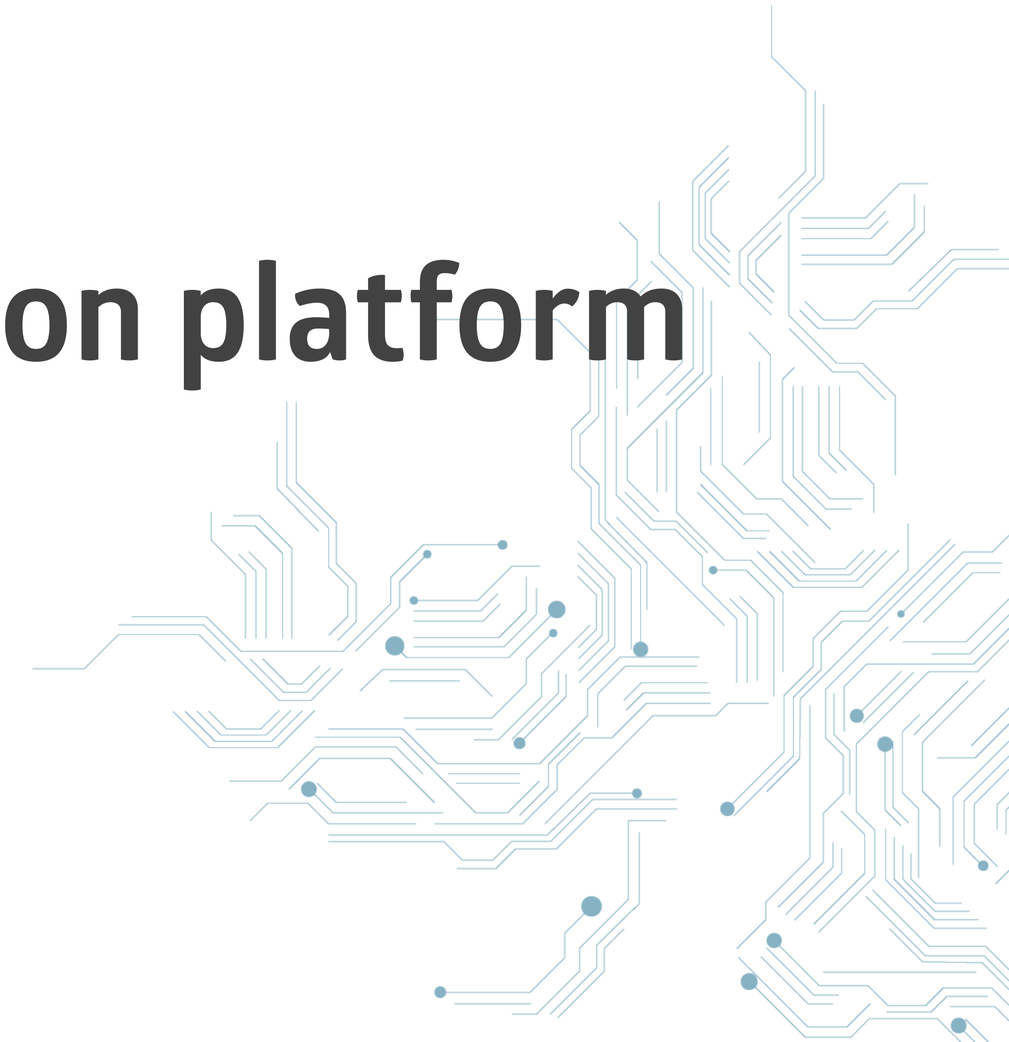
SysEleven
engage · build · run

Kubernetes Workshop

Bernd May & Max Rosin

kubernetes

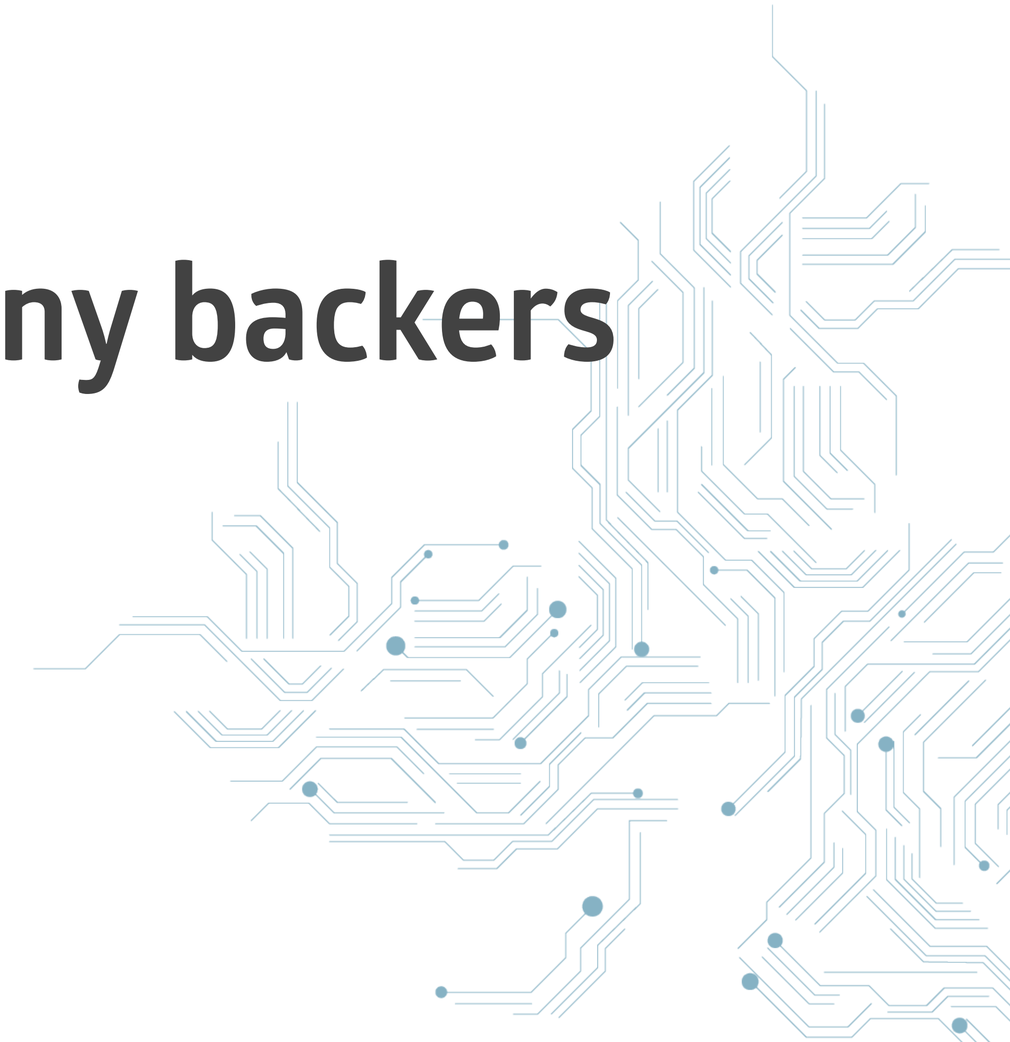
Container orchestration platform



Deploy, run and scale your services in isolated containers

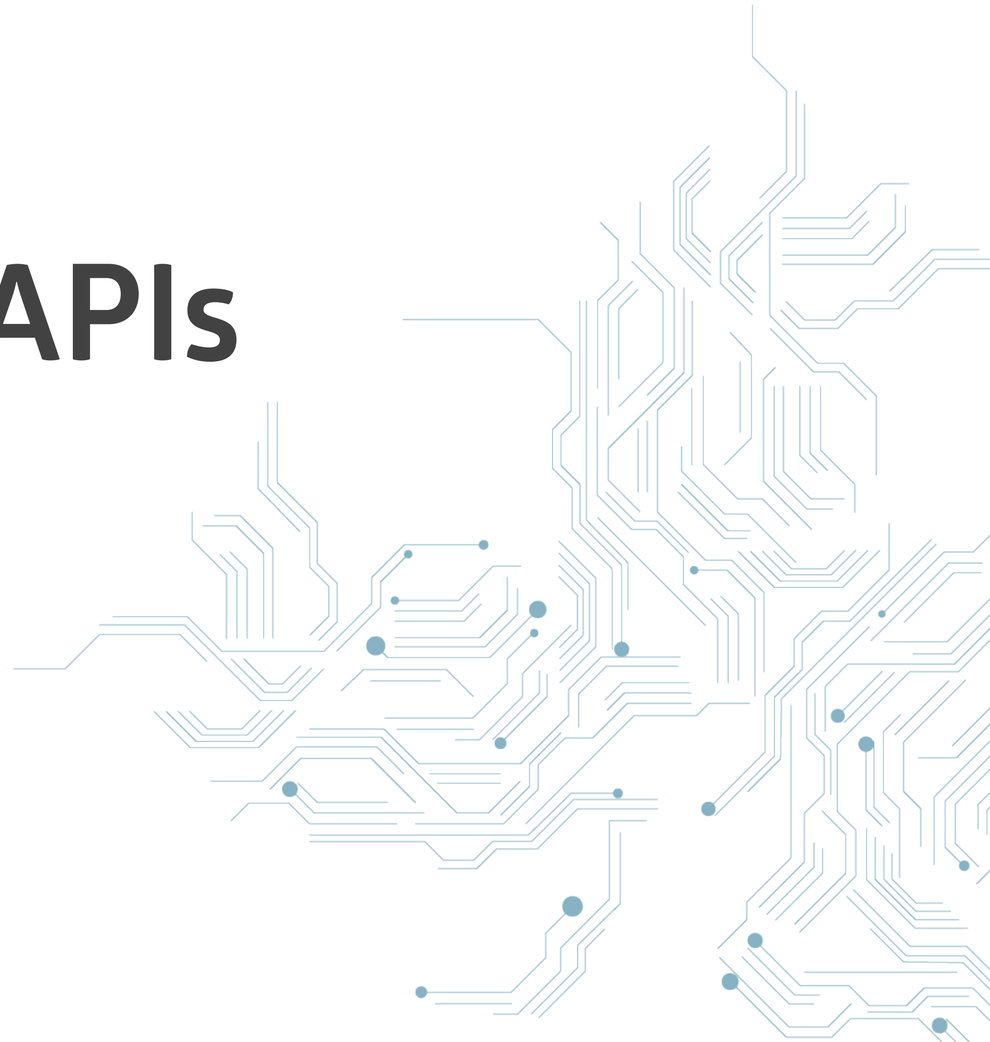
Very Powerful

Lot’s of large company backers



No vendor lock in

Standardized APIs



Runs on

Your laptop

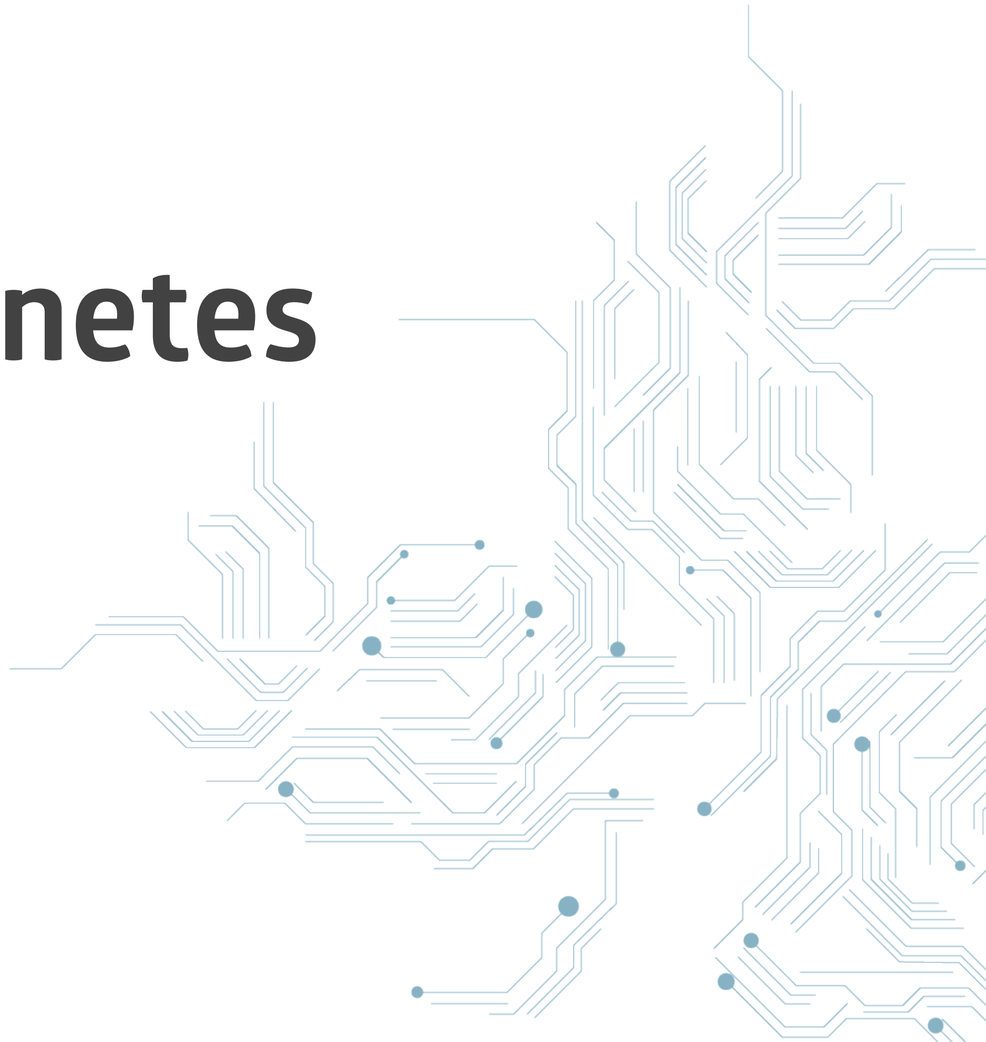

Kubernetes
General
File Sharing
Disk
Advanced
Proxies
Daemon
Kubernetes
Reset
Enable Kubernetes
Start a Kubernetes single-node cluster when starting Docker Desktop.
Deploy Docker Stacks to Kubernetes by default
Make Kubernetes the default orchestrator for "docker stack" commands
(changes "~/.docker/config.json")
Show system containers (advanced)
Show Kubernetes internal containers when using Docker commands.
Apply
Docker Desktop is running

Bare metal

Cloud Providers

And if you don't want to install and maintain Kubernetes yourself

Managed Kubernetes



CNCF Cloud Native Interactive Landscape

CLOUD NATIVE COMPUTING FOUNDATION

The Cloud Native Trail Map (png, pdf) is CNCF's recommended path through the cloud native landscape. The cloud native landscape (png, pdf) and serverless landscape (png, pdf) are dynamically generated below. Please open a pull request to correct any issues. Greyed logos are not open source. Last Updated: 2019-02-12 06:44:45Z

You are viewing 33 cards with a total of 180 stars, market cap of $4.01T and funding of $1.19B.

Landscape | Card Mode | Serverless

Share 462

Platform - Certified Kubernetes - Hosted (33)

Alibaba Cloud Container Service for Kubernetes — MCap: $434B
Alibaba Cloud

Amazon Elastic Container Service for Kubernetes (EKS) — MCap: $782B
Amazon Web Services

