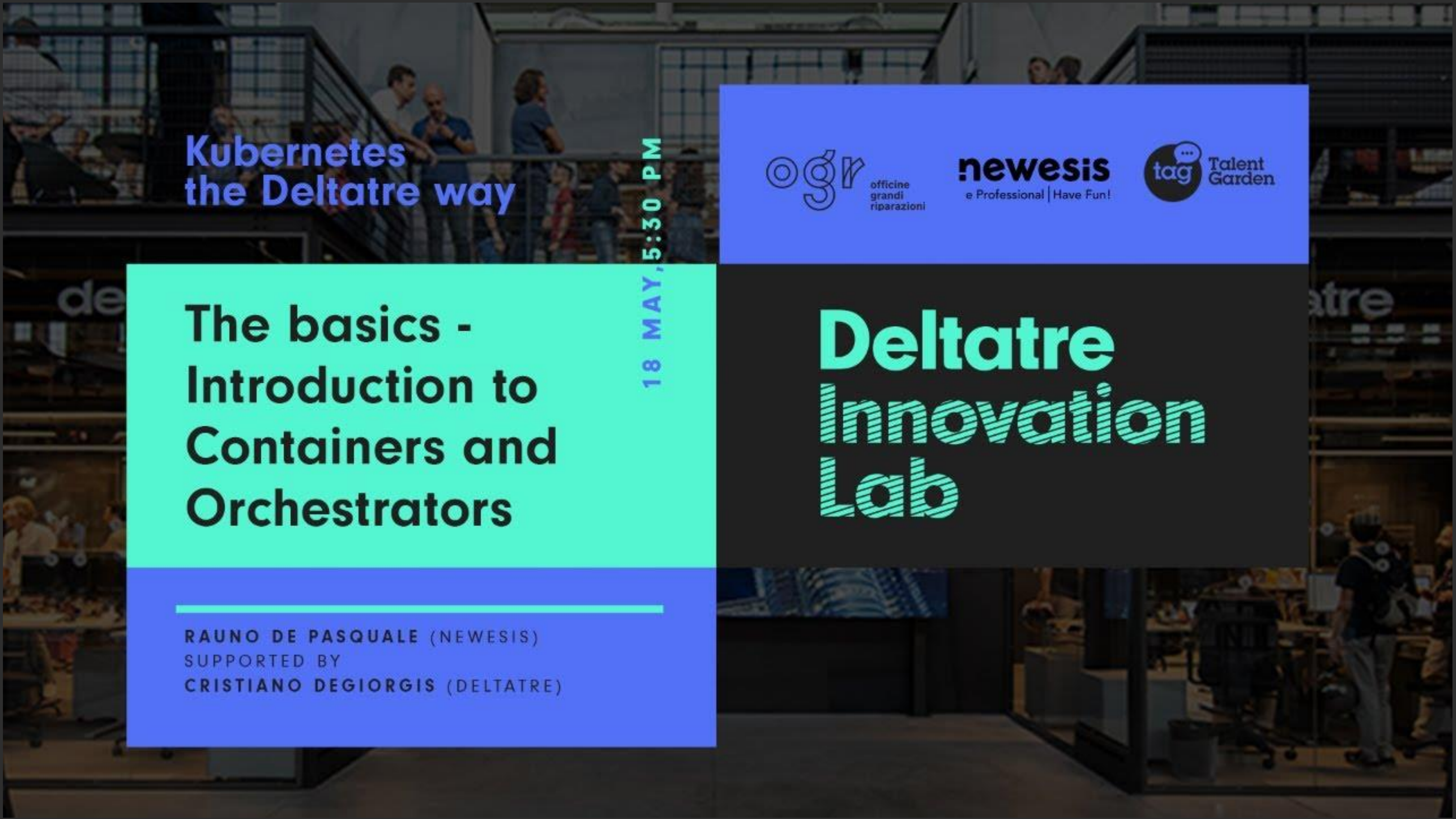


Kubernetes
the Deltatre way

18 MAY, 5:30 PM

The basics - Introduction to Containers and Orchestrators

RAUNO DE PASQUALE (NEWESIS)
SUPPORTED BY
CRISTIANO DEGIORGIS (DELTATRE)

ogr officine grandi riparazioni

newesis e Professional | Have Fun!

tag Talent Garden

Deltatre Innovation Lab

ABOUT US

- Linkedin: https://www.linkedin.com/in/rauno-de-pasquale-b075773
- Twitter: @RaunoDepa

- Rauno De Pasquale, Co-Founder and CTO at Newesis Srl, constantly trying to reconcile his degree in Philosophy with a passion for computer science. After almost 18 year at Deltatre, at the beginning of 2019 he creates Newesis, with the aim of simplifying the use of the most advanced services of Cloud platforms even in fields other than sports.

- Linkedin: https://www.linkedin.com/in/cristianodegiorgis/
- StackOverflow: https://stackoverflow.com/users/539684/crixo

- Cristiano Degiorgis, An enthusiastic *lehrling* in the IT world still feeling like Alice in wonderland after so many years being around.

AGENDA

- Knowing the context and the concepts behind the use of containers is essential to be able to proceed on the road that will lead you to master the Kubernetes and Cloud Native applications.
- This initial session covers basic skills to answer questions such as:
    - what is a container image?
    - Why did anyone feel the need for an orchestrator?
    - Are there alternatives to Docker and Kubernetes?
    - How does working with containers and Kubernetes connect to traditional virtualization?
- This session has the scope of providing the basic skills to be able to orientate in subsequent sessions where the ways of creating and running applications in the Kubernetes environment will be addressed.

- Speaker: Rauno De Pasquale (Newesis) supported by Cristiano DeGiorgis (Deltatre)
- Organised by: #DeltatreLab supported by #Newesis
- Powered by: #Deltatre
- Hashtags: #DeltatreK8S #Containers #Docker #Kubernetes #meetup #webinar

WHAT THIS SESSION IS NOT

- Training on what it is and how to use Docker
  - Wait for: Monday 25-May 17:30 --> 19:30 - Kubernetes the Deltatre way: Docker in Action
- Training on what it is and how to use Kubernetes
  - Wait for:
    - Wednesday 3-Jun 17:30 --> 19:30 - Kubernetes the Deltatre way: Kubernetes basics
    - Monday 8-Jun 17:30 --> 19:30 - Kubernetes the Deltatre way: Kubernetes advanced topics & Kind
    - Monday 15-Jun 17:30 --> 19:30 - Kubernetes the Deltatre way: Kubernetes CI/CD
    - Monday 22-Jun 17:30 --> 19:30 - Kubernetes the Deltatre way: Kubernetes extensibility: CRD & Operators

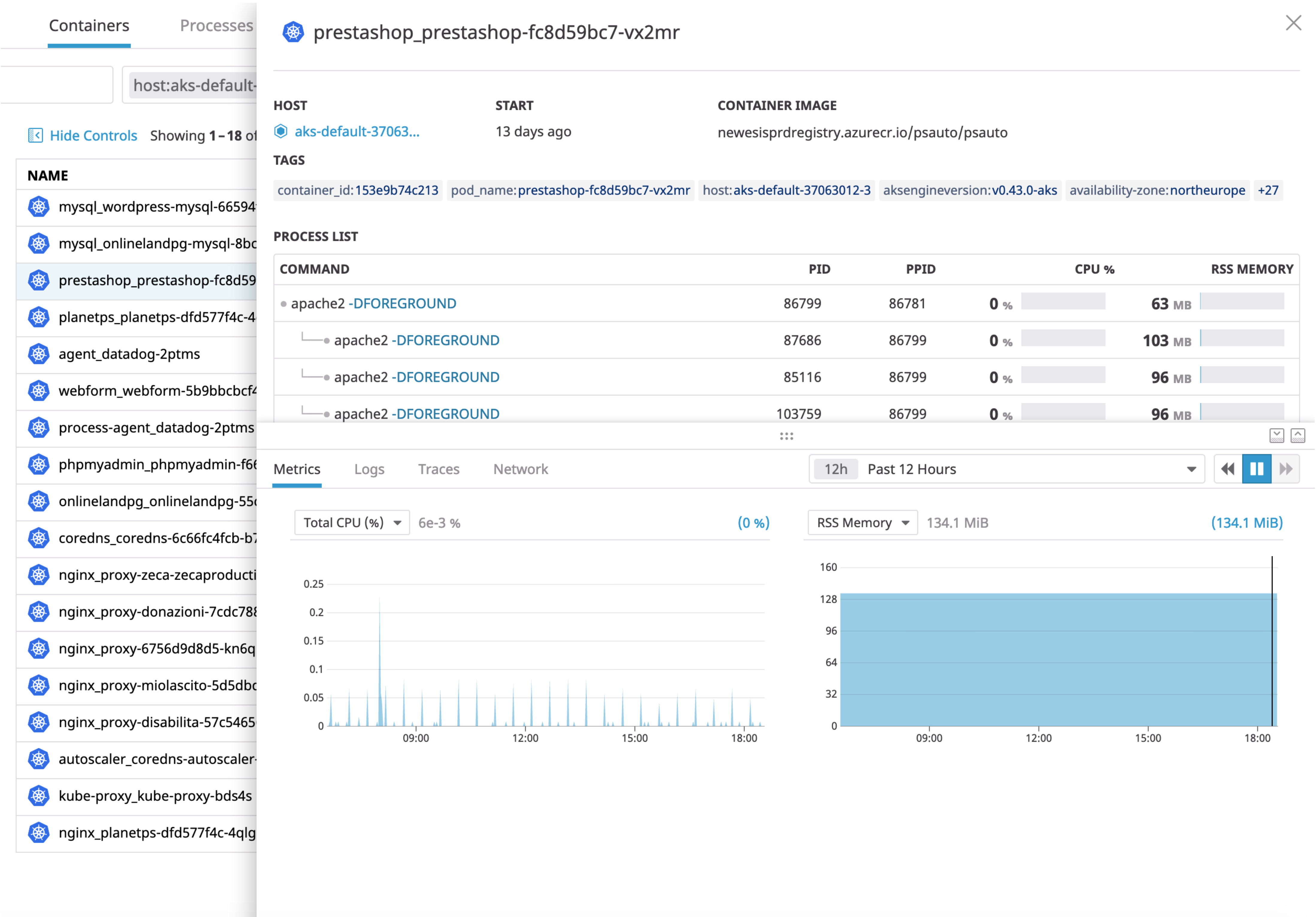
INTRODUCTION TO CONTAINERS

So, let's do an exercise with containers! Do you like containers?
Yes!!
Great! So read the exercise. And be careful, if you read the instructions really carefully, you'll be fine!
I have a trick to check if they can follow simple instructions
Oh yeah?
Yeah. One of the exercises asked them to run rm -rf / in a container...
Ohh... that's mean...
I mean, it's a good way of seeing who isn't capable of following simple instructions...
Sir, I have a problem...
Me too... my computer has frozen...
CommitStrip.com

WHAT IS A CONTAINER?

- “A container is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another. “ (Docker web site)
- “Containers offer a logical packaging mechanism in which applications can be abstracted from the environment in which they actually run.” (Google Cloud web site)
- “Linux containers are implementations of operating system-level virtualization for the Linux operating system.” “OS-level virtualization refers to an operating system paradigm in which the kernel allows the existence of multiple isolated user space instances. Such instances, called containers (Solaris, Docker), Zones (Solaris), virtual private servers (OpenVZ), partitions, virtual environments (VEs), virtual kernel (DragonFly BSD), or jails (FreeBSD jail or chroot jail),[1] may look like real computers from the point of view of programs running in them. “ (Wikipedia on Linux Containers and OS-Level virtualisation)
- “Isolated area of an OS with resource limits usage applied” (Nigel Poulton, book “Docker Deep Dive”)

A CONTAINER IS A GROUP OF PROCESSES ...



