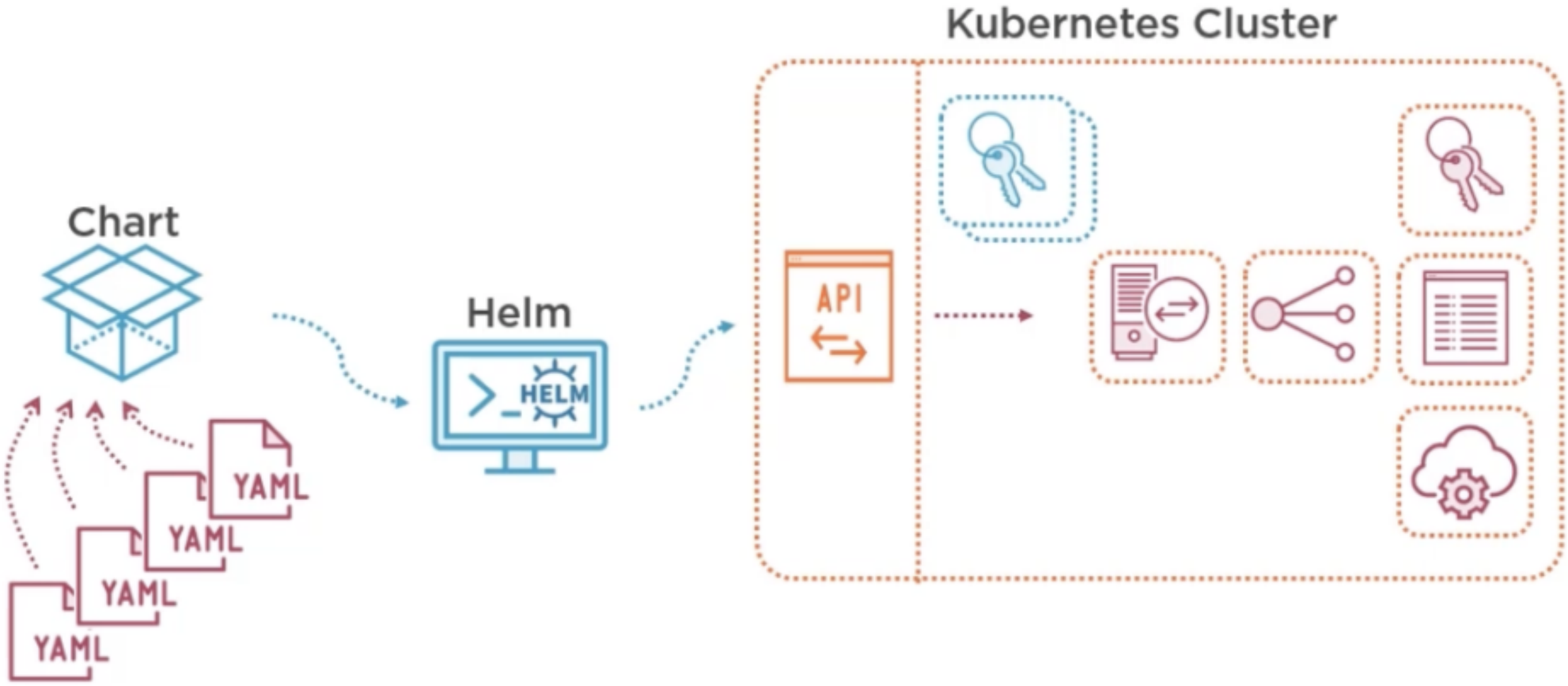
A CLI-supported framework that streamlines writing and deployment of Kubernetes configurations to multiple clusters. - ksonnet/ksonnet

www.ansible.com

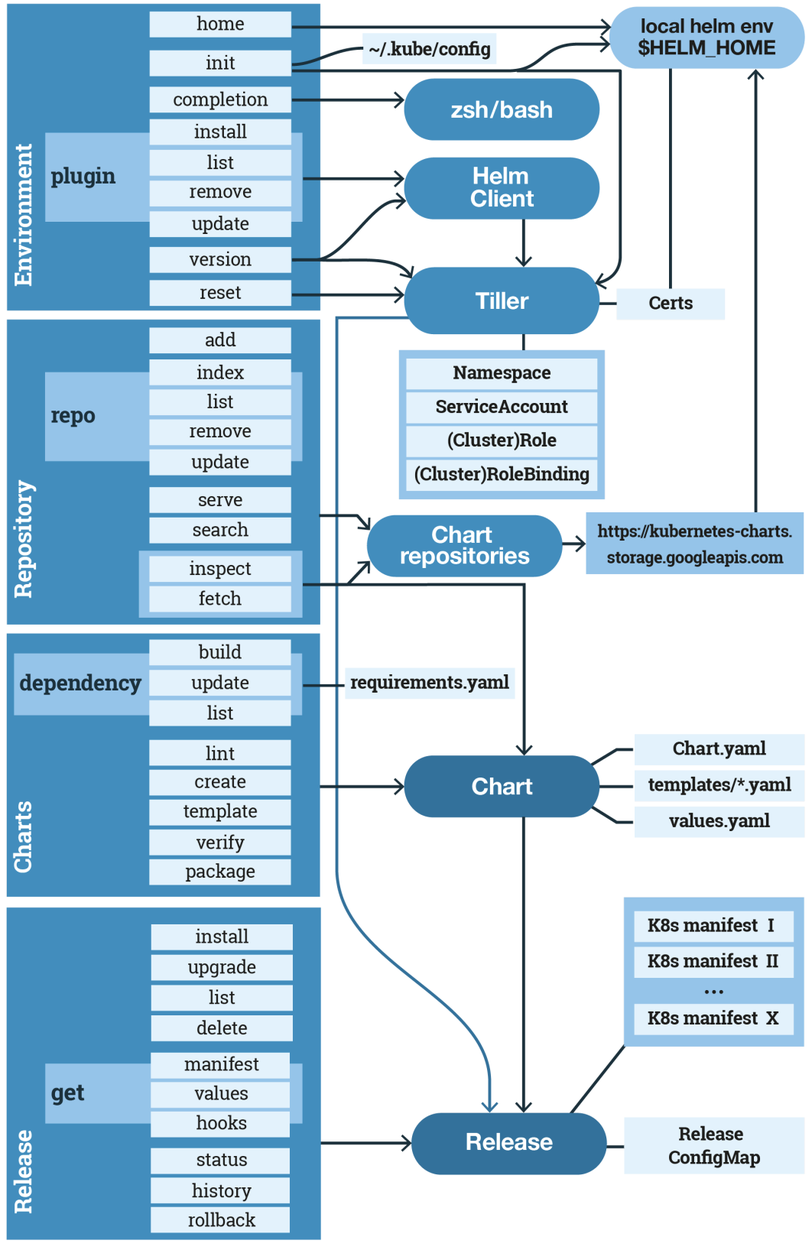
Automating Helm using Ansible

We recently released the kubernetes.core 1.1, our first Red Hat Certified Content Collection release, for general use. A big part of the new content that has been introduced is support for automating Helm operations.

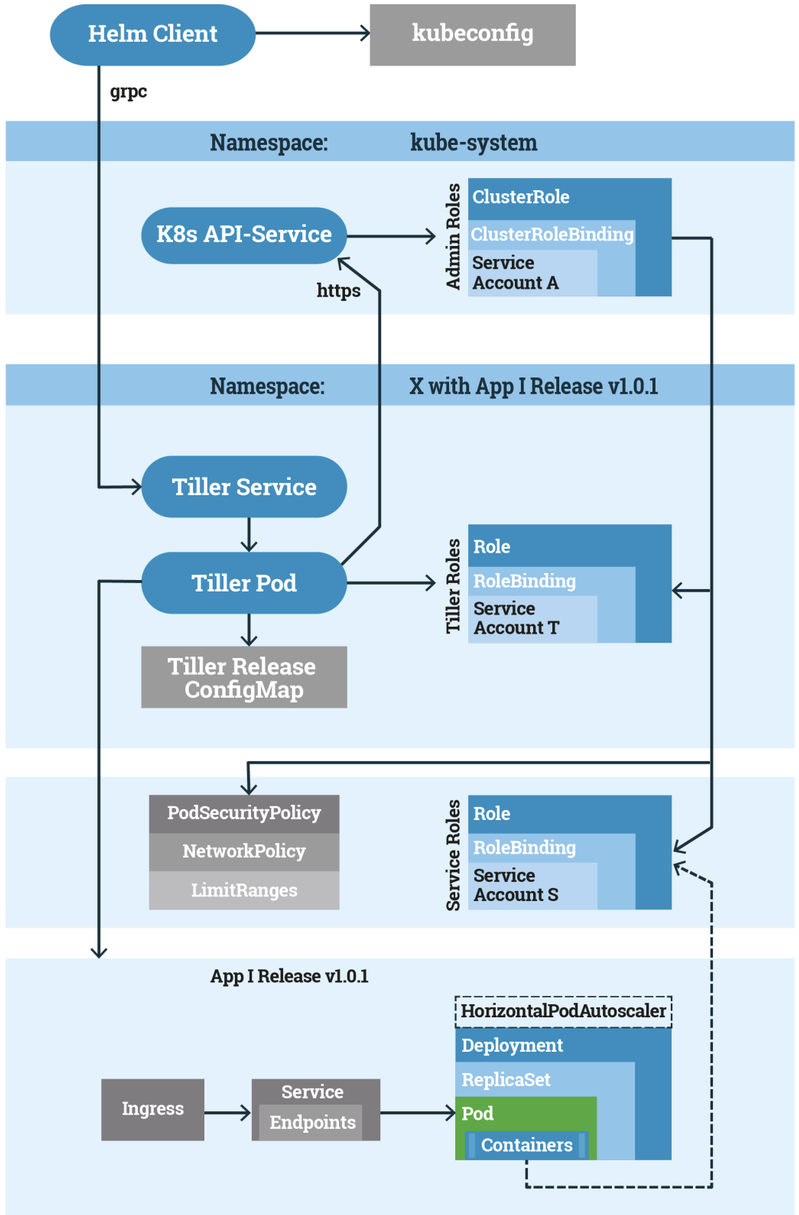
How Does Helm Work?



Design of Helm

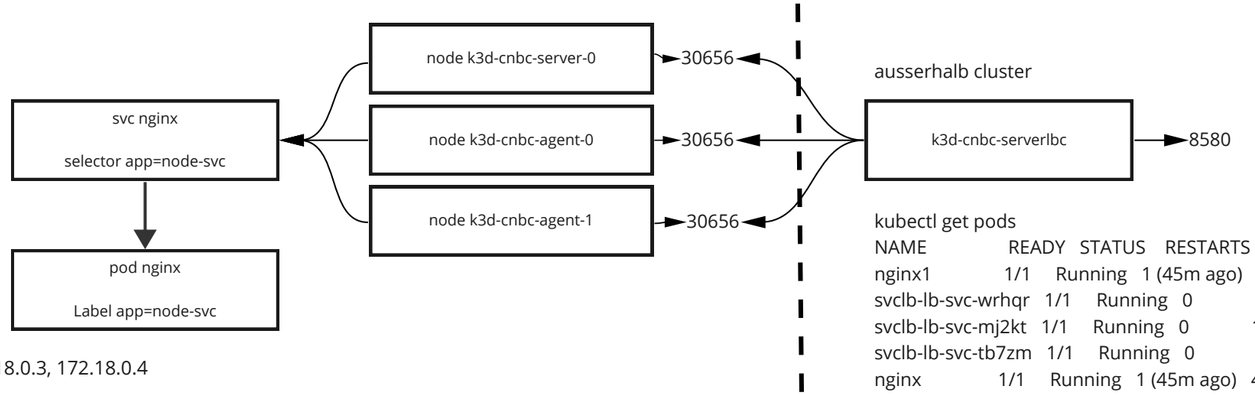


Helm Deployment



Loadbalancer

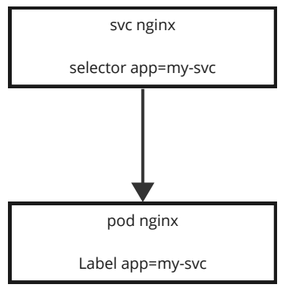
```
kubectl describe svc lb-svc
Name:                     lb-svc
Namespace:                nginx
Labels:                   app=lb-svc
Annotations:              <none>
Selector:                 app=lb-svc
Type:                     LoadBalancer
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.43.34.105
IPs:                      10.43.34.105
LoadBalancer Ingress:     172.18.0.2, 172.18.0.3, 172.18.0.4
Port:                     80-80  80/TCP
TargetPort:               80/TCP
NodePort:                 80-80  30506/TCP
Endpoints:                10.42.1.6:80
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
```




```
kubectl get pods
NAME              READY   STATUS    RESTARTS      AGE
nginx1            1/1     Running   1 (45m ago)   45m
svclb-lb-svc-wrhqr   1/1     Running   0             12m
svclb-lb-svc-mj2kt   1/1     Running   0             12m
svclb-lb-svc-tb7zm   1/1     Running   0             12m
nginx             1/1     Running   1 (45m ago)   45m
```

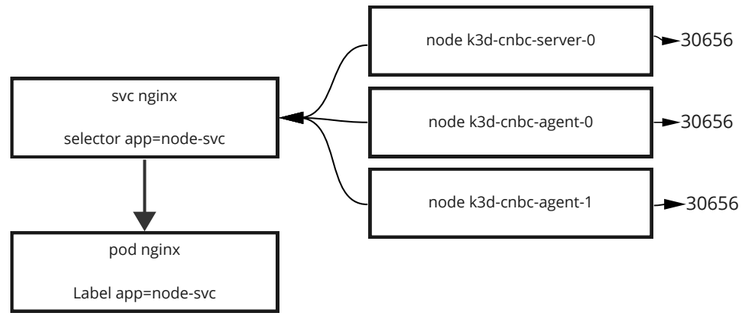
ClusterIP

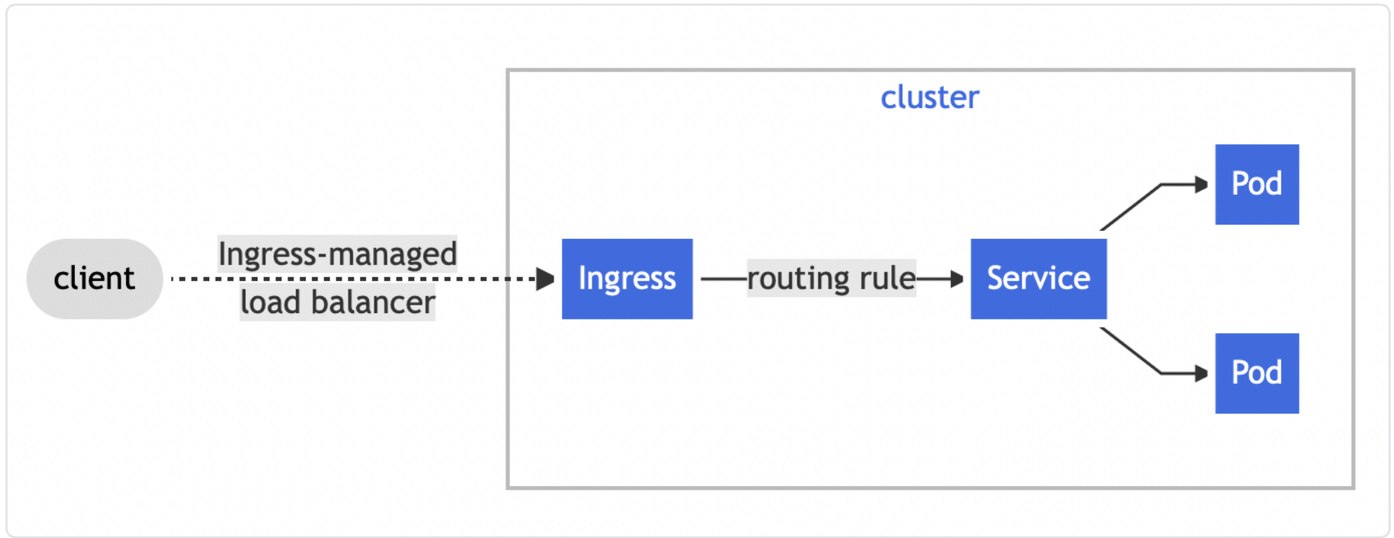
```
kubectl describe svc my-svc
Name:                     my-svc
Namespace:                nginx
Labels:                   app=my-svc
Annotations:              <none>
Selector:                 app=my-svc
Type:                     ClusterIP
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.43.11.78
IPs:                      10.43.11.78
Port:                     80-80  80/TCP
TargetPort:               80/TCP
Endpoints:                10.42.1.6:80
Session Affinity:         None
Events:                   <none>
```



NodePort

```
kubectl describe svc node-svc
Name:                     node-svc
Namespace:                nginx
Labels:                   app=node-svc
Annotations:              <none>
Selector:                 app=node-svc
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.43.91.238
IPs:                      10.43.91.238
Port:                     80-80  80/TCP
TargetPort:               80/TCP
NodePort:                 80-80  30656/TCP
Endpoints:                10.42.1.6:80
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
```