Azure (ACS) Engine — ★174
Microsoft — MCap: $808B

Azure Kubernetes Service (AKS) — MCap: $808B
Microsoft

Azure Stack (ACS) Engine — ★4
Microsoft — MCap: $808B

Baidu Cloud Container Engine — MCap: $58.6B
Baidu

BoCloud BeyondcentContainer — Funding: $15.3M
Bocloud

Catalyst Kubernetes Service — ★2
Catalyst Cloud

Cisco Container Platform — MCap: $214B
Cisco

DigitalOcean Kubernetes — Funding: $305M
DigitalOcean

EasyStack Kubernetes Service (EKS) — Funding: $110M
EasyStack

eBaoCloud
eBaoTech Corporation

eKing Cloud Container Platform
Hainan eKing Technology

ELASTX Private Kubernetes
ELASTX

Google Kubernetes Engine (GKE) — MCap: $763B
Google

HarmonyCloud Container Platform
Hangzhou Harmony Technology

Hasura — Funding: $1.6M
Hasura

Huawei Cloud Container Engine (CCE)
Huawei Technologies

IBM Cloud Kubernetes Service — MCap: $122B
IBM

Intellect FABRIC — MCap: $22.4B
Intellect Design Arena

Nirmata Managed Kubernetes
Nirmata

Nutanix Karbon — MCap: $9.48B
Nutanix

Oracle Container Engine — MCap: $184B
Oracle

OVH Managed Kubernetes Service — Funding: $737M
OVH.com

Rackspace Kubernetes-as-a-Service — Funding: $17.8M
Rackspace

Samsung SDS Kubernetes Service
Samsung SDS

SAP Certified Gardener — MCap: $126B
SAP

SysEleven MetaKube
SysEleven

Tencent Kubernetes Engine (TKE) — MCap: $421B
Tencent Holdings

TenxCloud Container Engine (TCE)
TenxCloud

VEXXHOST Kubernetes Container Service — 0
VEXXHOST

VMware Cloud PKS — MCap: $65.2B
VMware

ZTE TECS
ZTE

Easy setup

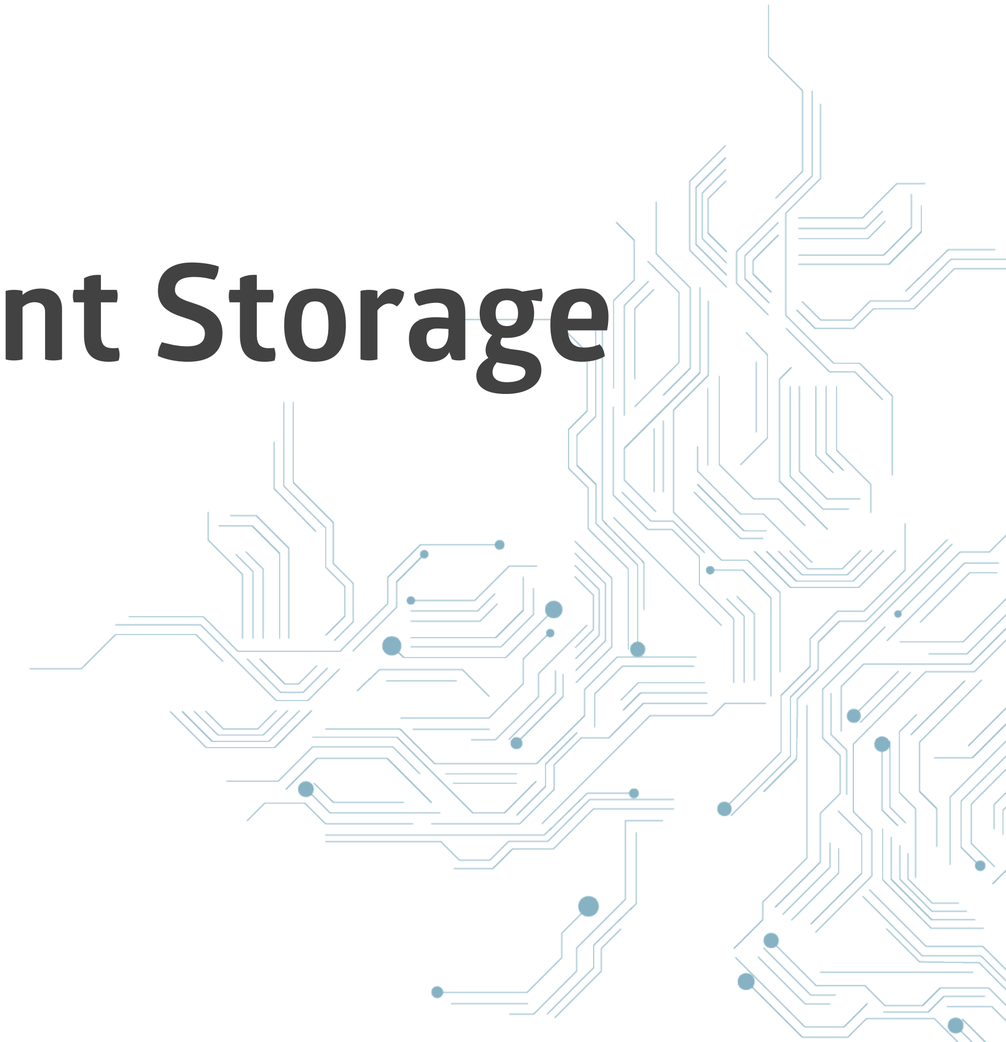
Easy upgrades

Easy scaling

Features

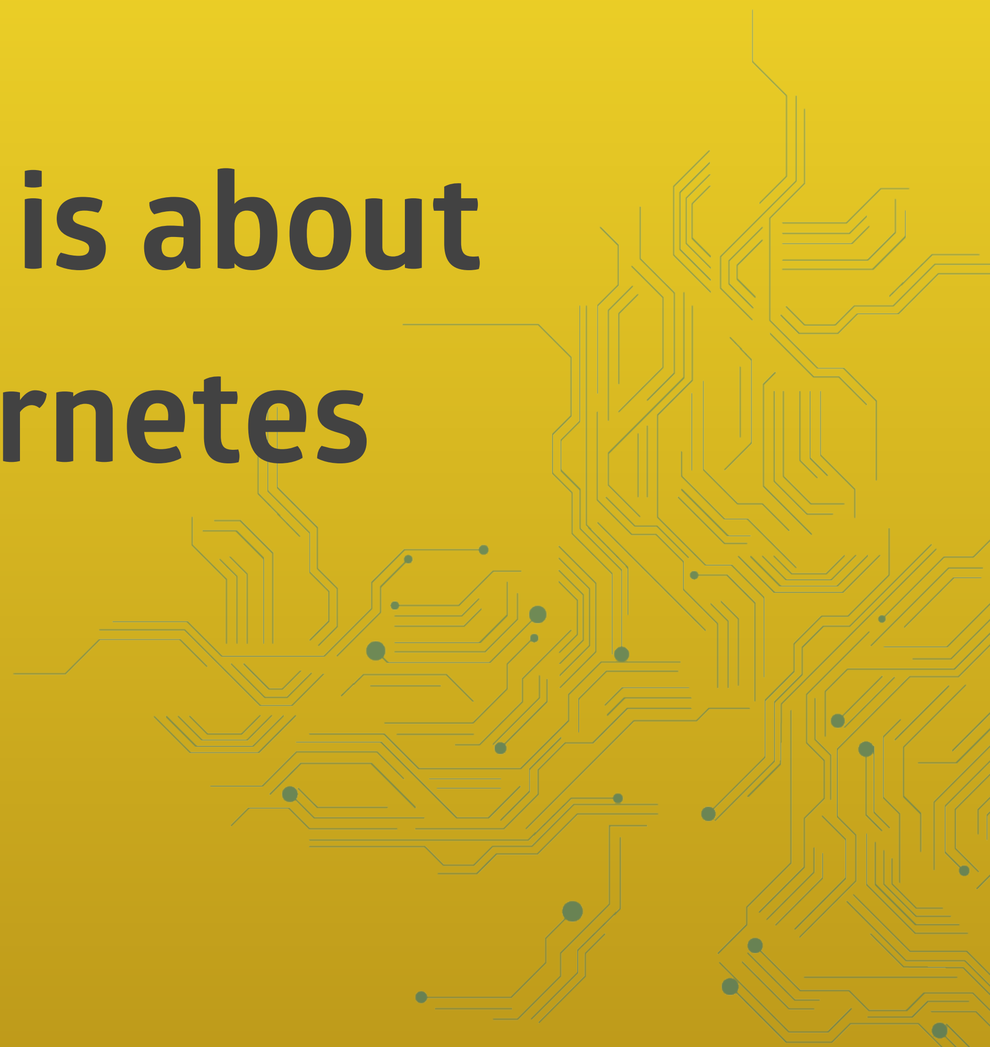
Load Balancing

Distributed Persistent Storage



Backups

But this workshop is about how to use Kubernetes

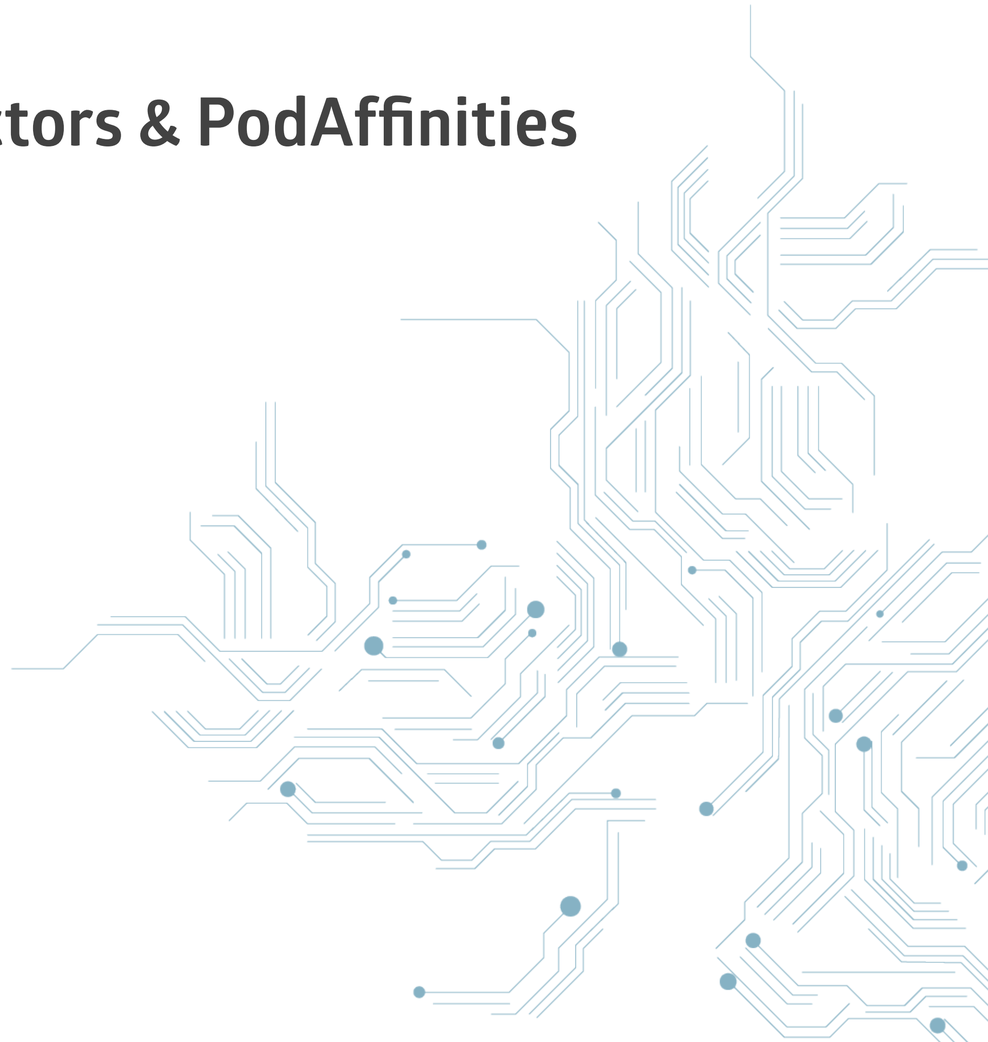


Learning curve

Agenda


METAKUBE

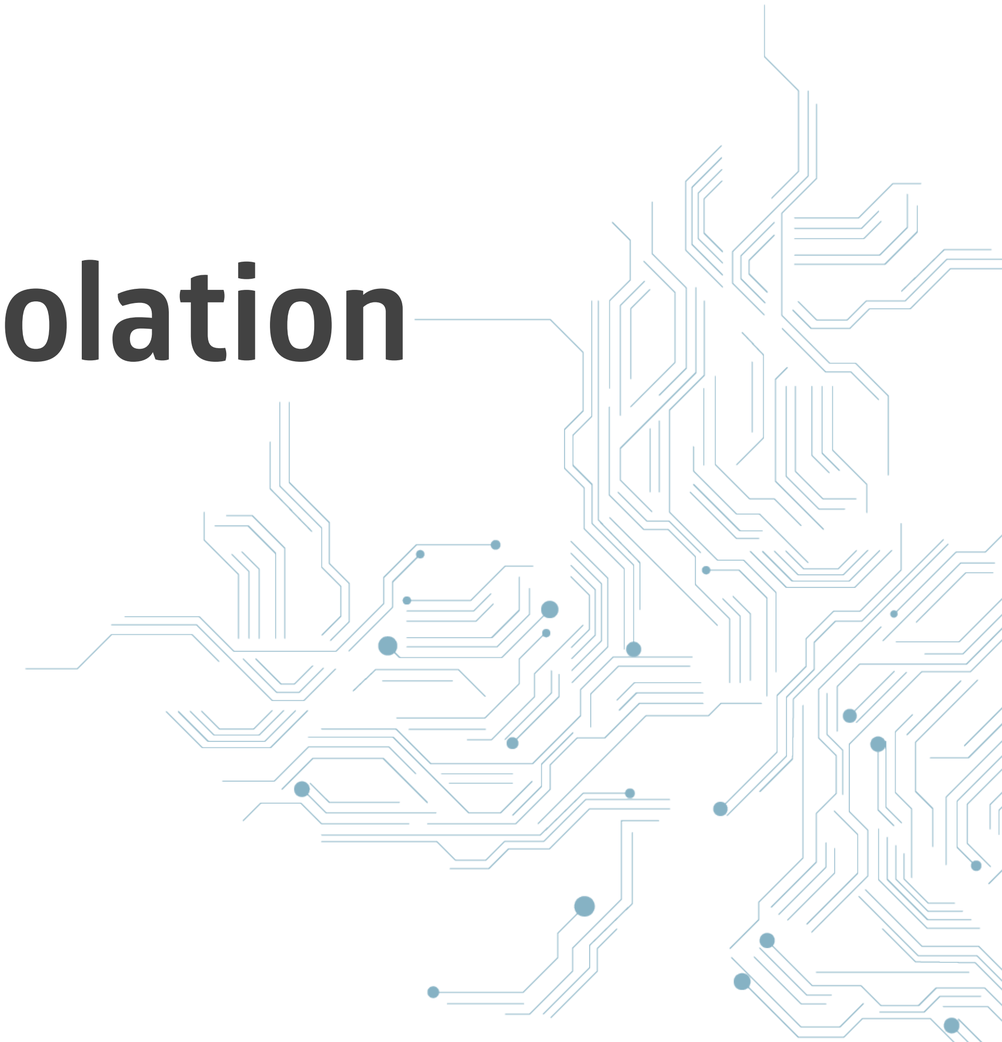
- Deployments
- Readiness and Liveness-Probes, NodeSelectors & PodAffinities
- ConfigMaps & Secrets
- External DNS
- Let'sEncrypt with cert-manager
- nginx-ingress-controller
- Helm



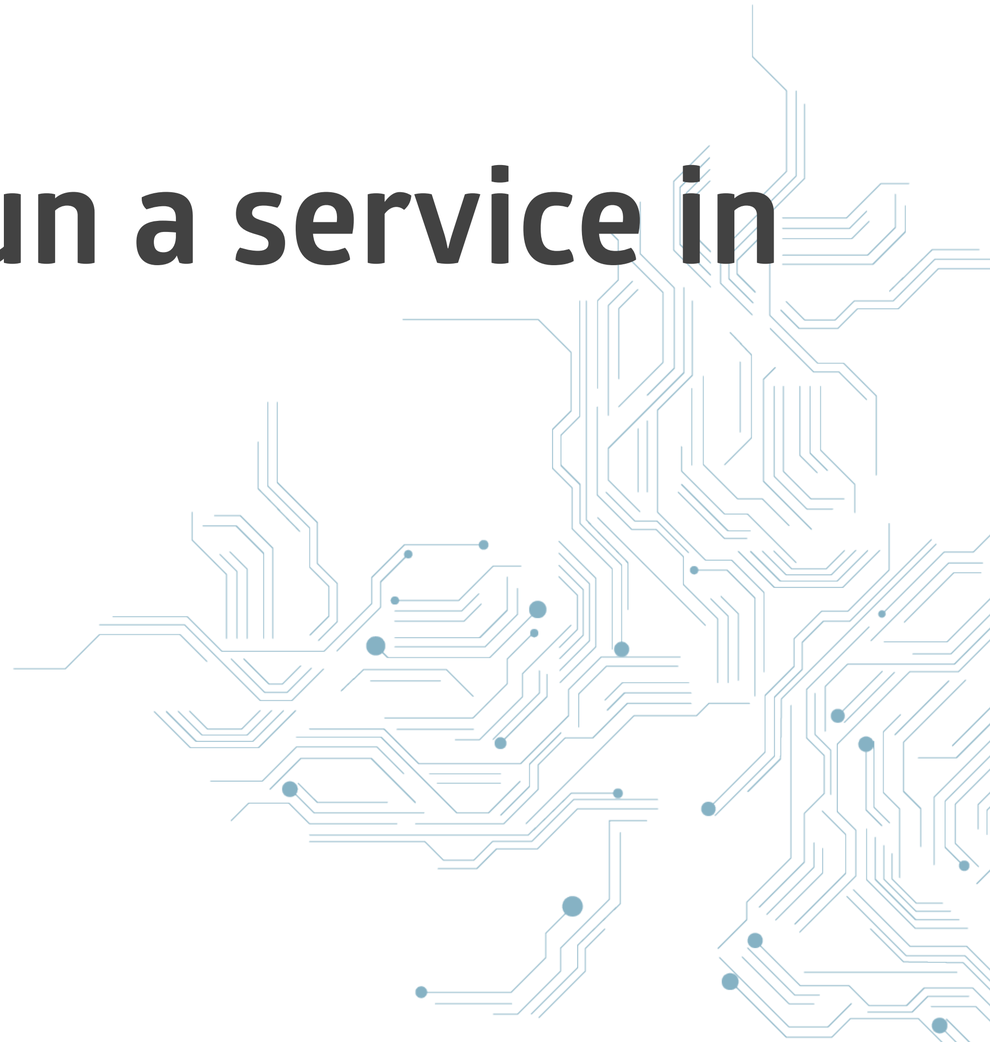
But first

Why containers?

Services run in isolation



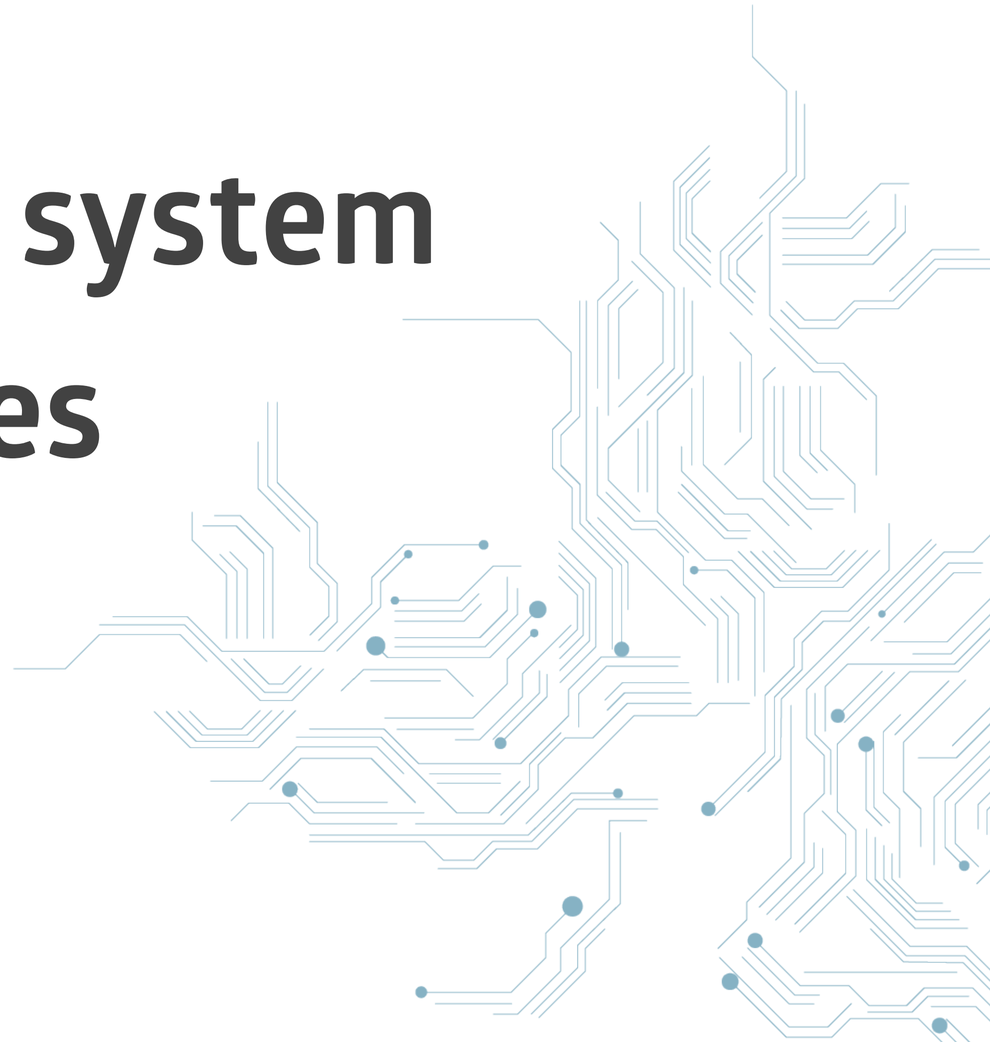
Everything needed to run a service in one image