Containers | Processes

host:aks-default-

Hide Controls Showing 1 – 18 of

NAME
mysql_wordpress-mysql-66594
mysql_onlinelandpg-mysql-8bc
prestashop_prestashop-fc8d59
planetps_planetps-dfd577f4c-4
agent_datadog-2ptms
webform_webform-5b9bbcbcf4
process-agent_datadog-2ptms
phpmyadmin_phpmyadmin-f6€
onlinelandpg_onlinelandpg-55
coredns_coredns-6c66fc4fcb-b7
nginx_proxy-zeca-zecaproducti
nginx_proxy-donazioni-7cdc78
nginx_proxy-6756d9d8d5-kn6q
nginx_proxy-miolascito-5d5dbc
nginx_proxy-disabilita-57c5465
autoscaler_coredns-autoscaler-
kube-proxy_kube-proxy-bds4s
nginx_planetps-dfd577f4c-4qlg

prestashop_prestashop-fc8d59bc7-vx2mr

HOST	START	CONTAINER IMAGE
aks-default-37063...	13 days ago	newesisprdregistry.azurecr.io/psauto/psauto

TAGS

container_id:153e9b74c213 pod_name:prestashop-fc8d59bc7-vx2mr host:aks-default-37063012-3 aksengineversion:v0.43.0-aks availability-zone:northeurope +27

PROCESS LIST

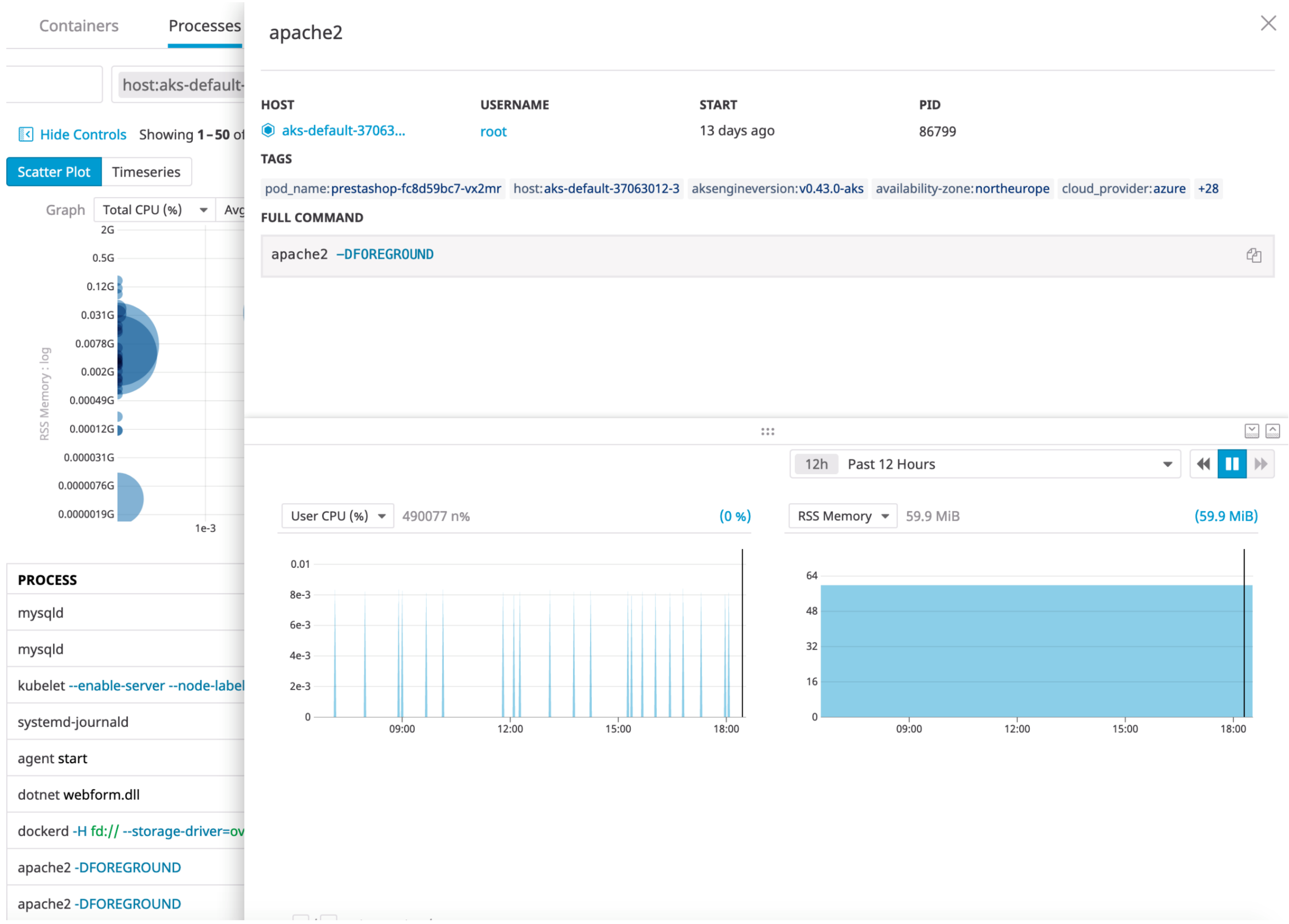
COMMAND	PID	PPID	CPU %	RSS MEMORY
apache2 -DFOREGROUND	86799	86781	0 %	63 MB
└─ apache2 -DFOREGROUND	87686	86799	0 %	103 MB
└─ apache2 -DFOREGROUND	85116	86799	0 %	96 MB
└─ apache2 -DFOREGROUND	103759	86799	0 %	96 MB

Metrics | Logs | Traces | Network

12h Past 12 Hours

Total CPU (%) 6e-3 % (0 %)

RSS Memory 134.1 MiB (134.1 MiB)



Containers | Processes

host:aks-default-

Hide Controls Showing 1 – 50 of

Scatter Plot | Timeseries

Graph Total CPU (%) Avg

PROCESS
mysqld
mysqld
kubelet --enable-server --node-label
systemd-journald
agent start
dotnet webform.dll
dockerd -H fd:// --storage-driver=ov
apache2 -DFOREGROUND
apache2 -DFOREGROUND

apache2

HOST	USERNAME	START	PID
aks-default-37063...	root	13 days ago	86799

TAGS

pod_name:prestashop-fc8d59bc7-vx2mr host:aks-default-37063012-3 aksengineversion:v0.43.0-aks availability-zone:northeurope cloud_provider:azure +28

FULL COMMAND

```
apache2 -DFOREGROUND
```

12h Past 12 Hours

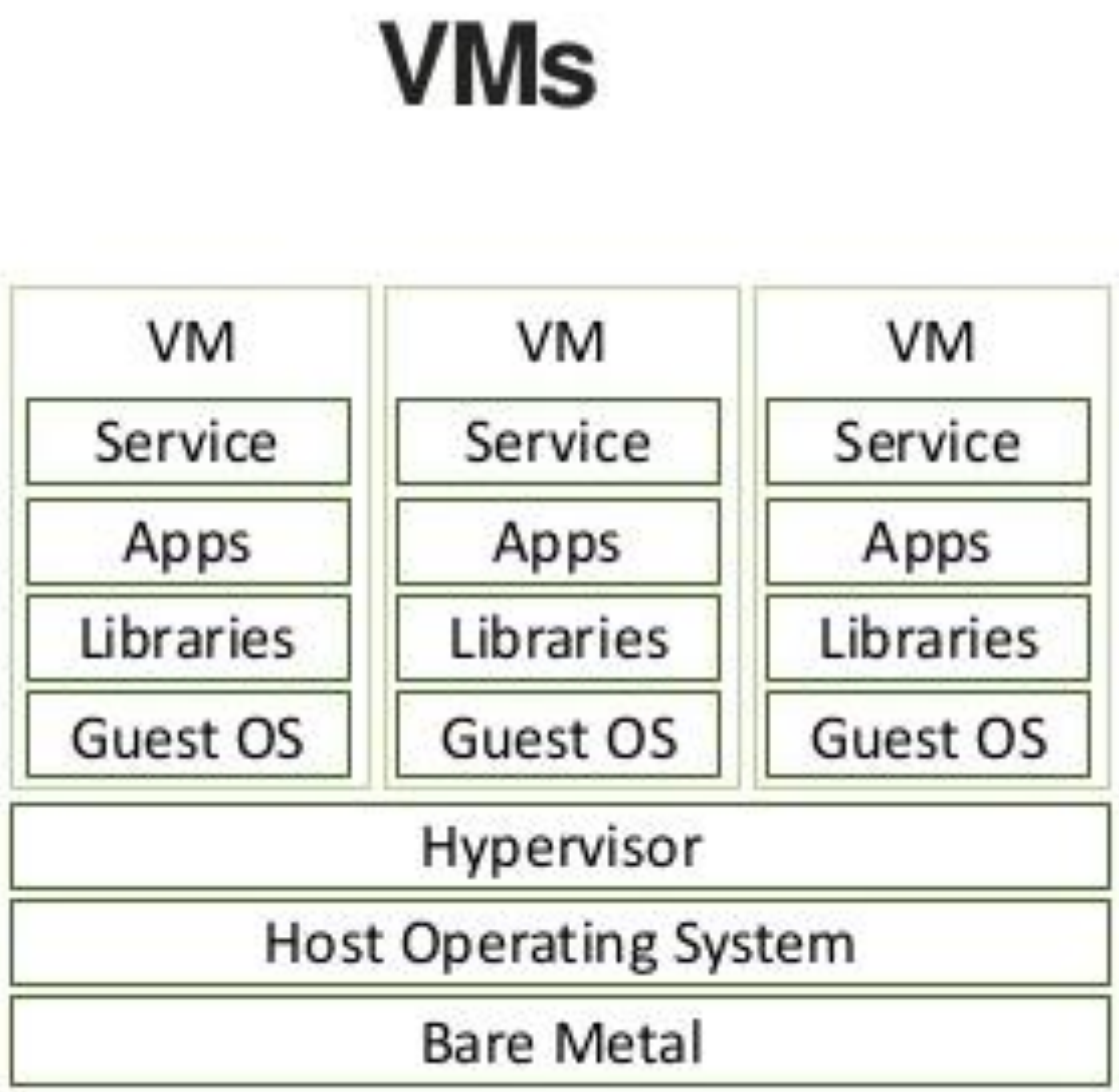
User CPU (%) 490077 n% (0 %)

RSS Memory 59.9 MiB (59.9 MiB)

… RESTRICTED TO A PRIVATE NAMESPACE

- “Namespaces are a feature of the Linux kernel that partitions kernel resources such that one set of processes sees one set of resources while another set of processes sees a different set of resources. The feature works by having the same namespace for a set of resources and processes, but those namespaces refer to distinct resources. Resources may exist in multiple spaces. Examples of such resources are process IDs, hostnames, user IDs, file names, and some names associated with network access, and interprocess communication. “ (Wikipedia – Linux namespaces)
- “cgroups (abbreviated from control groups) is a Linux kernel feature that limits, accounts for, and isolates the resource usage (CPU, memory, disk I/O, network, etc.) of a collection of processes..” (Wikipedia - Cgroups)
- “cgroups, which stands for control groups, are a kernel mechanism for limiting and measuring the total resources used by a group of processes running on a system. For example, you can apply CPU, memory, network or IO quotas. cgroups were originally developed by Paul Menage and Rohit Seth of Google, and their first features were merged into Linux 2.6.24.“ (Duncan Macrae - How Linux Kernel Cgroups And Namespaces Made Modern Containers Possible)
- “Namespaces are a kernel mechanism for limiting the visibility that a group of processes has of the rest of a system. For example you can limit visibility to certain process trees, network interfaces, user IDs or filesystem mounts. namespaces were originally developed by Eric Biederman, and the final major namespace was merged into Linux 3.8.” (Duncan Macrae - How Linux Kernel Cgroups And Namespaces Made Modern Containers Possible)