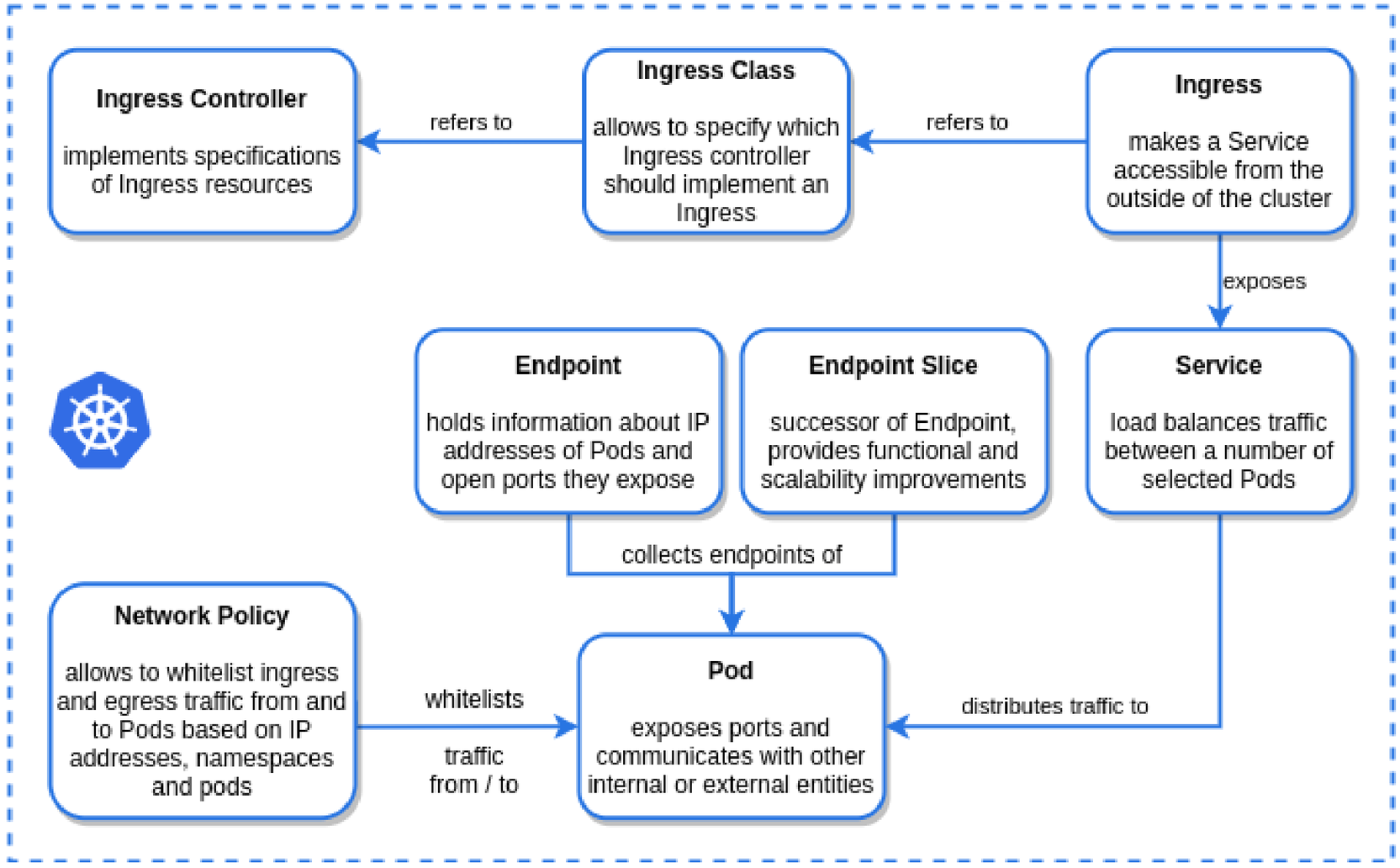


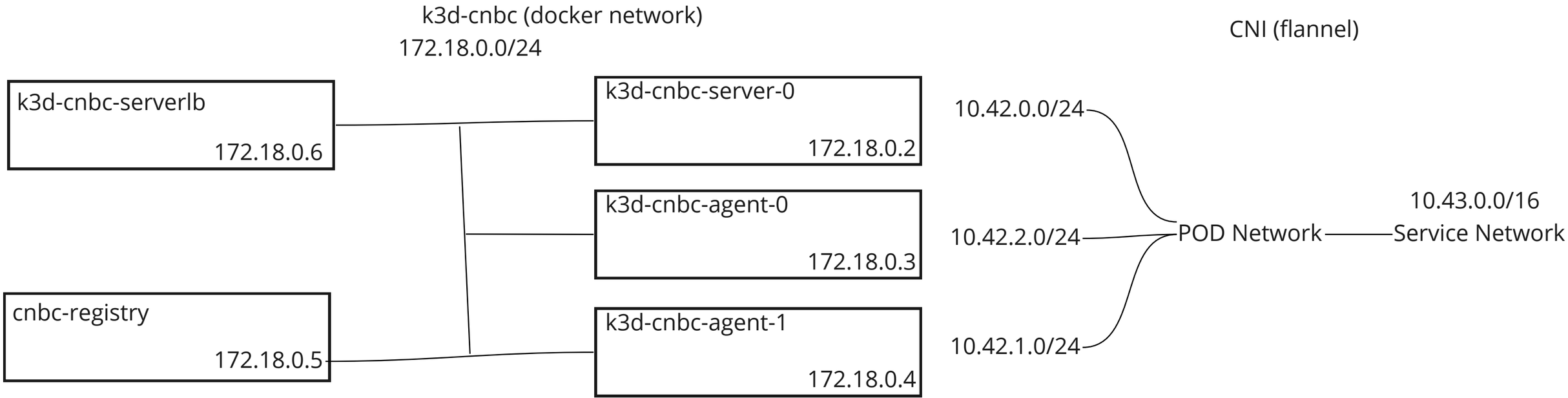


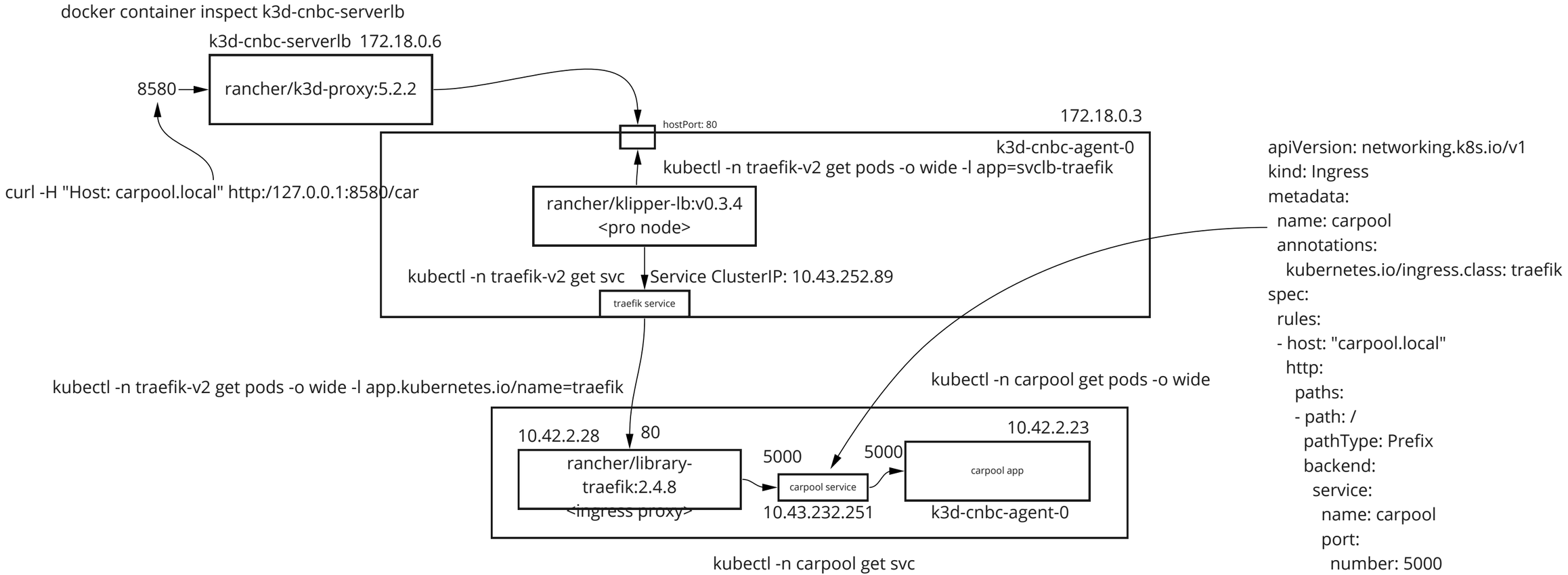

```
mkdir ~/traefik && cd ~/traefik
helm repo add traefik https://helm.traefik.io/traefik
helm repo update
kubectl create ns traefik-v2
helm install --namespace=traefik-v2 traefik traefik/traefik
kubectl -n traefik-v2 get all
```

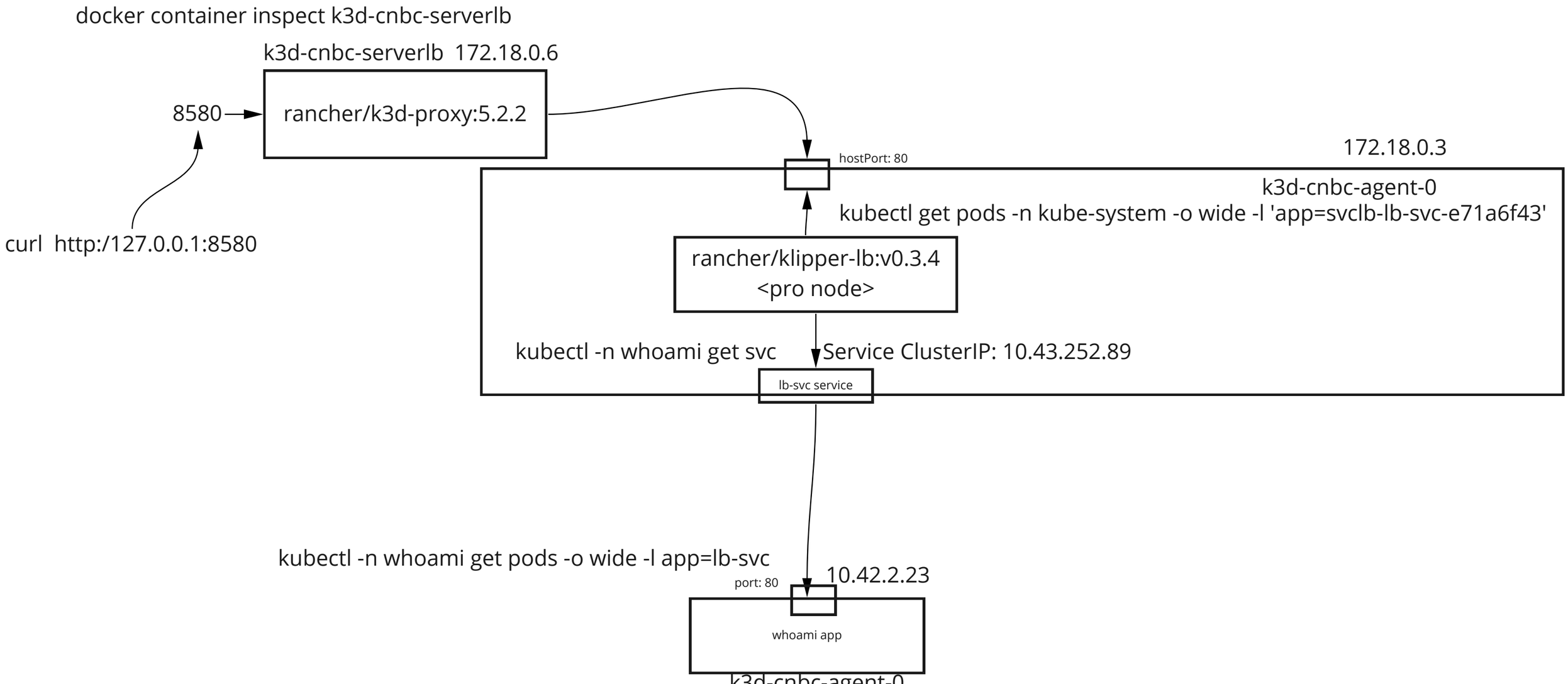
```
# Source: web/templates/ingress.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: release-name-web
  labels:
    helm.sh/chart: web-0.1.0
    app.kubernetes.io/name: web
    app.kubernetes.io/instance: release-name
    app.kubernetes.io/version: "1.16.0"
    app.kubernetes.io/managed-by: Helm
  annotations:
    kubernetes.io/ingress.class: traefik
spec:
  rules:
    - host: "web.local"
      http:
        paths:
          - path: /
            pathType: ImplementationSpecific
            backend:
              service:
                name: release-name-web
                port:
                  number: 80
```