Make things …

Easier to deploy

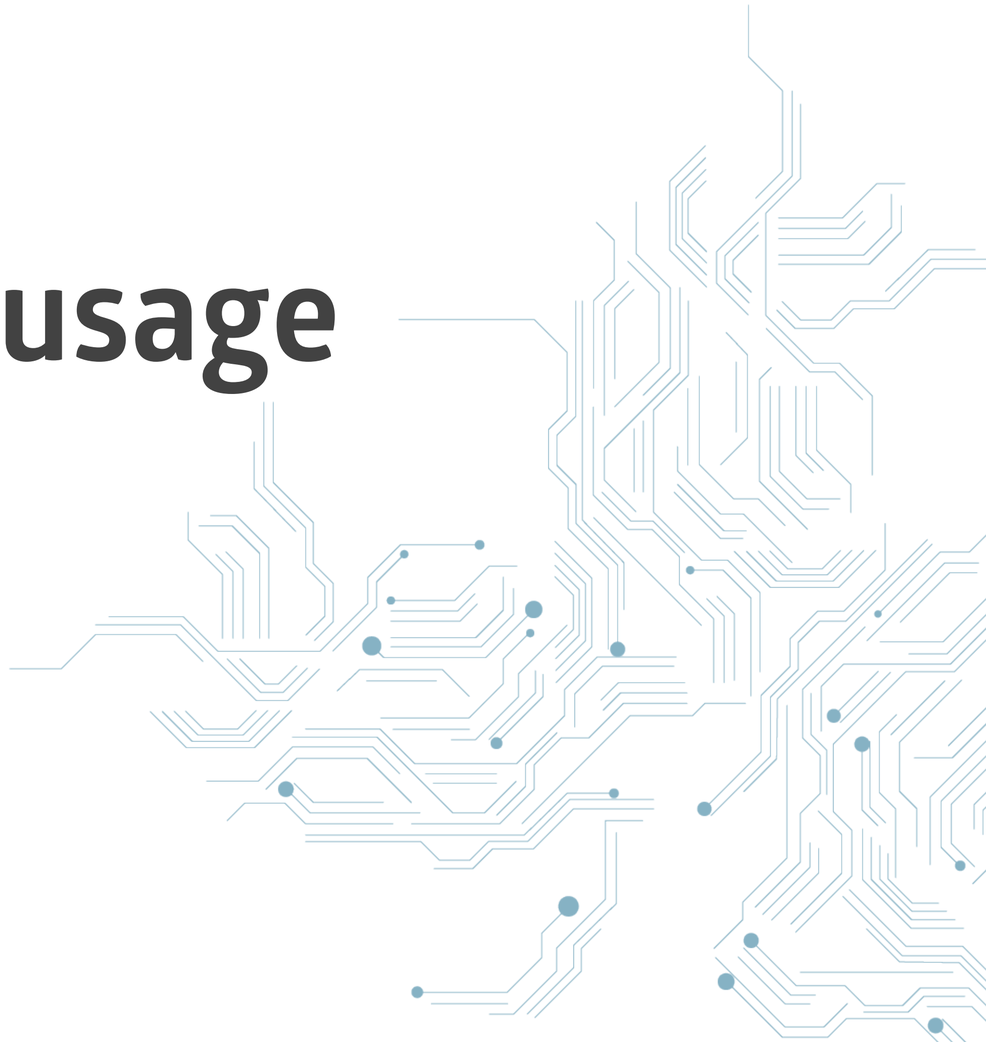
Easier to upgrade system dependencies

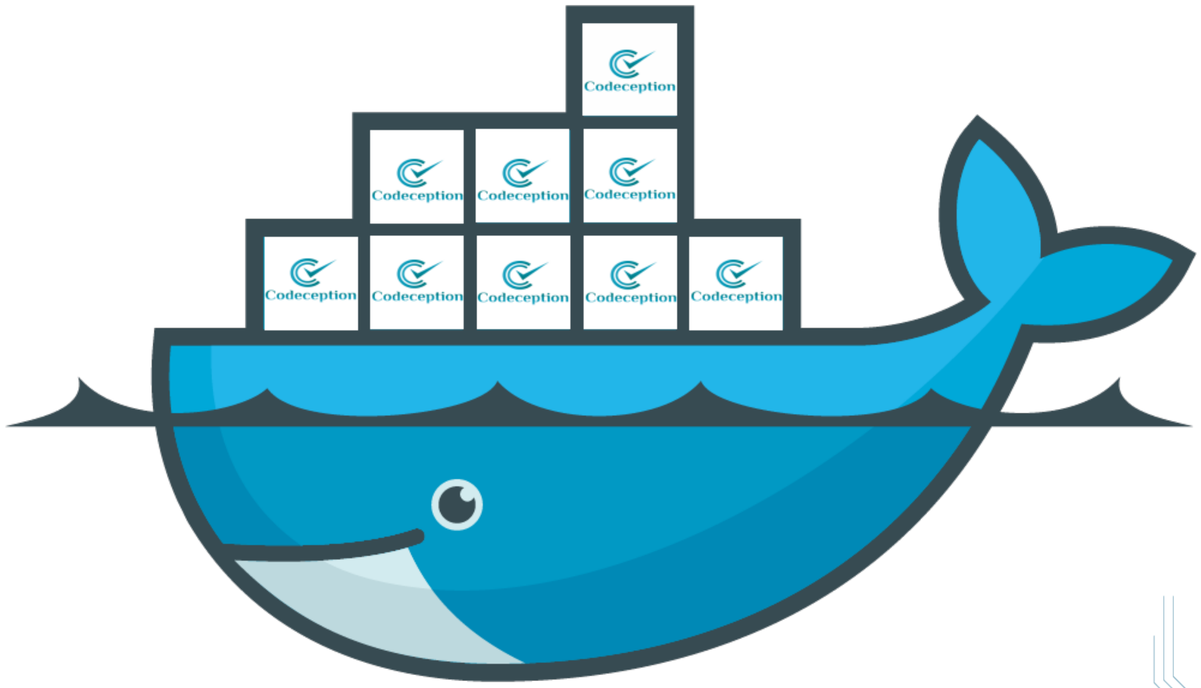


Easier to develop

Easier to scale

Better resource usage





docker

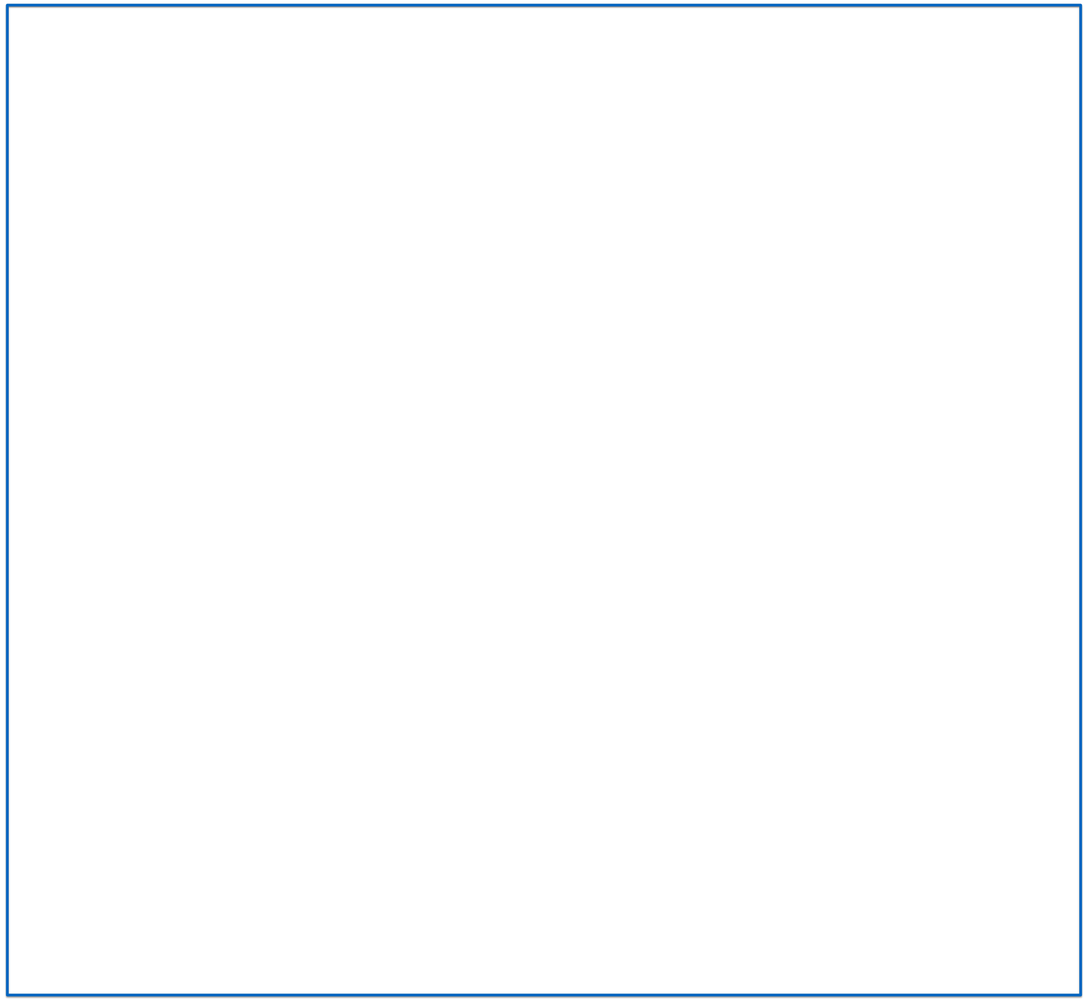
```
FROM php:7.2-apache
WORKDIR /var/www/html

RUN apt-get update -y && \
  apt-get install -y --no-install-recommends curl \
  rm -rf /var/lib/apt/lists/*

ENV TMP_DIR /tmp

COPY . /var/www/html/

EXPOSE 80
```

```
docker build -t gitlab.syseleven.de/syseleven/symfony-demo:2.0.0 .
```

```
docker run -p 8080:80 syseleven/symfony-demo:2.0.0
docker push syseleven/symfony-demo:2.0.0
```

Kubernetes helps you to run and deploy containers

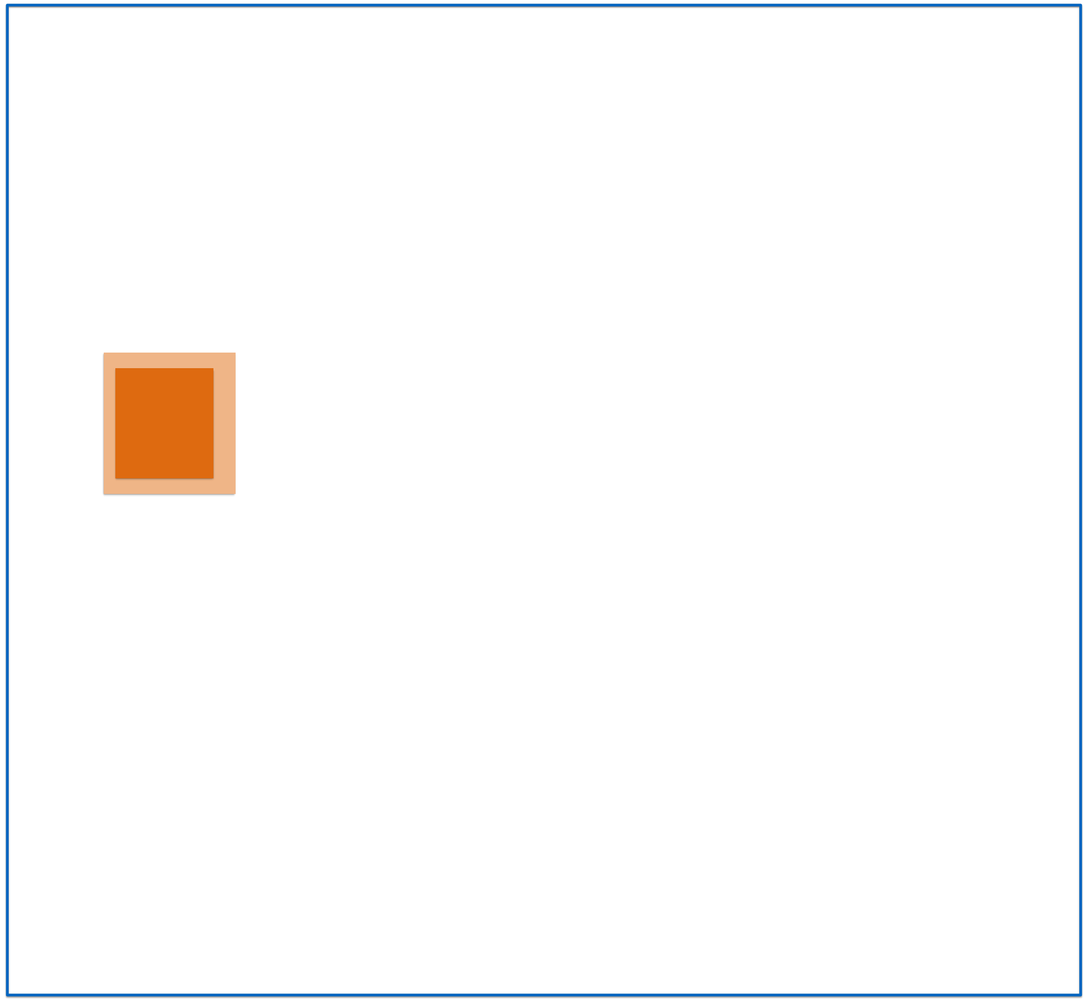
Let’s define some core concepts and terminology first



Kubernetes Cluster

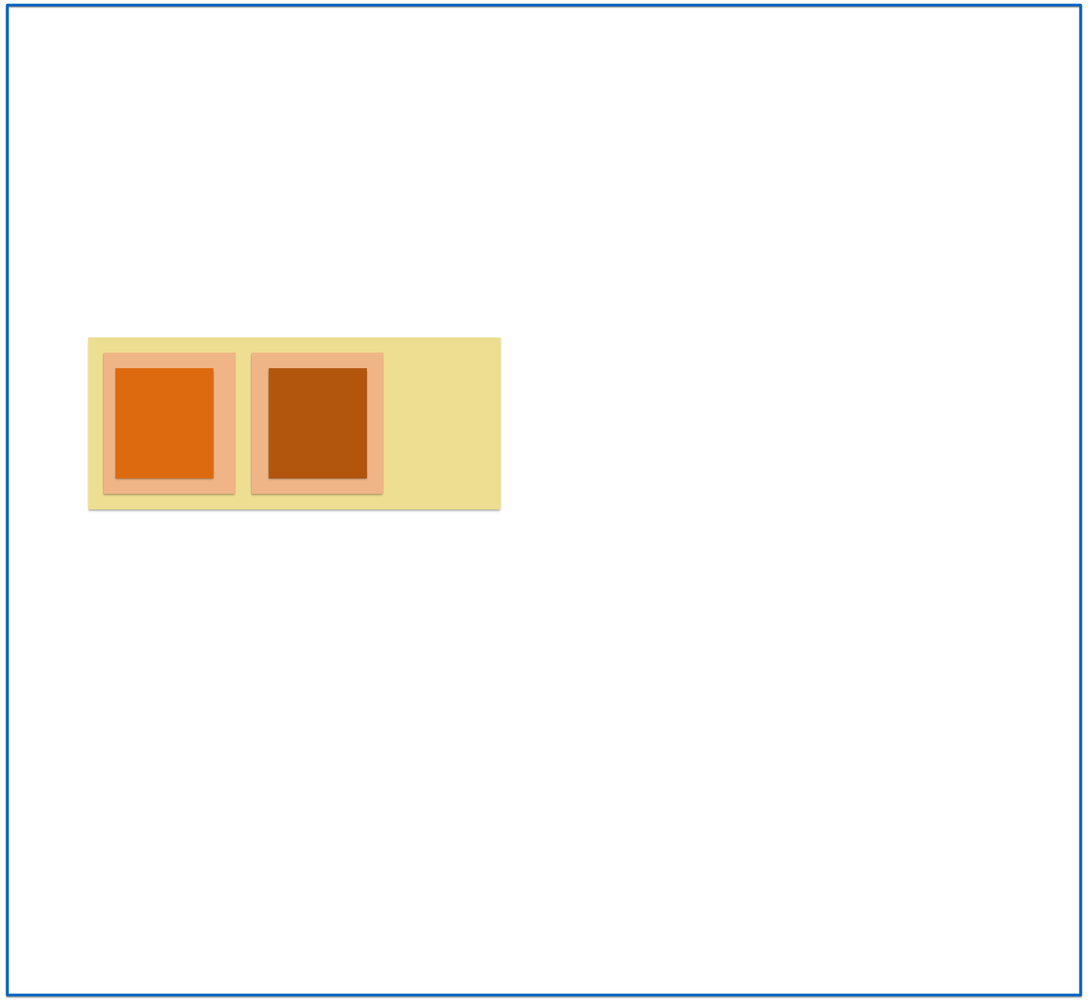
Image

- A docker image built from a Dockerfile that contains everything a service needs to run



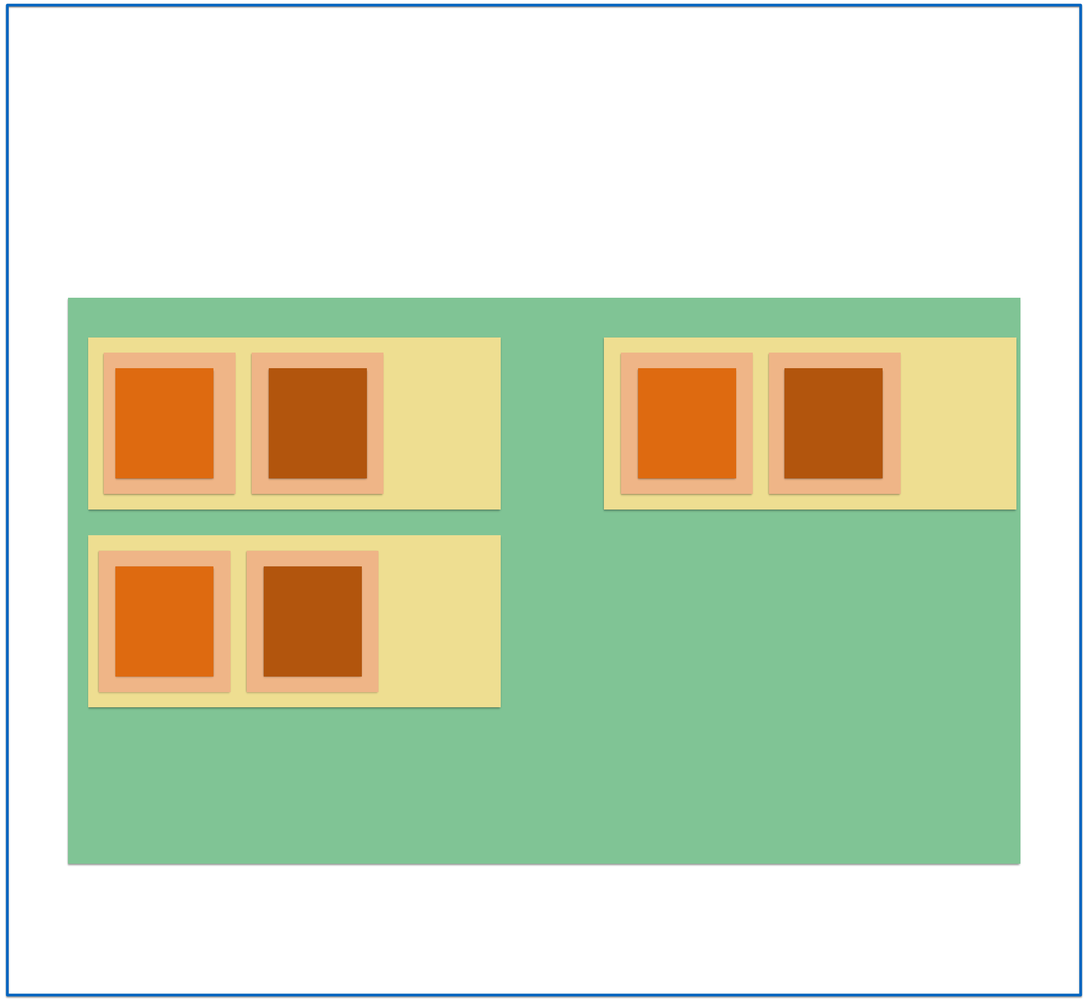
Container

- A container runs a docker image.
- Only 1 process can run inside of a container



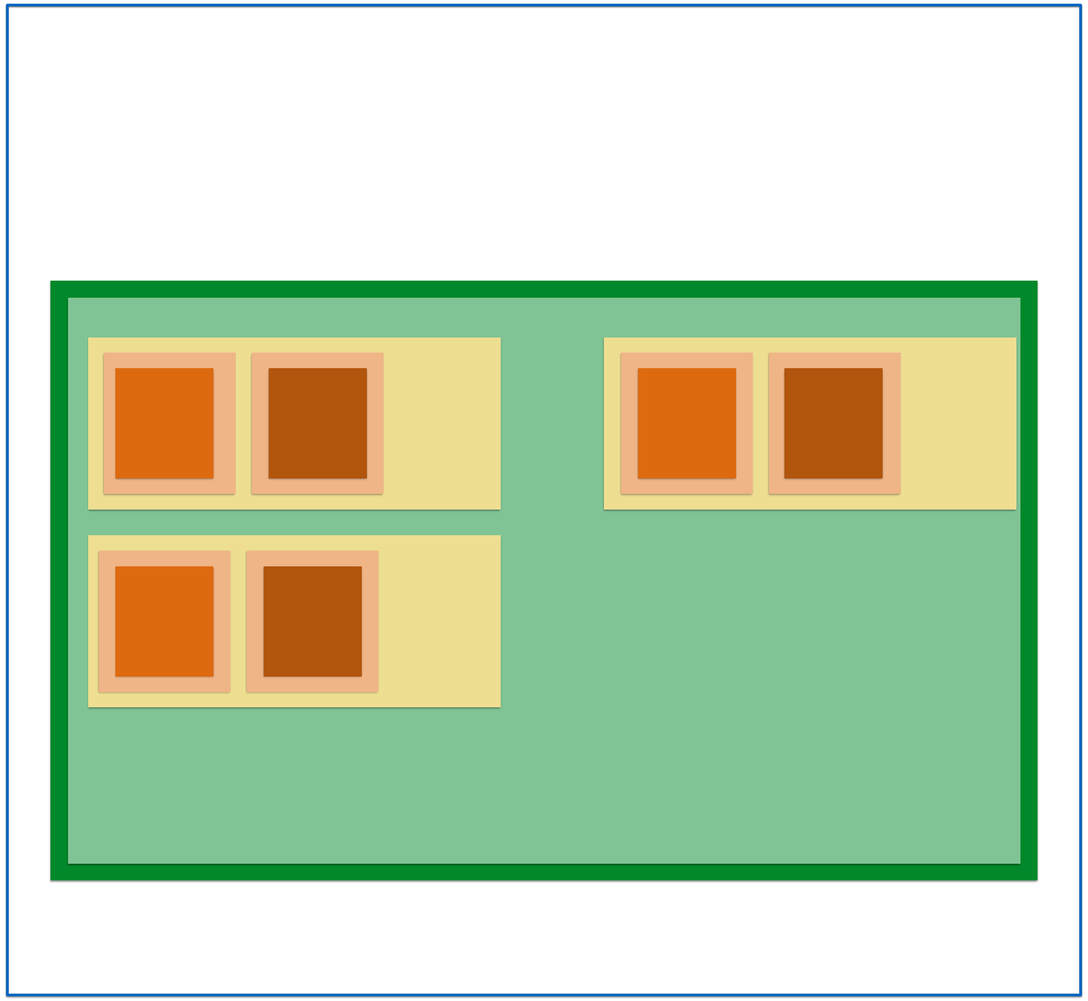
Pod

- A group of 1 or more containers
- Same port space
- Within a Pod: communication over localhost
- Every Pod has it's own IP
- All Pods can talk with each other
- IPs change all the time



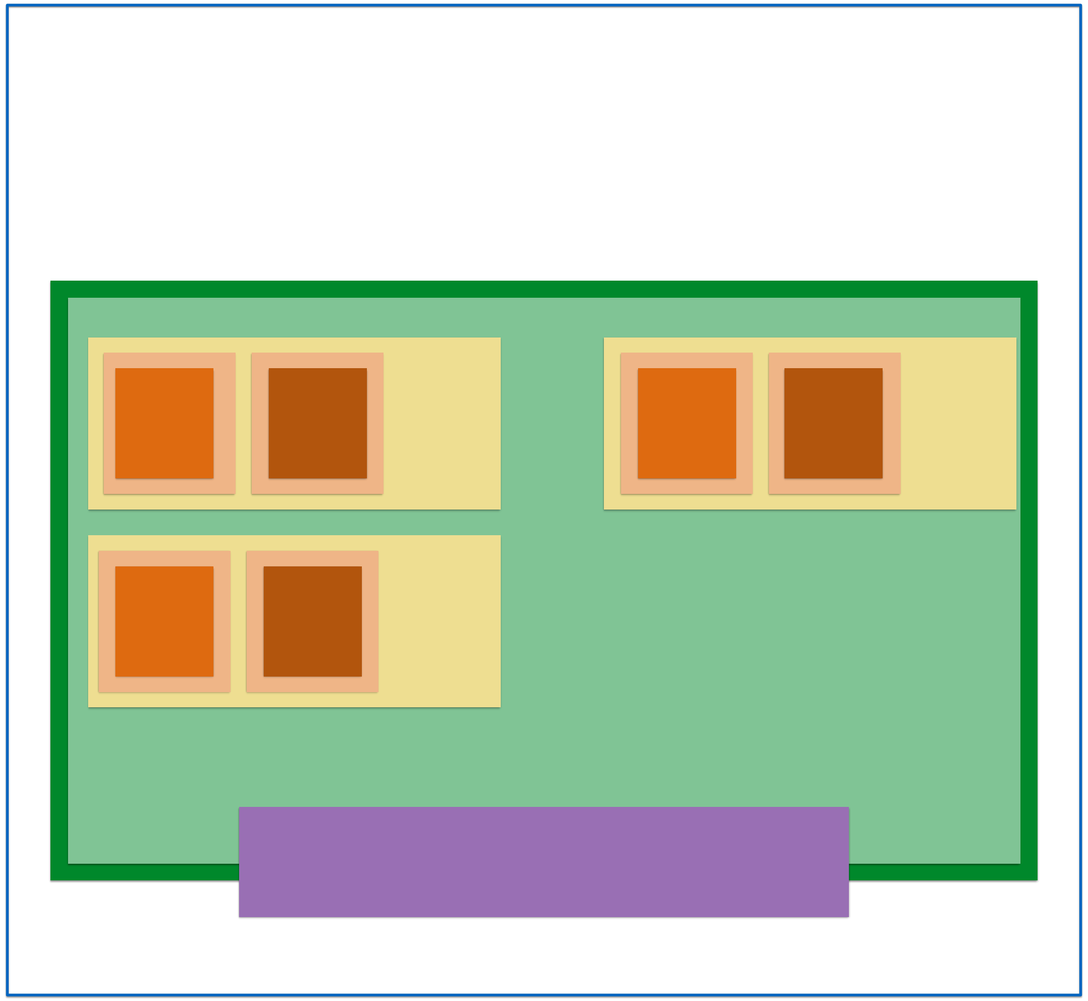
Replica Set

- Defines and manages how many instances of a pod should run
- ReplicaSet is tied to a specific definition of a Pod which is tied to specific image versions of the container
- Image versions in ReplicaSets can't be updated