CONTAINERS VS VIRTUAL MACHINES



VMs

VM | VM | VM

Service | Service | Service

Apps | Apps | Apps

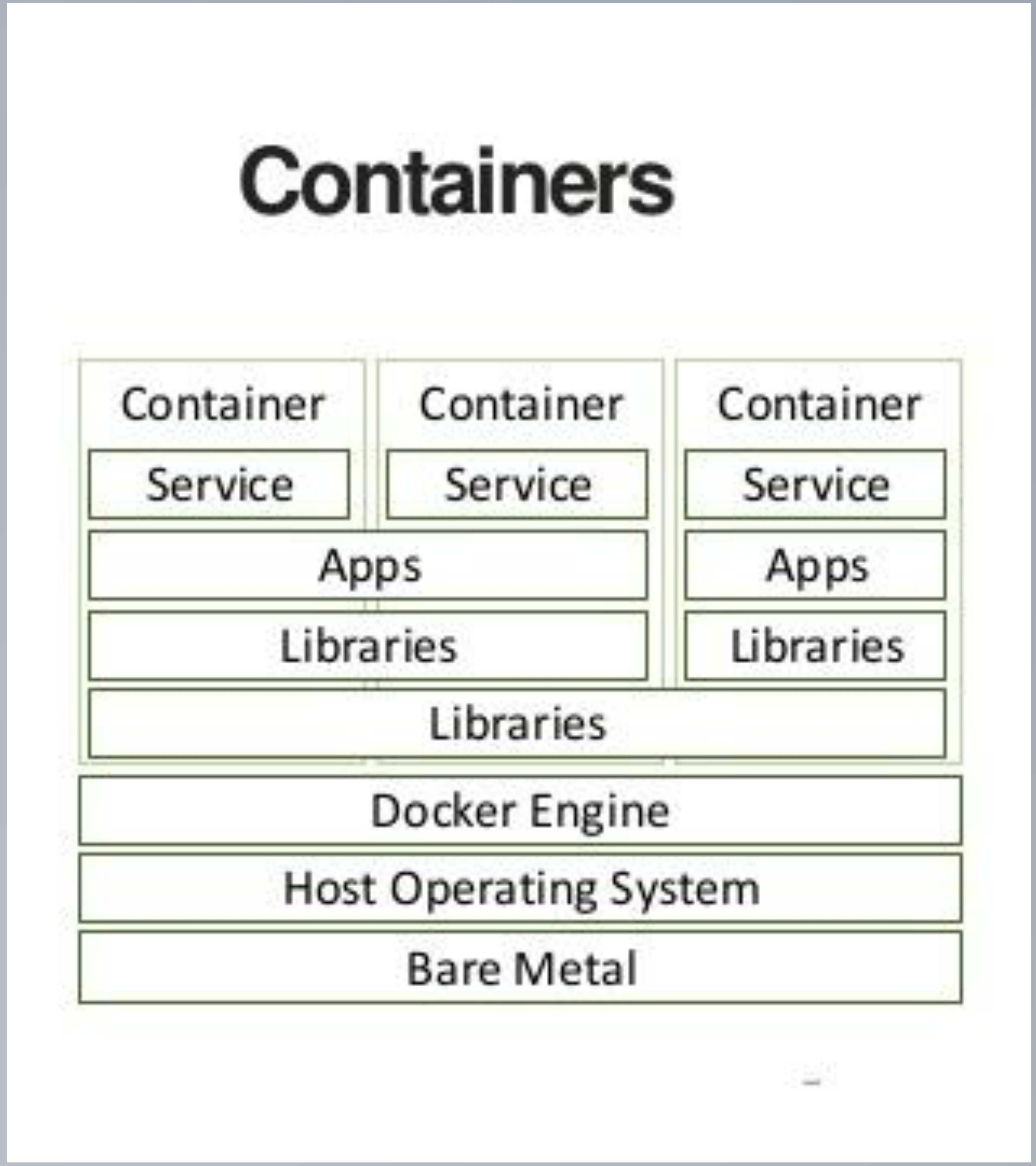
Libraries | Libraries | Libraries

Guest OS | Guest OS | Guest OS

Hypervisor

Host Operating System

Bare Metal



Containers

Container | Container | Container

Service | Service | Service

Apps | Apps

Libraries | Libraries

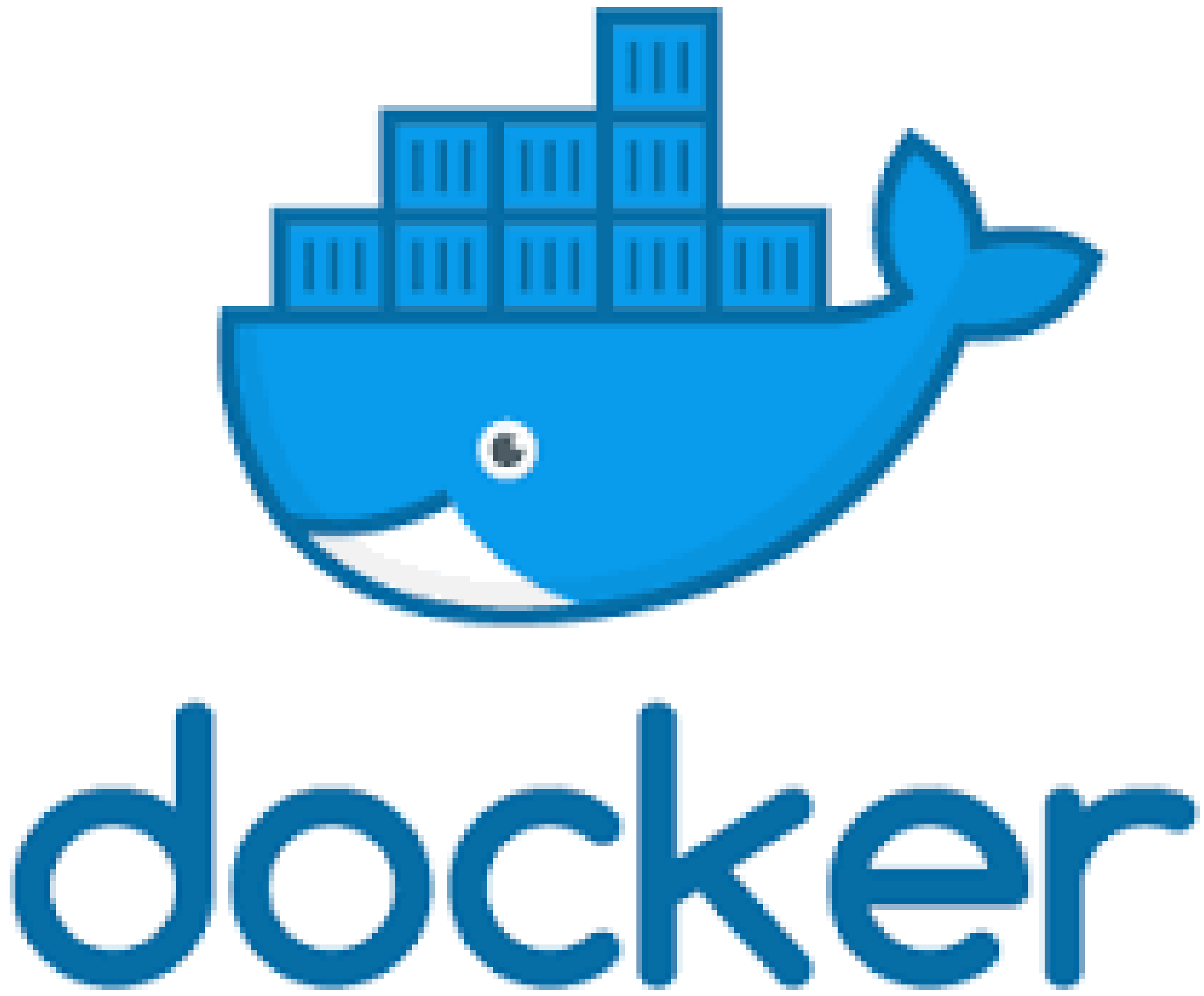
Libraries

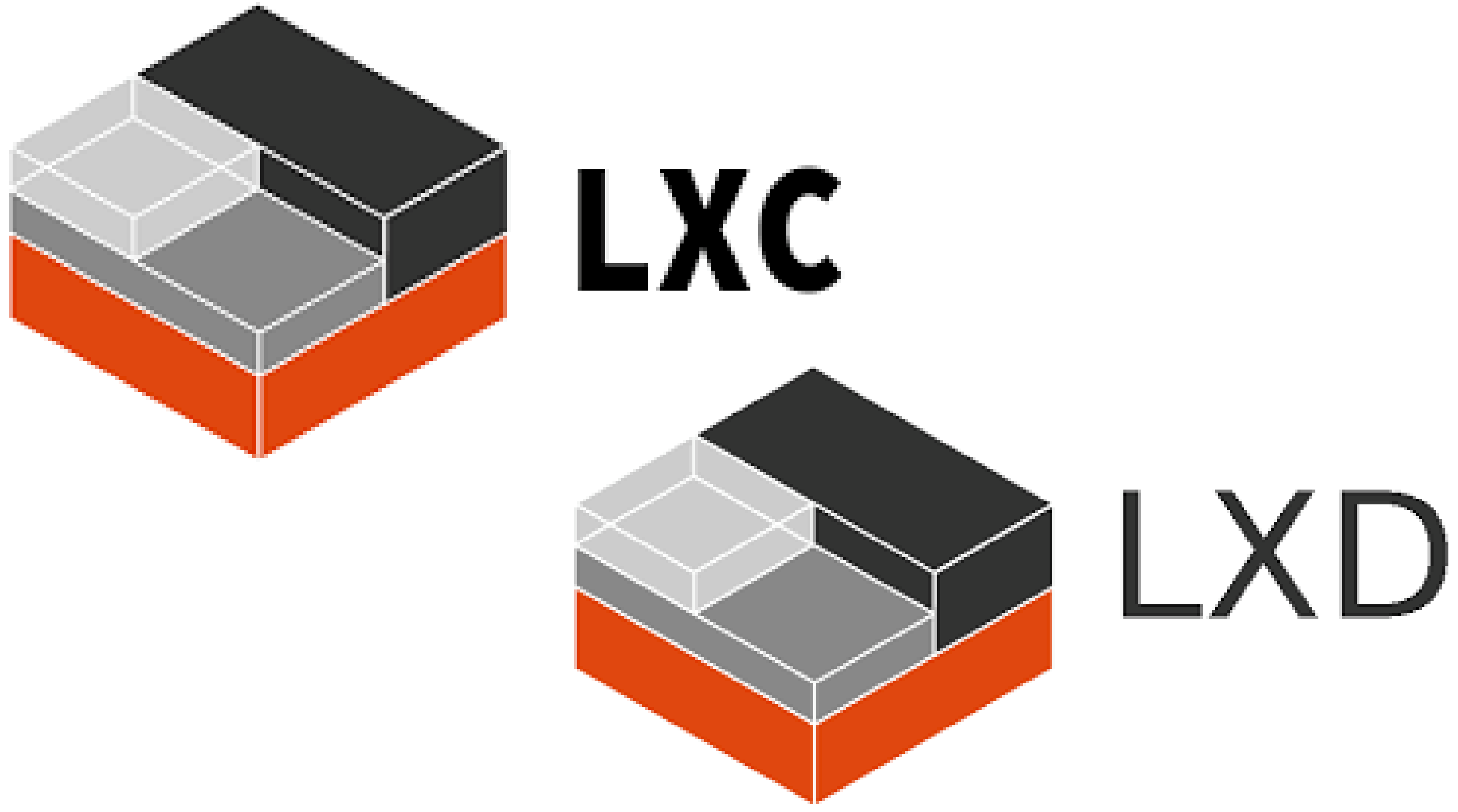
Docker Engine

Host Operating System

Bare Metal

DOES CONTAINER MEAN DOCKER?



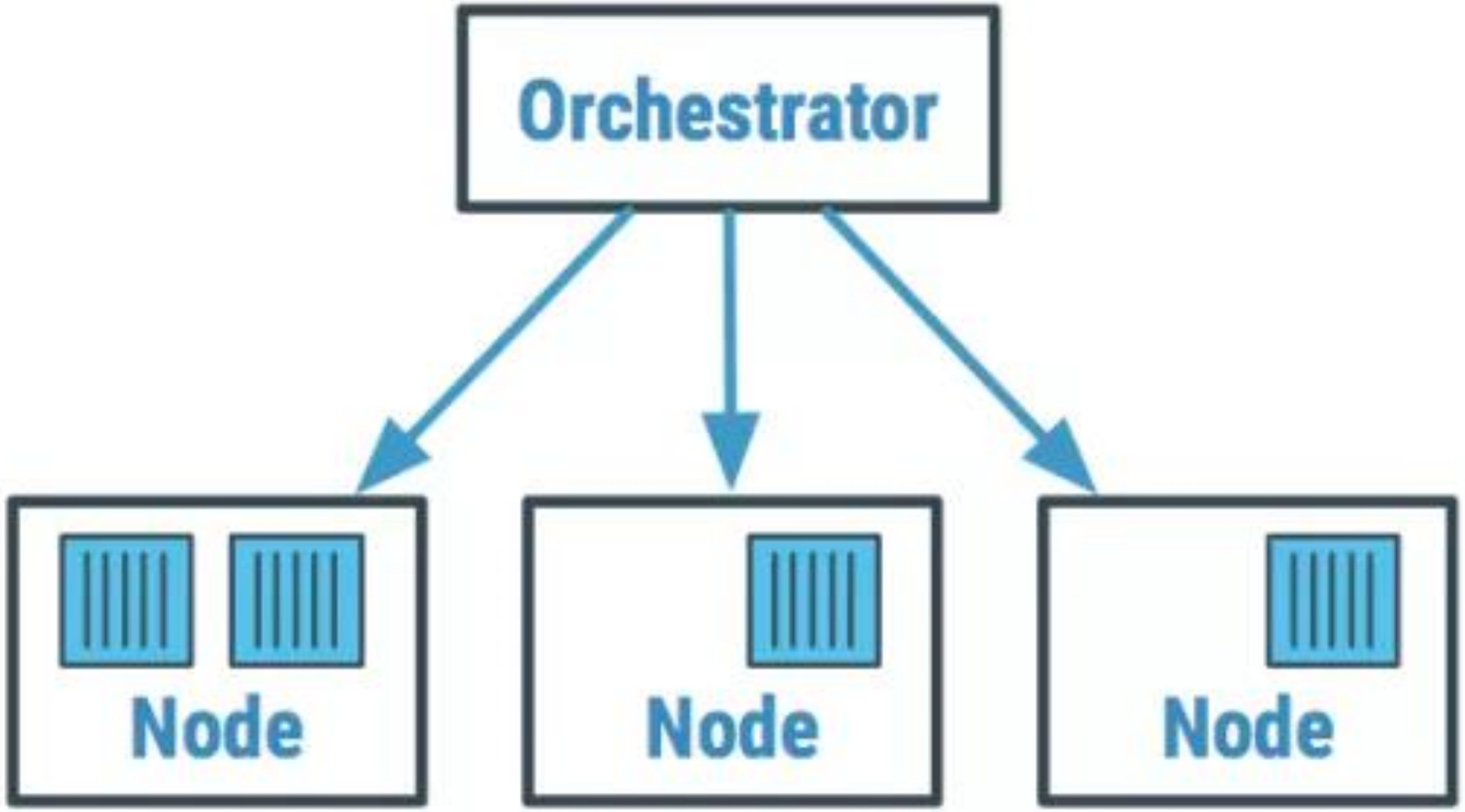


OK CONTAINERS BUT WHY ORCHESTRATORS?