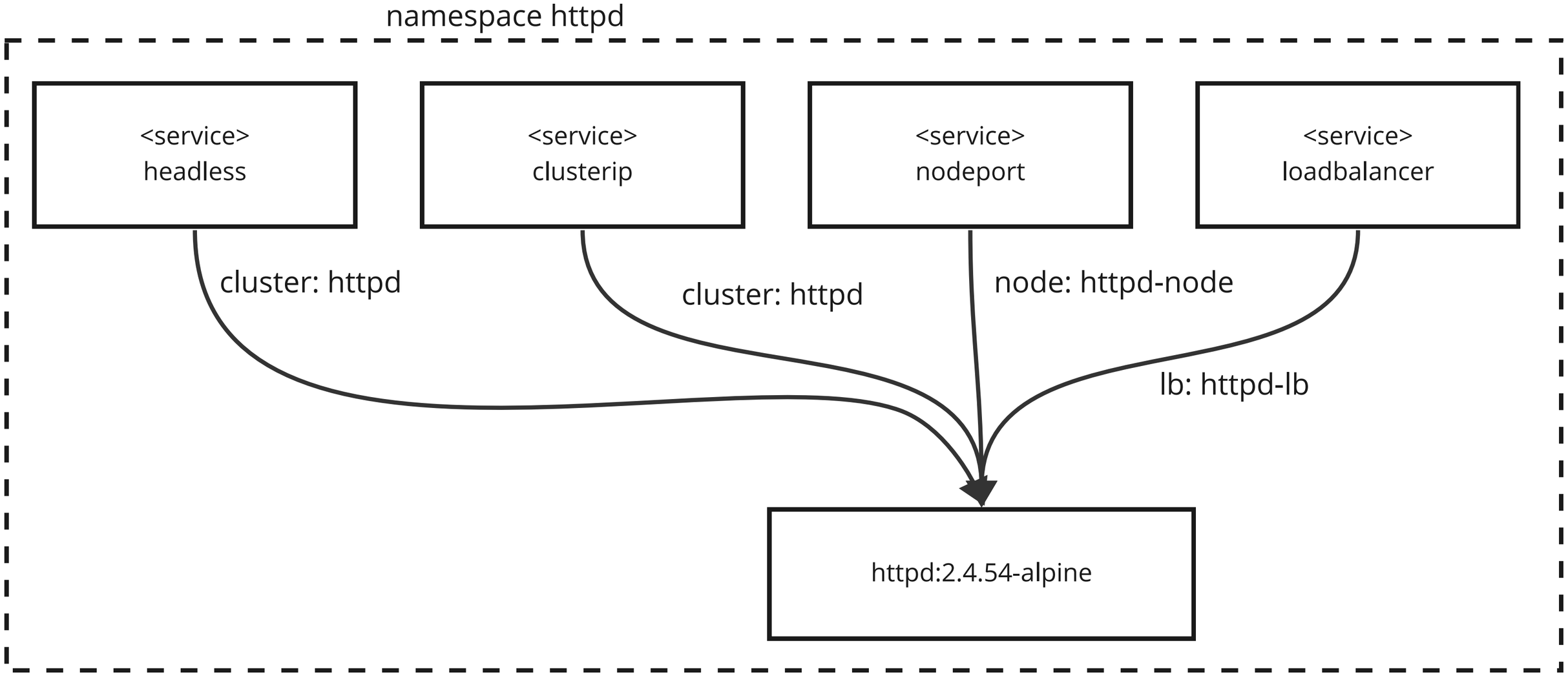
Kubernetes Networking Objects

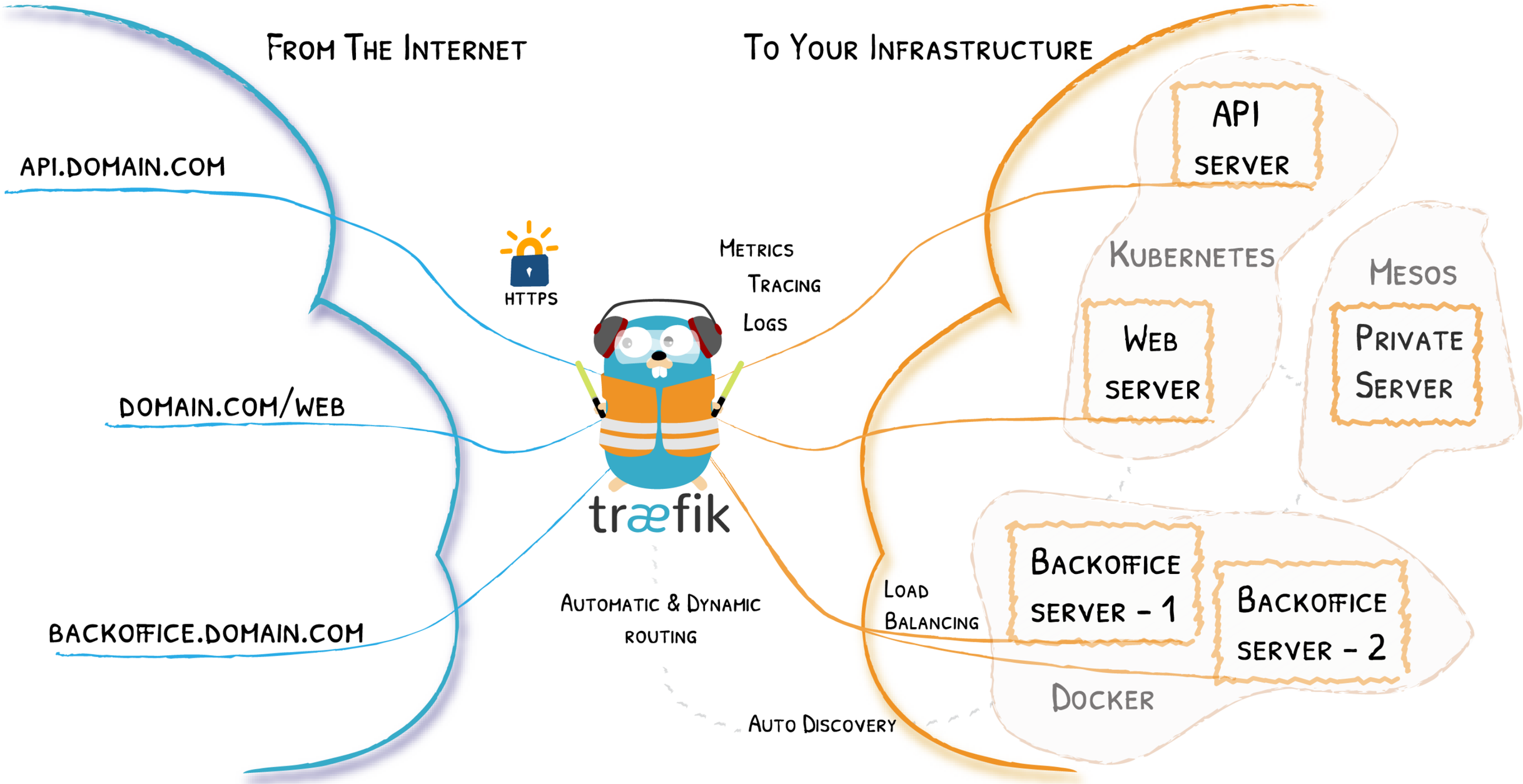


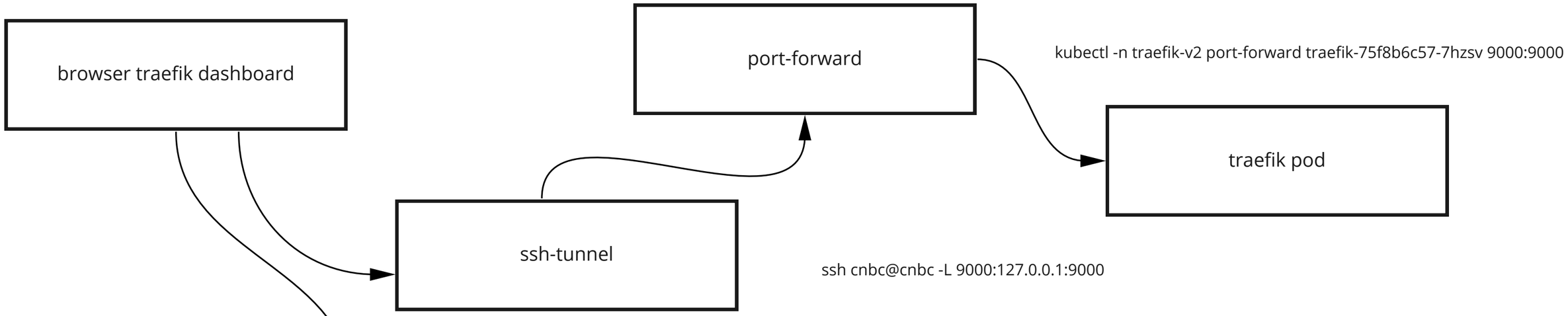

k3d-cnbc (docker network)
172.18.0.0/24
CNI (flannel)
k3d-cnbc-serverlb
172.18.0.6
k3d-cnbc-server-0
172.18.0.2
k3d-cnbc-agent-0
172.18.0.3
cnbc-registry
172.18.0.5
k3d-cnbc-agent-1
172.18.0.4
10.42.0.0/24
10.42.2.0/24
10.42.1.0/24
POD Network
10.43.0.0/16
Service Network

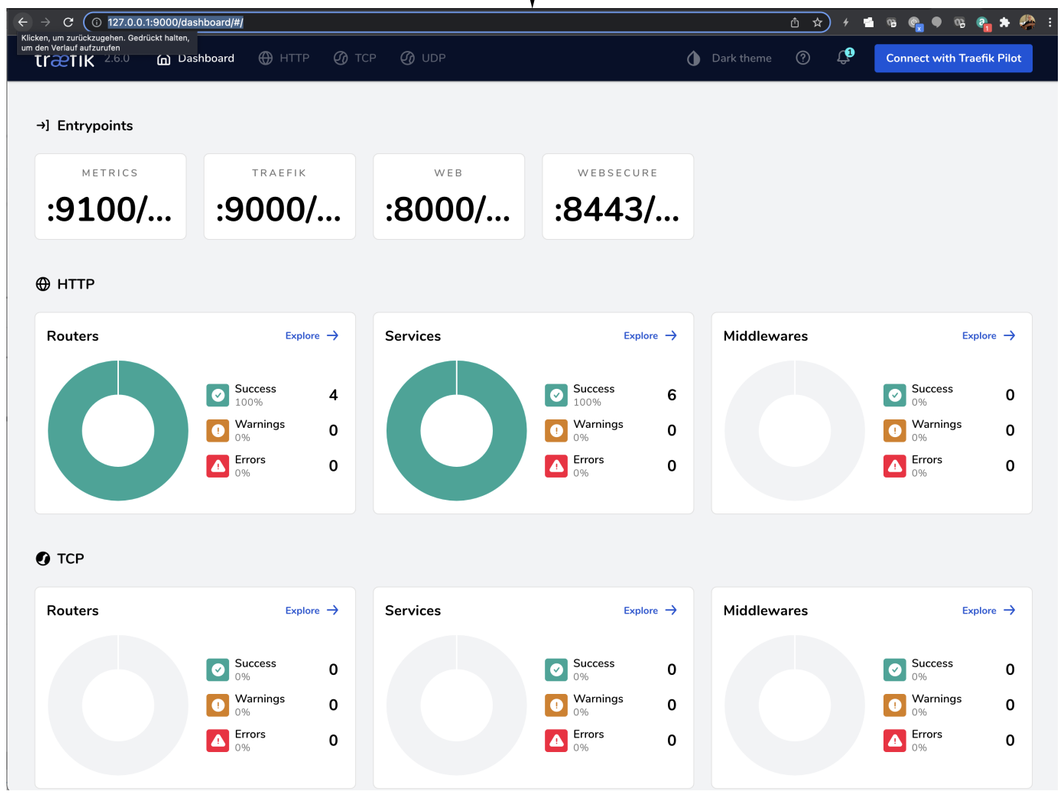

docker container inspect k3d-cnbc-serverlb
k3d-cnbc-serverlb 172.18.0.6
8580
rancher/k3d-proxy:5.2.2
curl -H "Host: carpool.local" http:/127.0.0.1:8580/car
hostPort: 80
172.18.0.3
k3d-cnbc-agent-0
kubectl -n traefik-v2 get pods -o wide -l app=svclb-traefik
rancher/klipper-lb:v0.3.4
<pro node>
kubectl -n traefik-v2 get svc
Service ClusterIP: 10.43.252.89
traefik service
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
name: carpool
annotations:
kubernetes.io/ingress.class: traefik
spec:
rules:
- host: "carpool.local"
http:
paths:
- path: /
pathType: Prefix
backend:
service:
name: carpool
port:
number: 5000
kubectl -n traefik-v2 get pods -o wide -l app.kubernetes.io/name=traefik
kubectl -n carpool get pods -o wide
10.42.2.28
80
rancher/library-traefik:2.4.8
<ingress proxy>
5000
carpool service
10.43.232.251
5000
10.42.2.23
carpool app
k3d-cnbc-agent-0
kubectl -n carpool get svc


docker container inspect k3d-cnbc-serverlb
k3d-cnbc-serverlb 172.18.0.6
8580
rancher/k3d-proxy:5.2.2
curl http:/127.0.0.1:8580
hostPort: 80
172.18.0.3
k3d-cnbc-agent-0
kubectl get pods -n kube-system -o wide -l 'app=svclb-lb-svc-e71a6f43'
rancher/klipper-lb:v0.3.4
<pro node>
kubectl -n whoami get svc
Service ClusterIP: 10.43.252.89
lb-svc service
kubectl -n whoami get pods -o wide -l app=lb-svc
10.42.2.23
port: 80
whoami app
k3d-cnbc-agent-0

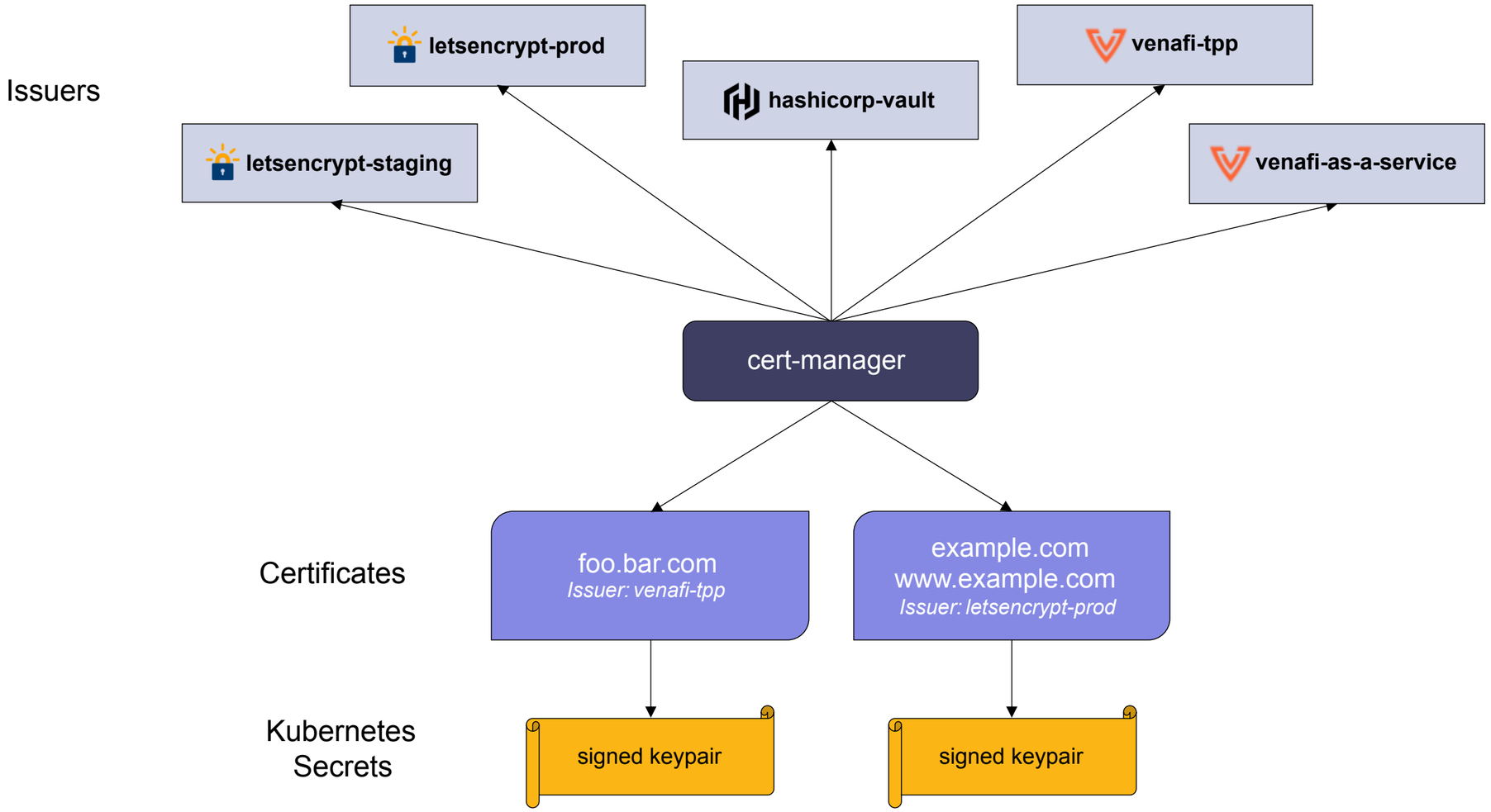

namespace httpd
<service>
headless
<service>
clusterip
<service>
nodeport
<service>
loadbalancer
cluster: httpd
cluster: httpd
node: httpd-node
lb: httpd-lb
httpd:2.4.54-alpine

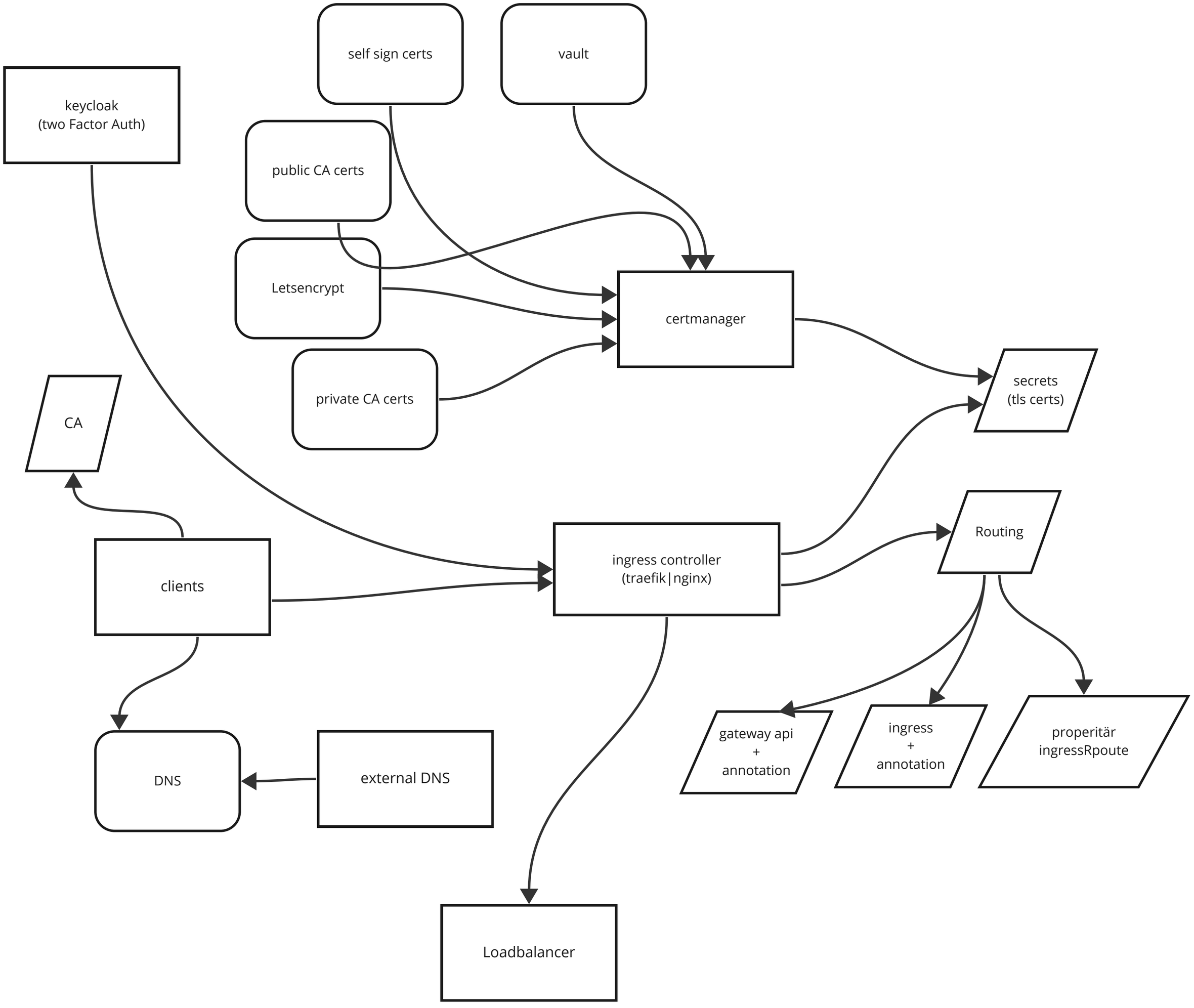

From The Internet
To Your Infrastructure
api.domain.com
domain.com/web
backoffice.domain.com
HTTPS
Metrics
Tracing
Logs
træfik
Automatic & Dynamic
routing
Auto Discovery
Load
Balancing
API
server
Kubernetes
Web
server
Mesos
Private
Server
Backoffice
server - 1
Backoffice
server - 2
Docker


browser traefik dashboard
port-forward
kubectl -n traefik-v2 port-forward traefik-75f8b6c57-7hzsv 9000:9000
traefik pod
ssh-tunnel
ssh cnbc@cnbc -L 9000:127.0.0.1:9000
http://127.0.0.1:9000/dashboard/#/