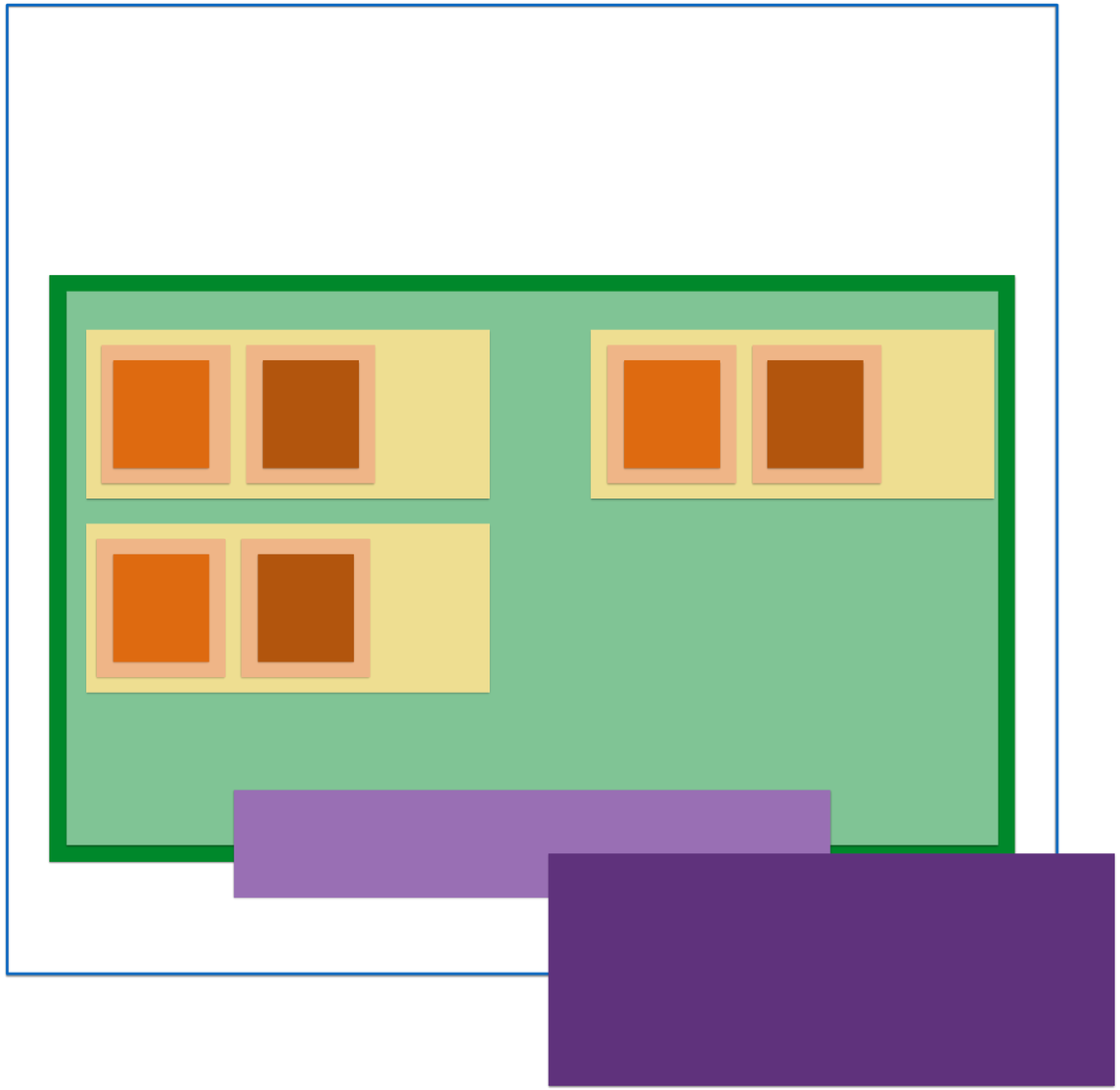
Deployment

- Manages updates and rollbacks of replica sets



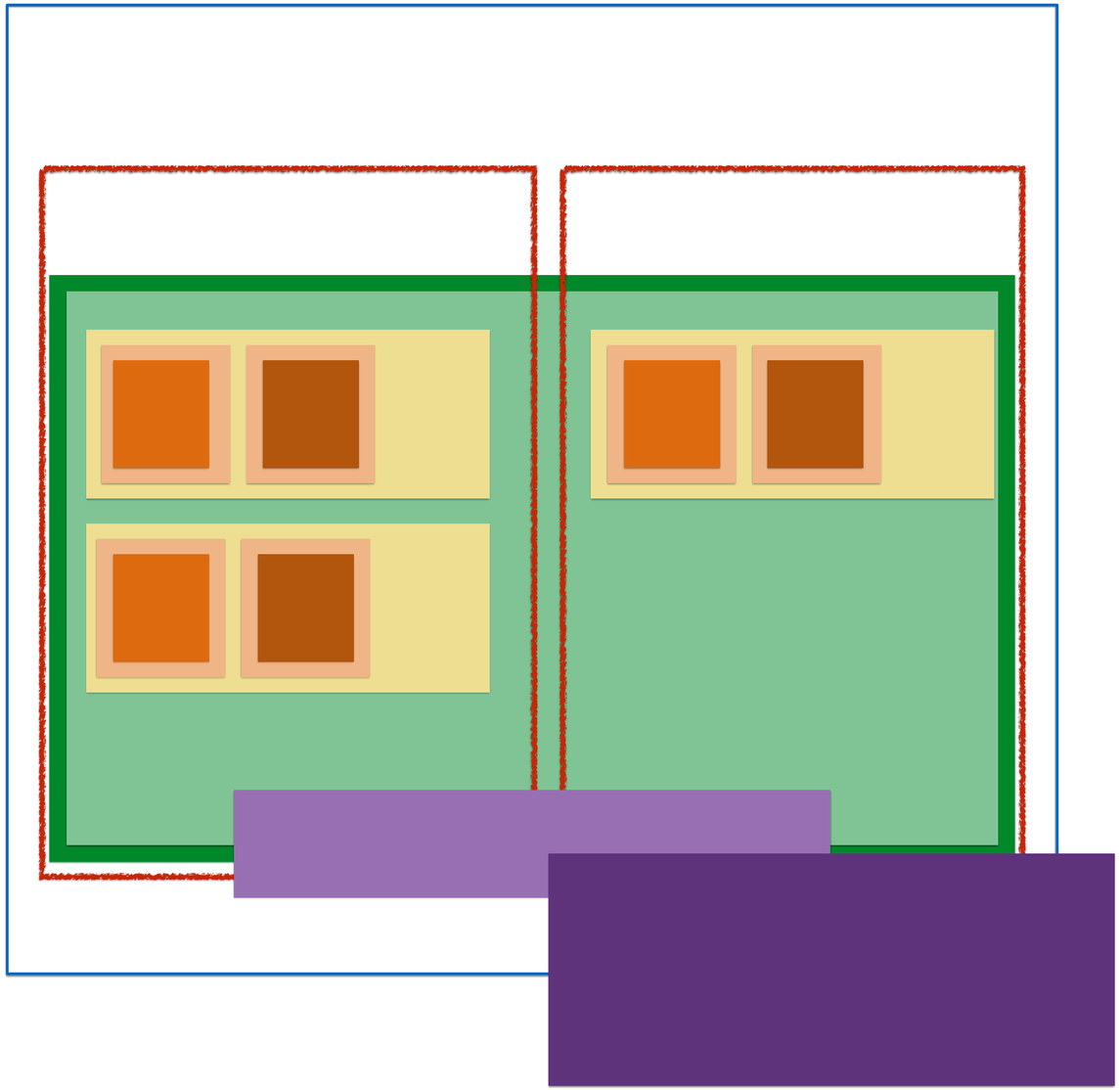
Service

- Internal LoadBalancer
- Makes all pods matching a set of labels accessible through a stable, internal IP address
- You can attach external IP address through an cloud LoadBalancer



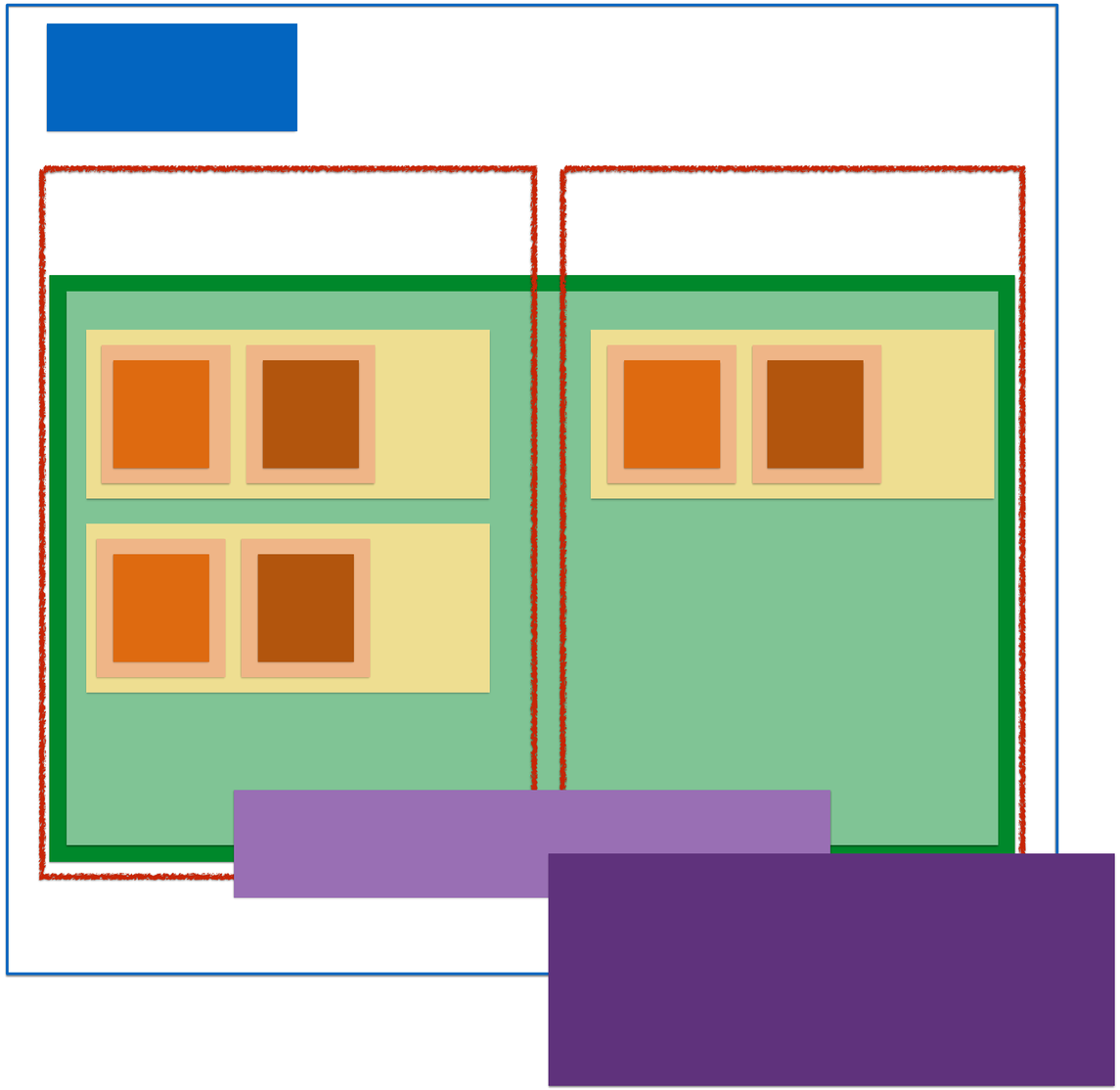
Ingress

- Makes a service accessible to the outside of Kubernetes through an ingress controller (e.g. nginx)
- Traffic is routed by routing rules, usually Host header



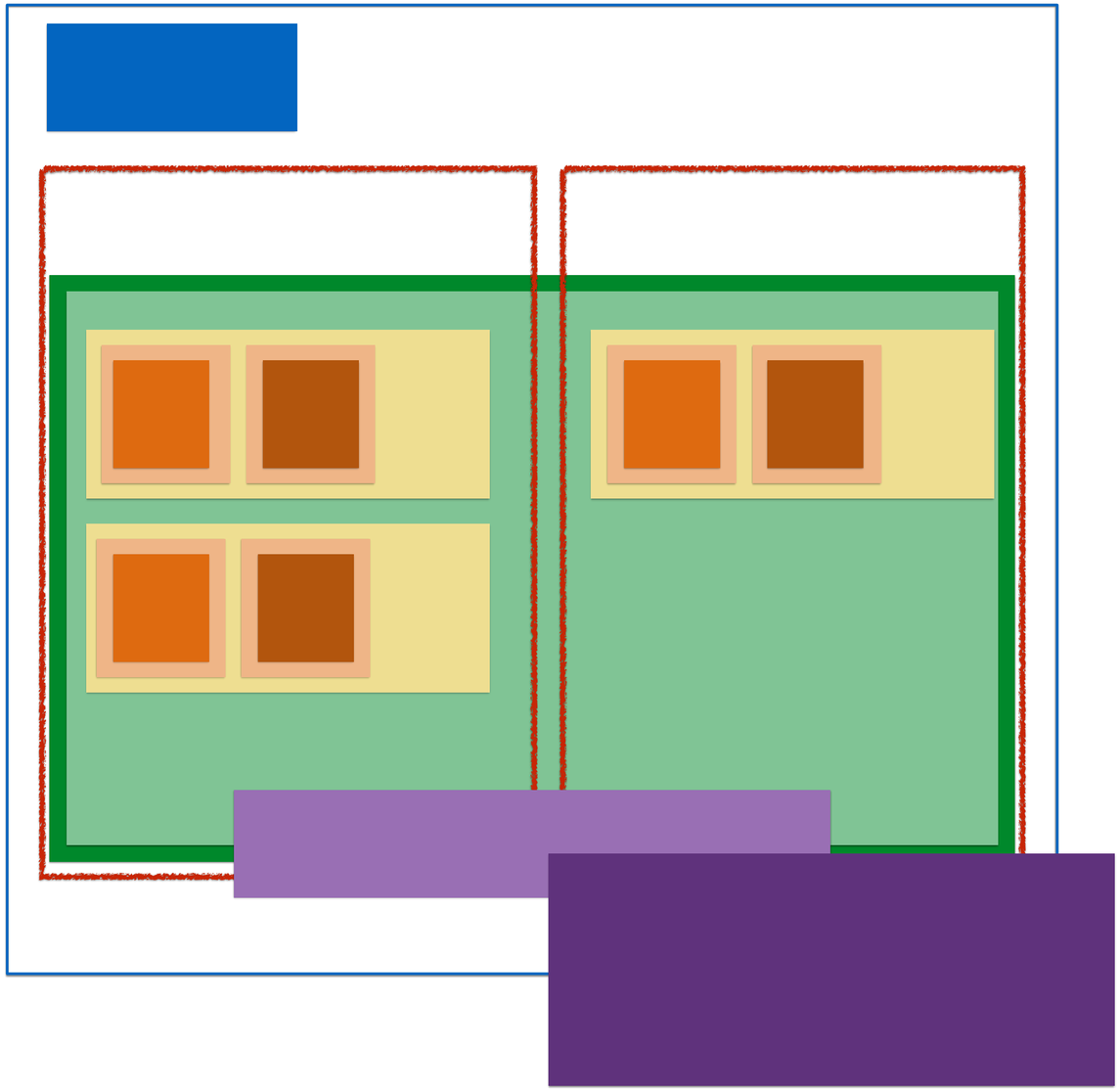
Node

- A physical server
- Containers get distributed automatically



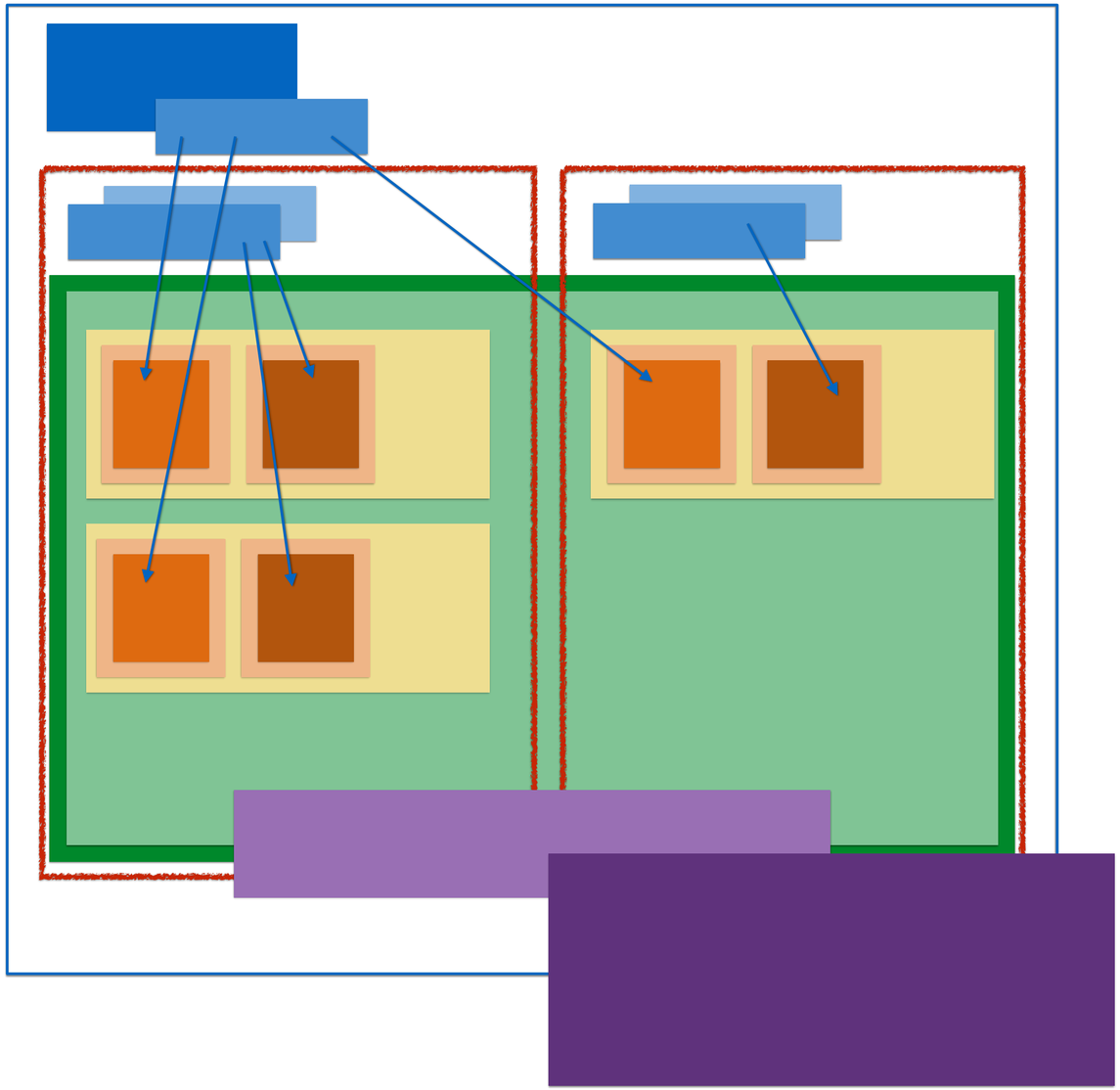
ConfigMap

- Key/Value storage for configuration



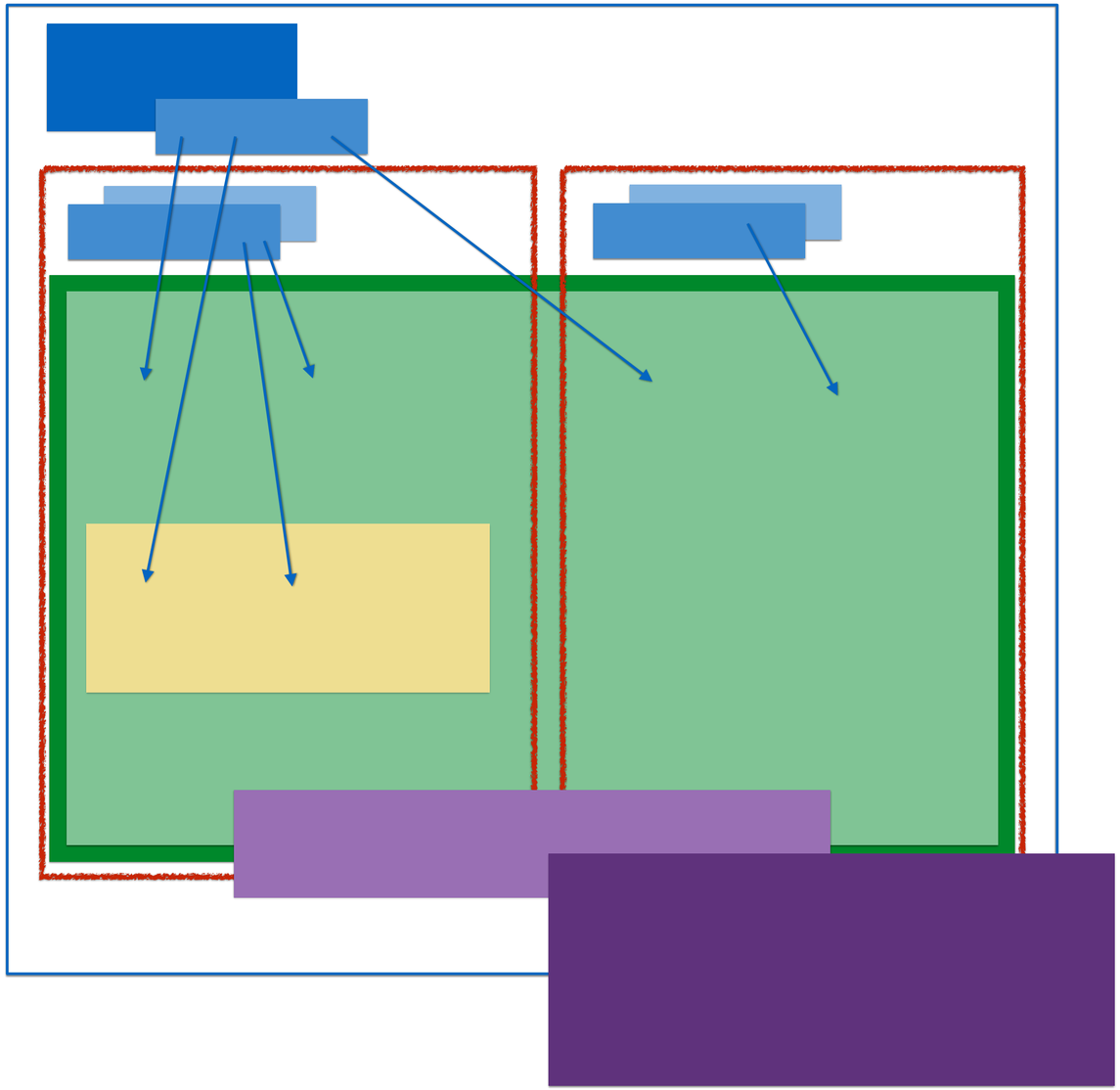
Secret

- Key/Value storage for configuration, usually passwords.