I think I've decided on a stack for this project, but I still have some doubts...
Don't think about it too much, just go for it
OK it's working great, but I keep wondering if it would have been better with a different stack
Who knows
Maybe it would have worked better with a different stack?
GO-LIVE PARTY
NO-ONE CARES! Everything is running smoothly in production, that's all that matters
LATER
I can't help thinking... maybe a different stack...
Oh Mom! Give it up already!
CommitStrip.com

CMA CG

ORCHESTRATORS



DOES ORCHESTRATOR MEAN KUBERNETES?

Container Orchestration Tools

Marathon (Mesosphere)

Docker Swarm

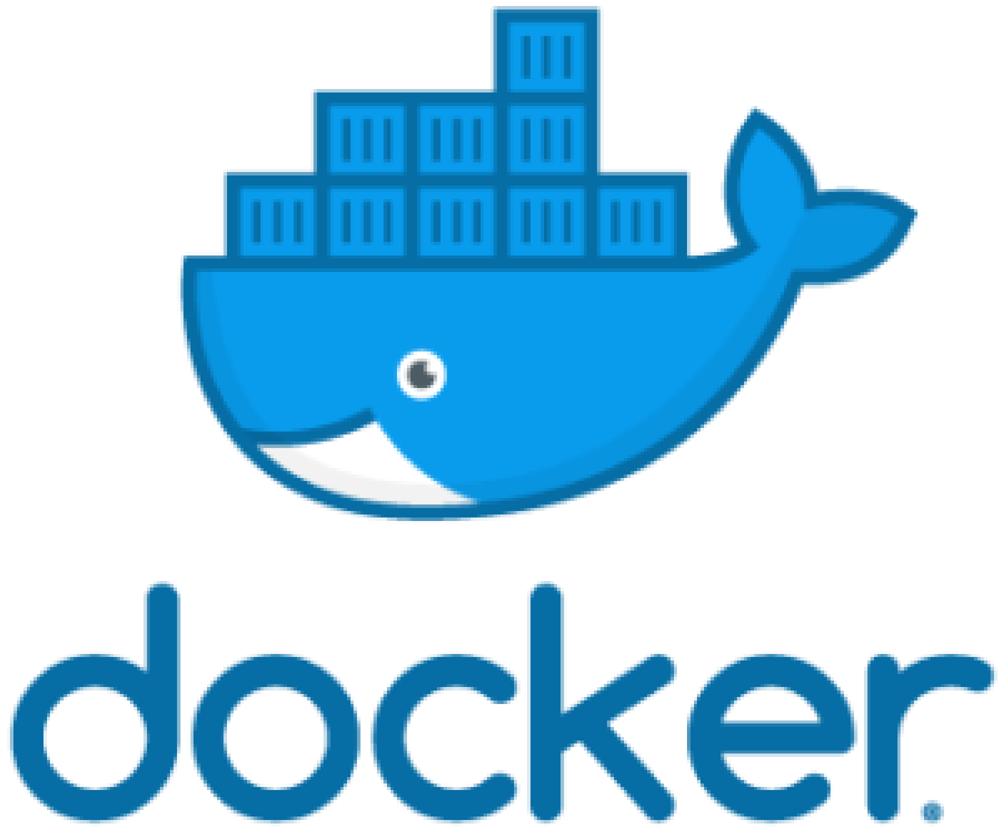
Nomad (HashiCorp)

Kubernetes

WHY DOCKER AND KUBERNETES?

Well, as per usual I don't know which talk to choose...
There's a session on the future of web hosting which sounds good... or maybe the one about virtual assistants...
Don't overcomplicate it, I've found the perfect talk for us
HOW TO BUILD A CHATBOT FOR BLOCKCHAIN SMART CONTRACTS USING GRAPHQL WITH SERVERLESS APIS SUPPORTED BY A SERVICE MESH
Perfect!
CommitStrip.com

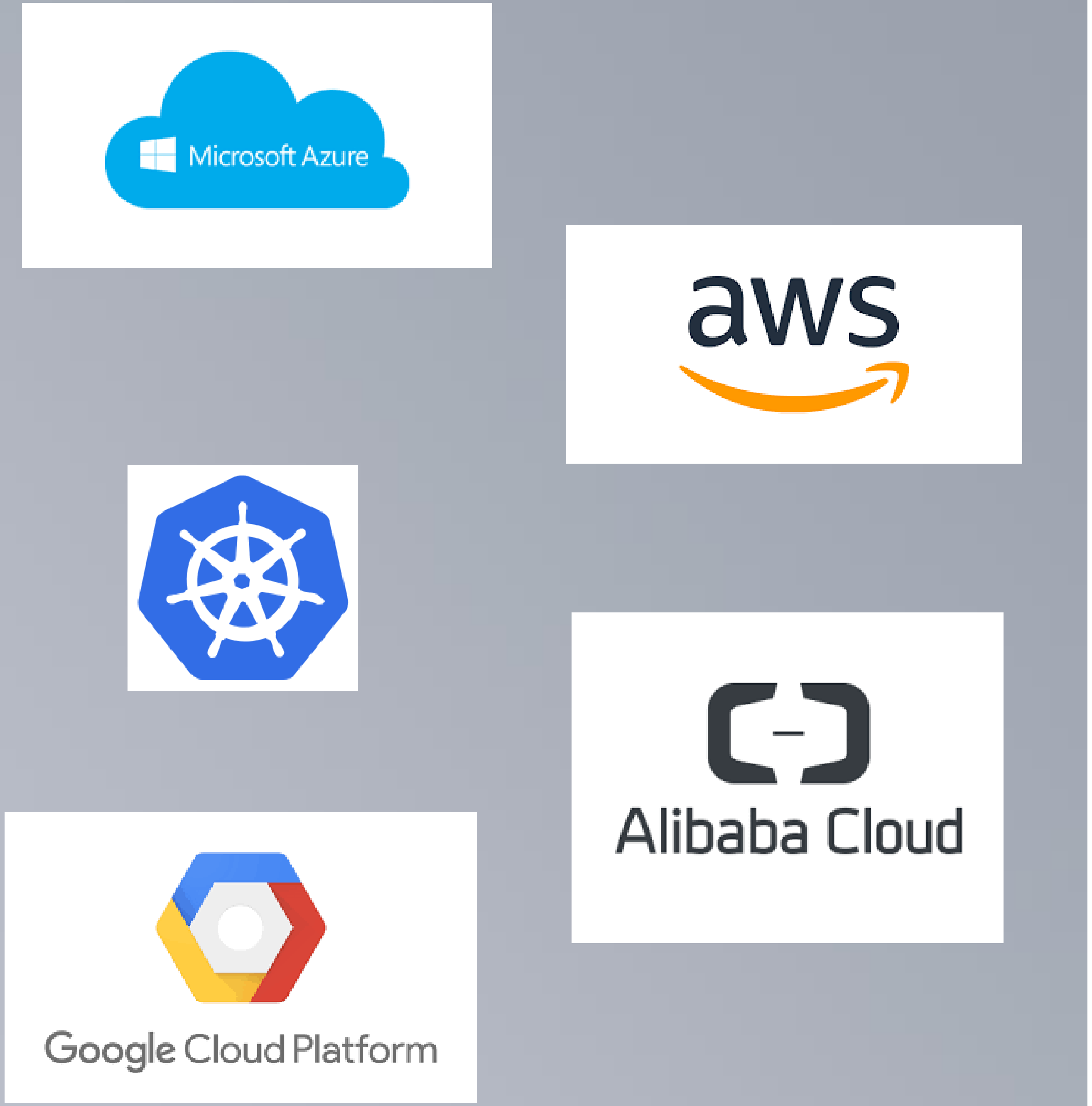
DELTATRE STRATEGY



- Docker and Kubernetes have the larger communities and larger adoption
- Fully supported by all major Cloud providers
- Fully supported for an OnPremises configuration
- Part of the Open Containers Initiatives
- Part of the Cloud Native Computing Foundation
- Docker supports Kubernetes (now part of the Entepirse Edition)
- Docker support migration from Swarm to Kubernetes
- Google Borg as foundation of Kubernetes

CLOUD AGNOSTIC

Portable Solutions

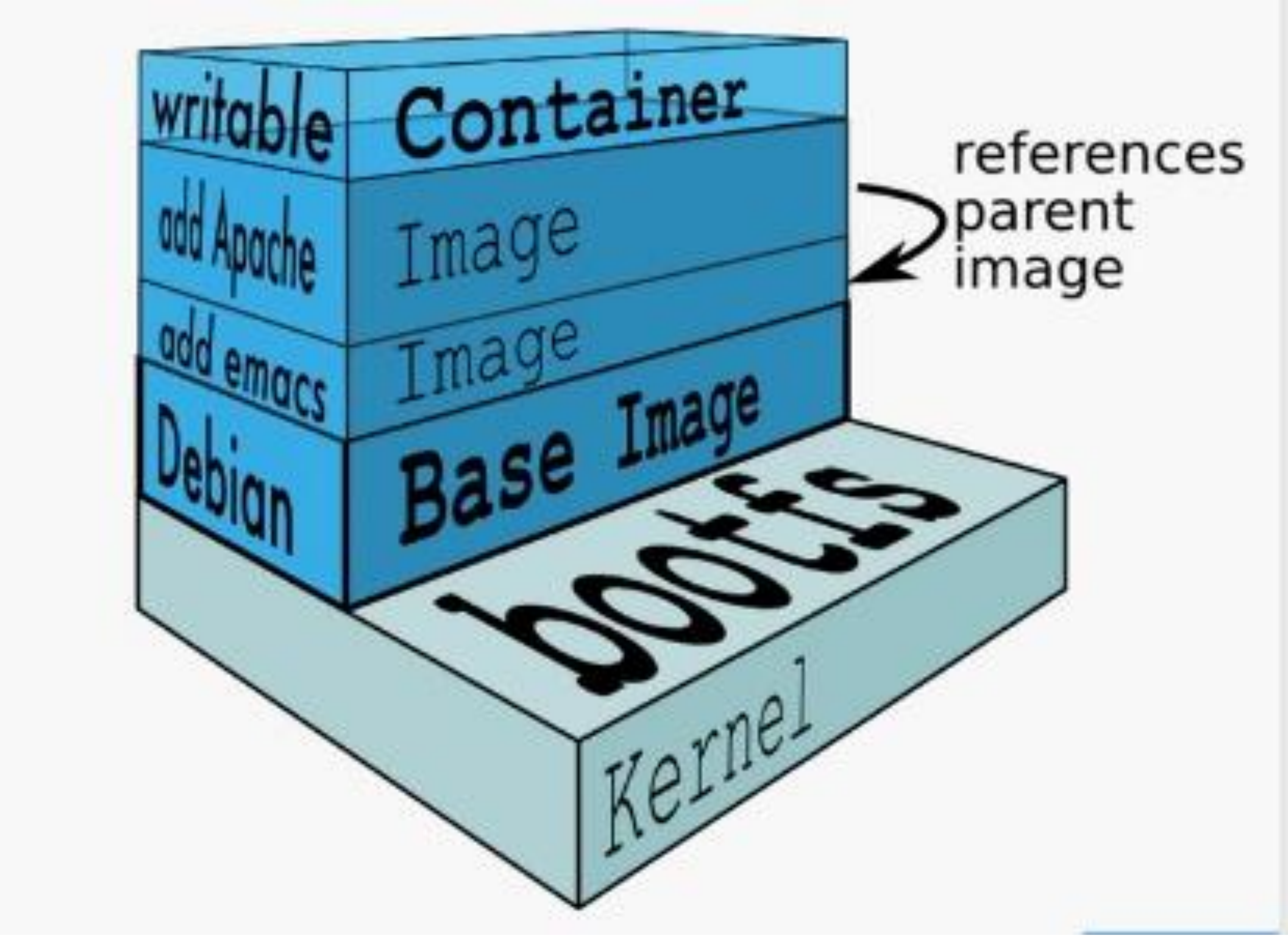


- Reusable components and products must be Cloud Agnostic
- Container images able to run on Linux OS
  - NodeJS
  - .Net Core
- Docker images and Kubernetes based deployments
- MongoDB and in general intensive IO applications to be installed into VMs and not as containers
- Usage of PaaS only if replaceable with alternatives (e.g. CosmosDB in Azure is ok if development is done to preserve compatibility with MongoDB)