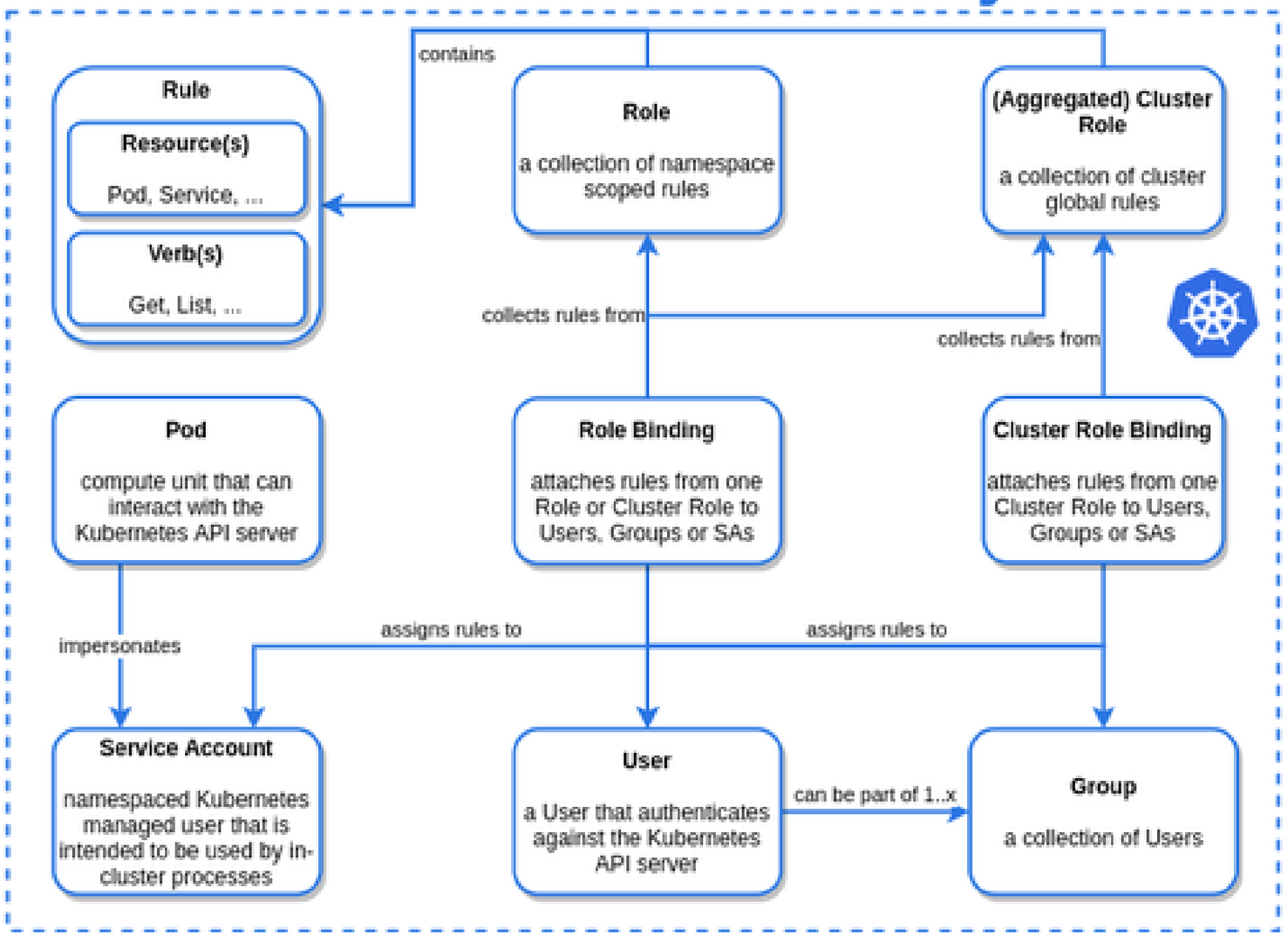

127.0.0.1:9000/dashboard/#/
Klicken, um zurückzugehen. Gedrückt halten, um den Verlauf aufzurufen
Dashboard
HTTP
TCP
UDP
Dark theme
Connect with Traefik Pilot
Entrypoints
METRICS
:9100/...
TRAEFIK
:9000/...
WEB
:8000/...
WEBSECURE
:8443/...
HTTP
Routers
Explore
Success 100% 4
Warnings 0% 0
Errors 0% 0
Services
Explore
Success 100% 6
Warnings 0% 0
Errors 0% 0
Middlewares
Explore
Success 0% 0
Warnings 0% 0
Errors 0% 0
TCP
Routers
Explore
Success 0% 0
Warnings 0% 0
Errors 0% 0
Services
Explore
Success 0% 0
Warnings 0% 0
Errors 0% 0
Middlewares
Explore
Success 0% 0
Warnings 0% 0
Errors 0% 0

https://cert-manager.io/docs/

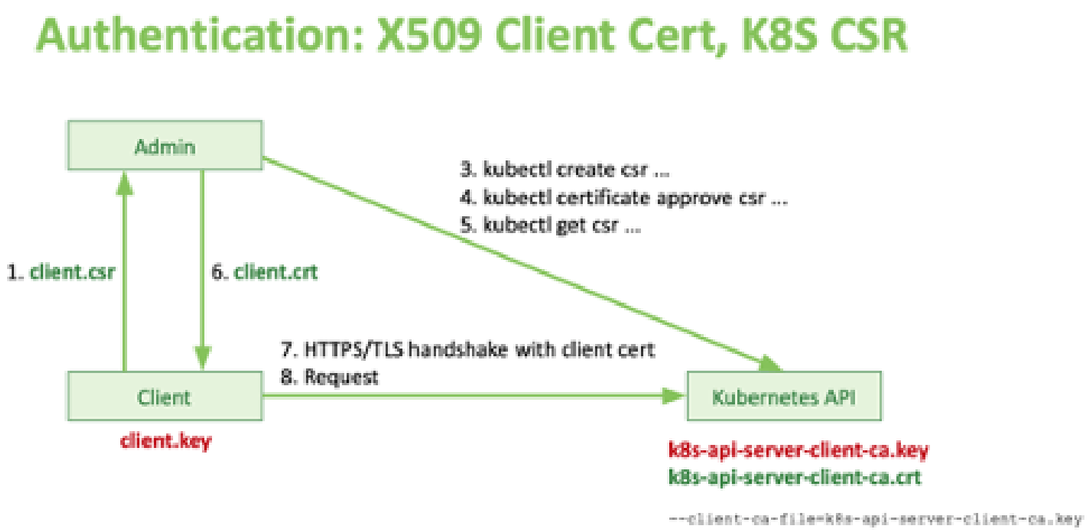



self sign certs
vault
keycloak
(two Factor Auth)
public CA certs
Letsencrypt
certmanager
private CA certs
secrets
(tls certs)
CA
clients
ingress controller
(traefik|nginx)
Routing
gateway api
+
annotation
ingress
+
annotation
properitär
ingressRpoute
DNS
external DNS
Loadbalancer

Kubernetes RBAC Objects



Client Certs

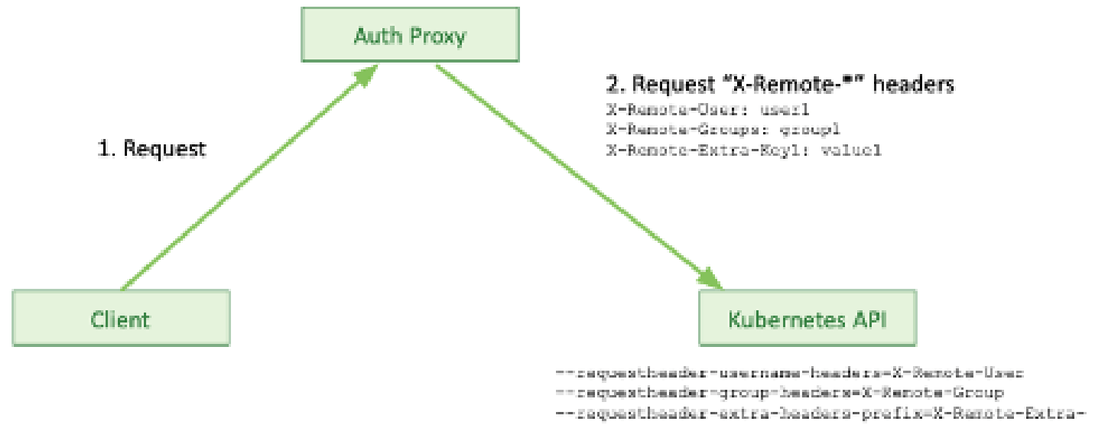


Service Account

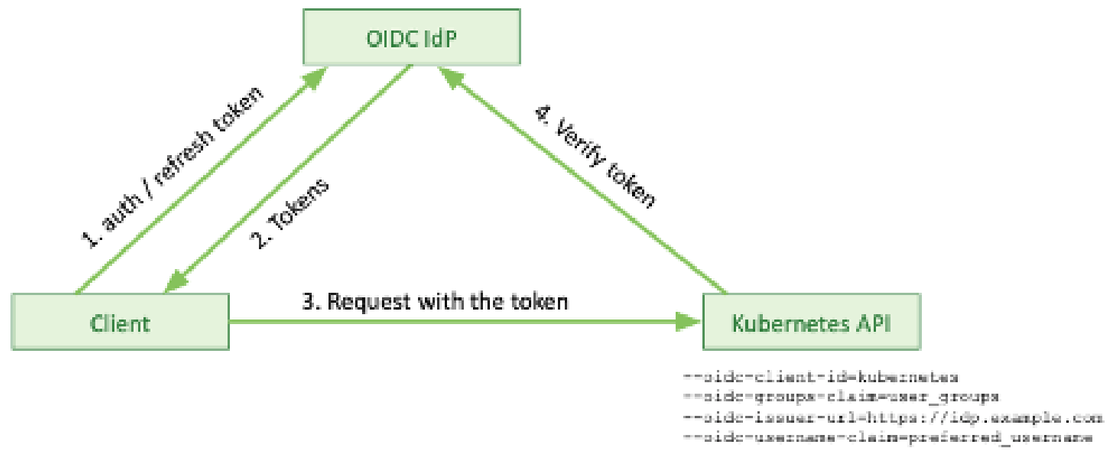
Impersonate Header


```
kubectl get nodes \
    --as "system:serviceaccount:default:sa1" \
    --as-group g1 \
    --as-group g2
```

Authenticating Proxy

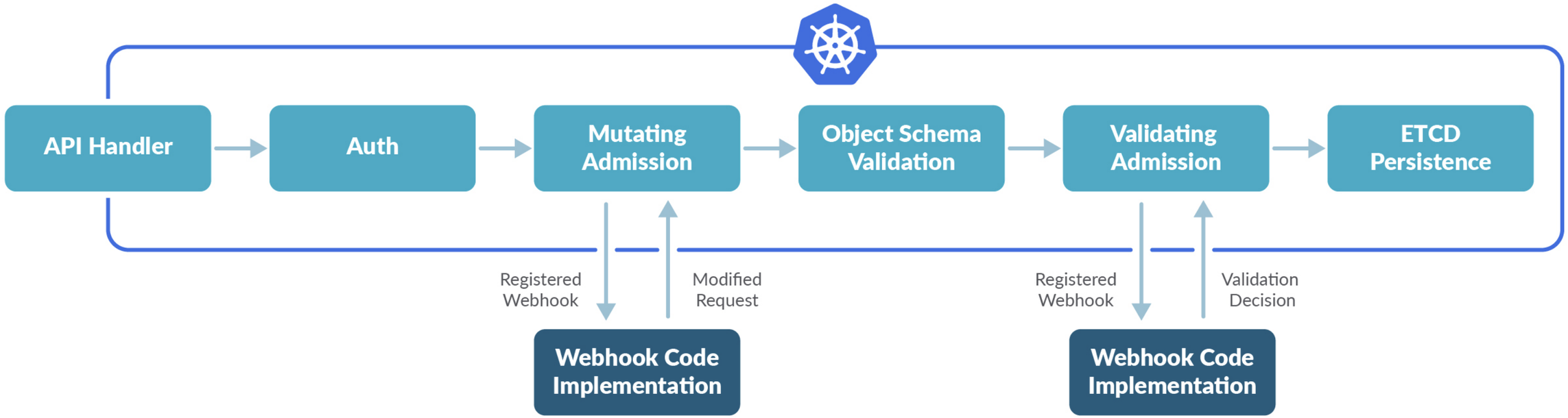


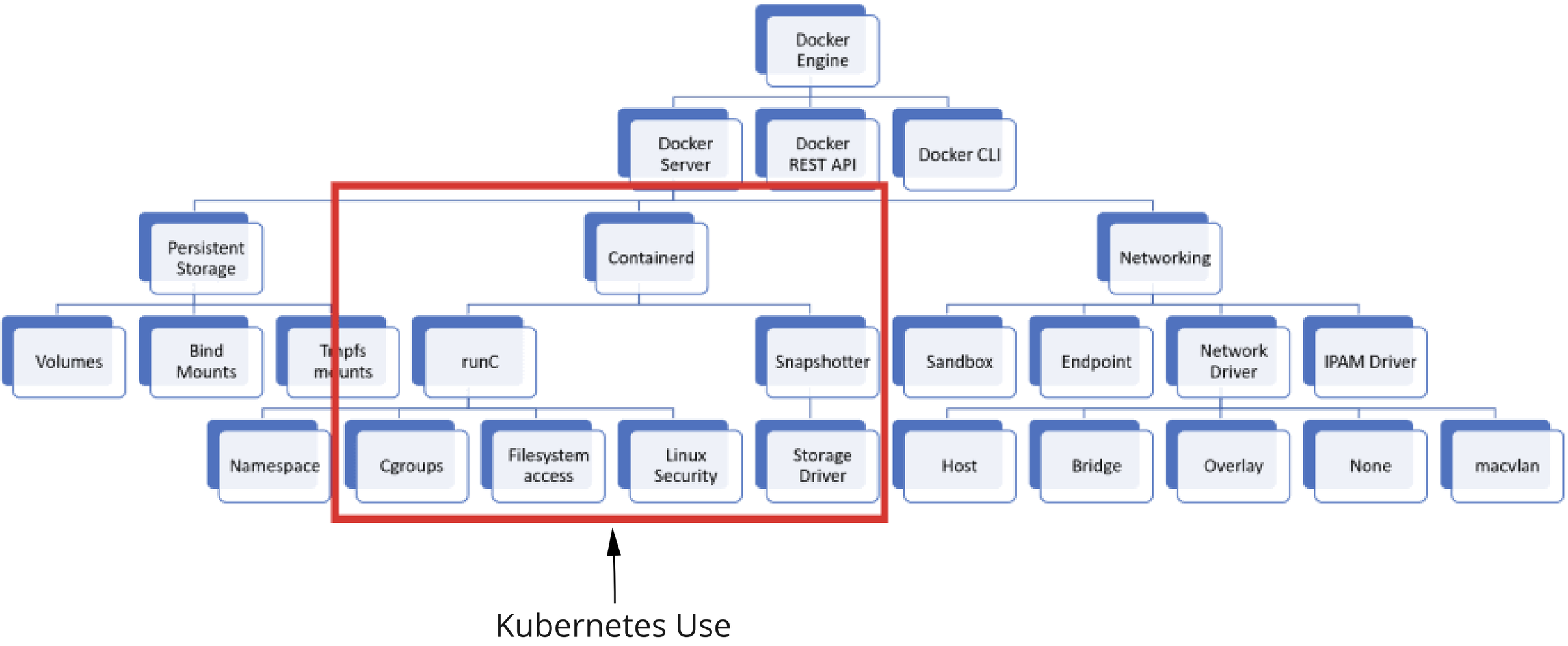
OIDC Token



https://www.cncf.io/blog/2020/07/31/kubernetes-rbac-101-authentication/

https://kubernetes.io/docs/reference/access-authn-authz/authentication/


API Handler
Auth
Mutating Admission
Object Schema Validation
Validating Admission
ETCD Persistence
Registered Webhook
Modified Request
Registered Webhook
Validation Decision
Webhook Code Implementation
Webhook Code Implementation


Docker Engine
Docker Server
Docker REST API
Docker CLI
Persistent Storage
Containerd
Networking
Volumes
Bind Mounts
runC
Snapshotter
Sandbox
Endpoint
Network Driver
IPAM Driver
Namespace
Cgroups
Filesystem access
Linux Security
Storage Driver
Host
Bridge
Overlay
None
macvlan
Kubernetes Use

application/vnd.docker.distribution.manifest.list.v2+json

container images v2

linux windows

container images v2 — arm

container images v2 — arm64

container images v2 — amd64

application/vnd.docker.container.image.v1+json

container images

Registry

Image Repository cnbc/appb

Image Repository cnbc/appa

alpine:sha265@askjdklasjdlkajsdljasldjaljdlkajsdla

tag C

tag A

tag B

20210423

latest

20210422

Image layer 1
<hash key>

Image layer 1.1

Image layer 2

Image layer 3

opencontainers.org

Open Container Initiative - Open Container Initiative

alpine:3.12

QEMU is a generic and open source machine emulator and virtualizer.