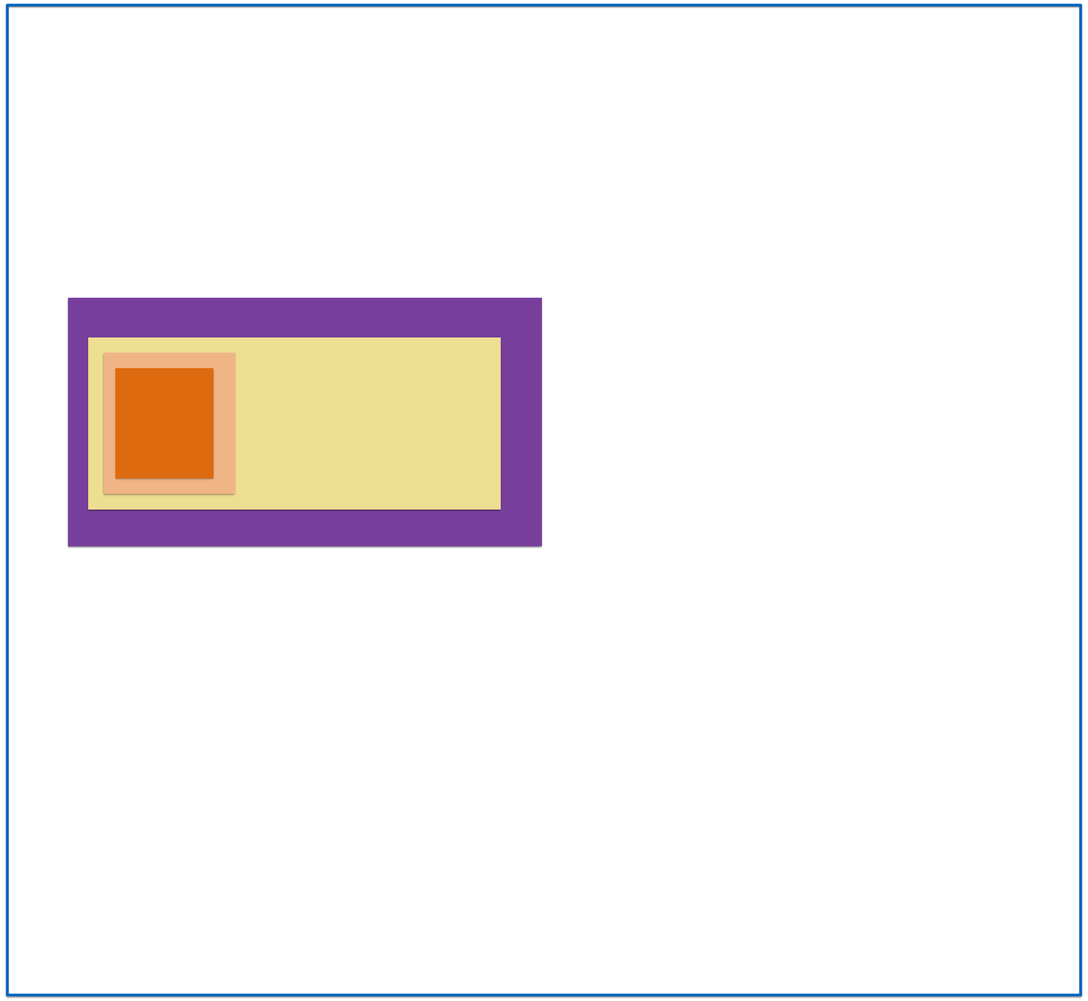
Volumes

- Volumes can be mounted into a container to access a ConfigMap, Secret, persistent volumes with network storage or a folder on the node



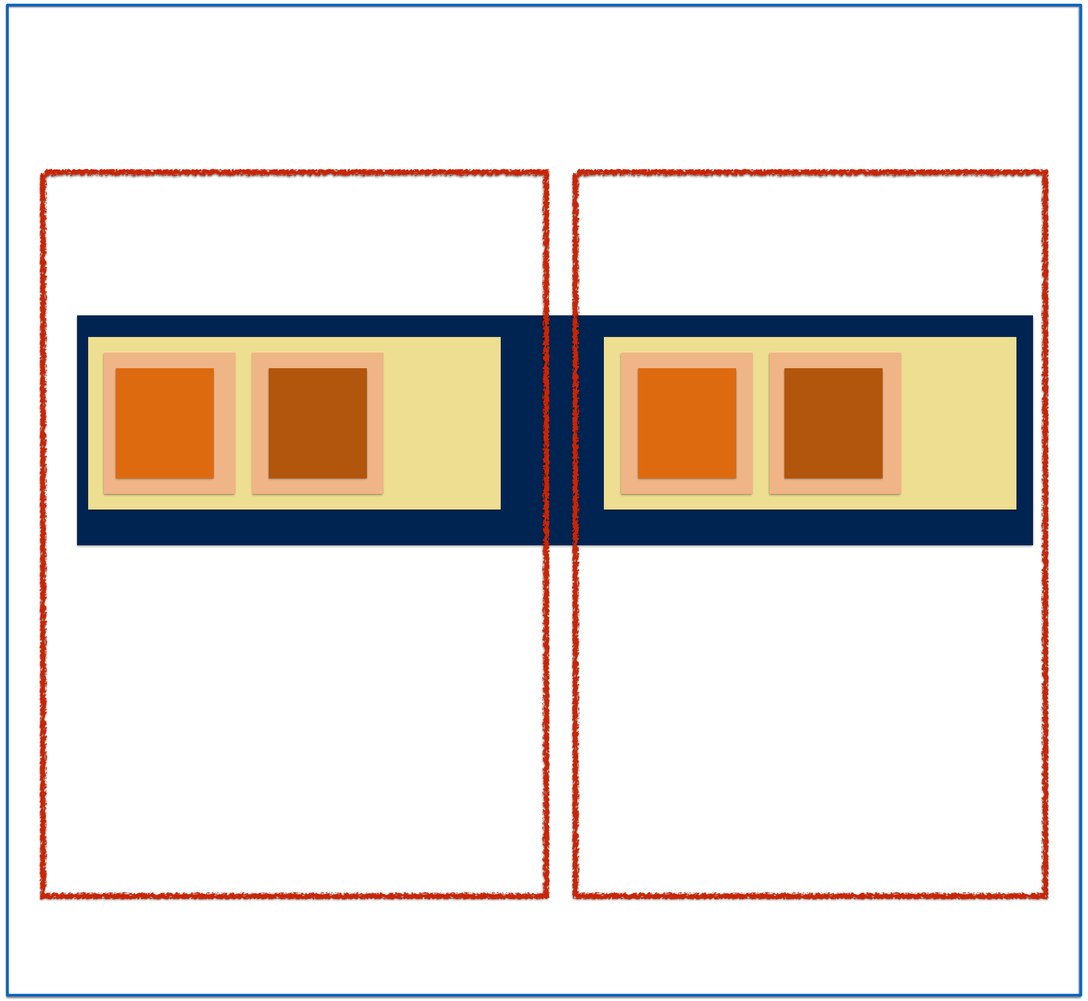
Namespaces

- Dedicated environment to deploy services in



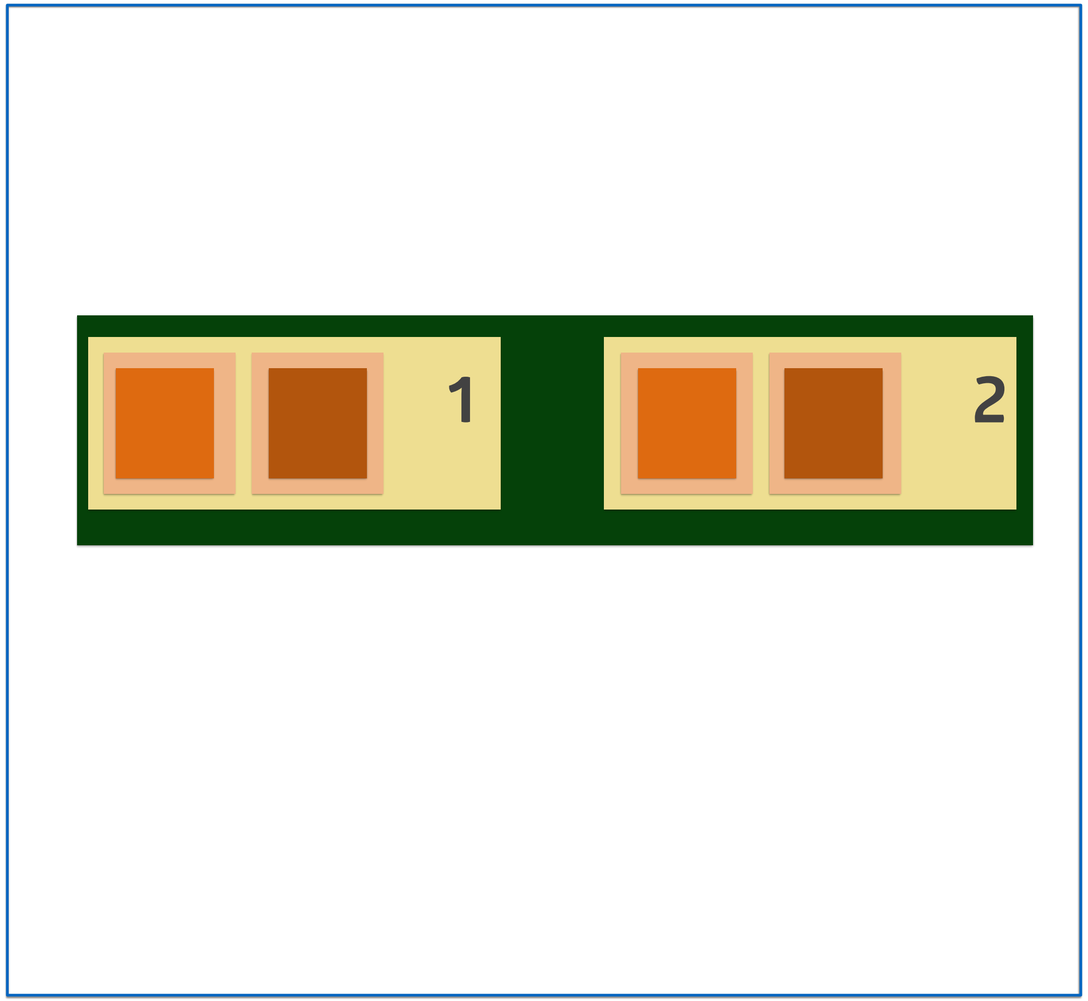
CronJob

- Includes a Pod that is started in a regular interval
- Process in the container should finish at some point



DaemonSet

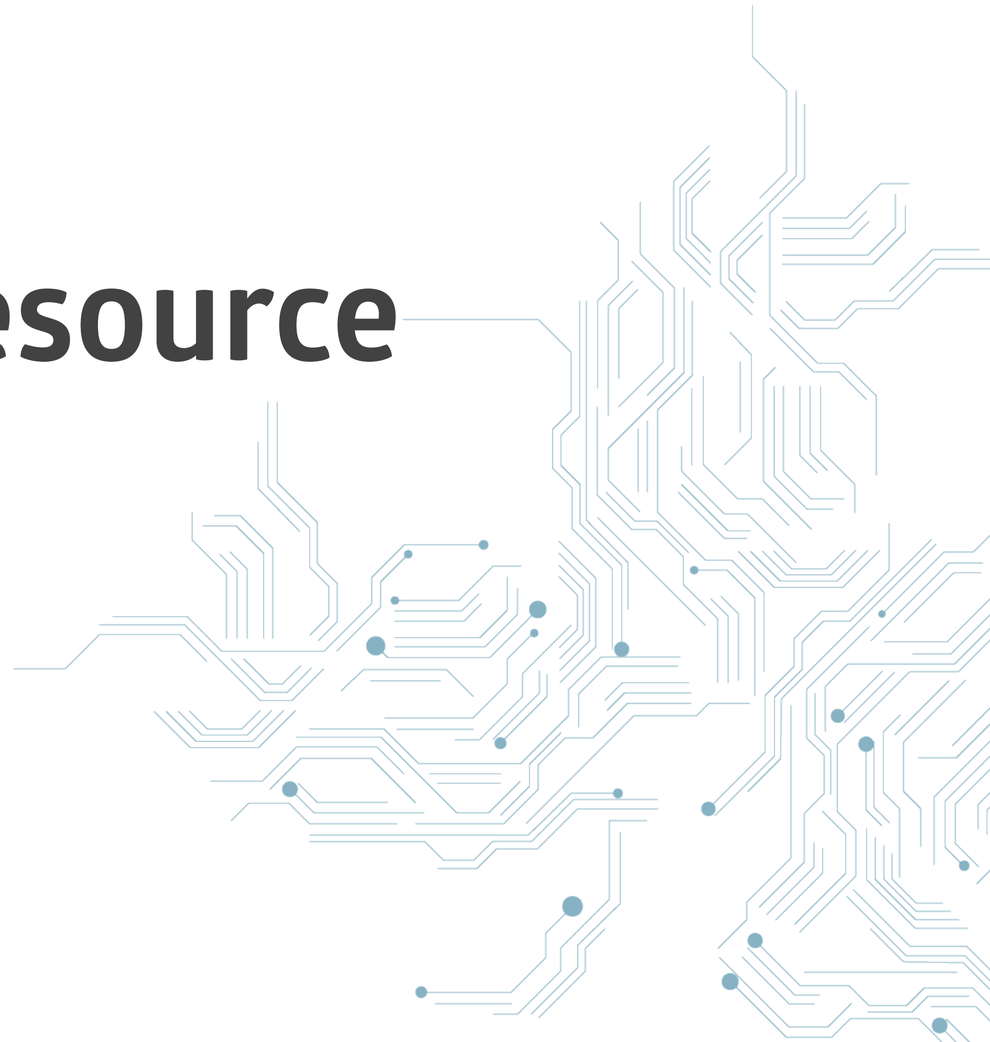
- Defines Pod that should run once on every Node
- Useful for monitoring or logging daemons



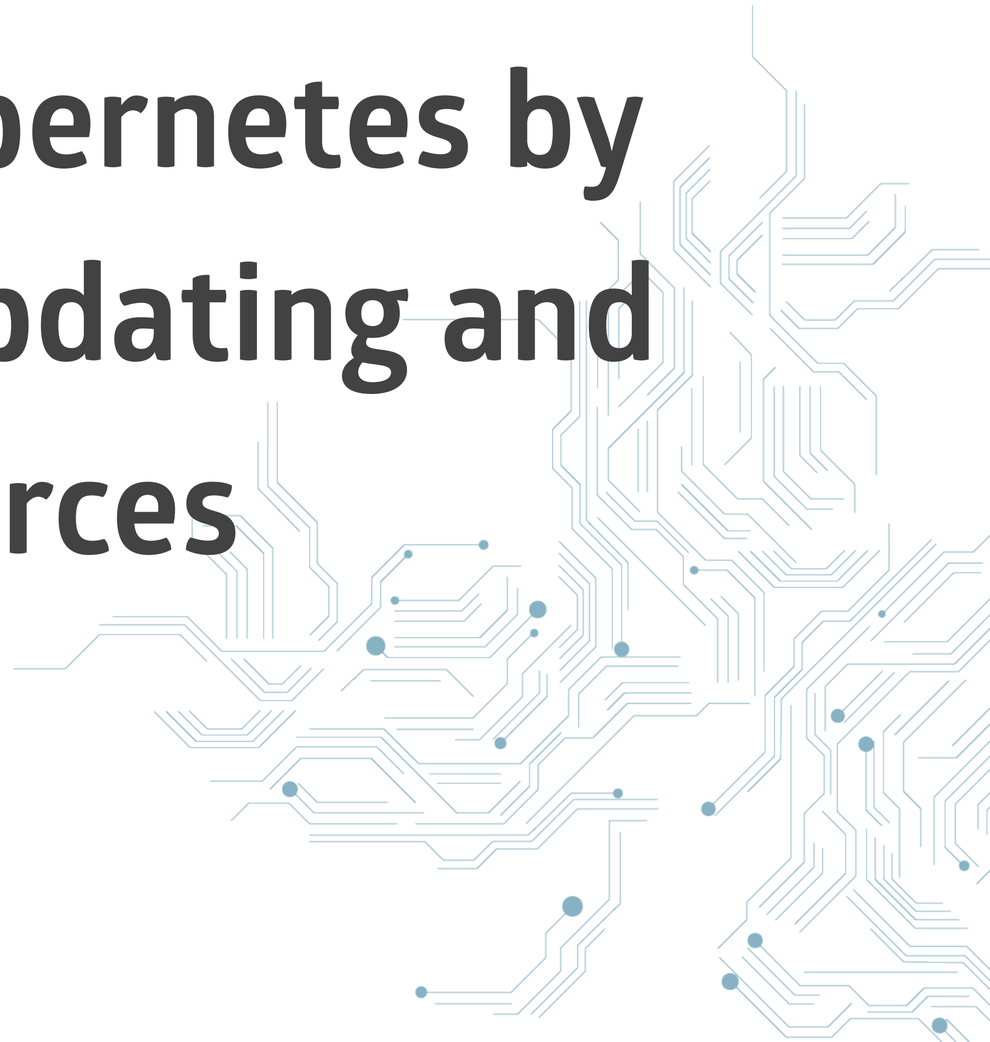
StatefulSet

- Ensures that Pods are started and run in a specific order
- Each Pod of a StatefulSet can have its own persistent volume
- Pod names stay the same

Everything is a resource

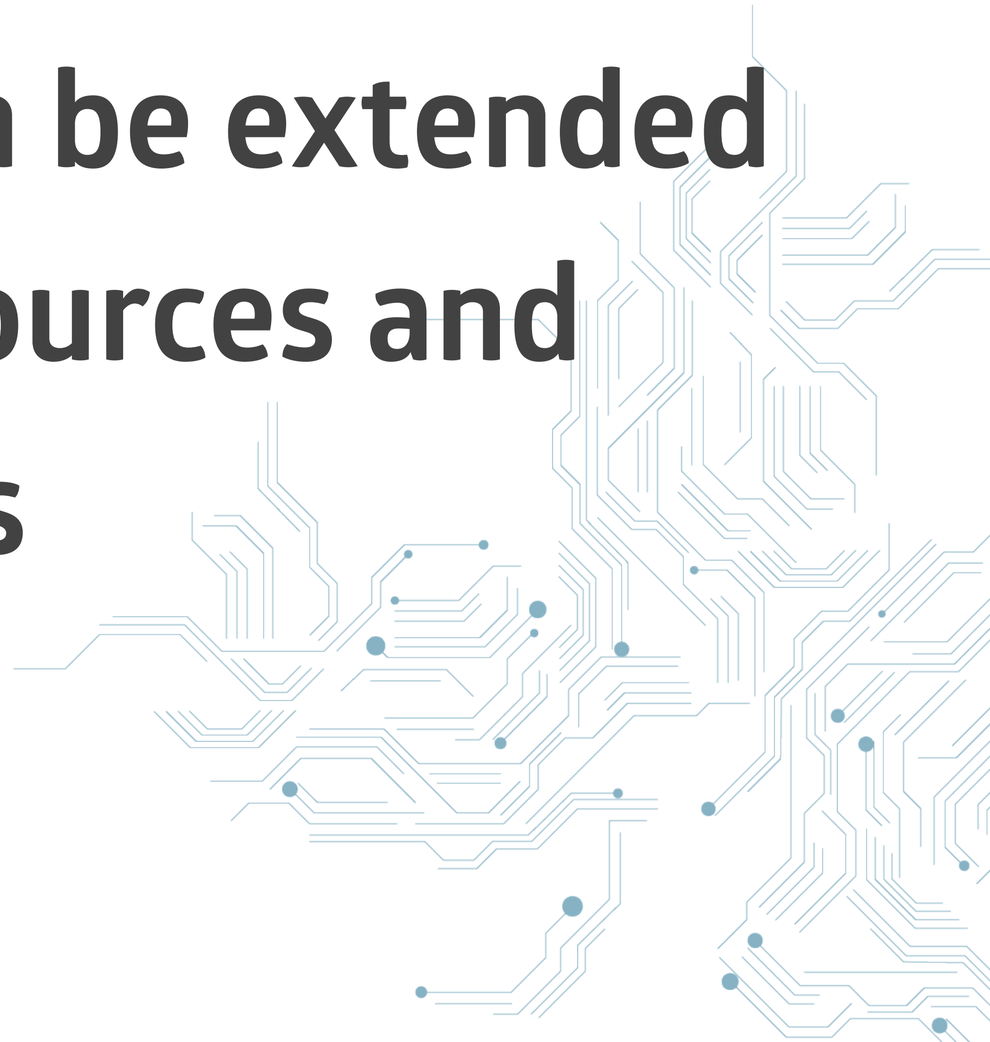


You interact with Kubernetes by creating, receiving, updating and deleting resources



Kubernetes has controllers to listen on these interactions and get the cluster in the desired state

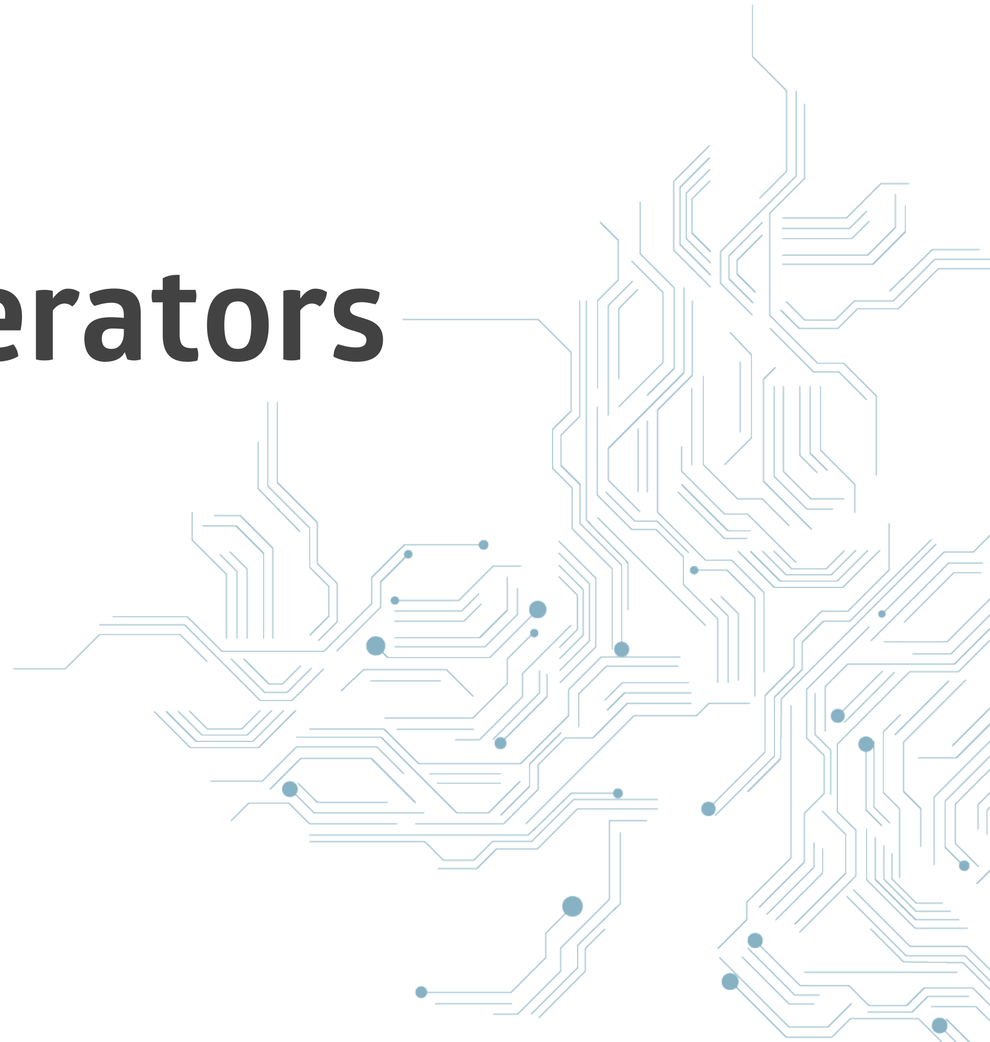
The Kubernetes API can be extended with additional Resources and Controllers



CustomResourceDefinitions

Certificate, Backup, Restore, MySQLCluster, Function, ...