Docker

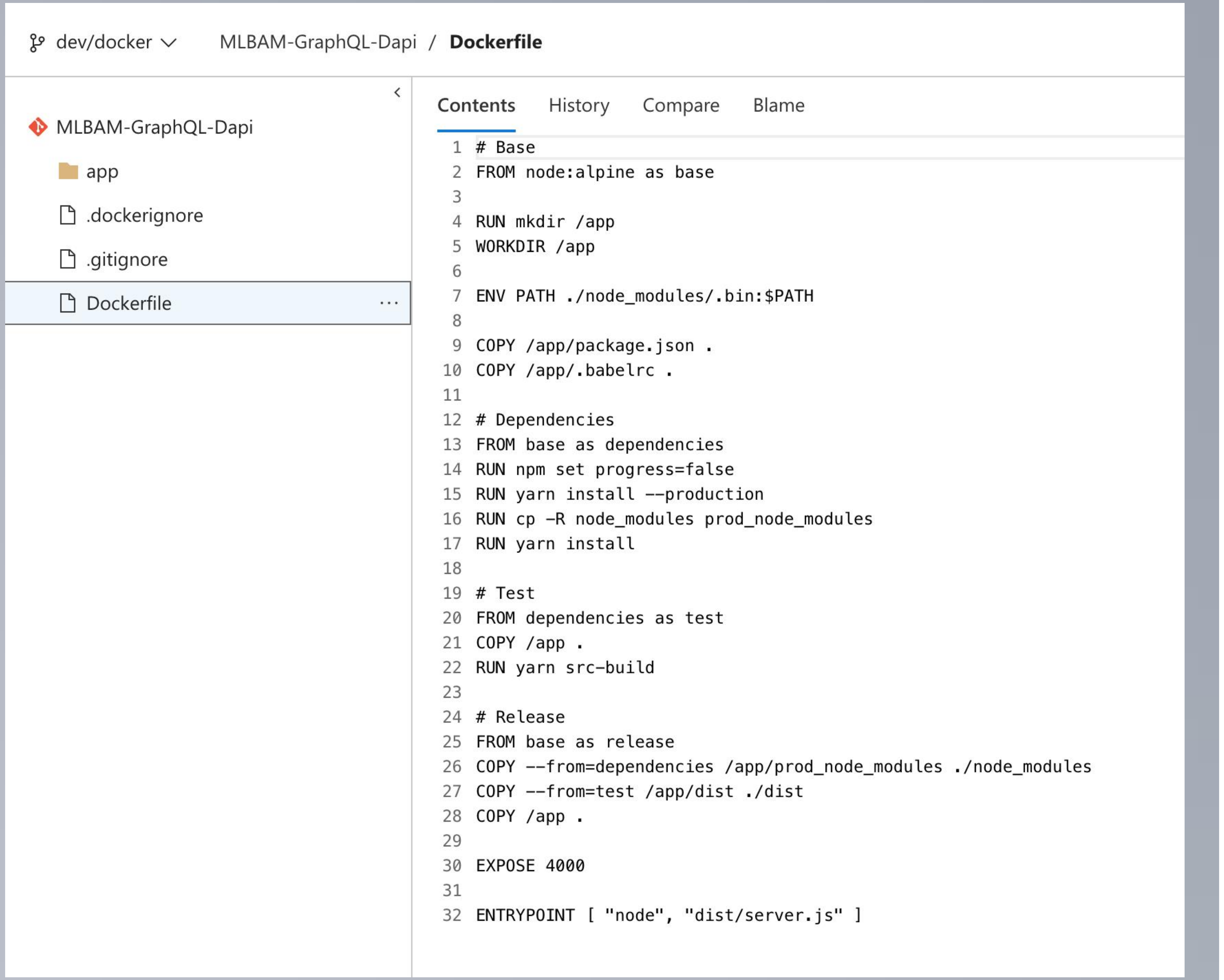
DOCKER BASICS

- ➤ Dockerfile
  - ➤ Source code of an image
- ➤ Image
  - ➤ Immutable package of application and its dependencies
  - ➤ Composed by multiple layers
- ➤ Container
  - ➤ Running instance of an image
- ➤ Registry
  - ➤ Repository of images
- ➤ Docker Daemon
  - ➤ Build images
  - ➤ Run Containers
- ➤ Docker CLI



DOCKERFILE

Image build instruction



```
# Base
FROM node:alpine as base

RUN mkdir /app
WORKDIR /app

ENV PATH ./node_modules/.bin:$PATH

COPY /app/package.json .
COPY /app/.babelrc .

# Dependencies
FROM base as dependencies
RUN npm set progress=false
RUN yarn install --production
RUN cp -R node_modules prod_node_modules
RUN yarn install

# Test
FROM dependencies as test
COPY /app .
RUN yarn src-build

# Release
FROM base as release
COPY --from=dependencies /app/prod_node_modules ./node_modules
COPY --from=test /app/dist ./dist
COPY /app .

EXPOSE 4000

ENTRYPOINT [ "node", "dist/server.js" ]
```

- ➤A dockerfile contains the instruction for the docker build process on how to create a new image
- ➤Build of an image is done by executing command inside a container
- ➤A container is the execution of an image
- ➤Multi-Stage builds should be used to optimise image creation process and image size

Kubernetes

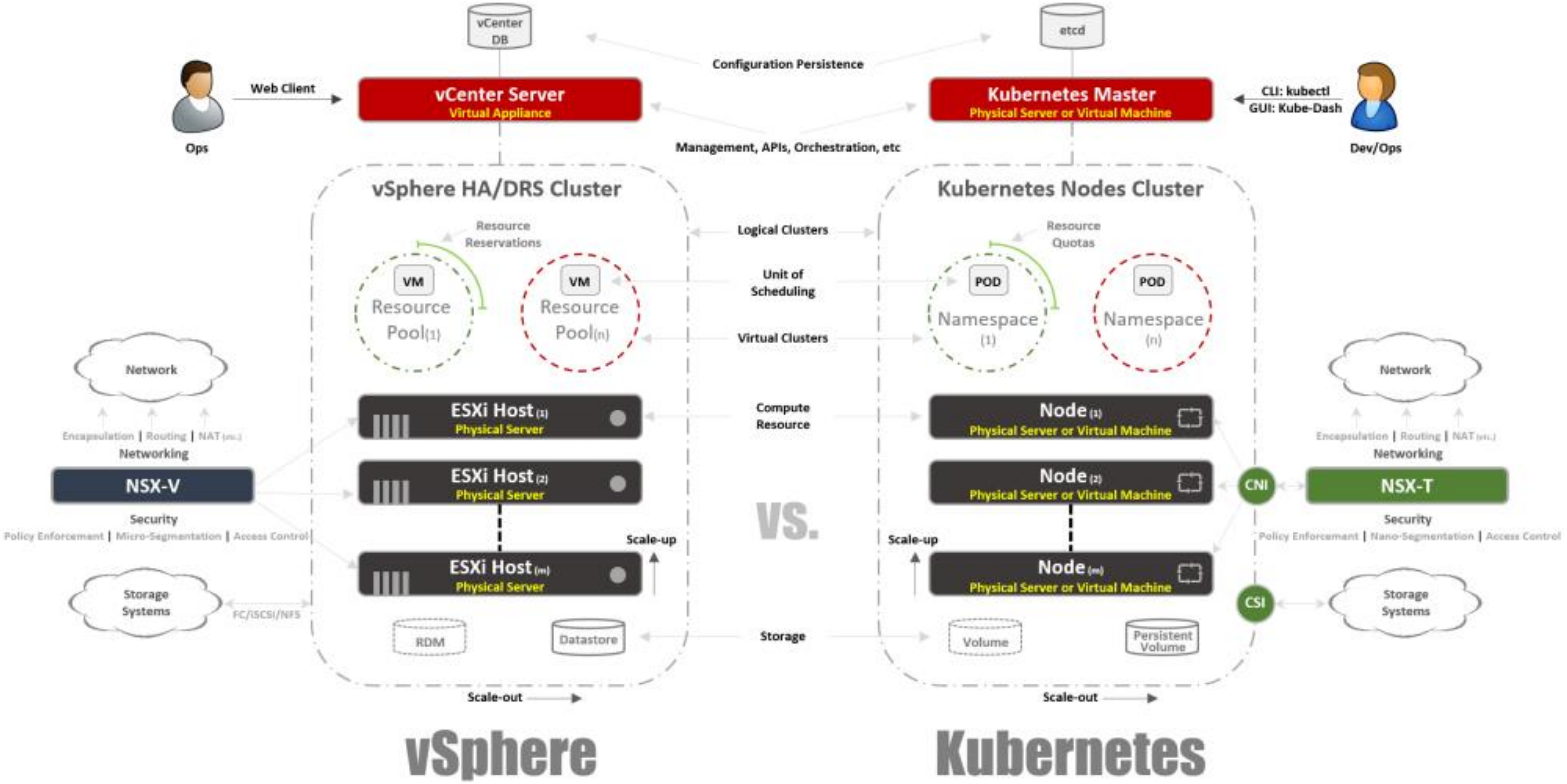
KUBERNETES – THE ORIGIN

- Greek for “Helmsman”; also the root of the word “governor” and “cybernetic”
- Orchestrator for containers
- Builds on Docker containers
- Also supporting other container technologies
- Multi-cloud and bare-metal environments
- Inspired and informed by Google’s experiences and internal systems
- 100% Open Source, written in Go
- Created by three Google employees initially during the summer of 2014; grew exponentially and became the first project to get donated to the CNCF
- Release 1.0 21st July 2015

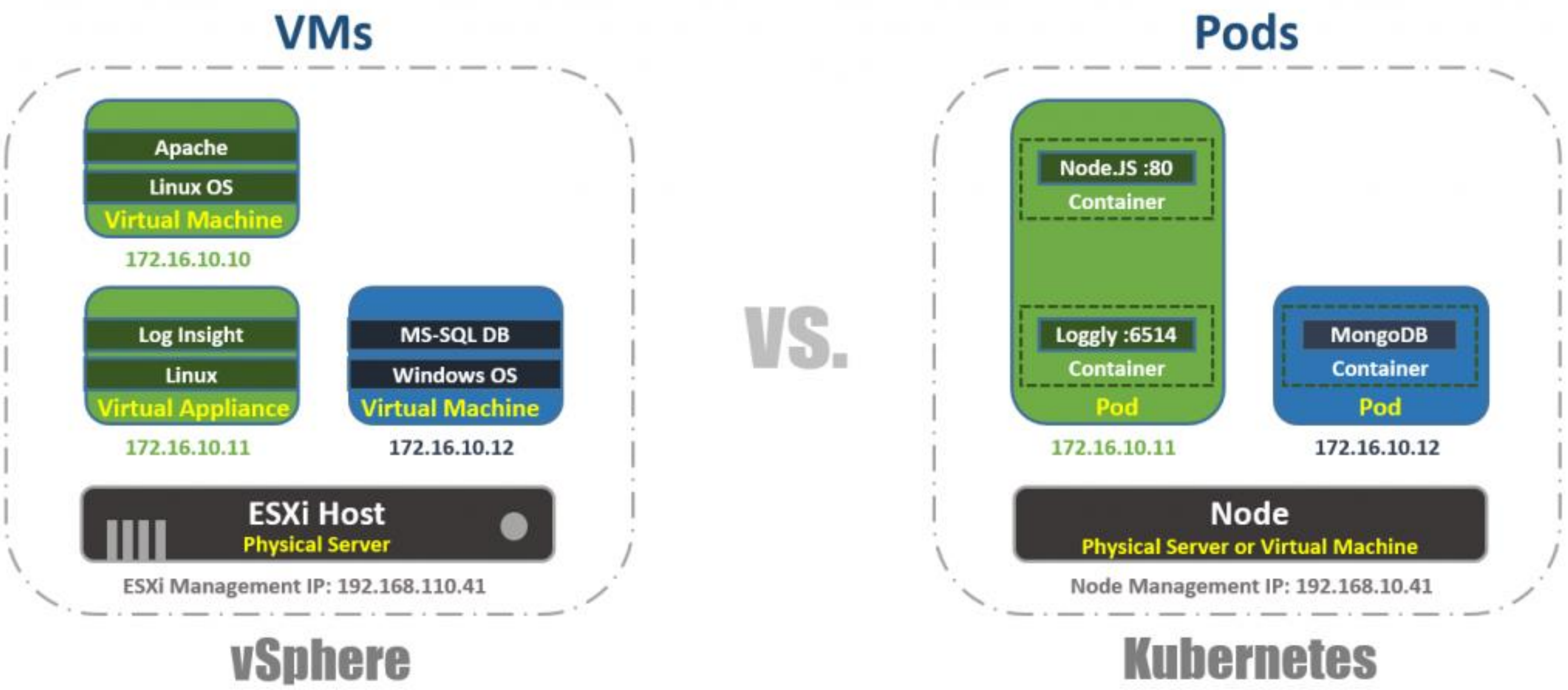
KUBERNETES – THE BASIC CONCEPTS

- It all started with Google growing and experiencing problems on managing the new scale of hardware and software
- The Datacentre as a Computer (https://research.google/pubs/pub35290/)
- Abstract completely hardware (software defined datacentre)
- Abstract completely from network (software define network)
- Declarative application deployment (deploy is documentation)
- Self-Healing system based on desired state
- Ability to configure rules for automatic scaling
- Designed for multi-tenant
- Designed for integration (“API first” approach)