www.qemu.org

QEMU

```
docker container run --platform=linux/arm -ti alpine:3.12 /bin/sh
/ # uname -a
Linux 2abafab20995 4.19.121-linuxkit #1 SMP Tue Dec 1 17:50:32 UTC 2020 armv7l Linux
/ #
```

cloud native application bundles

cnab.io

CNAB: Cloud Native Application Bundles

helm charts

helm.sh

Helm

Helm - The Kubernetes Package Manager.

github.com

distribution/distribution

The toolkit to pack, ship, store, and deliver container content - distribution/distribution

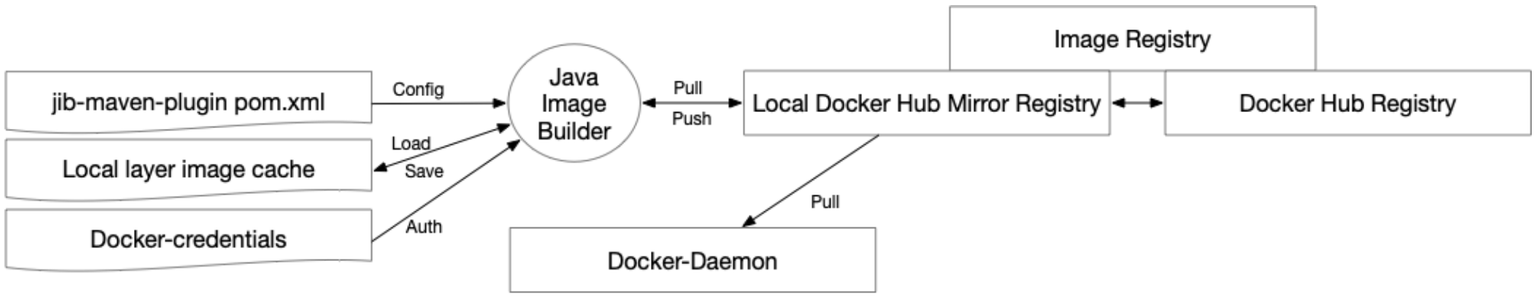
github.com

opencontainers/distributic spec

OCI Distribution Specification. Contribute to opencontainers/distribution-spec development by creating an account on GitHub.

chartmuseum.com

ChartMuseum - Helm Chart Repository

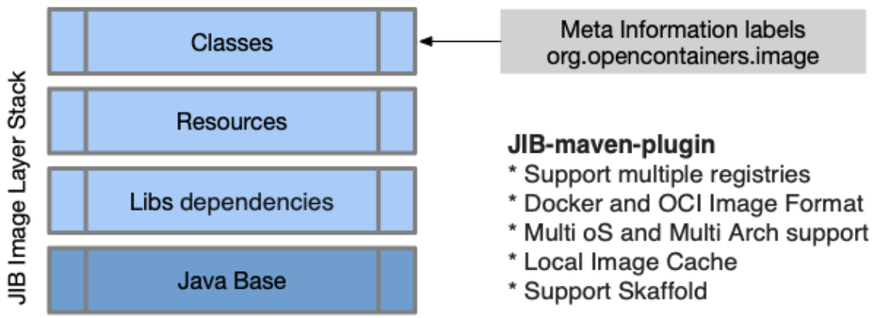


Java Image Builder (JIB)

Simple
* No Dockerfile
* Daemonless
* Support Java Native

Fast
* Support distinct layers
* Use registry caching
* Rebuild & push only layer that changed

Reproducible
* No Timestamp
* Use only Maven and Gradle build metadata
* Can use Distroless Base Java



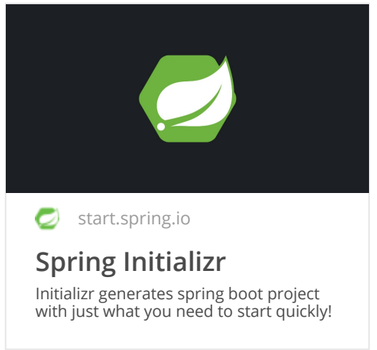
JIB-maven-plugin
* Support multiple registries
* Docker and OCI Image Format
* Multi oS and Multi Arch support
* Local Image Cache
* Support Skaffold

https://github.com/GoogleContainerTools/jib/tree/master/jib-maven-plugin

https://github.com/GoogleContainerTools/jib/blob/master/README.md

https://cloudplatform.googleblog.com/2018/07/introducing-jib-build-java-docker-images-better.html

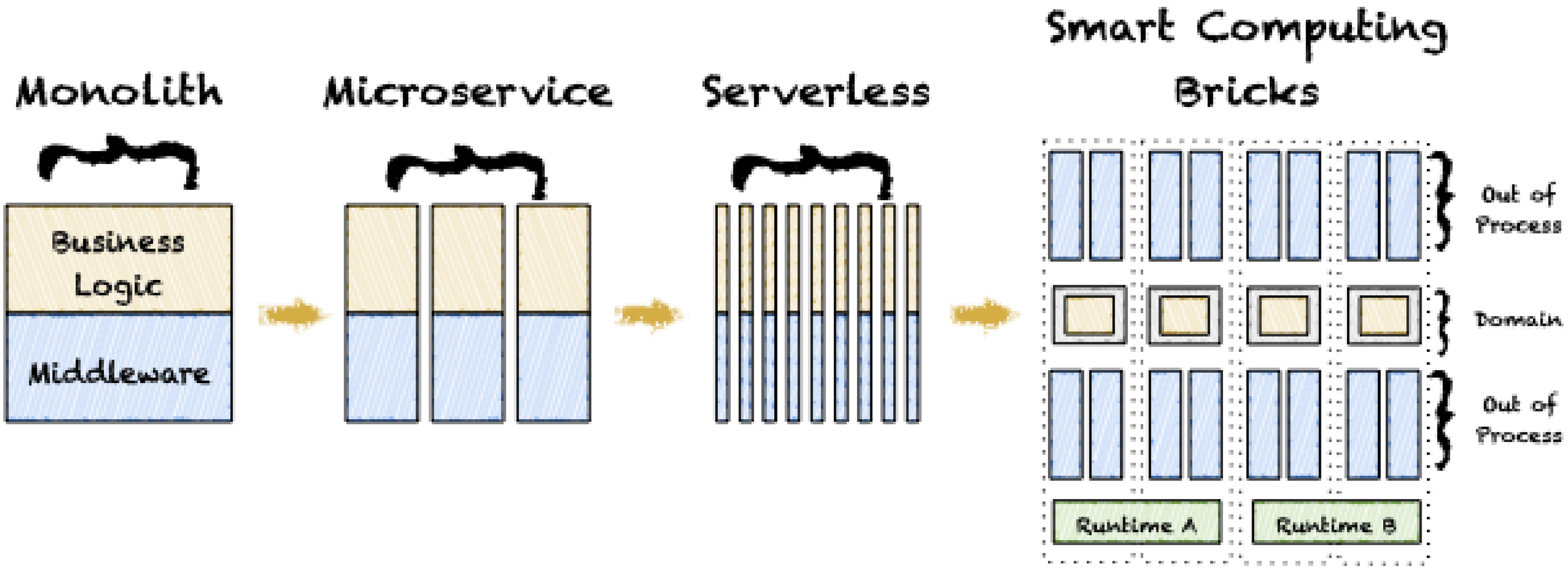
https://github.com/GoogleContainerTools/distroless

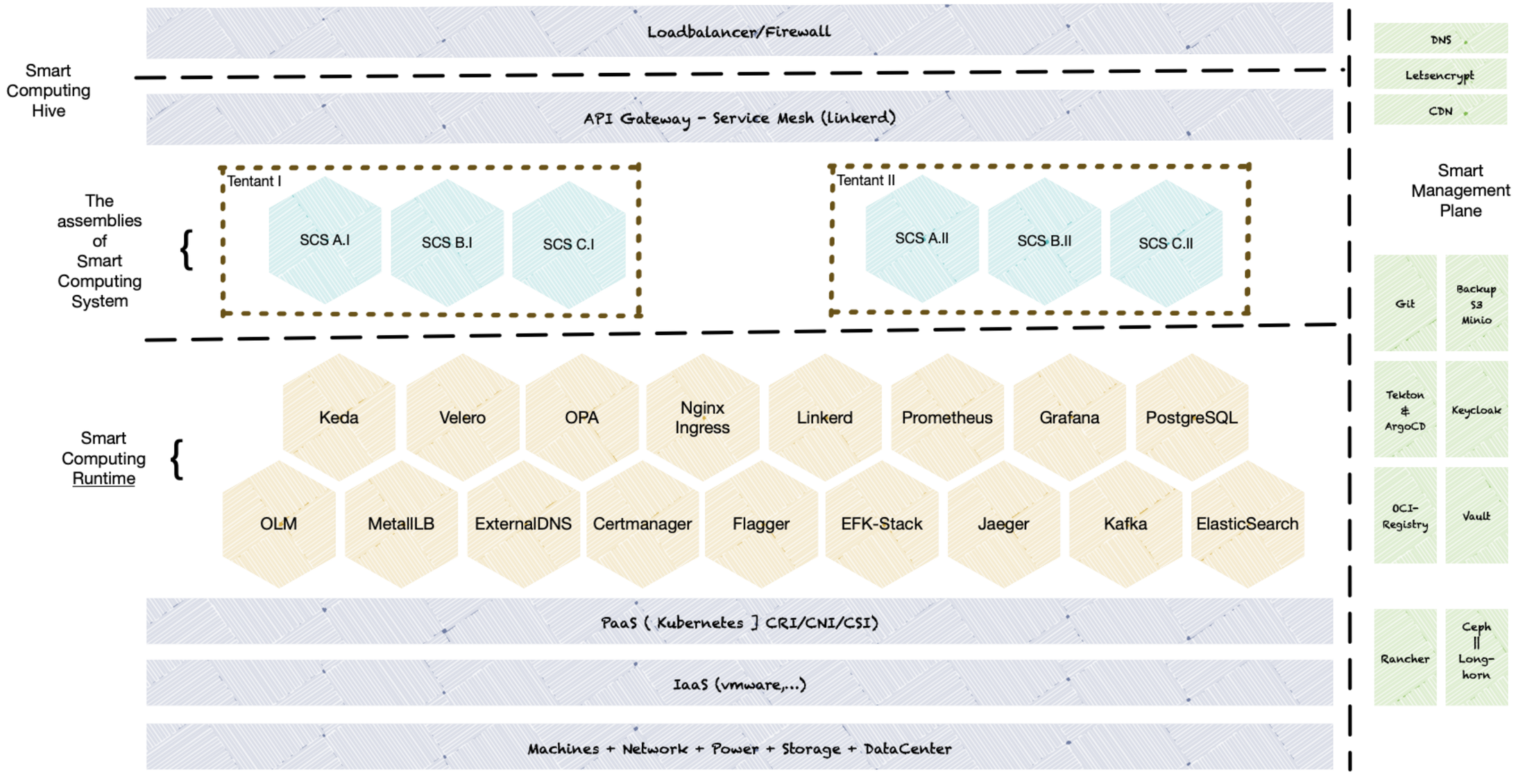


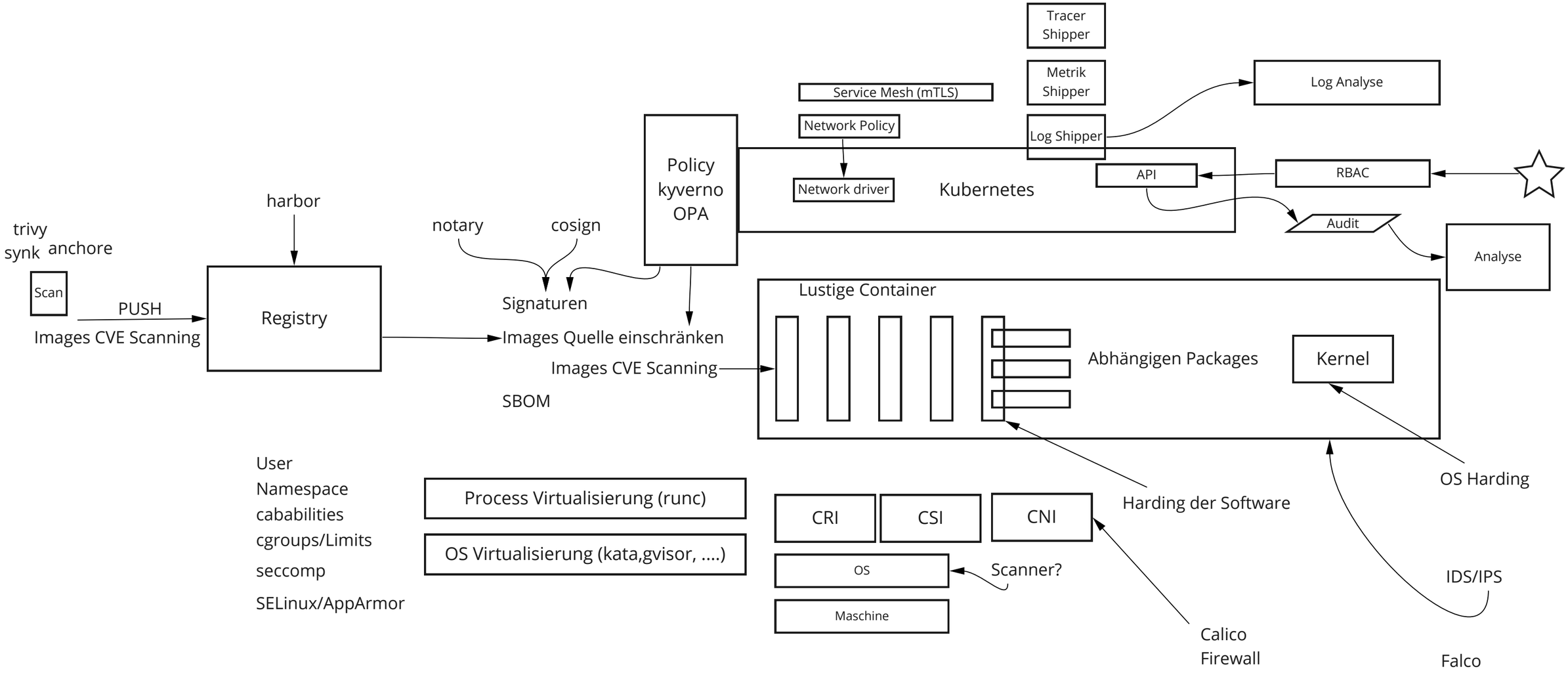

```
<plugin>
 <groupId>com.google.cloud.tools</groupId>
          <artifactId>jib-maven-plugin</artifactId>
          <version>2.5.2</version>
          <configuration>
            <from>
              <image>adoptopenjdk/openjdk11</image>
            </from>
            <to>
              <image>bee42/${project.artifactId}</image>
              <tags>
                <tag>latest</tag>
                <tag>${project.version}</tag>
              </tags>
            </to>
            <container>
              <ports>
                <port>8080</port>
              </ports>
            </container>
            <allowInsecureRegistries>true</allowInsecureRegistries>
          </configuration>
        </plugin>
```

```
package com.example.demo.controller;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HelloWorldController {
@GetMapping(path = "/helloworld")
public String helloWorld() {
 return "Hello World!";
}
}
```