Controllers / Operators



```
kind: Deployment
apiVersion: extensions/v1beta1
metadata:
  name: symfony-demo
spec:
 template:
    spec:
      containers:
        - name: symfony-demo
          image: symfony-demo:1.1.0
          ports:
            - containerPort: 80
```

```
$ kubectl apply -f deployment.yaml
```

```
$ kubectl get deployments
NAME           DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
symfony-demo   1         1         1            1           21h
```

```
$ kubectl get deployment symfony-demo -o yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    ...
spec:
 ...
 template:
     ...
     spec:
       containers:
         - name: symfony-demo
```

```
$ kubectl delete deployment symfony-demo
```

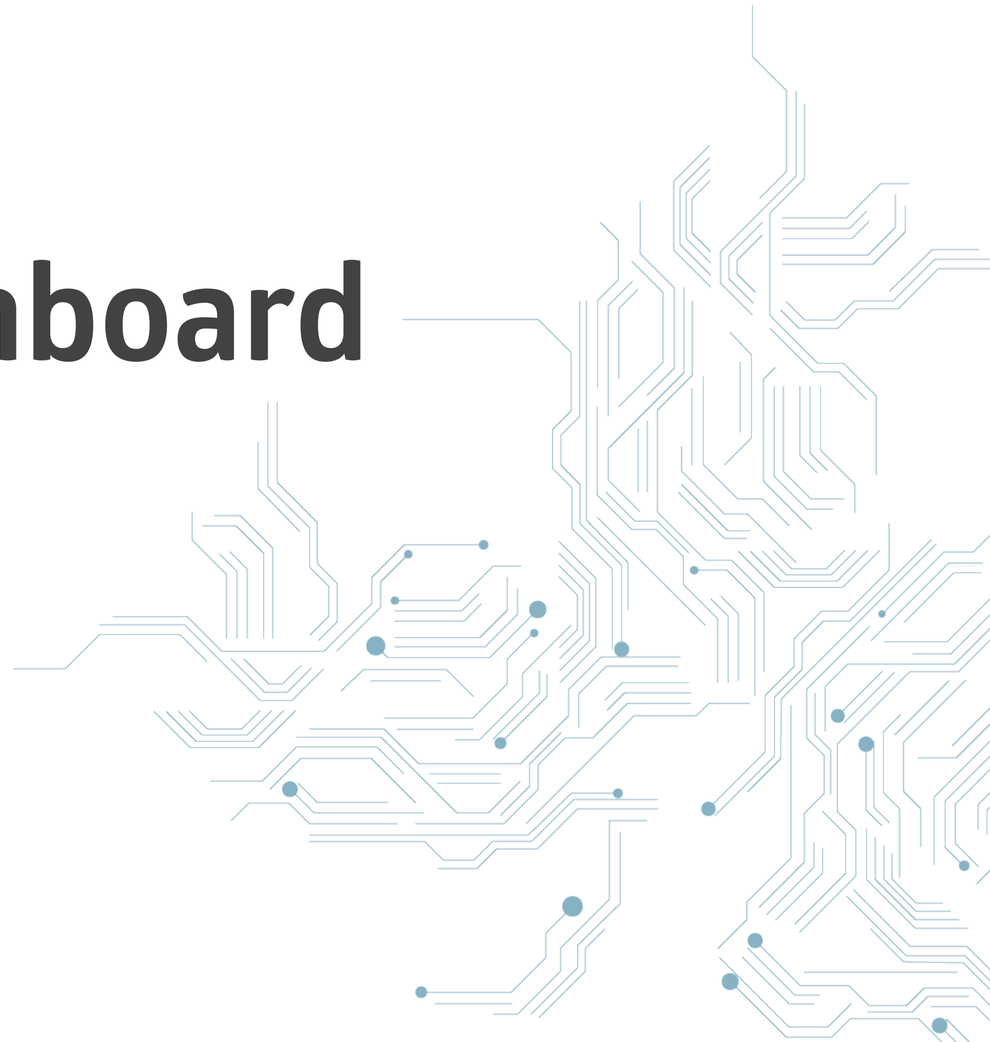
Tooling

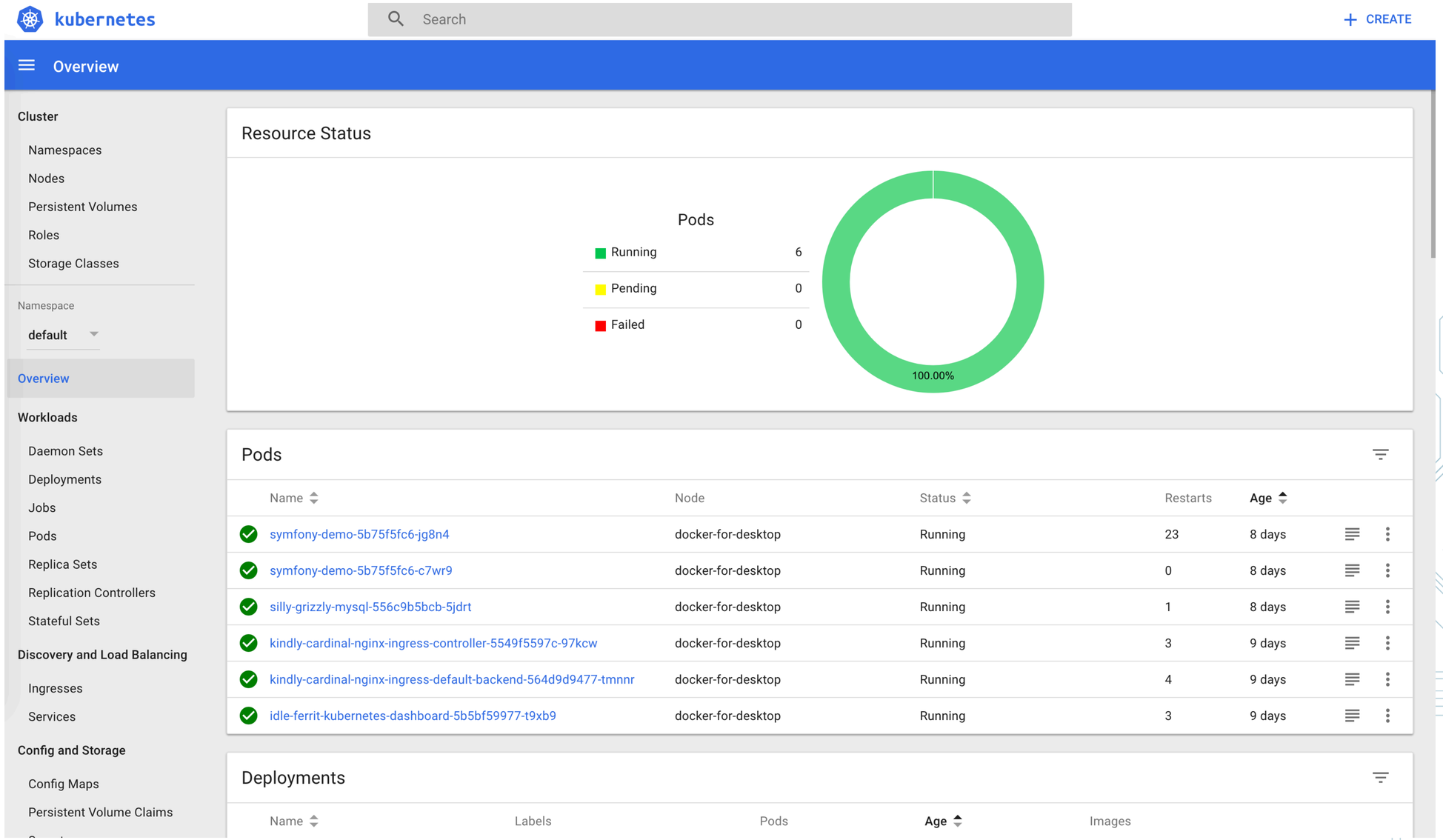
kubectl

REST API

```
$ kubectl proxy --port=8080
$ curl http://localhost:8080/api/v1/namespaces/default/pods
{
  "kind": "PodList",
  "apiVersion": "v1",
  "metadata": {
    "selfLink": "/api/v1/namespaces/default/pods",
    "resourceVersion": "336834"
  },
  "items": [
    {
      "metadata": {
        "name": "kubernetes-dashboard-5b5bf59977-t9xb9",
```

kubernetes-dashboard




kubernetes
Search
CREATE
Overview
Cluster
Namespaces
Nodes
Persistent Volumes
Roles
Storage Classes
Namespace
default
Overview
Workloads
Daemon Sets
Deployments
Jobs
Pods
Replica Sets
Replication Controllers
Stateful Sets
Discovery and Load Balancing
Ingresses
Services
Config and Storage
Config Maps
Persistent Volume Claims
Resource Status
Pods
Running 6
Pending 0
Failed 0
100.00%
Pods
Name
Node
Status
Restarts
Age
symfony-demo-5b75f5fc6-jg8n4 docker-for-desktop Running 23 8 days
symfony-demo-5b75f5fc6-c7wr9 docker-for-desktop Running 0 8 days
silly-grizzly-mysql-556c9b5bcb-5jdrt docker-for-desktop Running 1 8 days
kindly-cardinal-nginx-ingress-controller-5549f5597c-97kcw docker-for-desktop Running 3 9 days
kindly-cardinal-nginx-ingress-default-backend-564d9d9477-tmnnr docker-for-desktop Running 4 9 days
idle-ferrit-kubernetes-dashboard-5b5bf59977-t9xb9 docker-for-desktop Running 3 9 days
Deployments
Name
Labels
Pods
Age
Images

Helm

The package manager for Kubernetes

```
$ helm install stable/wordpress
```

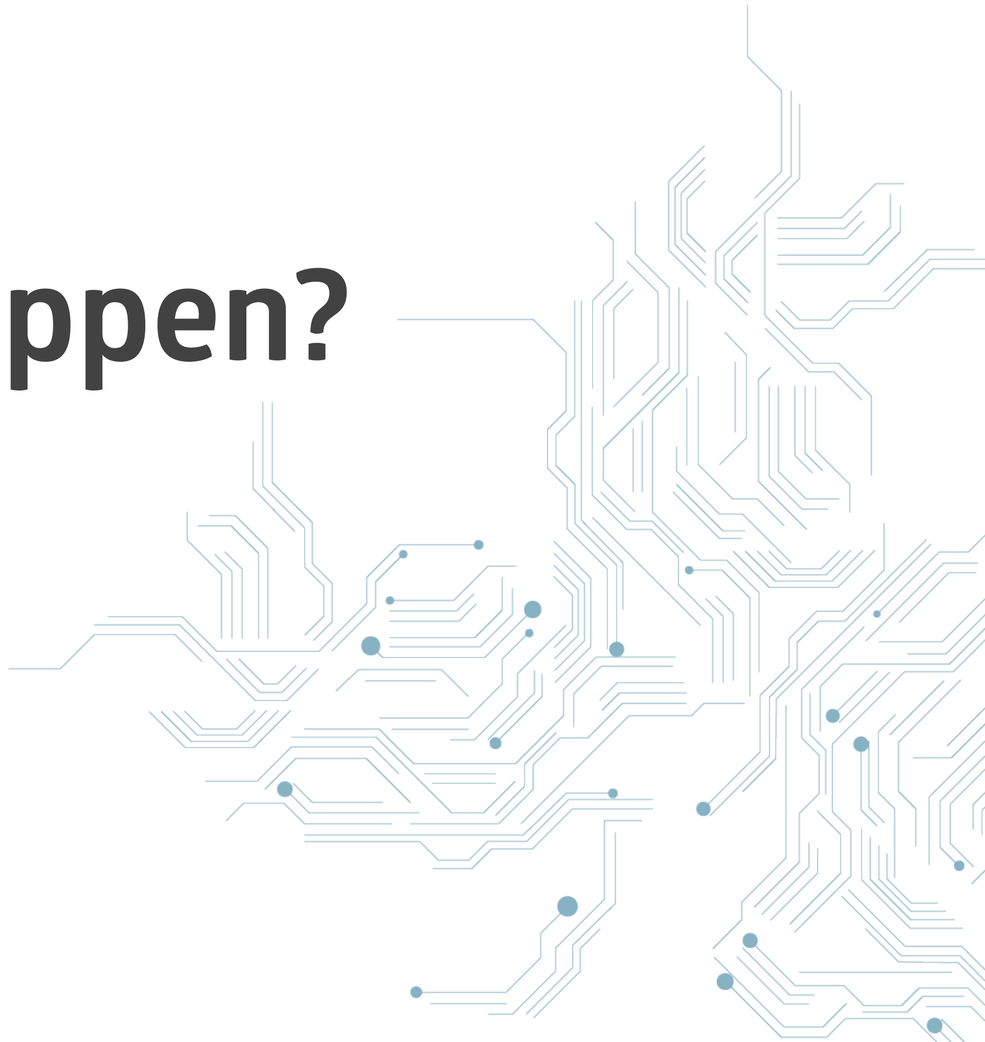
hands-on

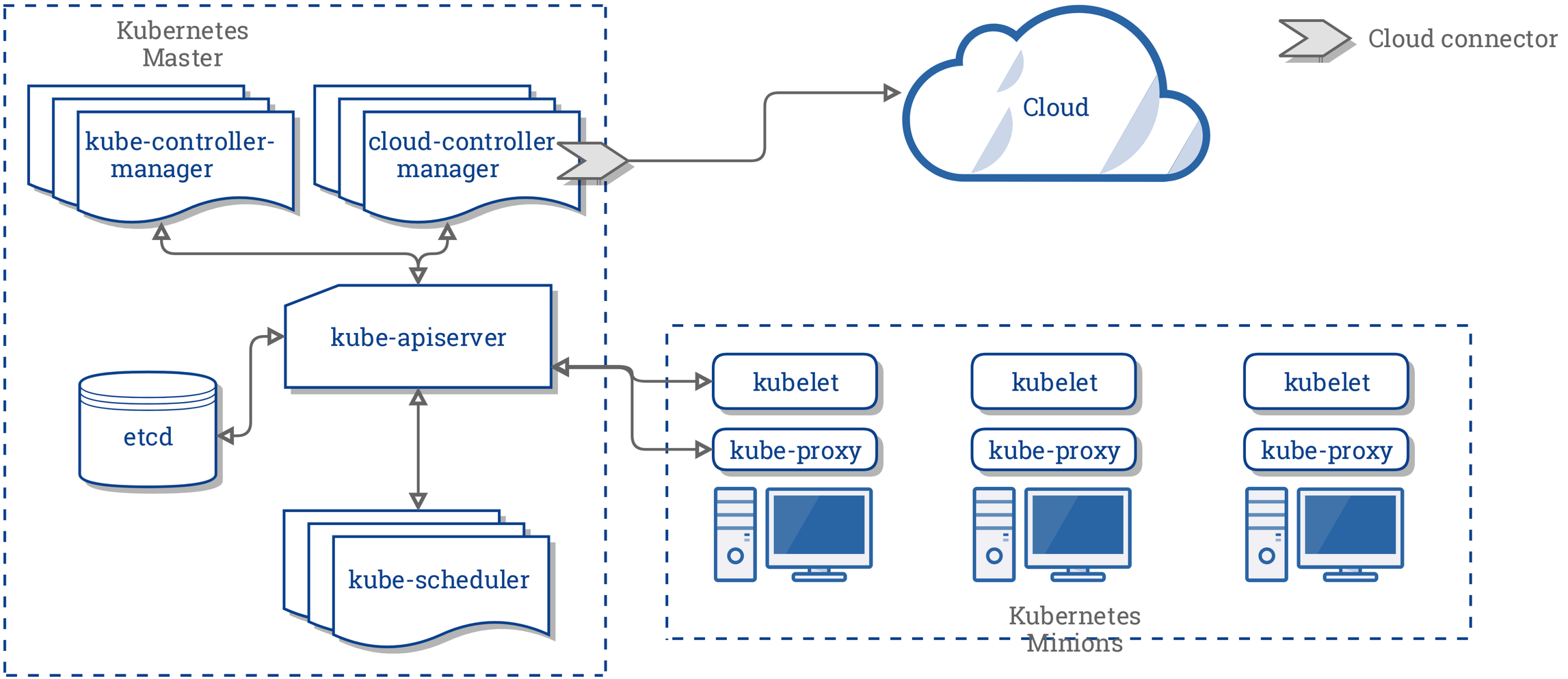
Install kubectl:

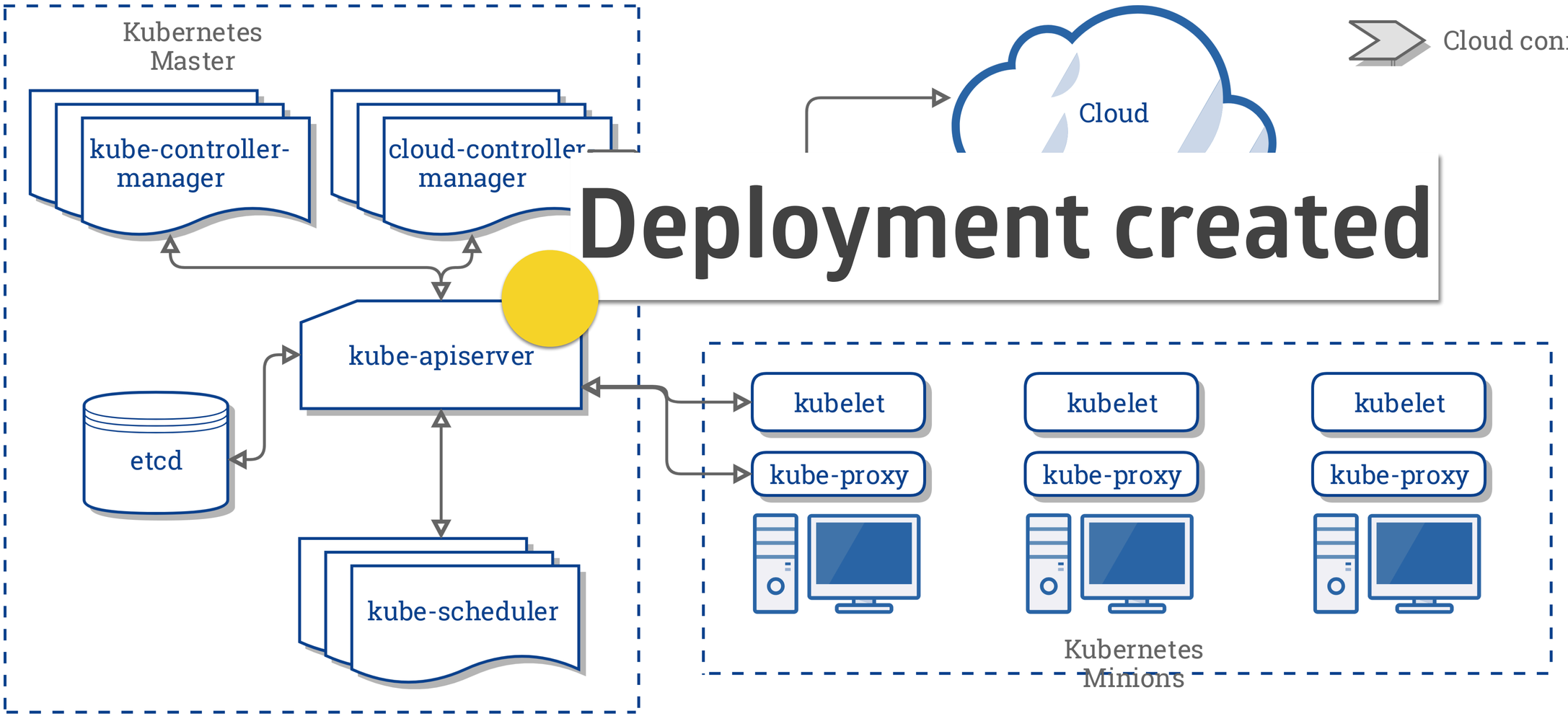
https://kubernetes.io/docs/tasks/tools/install-kubectl/

1 - Deploying a simple Web Application

What did just happen?