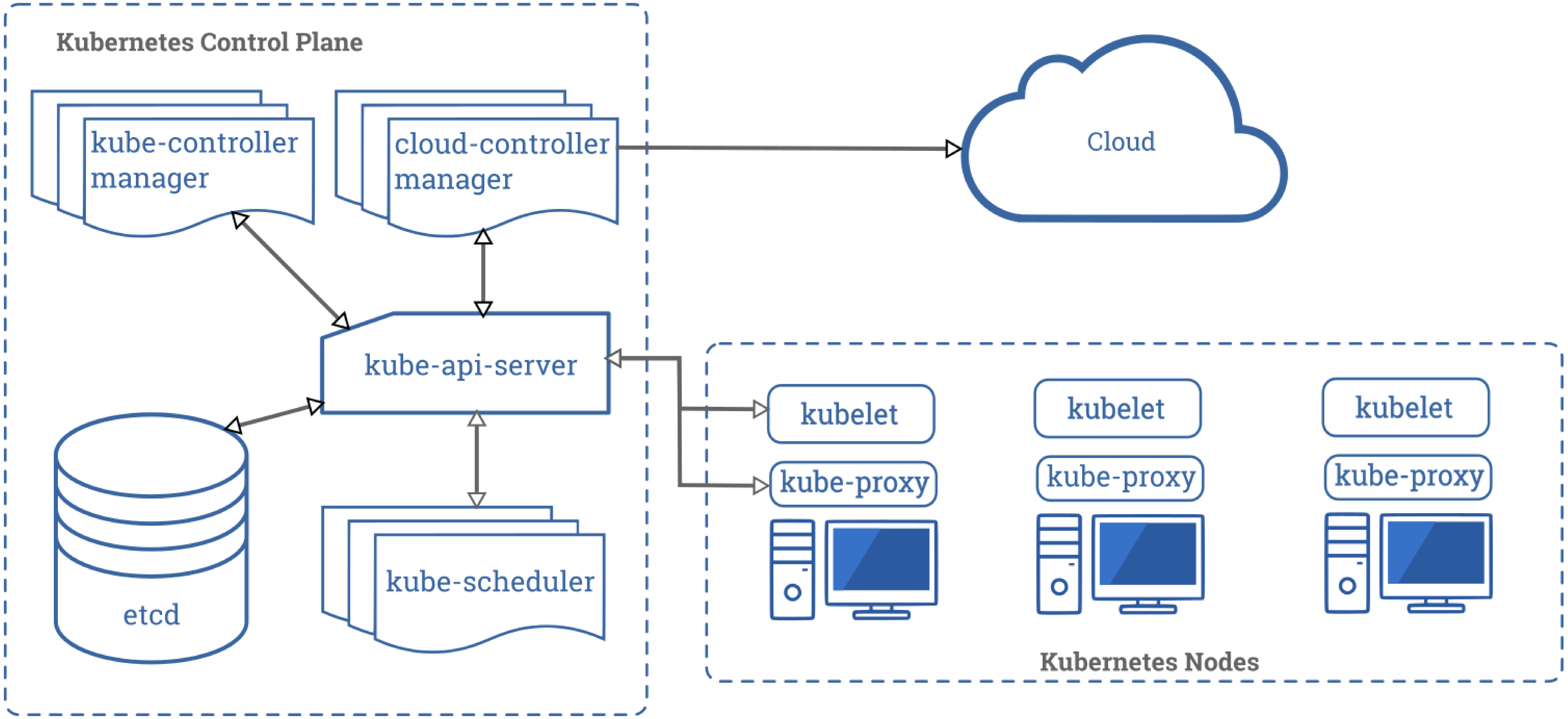
KUBERNETES VS VIRTUAL MACHINES



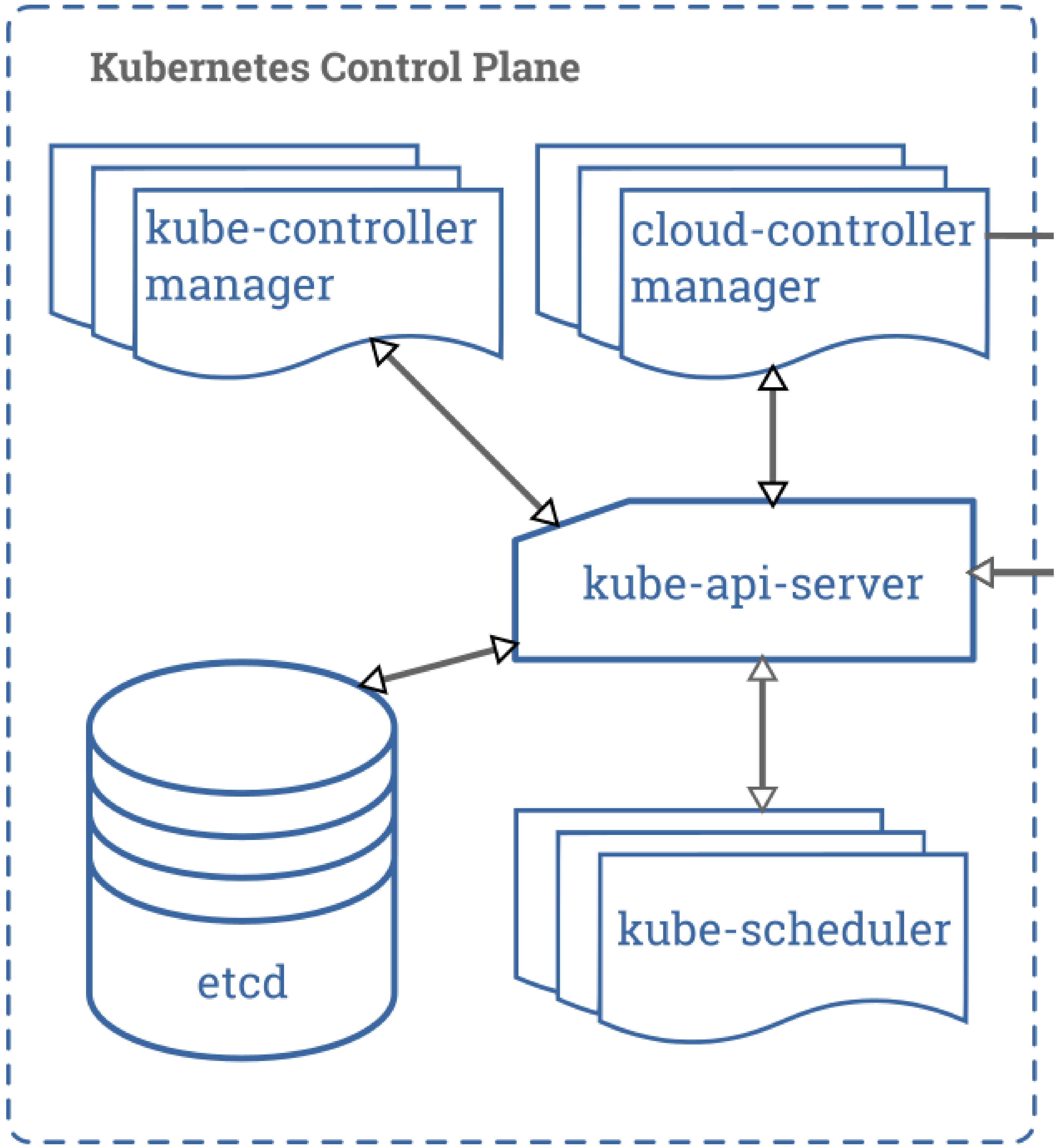
KUBERNETES VS VIRTUAL MACHINES



KUBERNETES ARCHITECTURE



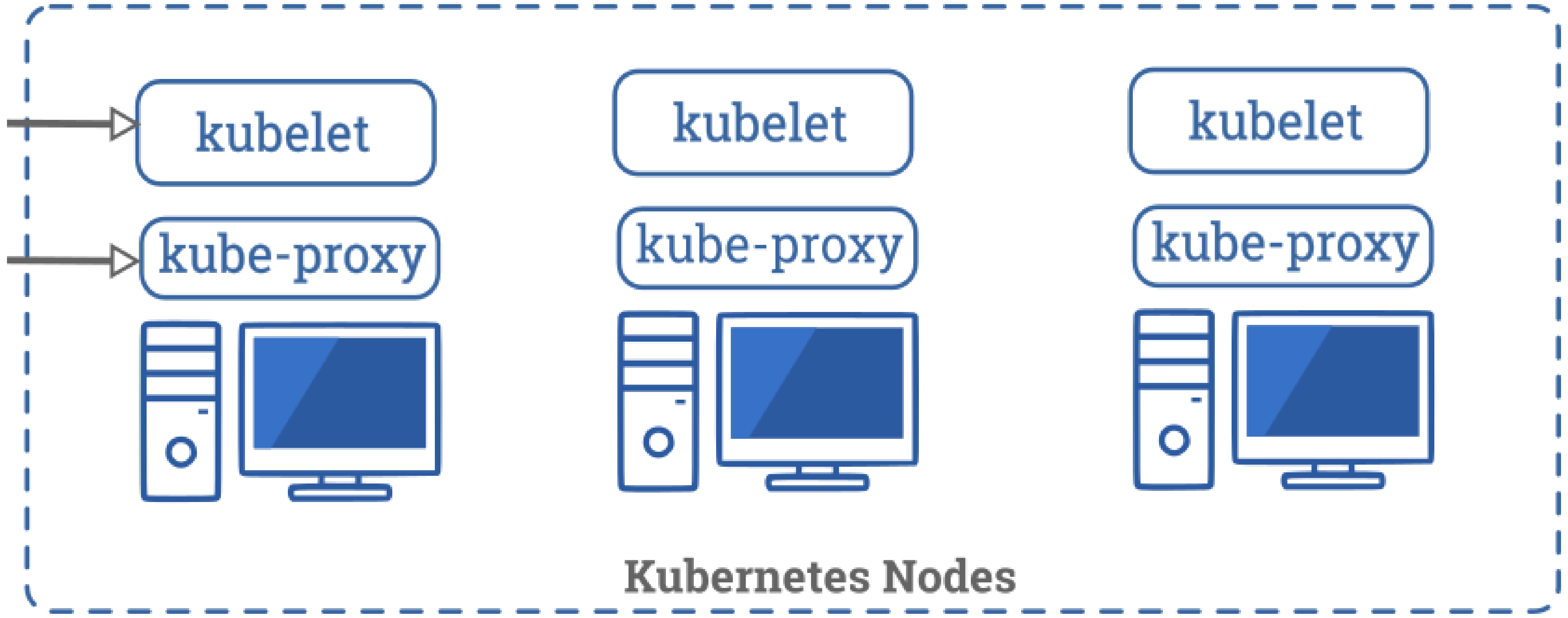
KUBERNETES ARCHITECTURE



- Etcd
  - The etcd project, developed by the team at CoreOS, is a lightweight, distributed key-value store that can be configured to span across multiple nodes.
  - Kubernetes uses etcd to store configuration data that can be accessed by each of the nodes in the cluster.
- Kube-apiserver
  - This is the main management point of the entire cluster as it allows a user to configure Kubernetes' workloads and organizational units
  - The API server implements a RESTful interface
- Kube-controller-manager
  - It manages different controllers that regulate the state of the cluster, manage workload life cycles, and perform routine tasks.
  - When a change is seen, the controller reads the new information and implements the procedure that fulfills the desired state.
- Kube-scheduler
  - The process that actually assigns workloads to specific nodes
  - The scheduler is responsible for tracking available capacity on each host to make sure that workloads are not scheduled in excess of the available resources.