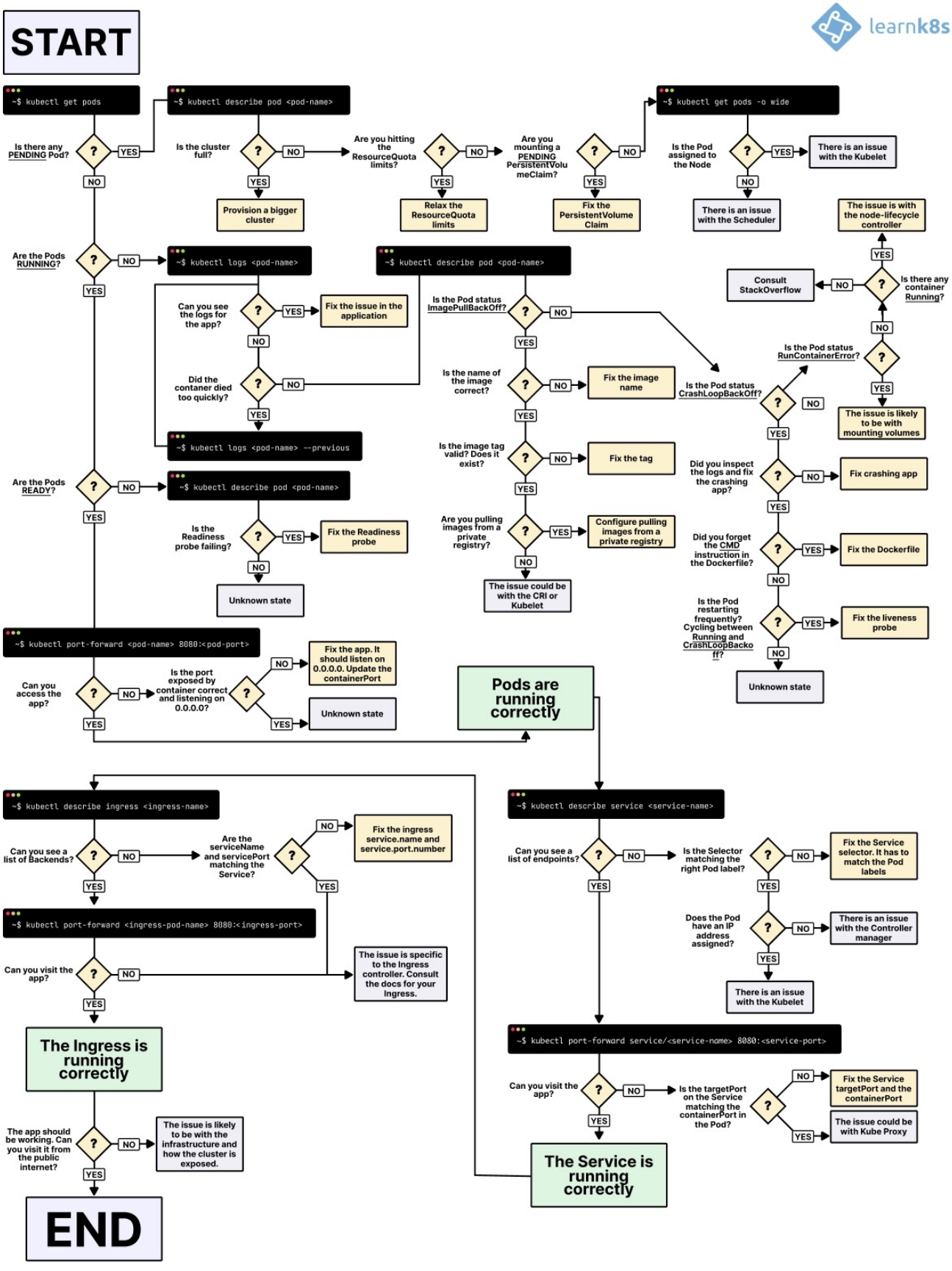
Smart Composition Architecture

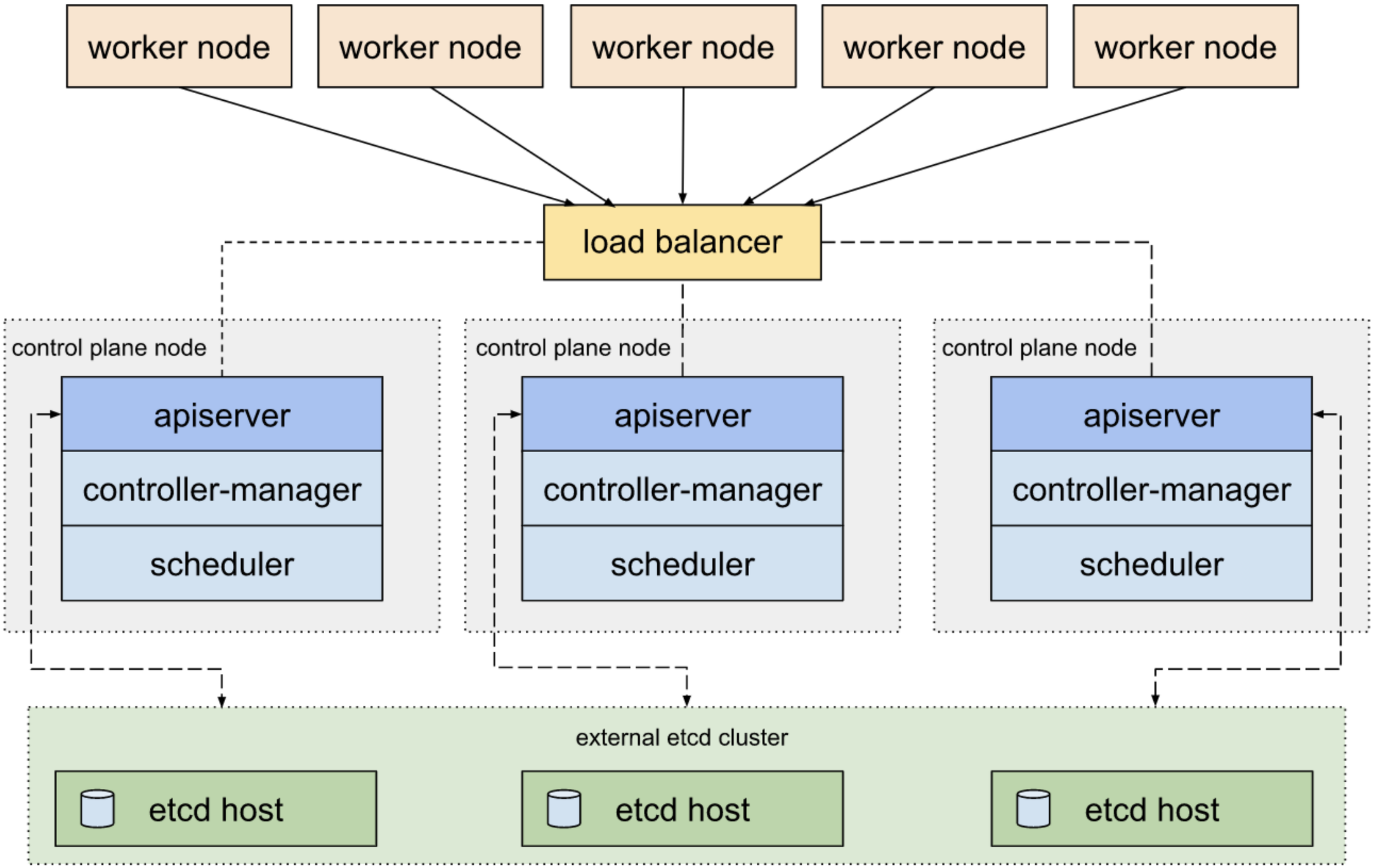





Tracer Shipper
Metrik Shipper
Log Shipper
Log Analyse
Service Mesh (mTLS)
Network Policy
Network driver
Policy kyverno OPA
Kubernetes
API
RBAC
Audit
Analyse
harbor
trivy synk
anchore
Scan
PUSH
Images CVE Scanning
Registry
notary
cosign
Signaturen
Images Quelle einschränken
Images CVE Scanning
SBOM
Lustige Container
Abhängigen Packages
Kernel
User
Namespace
cababilities
cgroups/Limits
seccomp
SELinux/AppArmor
Process Virtualisierung (runc)
OS Virtualisierung (kata,gvisor, ...)
CRI
CSI
CNI
OS
Maschine
Scanner?
Harding der Software
OS Harding
IDS/IPS
Calico Firewall
Falco

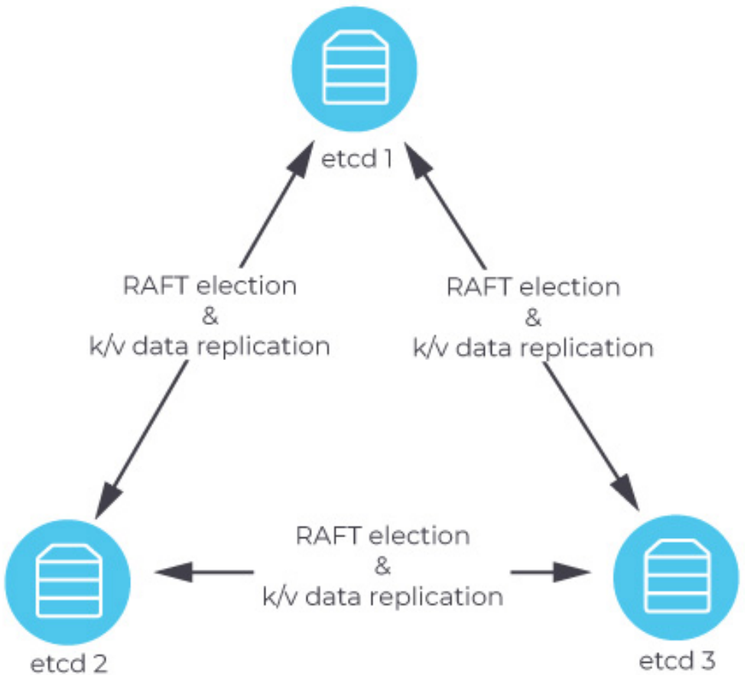
https://learnk8s.io/troubleshooting-deployments




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

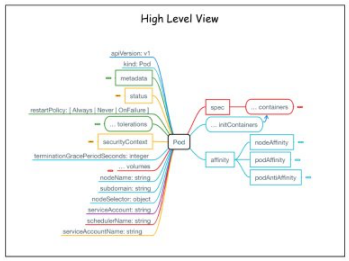
simple etcd cluster:

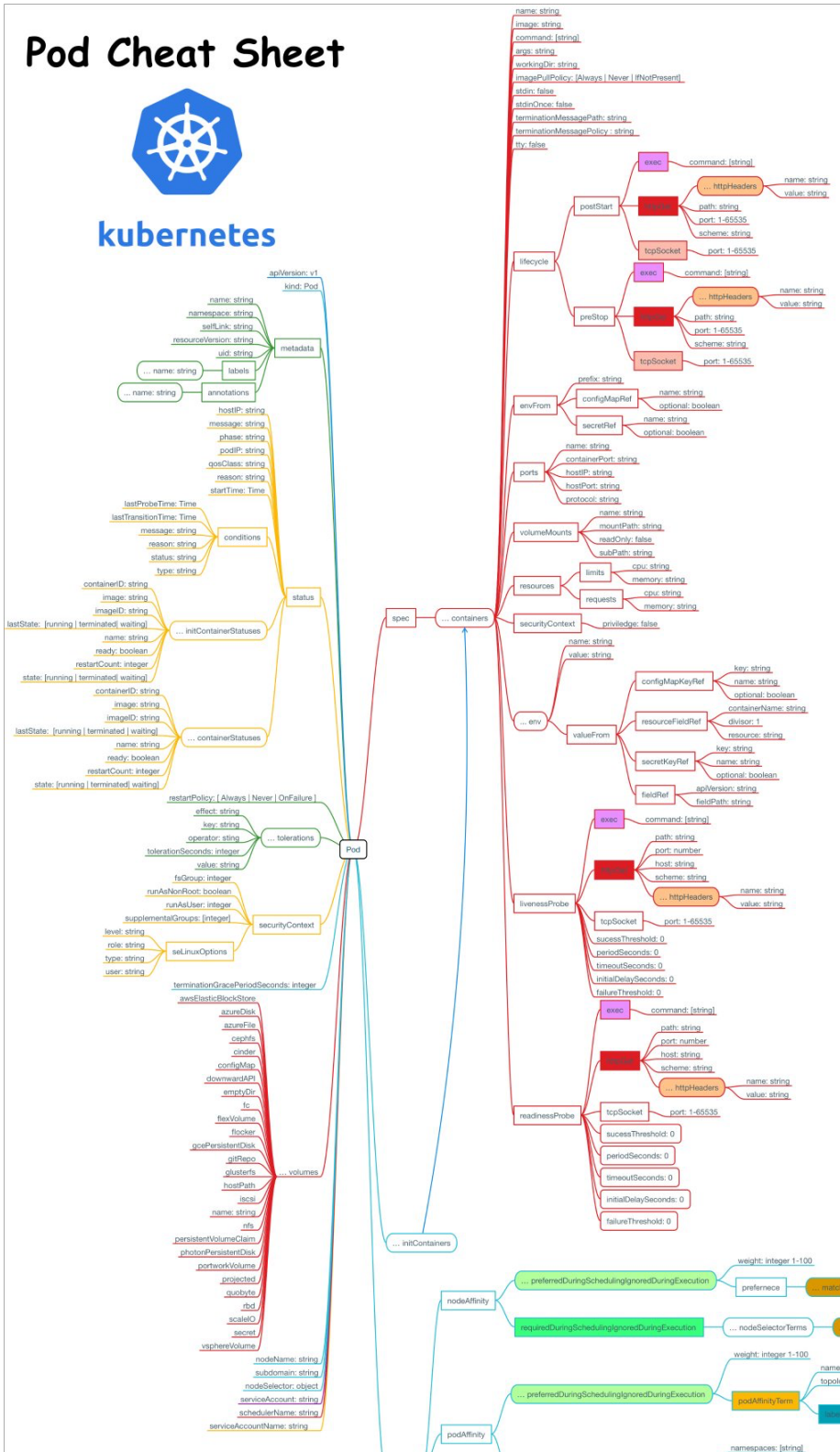
CYBERTEC
DATA SCIENCE & POSTGRESQL



Pod Cheat Sheet

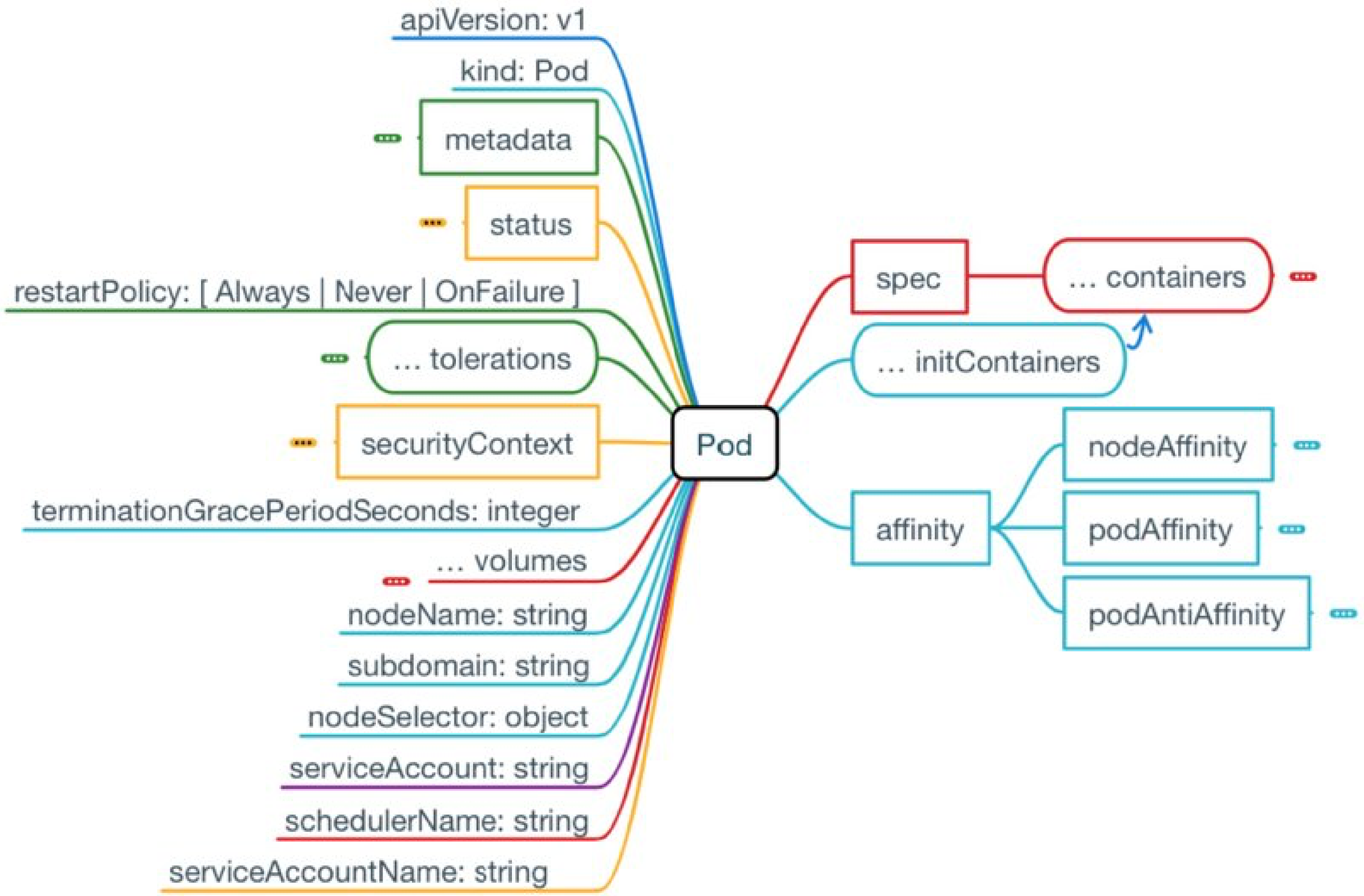
Reference https://kubernetes.io/docs/api-reference/v1.6/