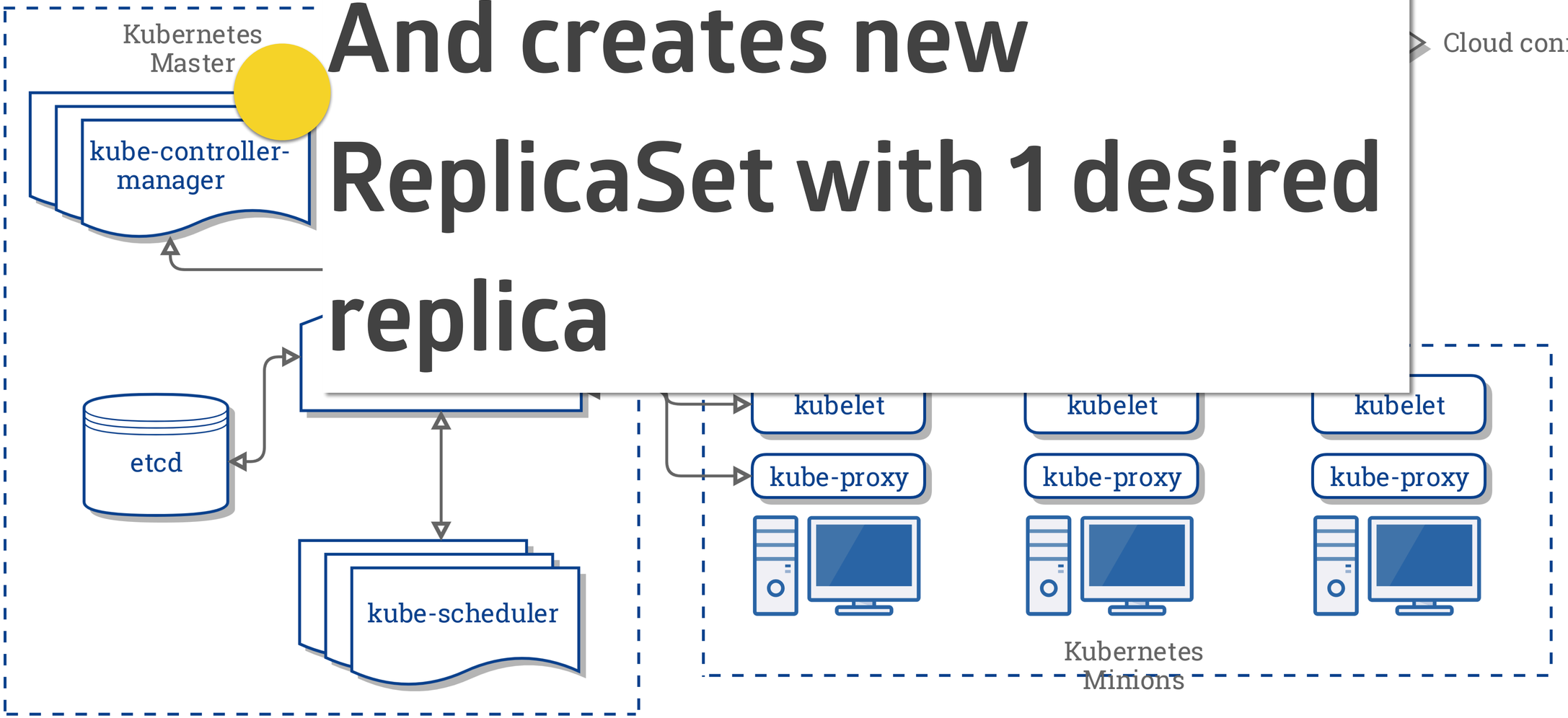



Kubernetes
Master
kube-controller-
manager
cloud-controller
manager
Cloud
Cloud connector
kube-apiserver
etcd
kube-scheduler
kubelet
kube-proxy
kubelet
kube-proxy
kubelet
kube-proxy
Kubernetes
Minions

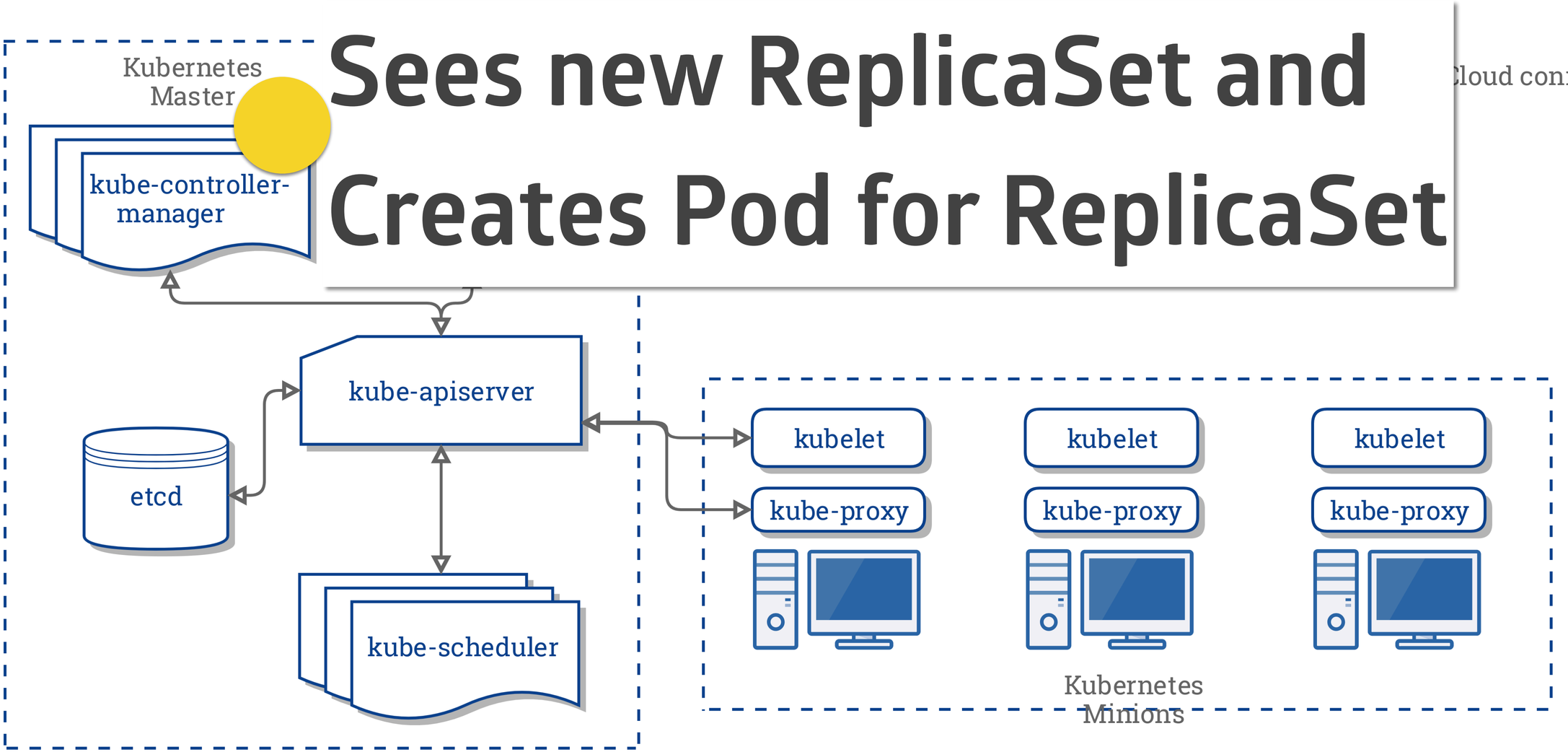

Kubernetes
Master
kube-controller-
manager
cloud-controller-
manager
kube-apiserver
etcd
kube-scheduler
Cloud
Cloud connector
Deployment created
kubelet
kubelet
kubelet
kube-proxy
kube-proxy
kube-proxy
Kubernetes
Minions