KUBERNETES ARCHITECTURE

- Container Runtime
  - Typically Docker
  - Rkt and runC supported
- Kubelet
  - The kubelet service communicates with the master components to authenticate to the cluster and receive commands and work
  - The kubelet process then assumes responsibility for maintaining the state of the work on the node server.
- Kube-Proxy
  - To manage individual host subnetting and make services available to other components



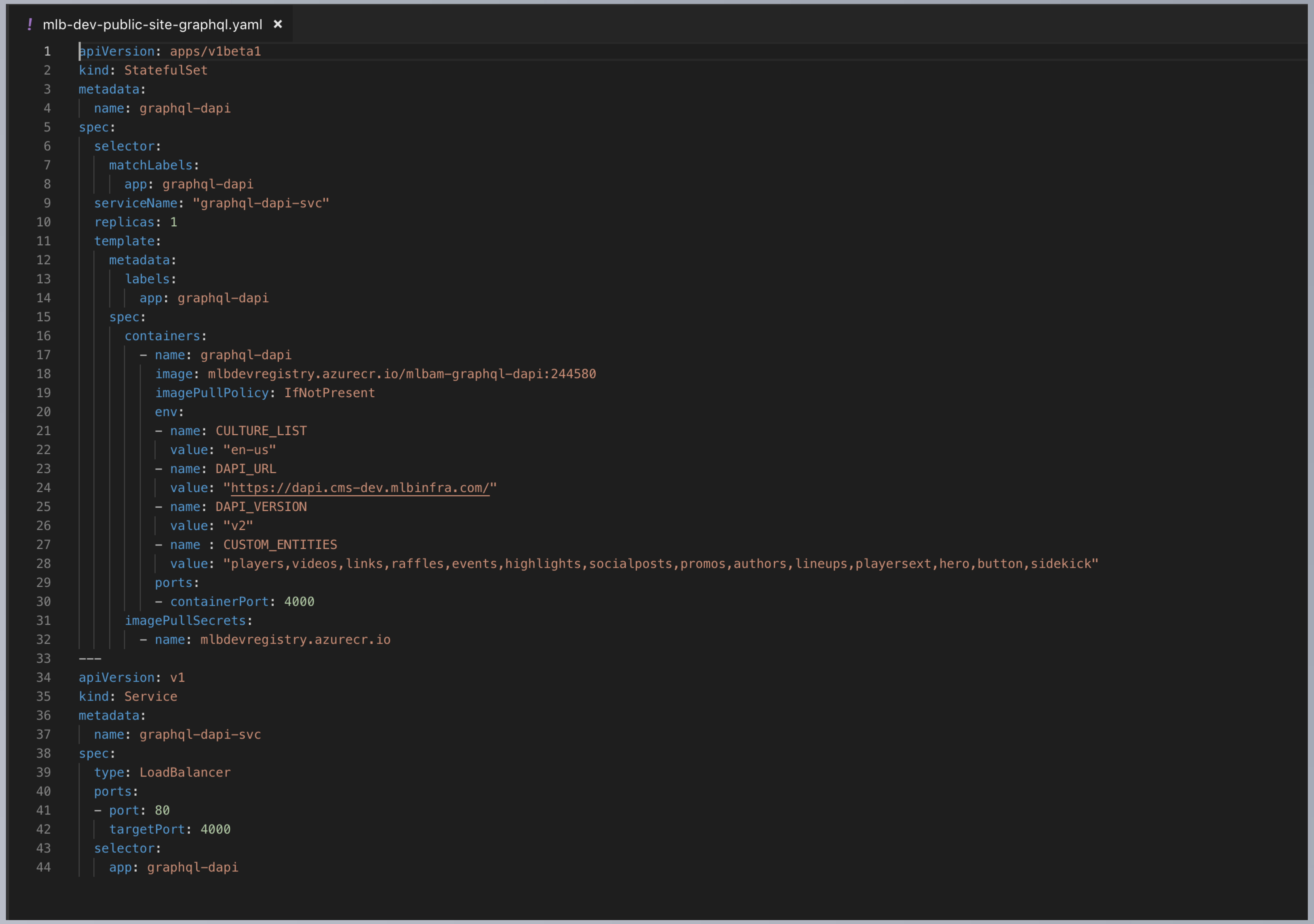
KUBERNETES – THE BASIC CONCEPTS

- Cluster - A collection of hosts that aggregate their available resources including cpu, ram, disk, and their devices into a usable pool.
- Master - The master(s) represent a collection of components that make up the control plane of Kubernetes. These components are responsible for all cluster decisions including both scheduling and responding to cluster events.
- Node - A single host, physical or virtual capable of running pods. A node is managed by the master(s), and at a minimum runs both kubelet and kube-proxy to be considered part of the cluster.
- Namespace - A logical cluster or environment. Primary method of dividing a cluster or scoping access.

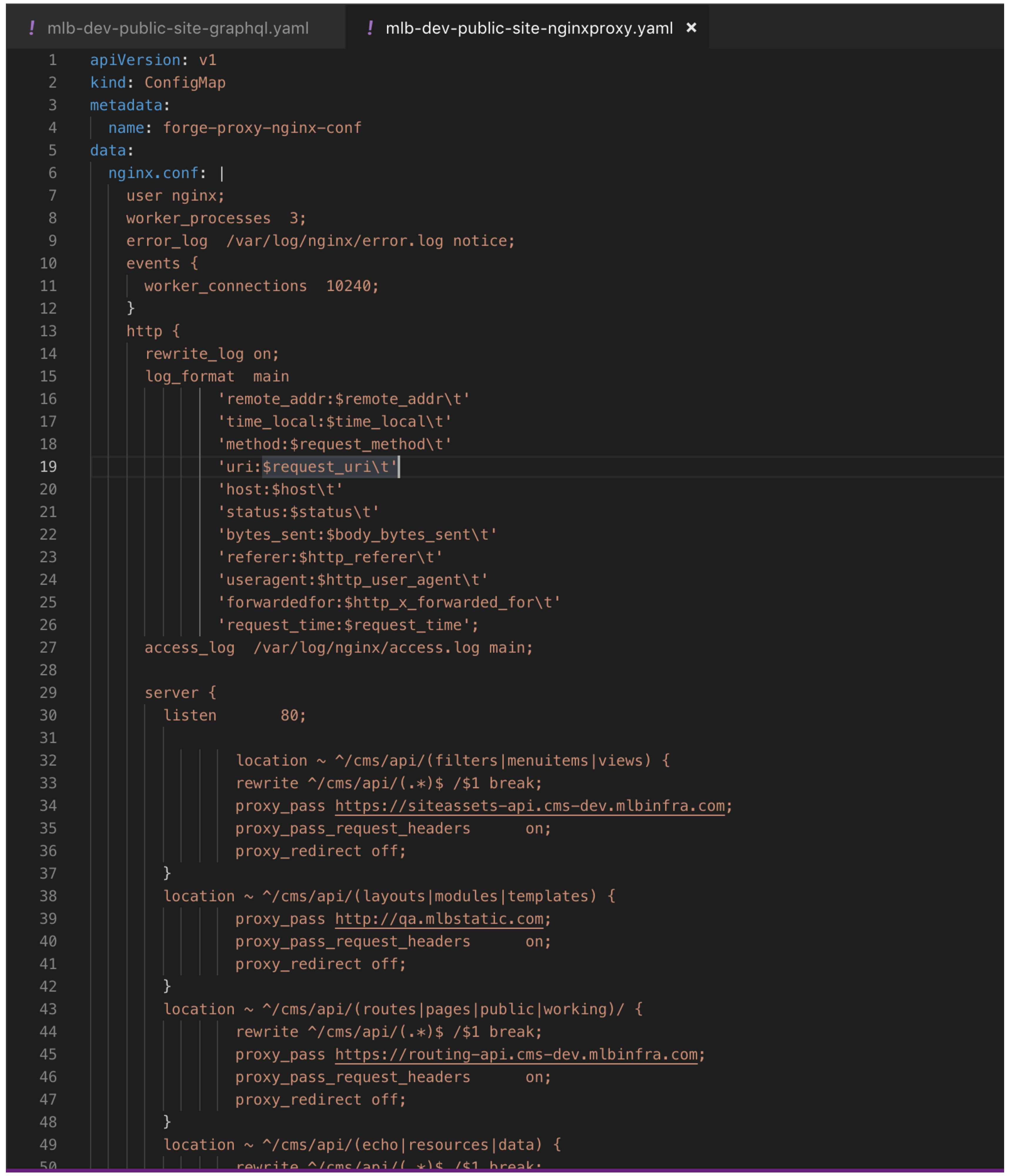
KUBERNETES – THE BASIC CONCEPTS

- Pod - A pod is the smallest unit of work or management resource within Kubernetes. It is comprised of one or more containers that share their storage, network, and context (namespace, cgroups etc).
- Deployment - A declarative method of managing stateless Pods and ReplicaSets. Provides rollback functionality in addition to more granular update control mechanisms.
- Service - Services provide a method of exposing and consuming L4 Pod network accessible resources. They use label selectors to map groups of pods and ports to a cluster-unique virtual IP.
- Volume - Storage that is tied to the Pod Lifecycle, consumable by one or more containers within the pod.
- ConfigMap - Externalized data stored within kubernetes that can be referenced as a commandline argument, environment variable, or injected as a file into a volume mount. Ideal for separating containerized application from configuration.
- Secret - Functionally identical to ConfigMaps, but stored encoded as base64, and encrypted at rest (if configured).

KUBERNETES – YAML FILES



```yaml
apiVersion: apps/v1beta1
kind: StatefulSet
metadata:
  name: graphql-dapi
spec:
  selector:
    matchLabels:
      app: graphql-dapi
  serviceName: "graphql-dapi-svc"
  replicas: 1
  template:
    metadata:
      labels:
        app: graphql-dapi
    spec:
      containers:
        - name: graphql-dapi
          image: mlbdevregistry.azurecr.io/mlbam-graphql-dapi:244580
          imagePullPolicy: IfNotPresent
          env:
          - name: CULTURE_LIST
            value: "en-us"
          - name: DAPI_URL
            value: "https://dapi.cms-dev.mlbinfra.com/"
          - name: DAPI_VERSION
            value: "v2"
          - name : CUSTOM_ENTITIES
            value: "players,videos,links,raffles,events,highlights,socialposts,promos,authors,lineups,playersext,hero,button,sidekick"
          ports:
          - containerPort: 4000
      imagePullSecrets:
        - name: mlbdevregistry.azurecr.io
---
apiVersion: v1
kind: Service
metadata:
  name: graphql-dapi-svc
spec:
  type: LoadBalancer
  ports:
  - port: 80
    targetPort: 4000
  selector:
    app: graphql-dapi
```