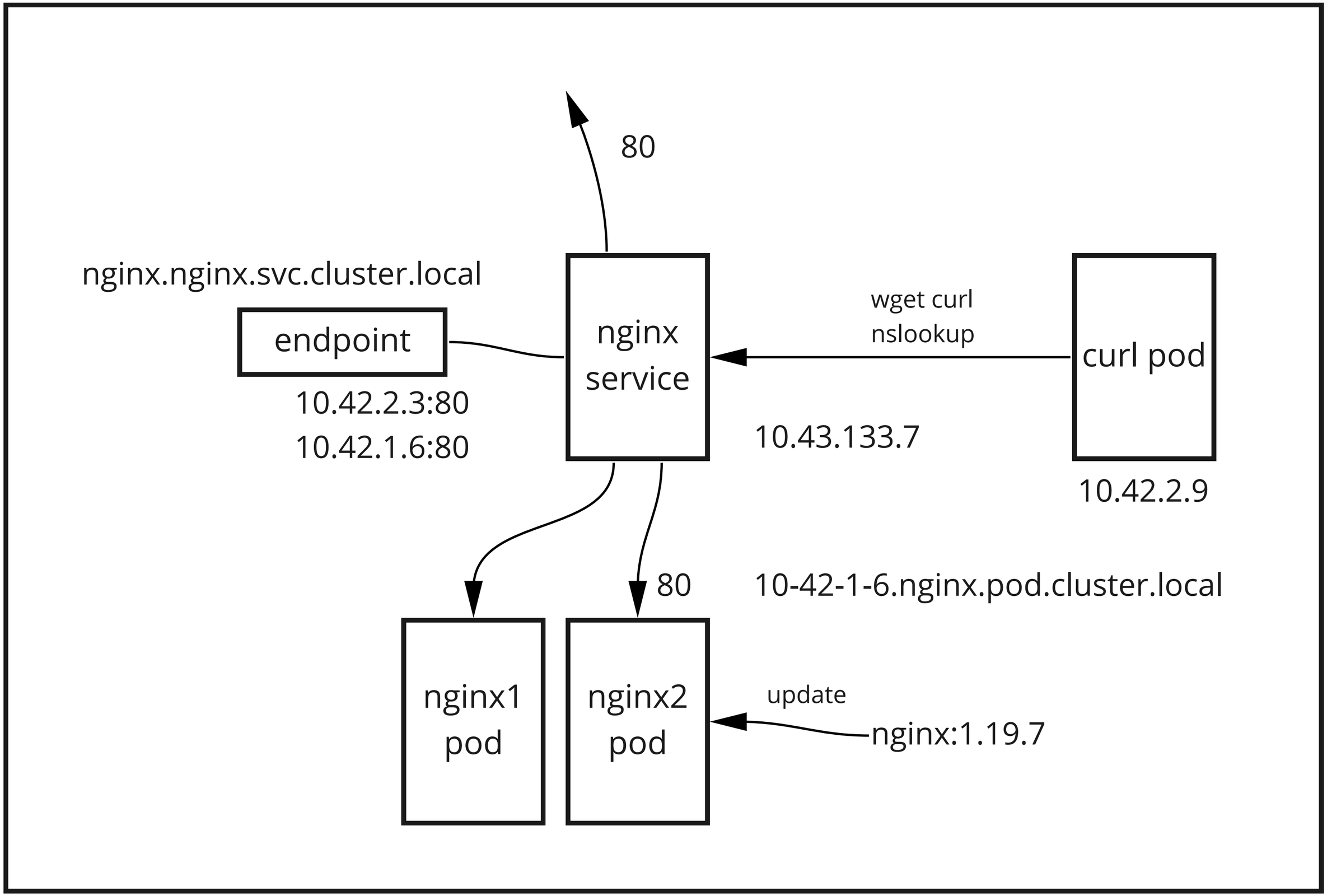


https://kubernauts.de/.galleries/downloads/kubernetes-pod-cheatsheet.png

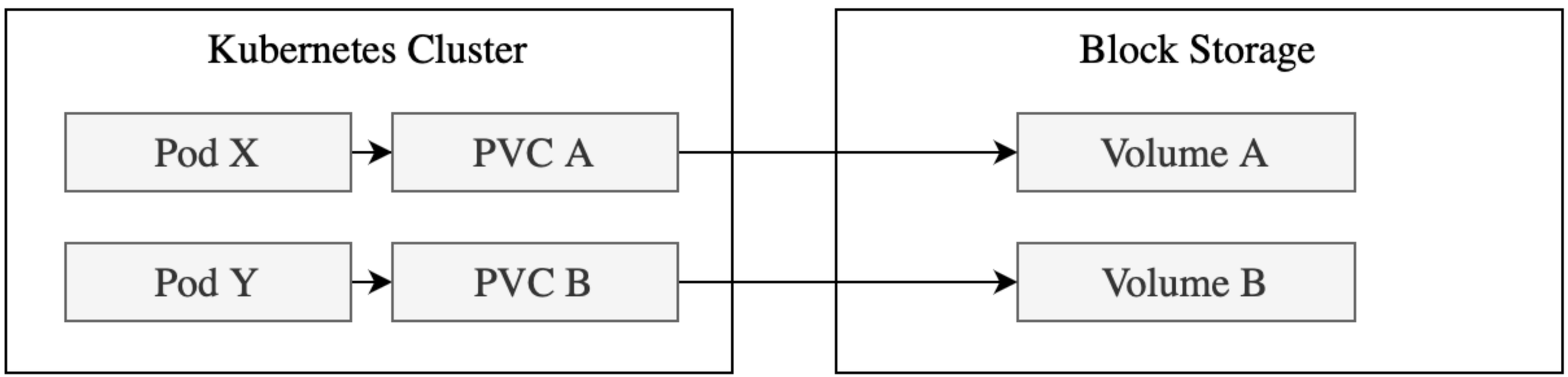
High Level View



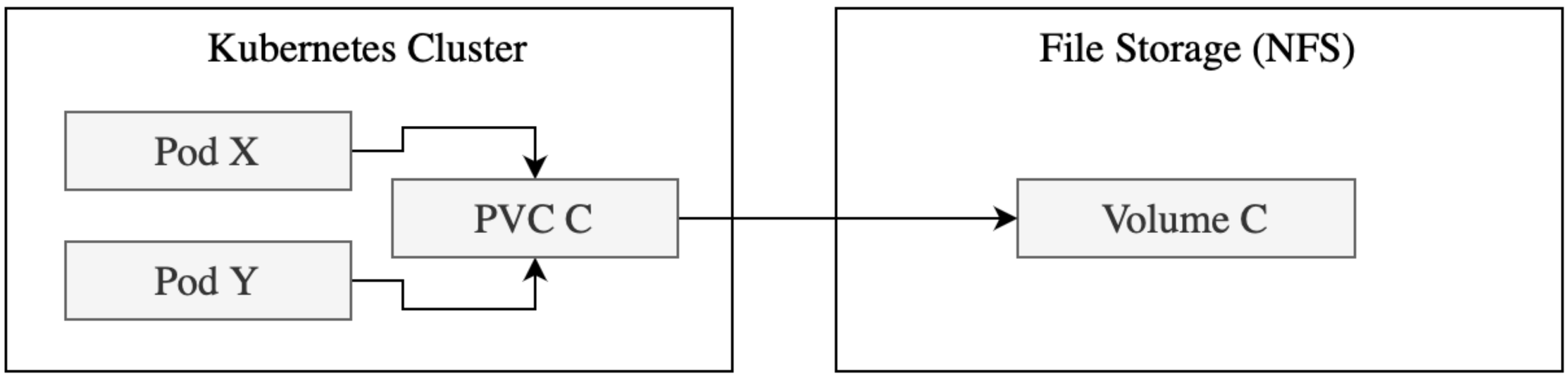
NGINX Namespace

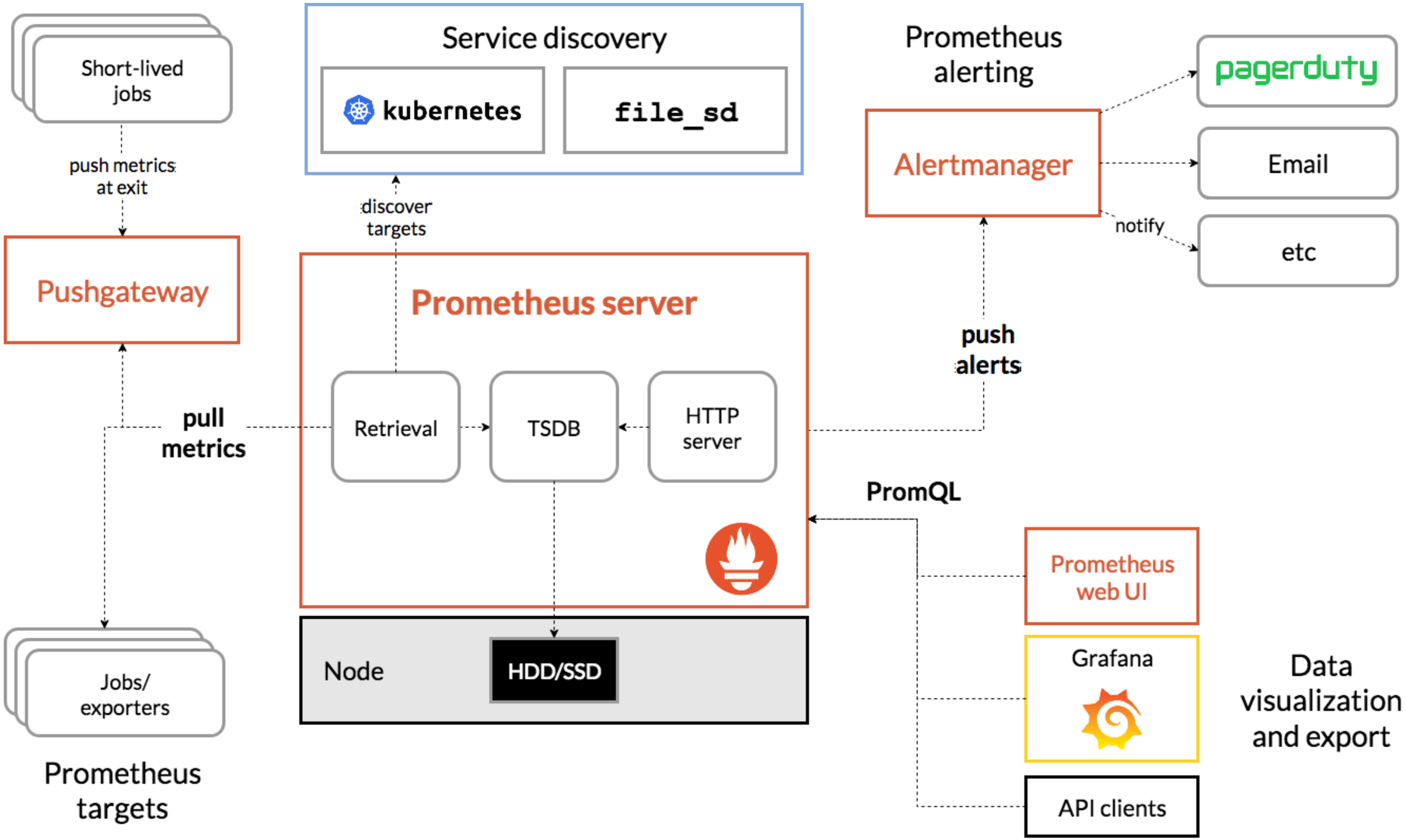


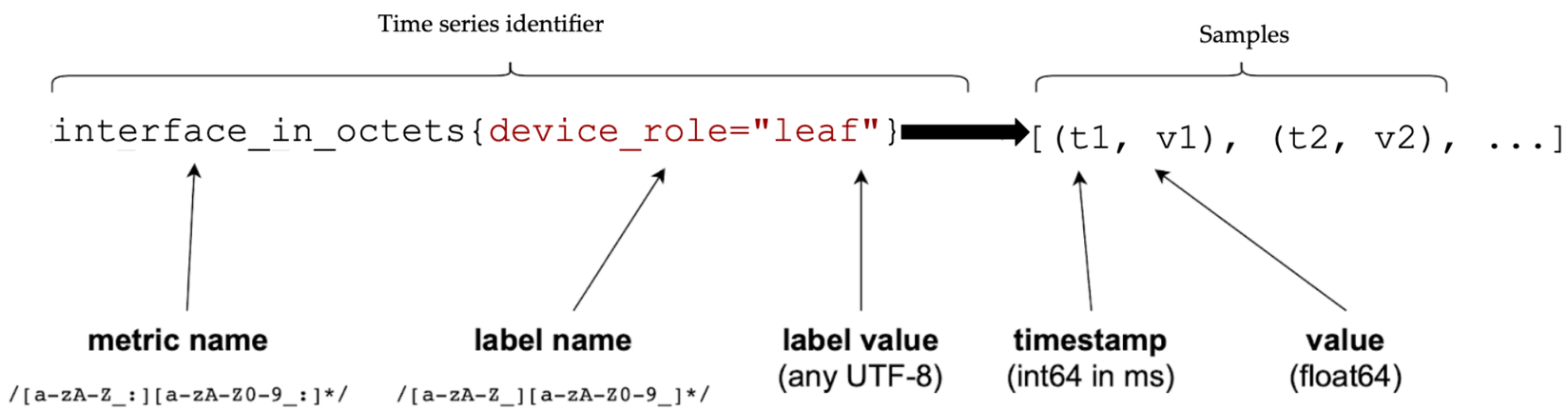
ReadWriteOnce

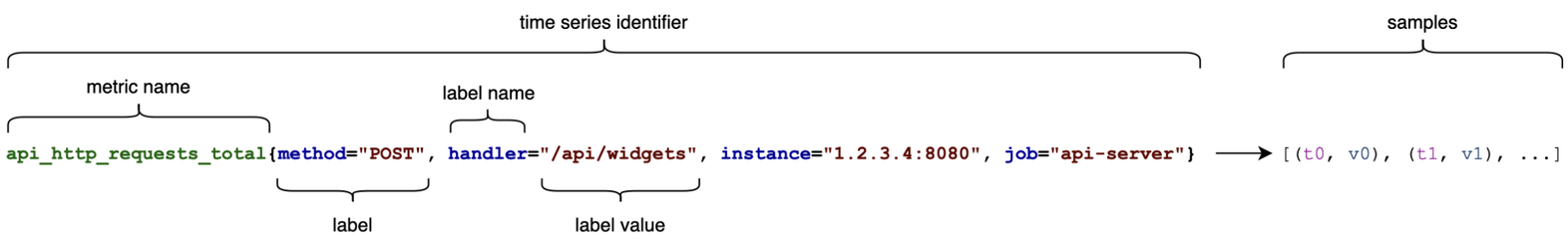


ReadWriteMany

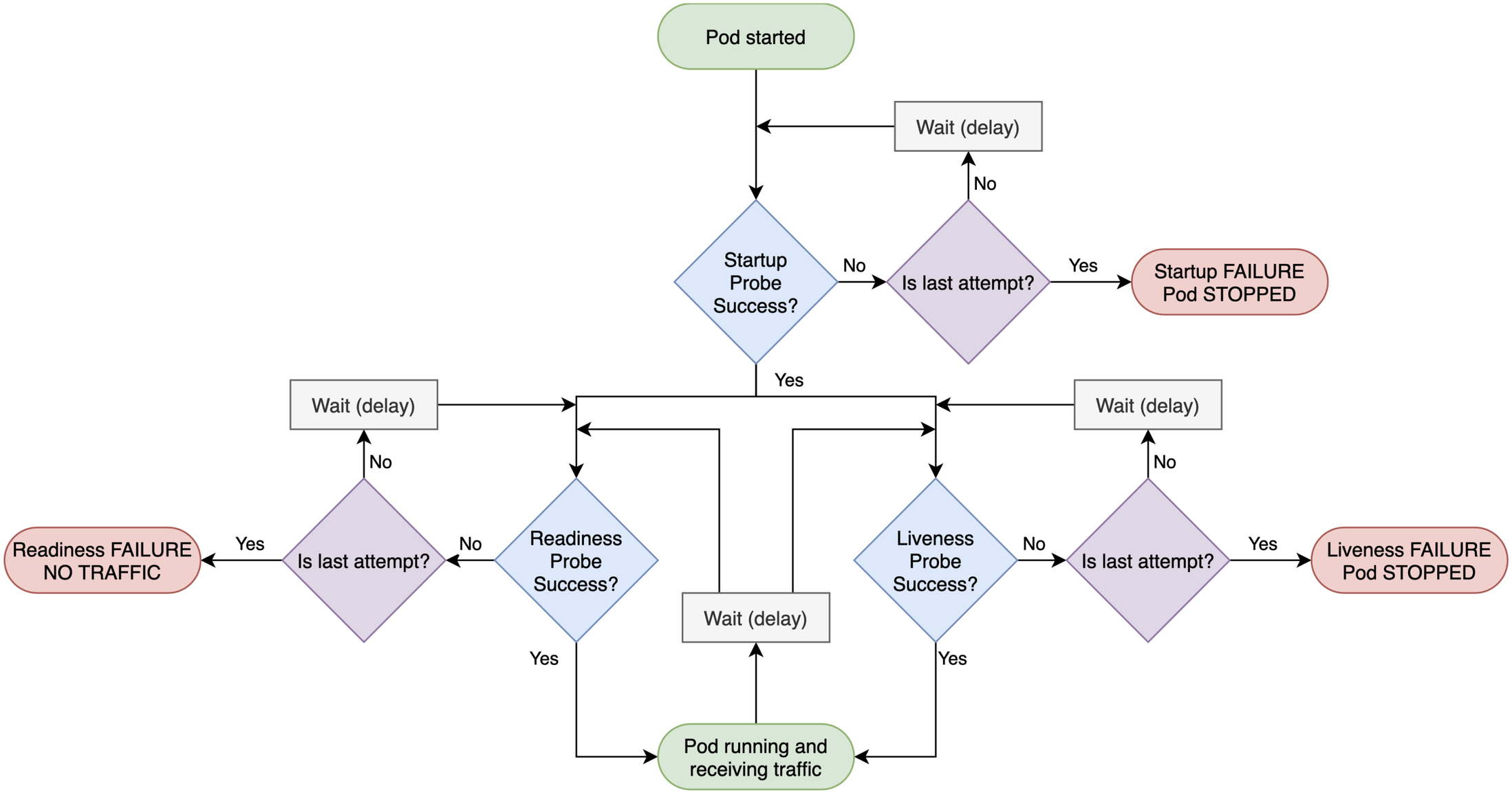



Short-lived jobs
push metrics at exit
Pushgateway
pull metrics
Service discovery
kubernetes
file_sd
discover targets
Prometheus server
Retrieval
TSDB
HTTP server
Node
HDD/SSD
Jobs/ exporters
Prometheus targets
Prometheus alerting
Alertmanager
pagerduty
Email
etc
notify
push alerts
PromQL
Prometheus web UI
Grafana
API clients
Data visualization and export