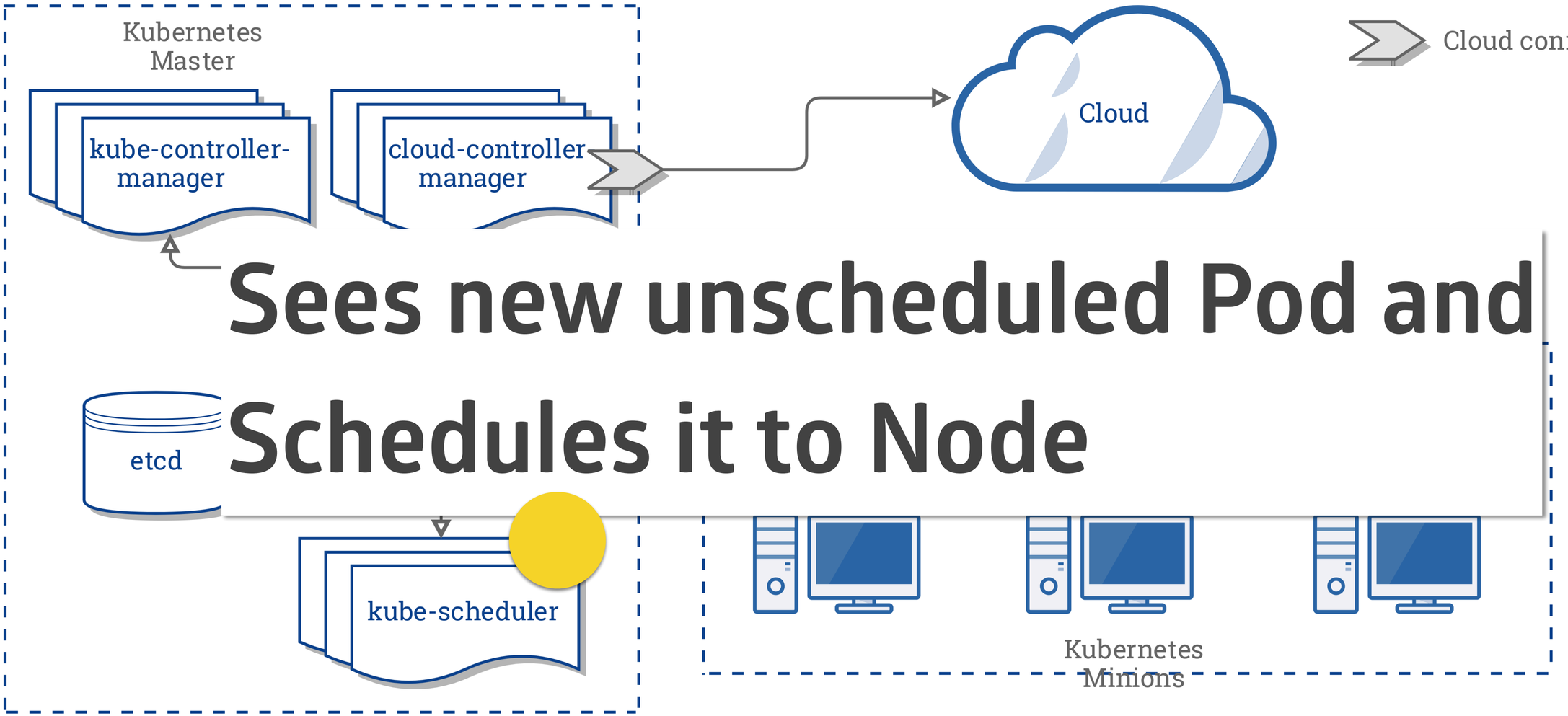
Sees new Deployment

And creates new ReplicaSet with 1 desired replica

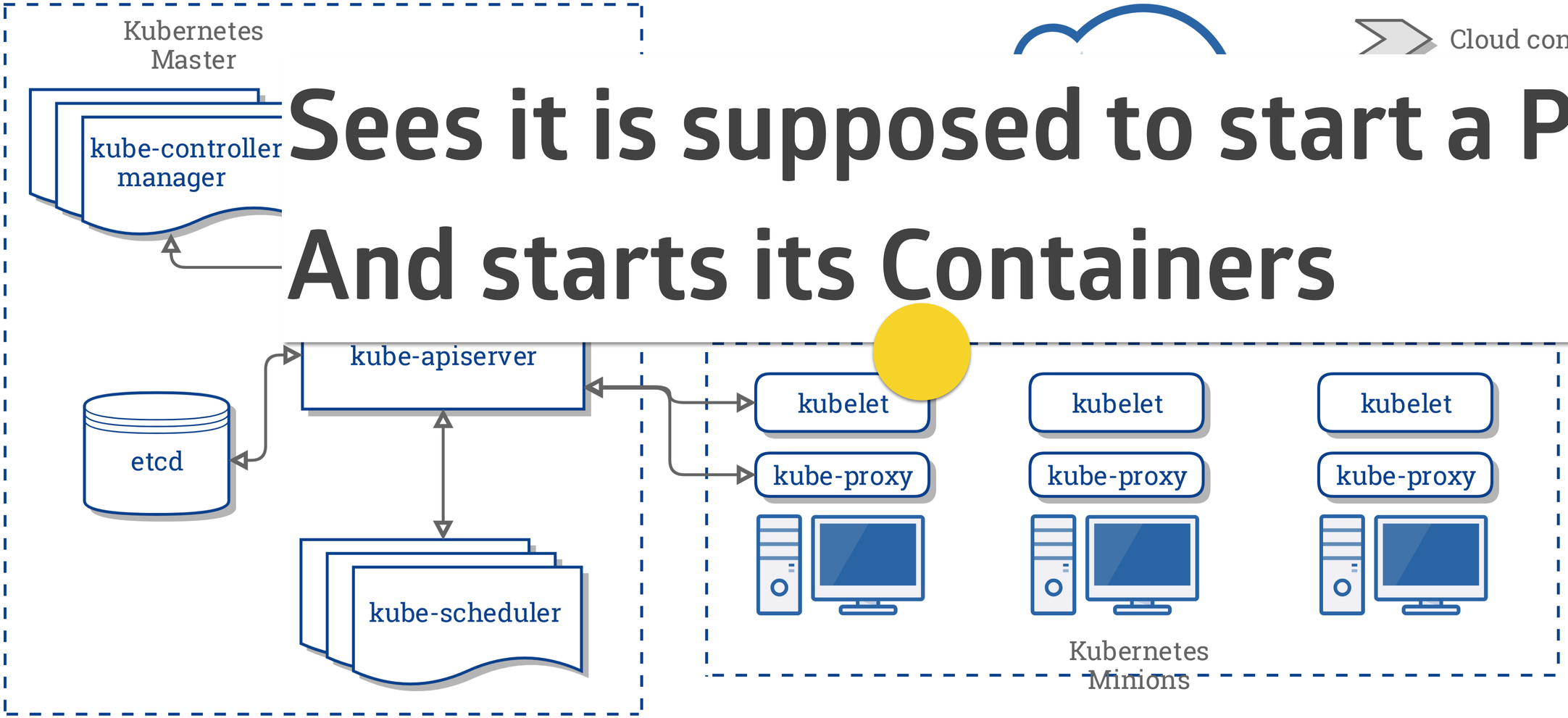


Sees new ReplicaSet and Creates Pod for ReplicaSet



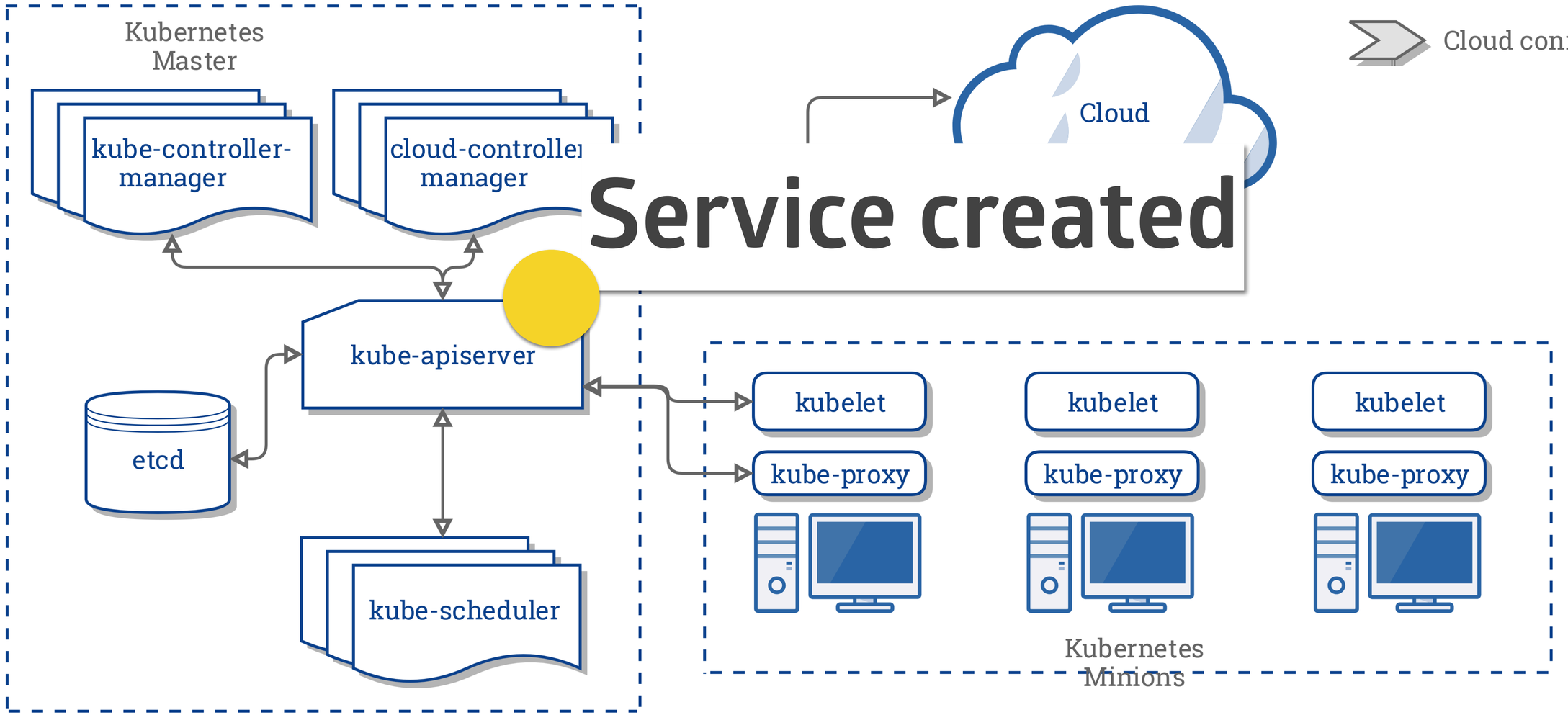


Sees new unscheduled Pod and Schedules it to Node

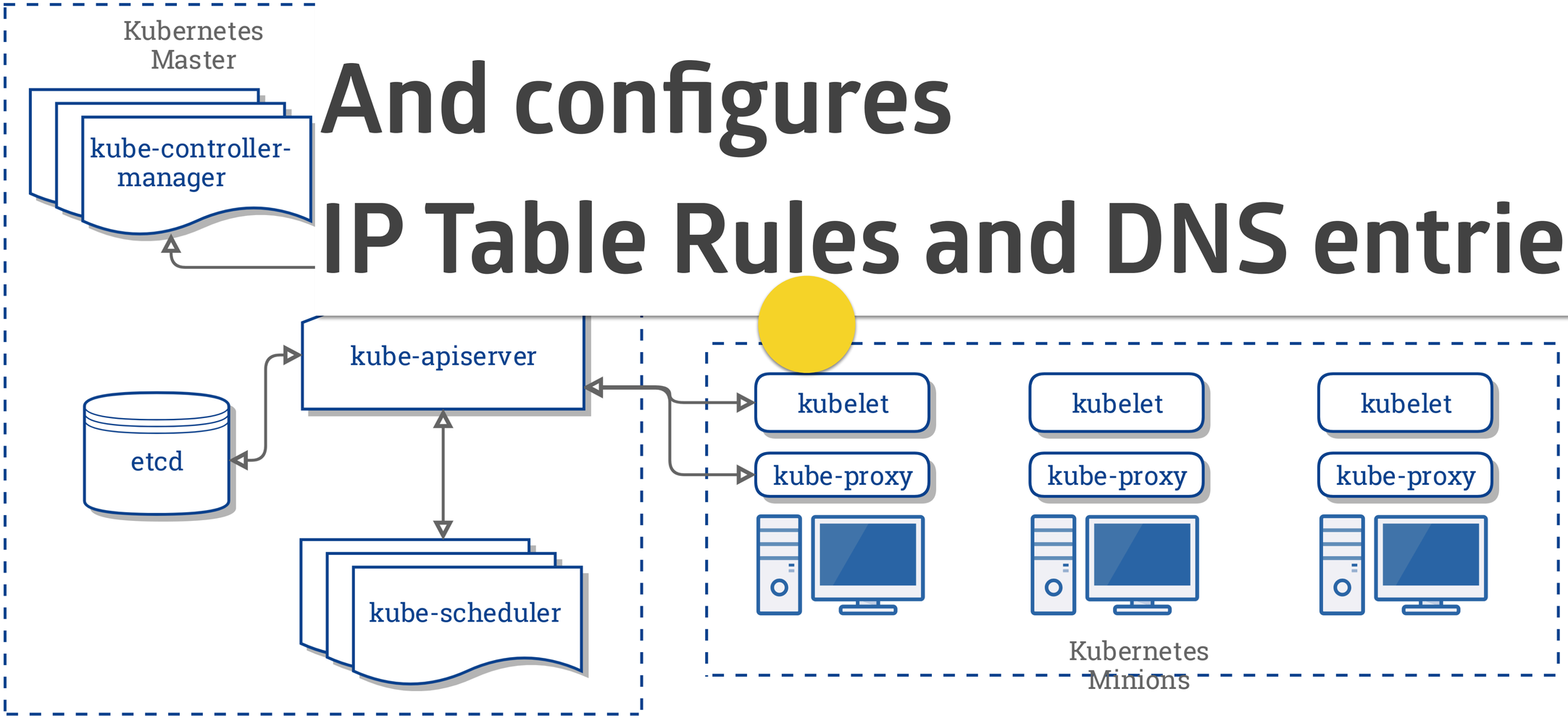


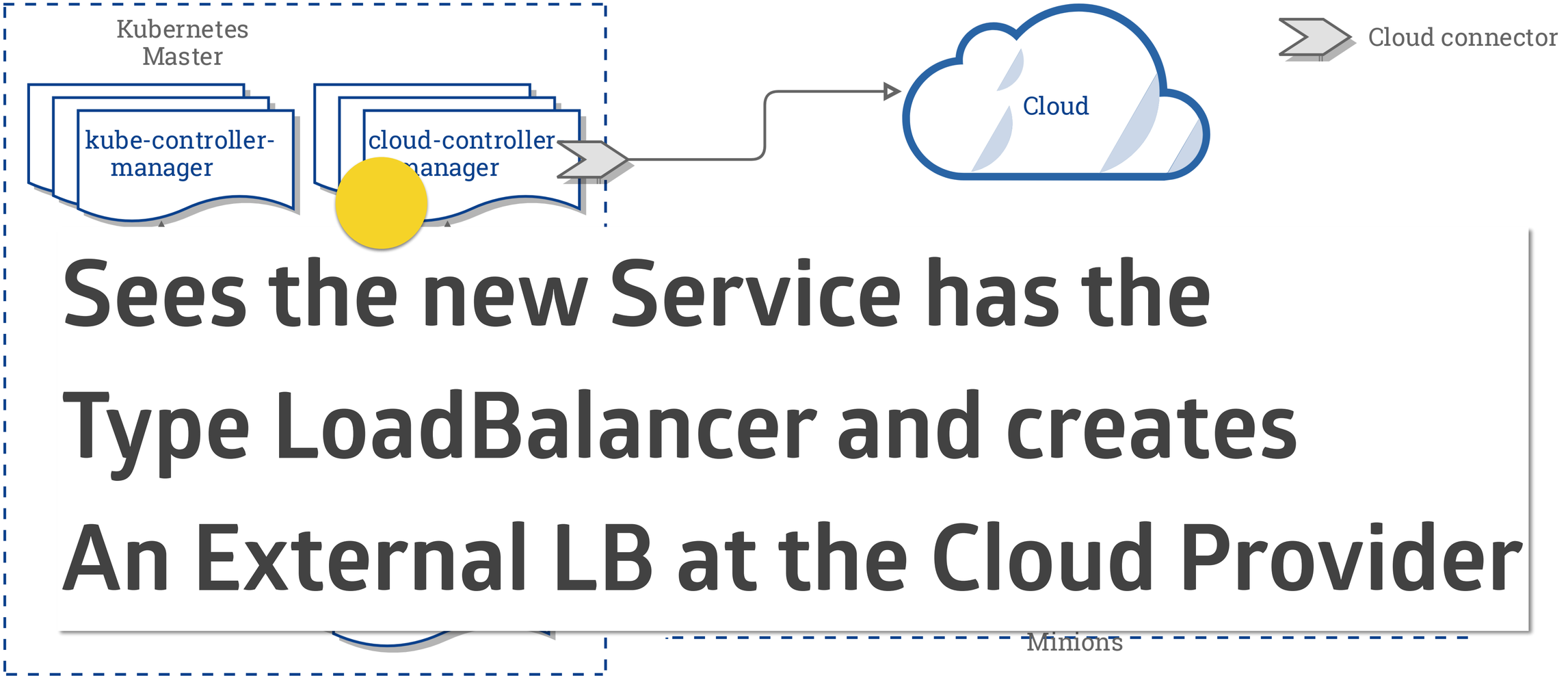
Sees it is supposed to start a Pod

And starts its Containers


Kubernetes
Master
kube-controller-
manager
cloud-controller
manager
Cloud
Service created
Cloud connector
kube-apiserver
etcd
kube-scheduler
kubelet
kube-proxy
kubelet
kube-proxy
kubelet
kube-proxy
Kubernetes
Minions

Sees the new Service
And configures
IP Table Rules and DNS entries

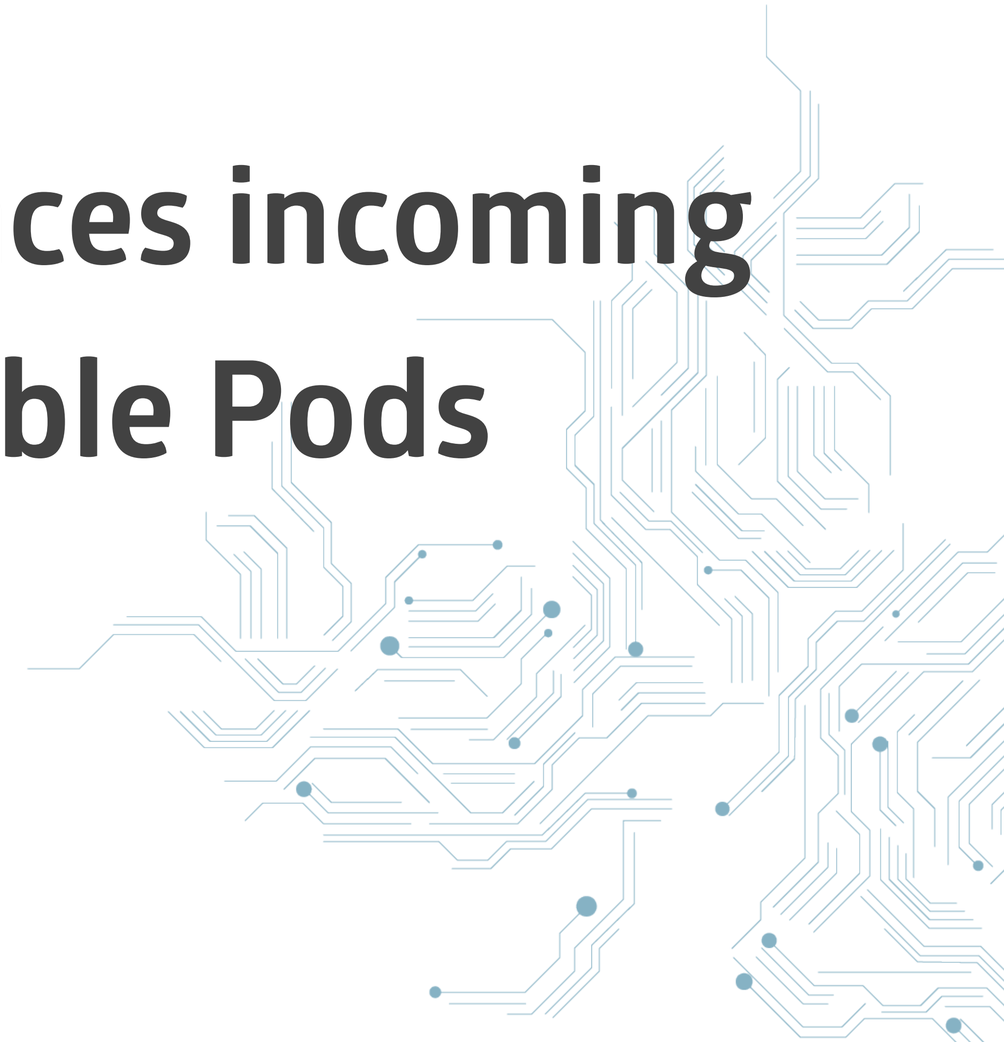




Sees the new Service has the Type LoadBalancer and creates An External LB at the Cloud Provider

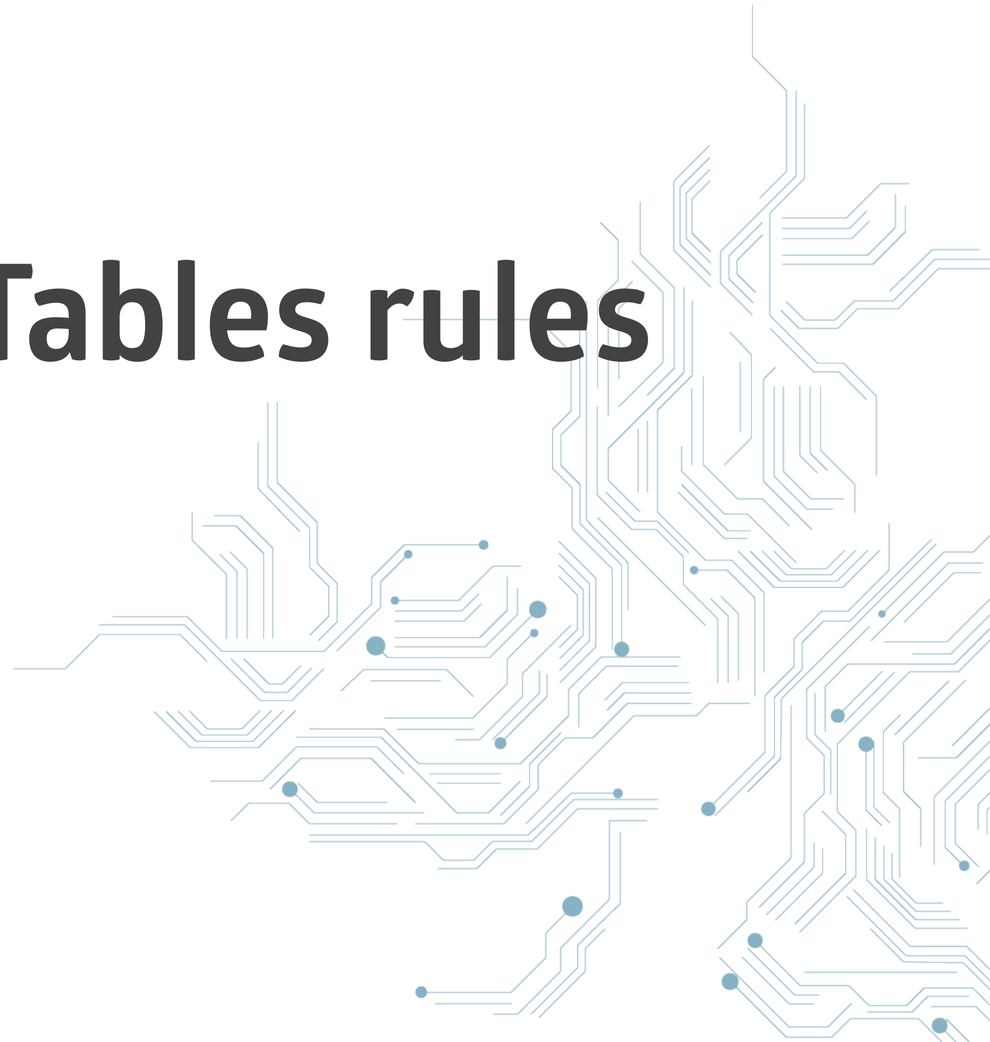
How is traffic routed to the Pod

The Service load-balances incoming traffic to all available Pods

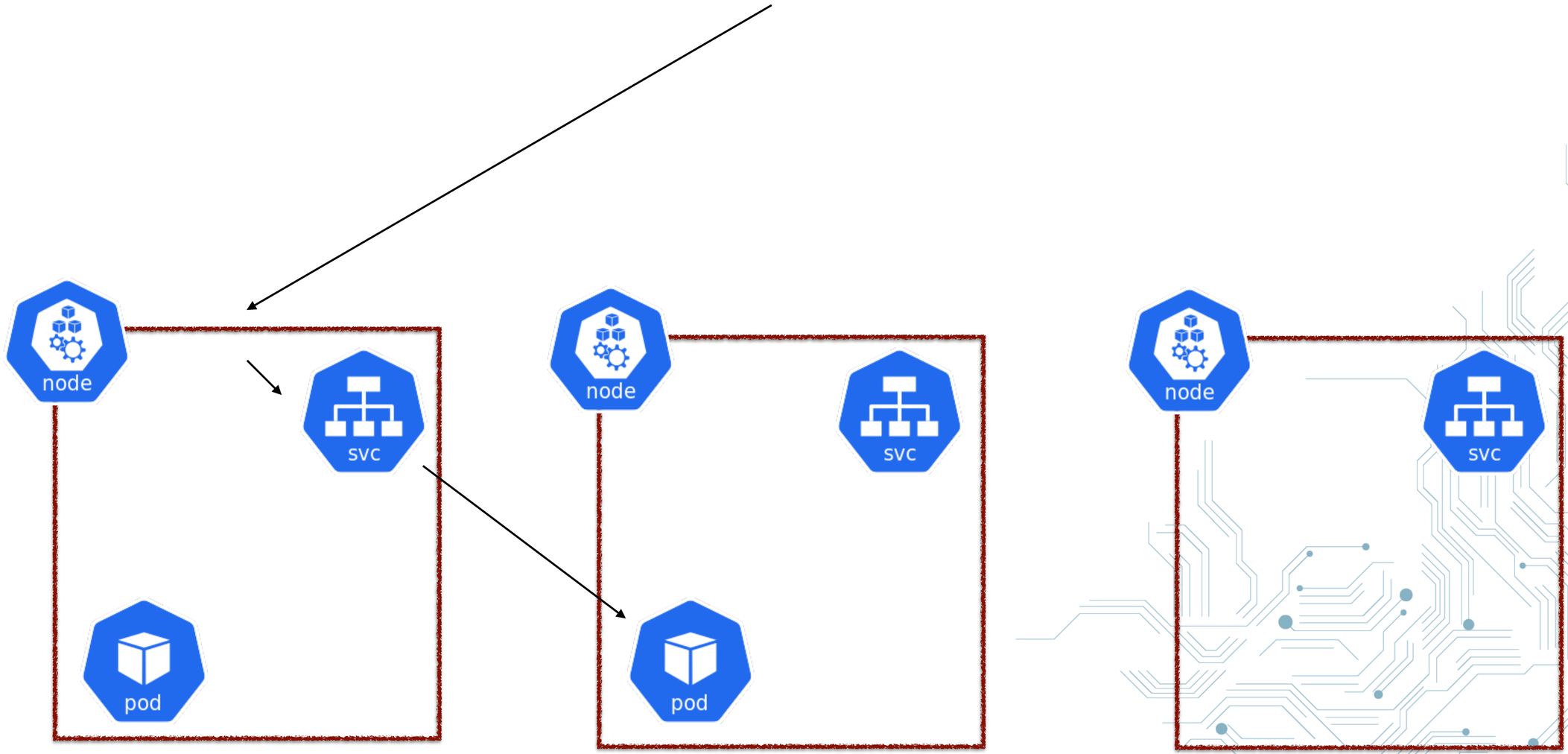


Every Service has a virtual IP

Round Robin with IP Tables rules

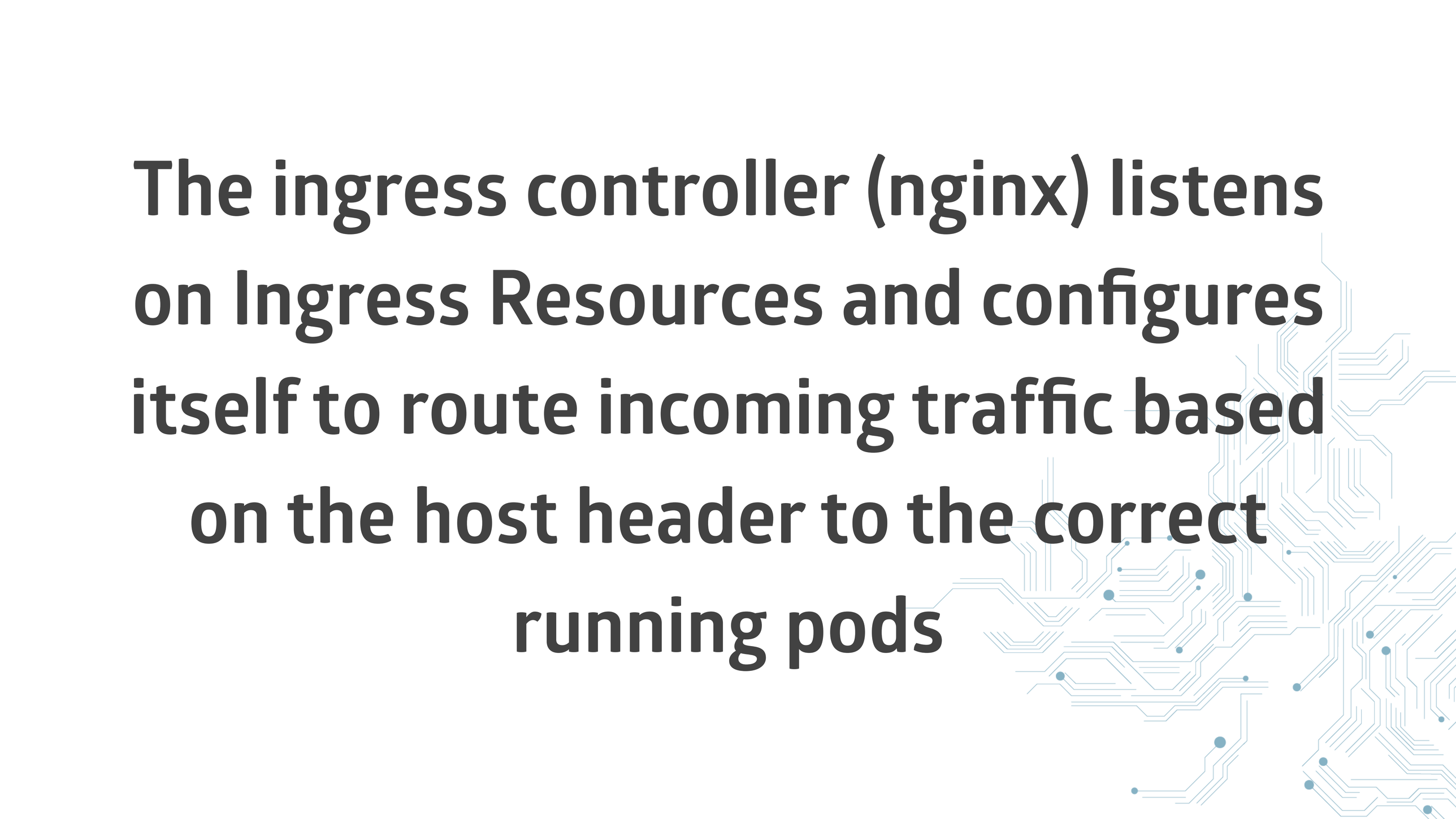


OpenStack LoadBalancer

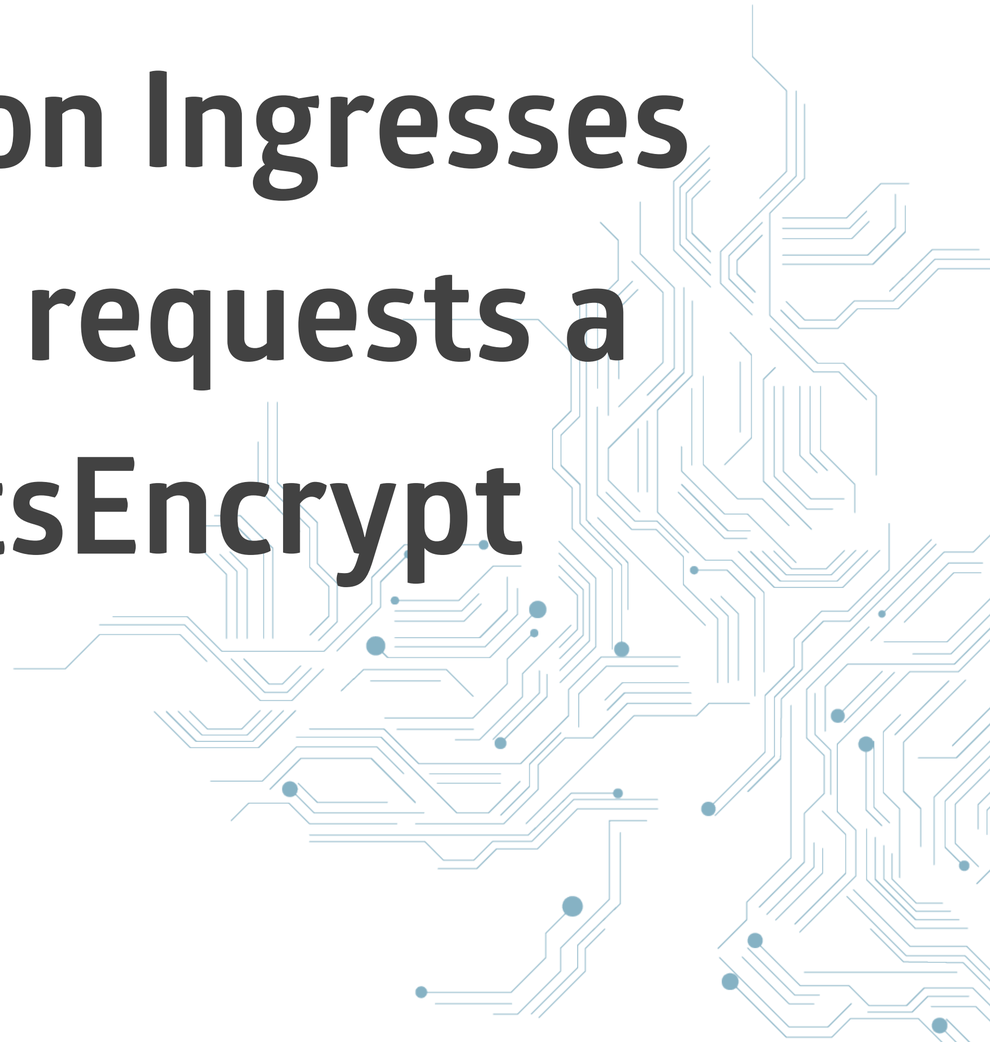


9 - Using an Ingress with TLS

The ingress controller (nginx) listens on Ingress Resources and configures itself to route incoming traffic based on the host header to the correct running pods



Cert-manager listens on Ingresses and if they want TLS, requests a certificate from LetsEncrypt



External-DNS listens on Ingresses and creates DNS entries at AWS Route 53

How is traffic routed to the Pod

OpenStack LoadBalancer