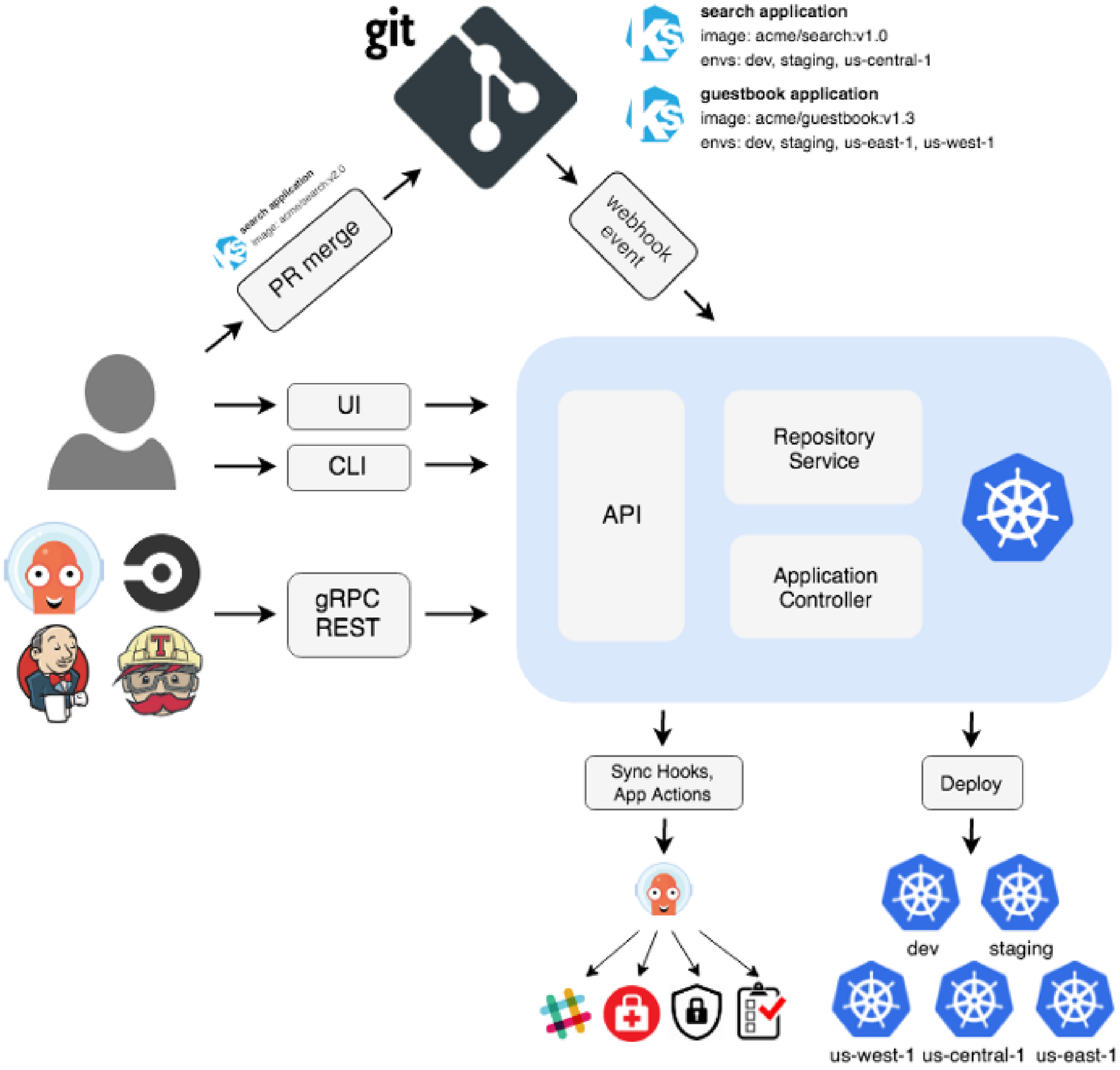


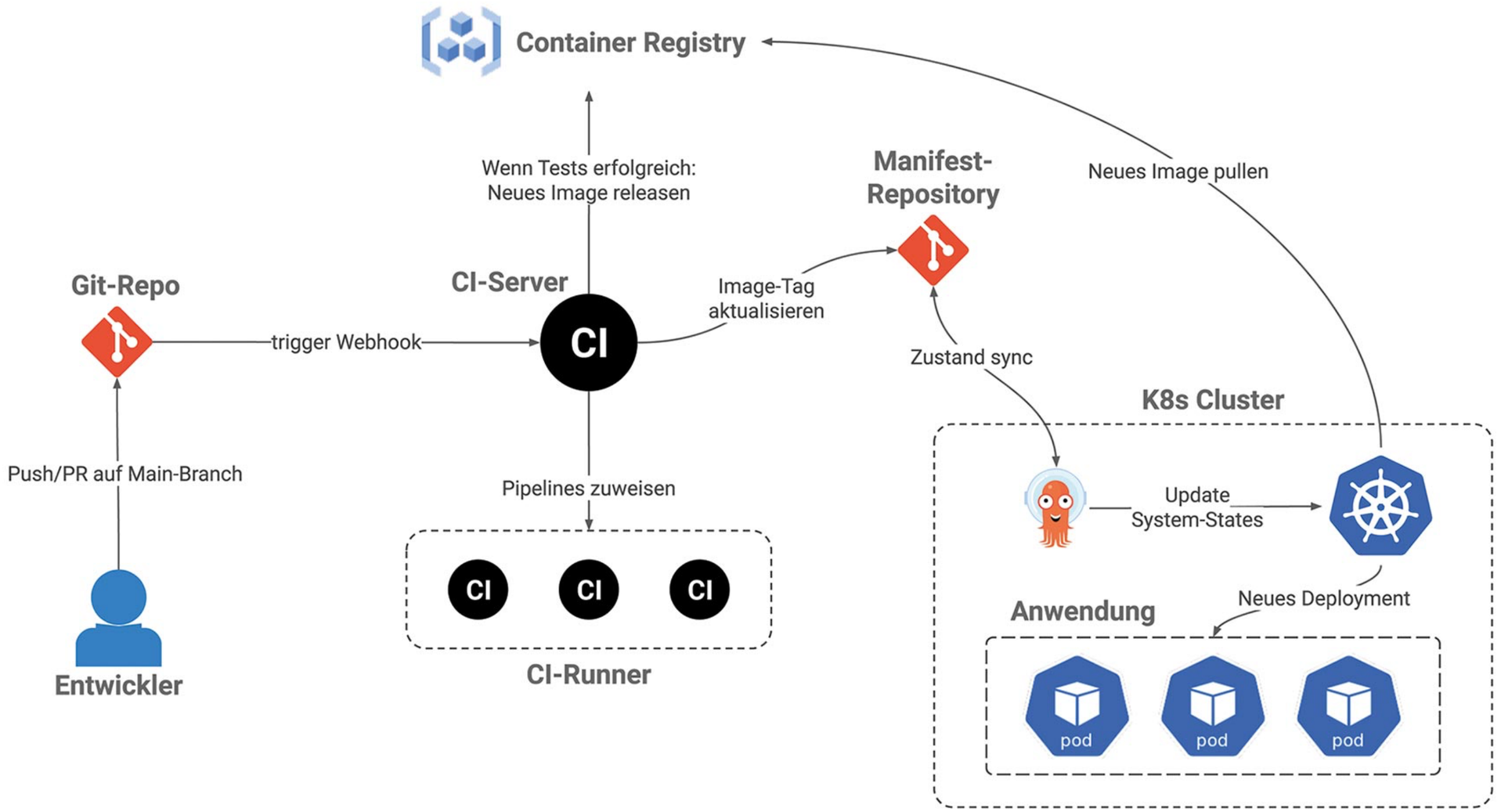


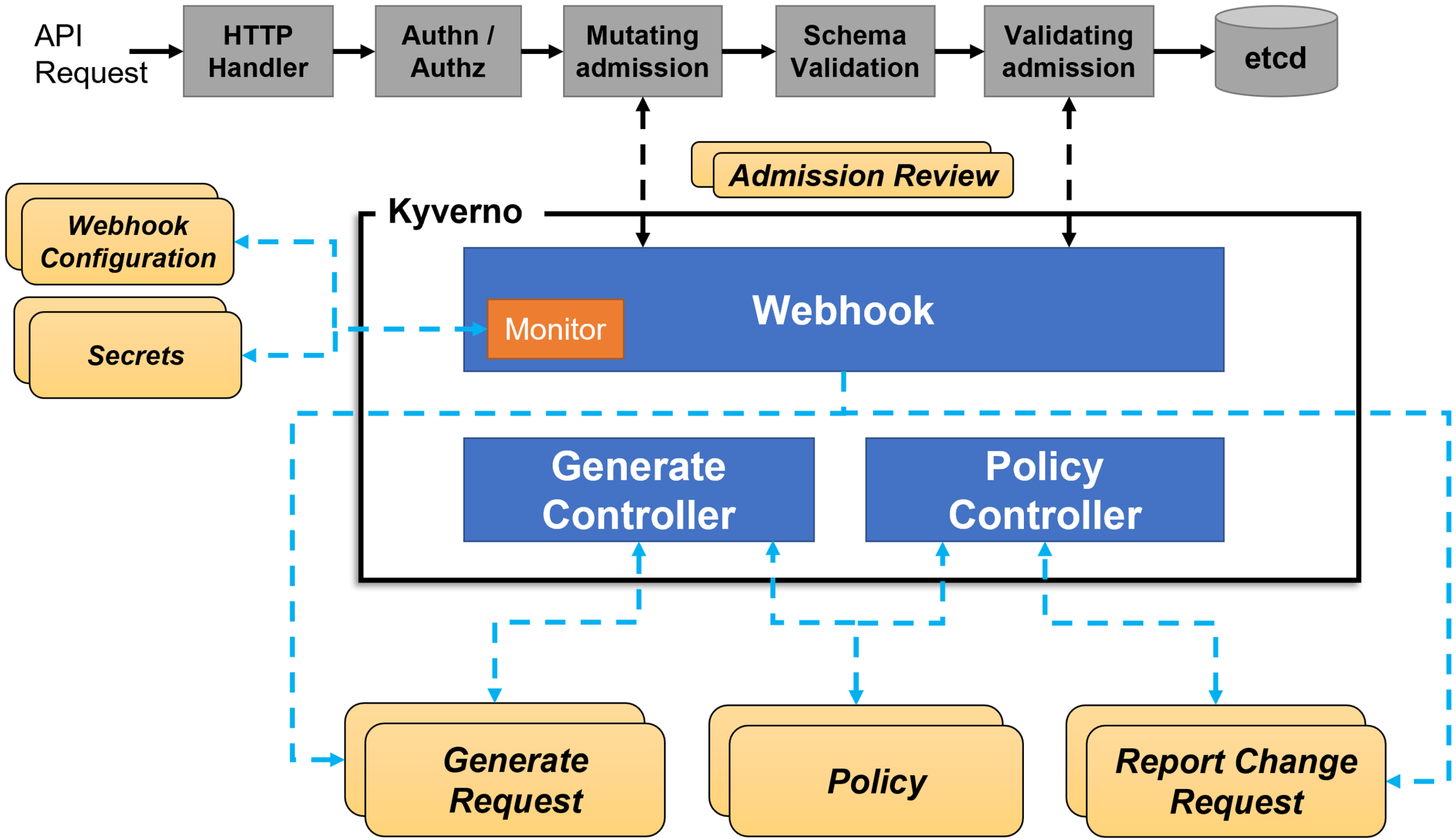
https://promlabs.com/blog/2020/07/02/selecting-data-in-promql



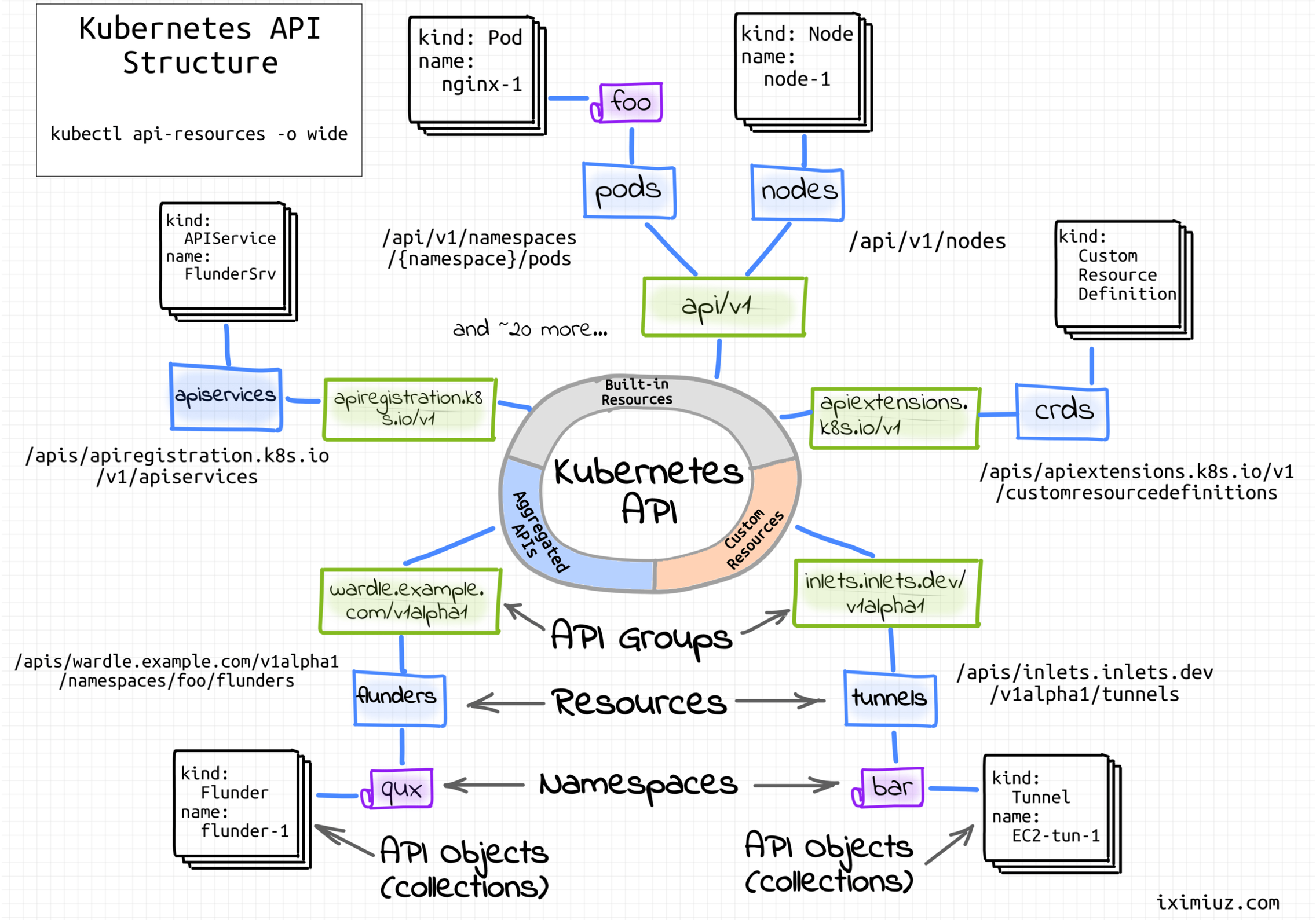
https://andrewlock.net/deploying-asp-net-core-applications-to-kubernetes-part-6-adding-health-checks-with-liveness-readiness-and-startup-probes/


git
search application
image: acme/search:v1.0
envs: dev, staging, us-central-1
guestbook application
image: acme/guestbook:v1.3
envs: dev, staging, us-east-1, us-west-1
search application
image: acme/search:v2.0
PR merge
webhook event
UI
CLI
gRPC
REST
API
Repository Service
Application Controller
Sync Hooks, App Actions
Deploy
dev
staging
us-west-1
us-central-1
us-east-1


Container Registry
Wenn Tests erfolgreich:
Neues Image releasen
Manifest-
Repository
Neues Image pullen
Git-Repo
CI-Server
trigger Webhook
CI
Image-Tag
aktualisieren
Zustand sync
K8s Cluster
Push/PR auf Main-Branch
Pipelines zuweisen
Update
System-States
Anwendung
Neues Deployment
CI
CI
CI
pod
pod
pod
Entwickler
CI-Runner


API Request
HTTP Handler
Authn / Authz
Mutating admission
Schema Validation
Validating admission
etcd
Admission Review
Kyverno
Webhook Configuration
Secrets
Monitor
Webhook
Generate Controller
Policy Controller
Generate Request
Policy
Report Change Request

- Personen
  - alexander
  - jannik
  - Maximilian
  - dennis
  - Joern
  - riccardo
  - dalibor
  - michael-t
  - michael-f
  - sascha
  - Peter
    - Mentor
    - Systemarchitekt
    - Apache Member
      - Apache Tomcat Committer
    - Geschäftsführer und Gründer der bee42
    - Tech
      - Java seit 1995
      - Container seit 2013
      - Container Trainings seit 2014
      - DevOps und SRE
      - Agile
      - Automation
      - Container on Embedded Machines - Raspberry PI
    - Hobby
      - Einradhockey
      - Wandern
      - Lesen



https://iximiuz.com/en/posts/kubernetes-api-structure-and-terminology/