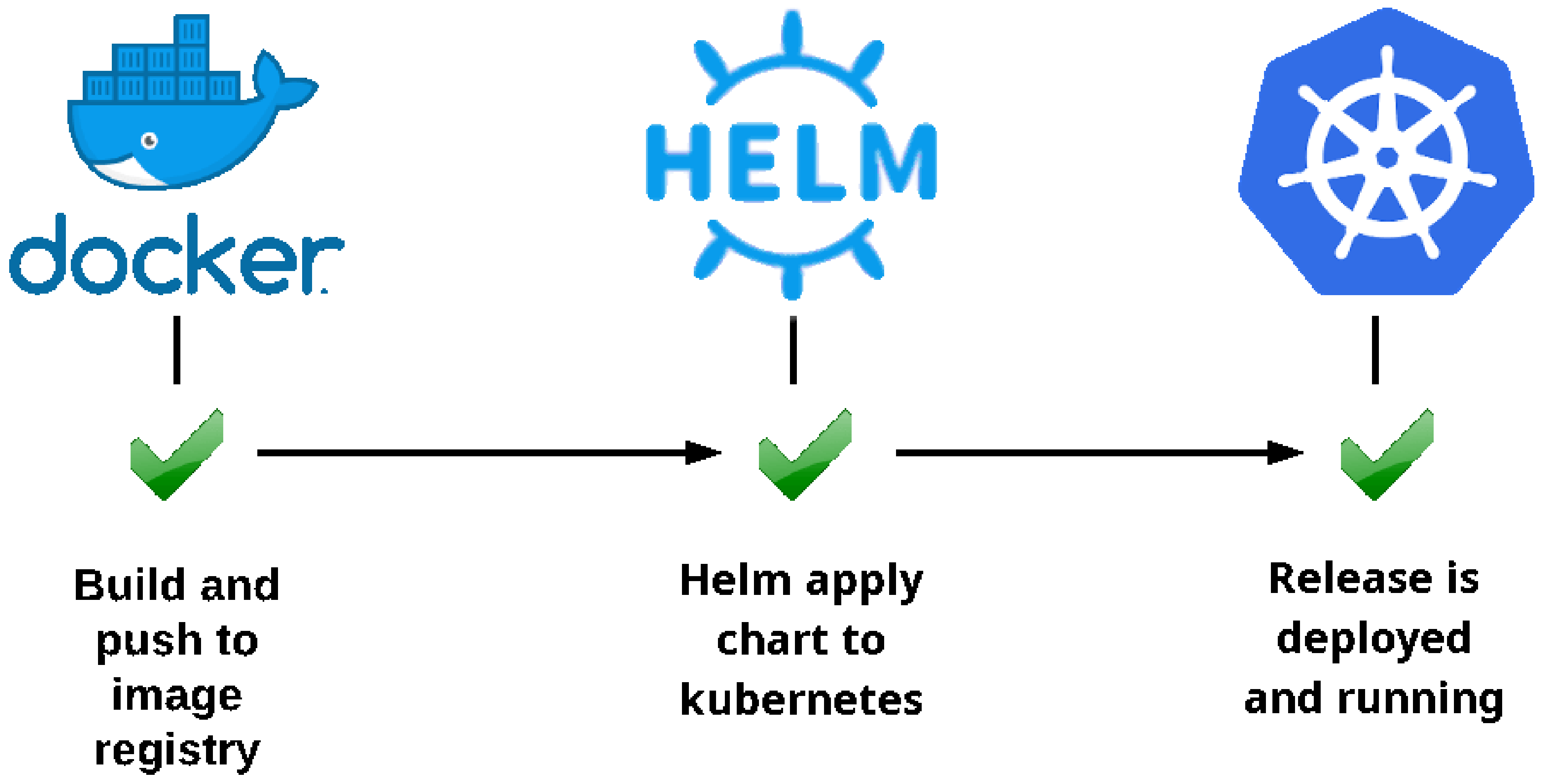

! mlb-dev-public-site-graphql.yaml
! mlb-dev-public-site-nginxproxy.yaml ×

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: forge-proxy-nginx-conf
data:
  nginx.conf: |
    user nginx;
    worker_processes  3;
    error_log  /var/log/nginx/error.log notice;
    events {
      worker_connections  10240;
    }
    http {
      rewrite_log on;
      log_format  main
              'remote_addr:$remote_addr\t'
              'time_local:$time_local\t'
              'method:$request_method\t'
              'uri:$request_uri\t'
              'host:$host\t'
              'status:$status\t'
              'bytes_sent:$body_bytes_sent\t'
              'referer:$http_referer\t'
              'useragent:$http_user_agent\t'
              'forwardedfor:$http_x_forwarded_for\t'
              'request_time:$request_time';
      access_log  /var/log/nginx/access.log main;

      server {
        listen       80;

                location ~ ^/cms/api/(filters|menuitems|views) {
                rewrite ^/cms/api/(.*)$ /$1 break;
                proxy_pass https://siteassets-api.cms-dev.mlbinfra.com;
                proxy_pass_request_headers      on;
                proxy_redirect off;
        }
        location ~ ^/cms/api/(layouts|modules|templates) {
                proxy_pass http://qa.mlbstatic.com;
                proxy_pass_request_headers      on;
                proxy_redirect off;
        }
        location ~ ^/cms/api/(routes|pages|public|working)/ {
                rewrite ^/cms/api/(.*)$ /$1 break;
                proxy_pass https://routing-api.cms-dev.mlbinfra.com;
                proxy_pass_request_headers      on;
                proxy_redirect off;
        }
        location ~ ^/cms/api/(echo|resources|data) {
                rewrite ^/cms/api/(.*)$ /$1 break;
```

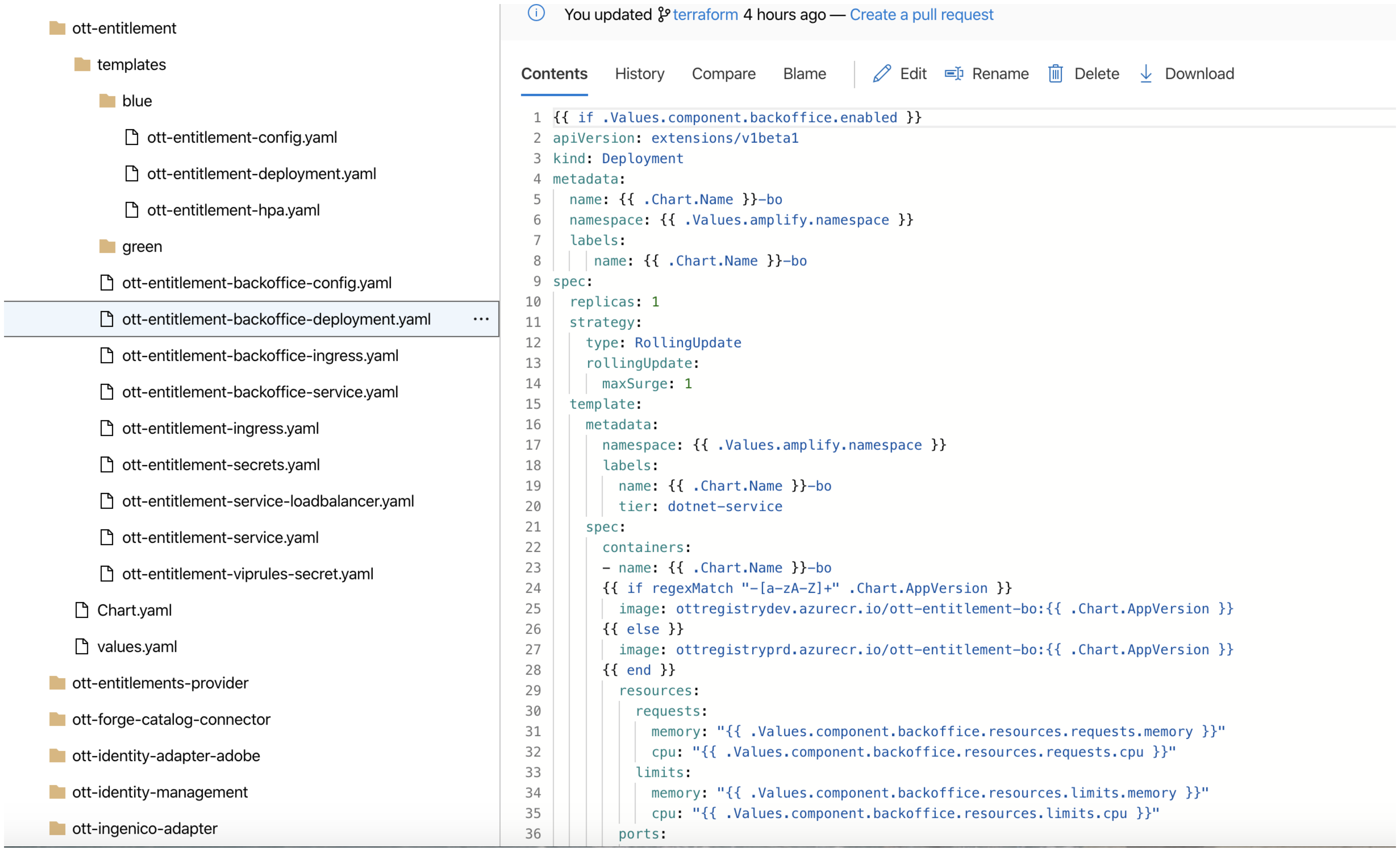
and more

HELM PACKAGE MANAGER

- Part of the Cloud Native Computing Foundation
- Designed to simply management of dependencies on Kubernetes deployments
- CHARTS: Helm packages, a few YAML configurations files
- Mostly standard Kubernetes YAML format
- Templates and Values yaml files used to abstract composition of Kubernetes YAML files with variables (e.g. by environment)
- Requirementes.yaml used to define dependencies


docker
HELM
Build and push to image registry
Helm apply chart to kubernetes
Release is deployed and running

HELM CHART EXAMPLE



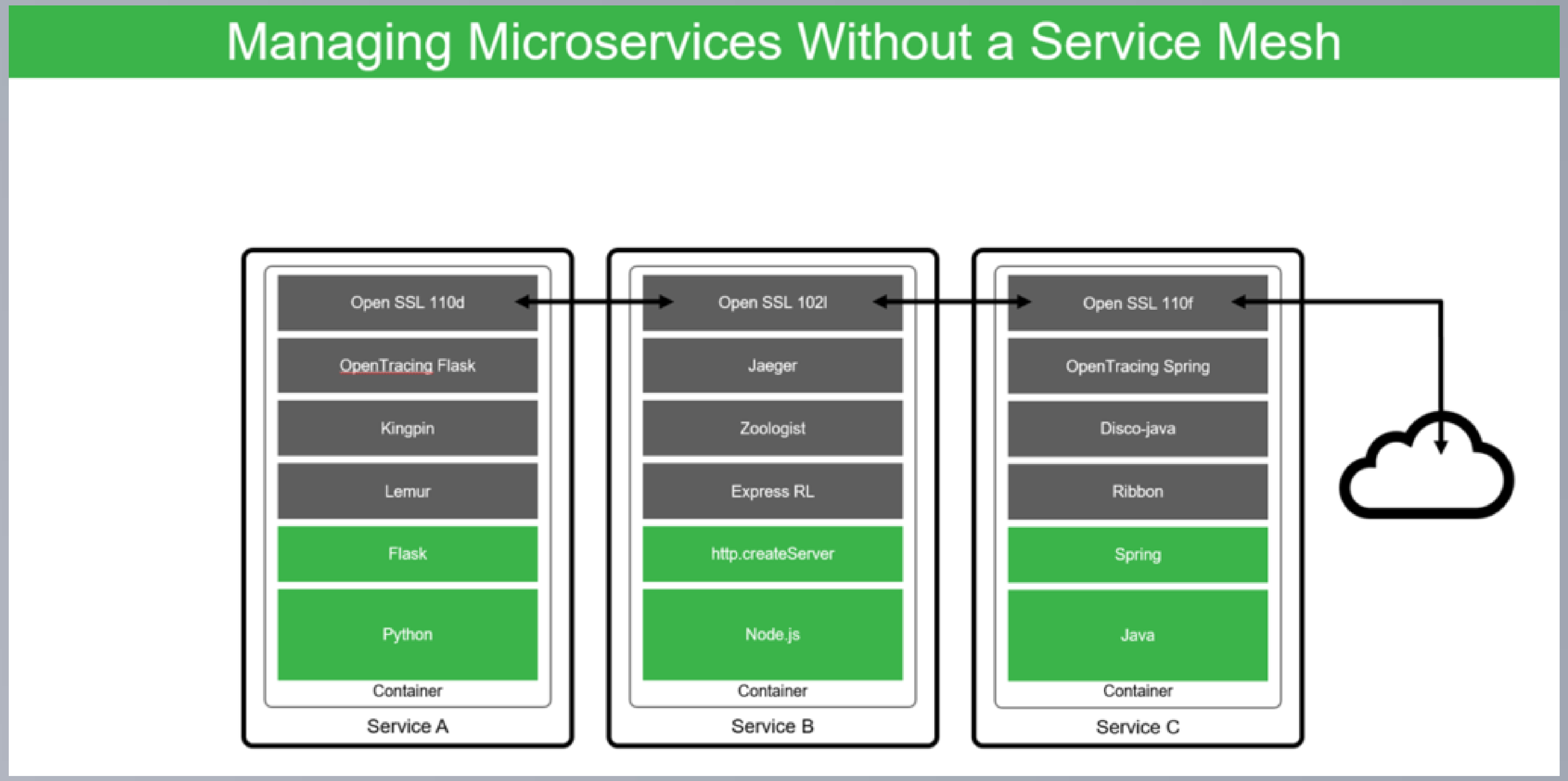
- ott-entitlement
  - templates
    - blue
      - ott-entitlement-config.yaml
      - ott-entitlement-deployment.yaml
      - ott-entitlement-hpa.yaml
    - green
    - ott-entitlement-backoffice-config.yaml
    - ott-entitlement-backoffice-deployment.yaml
    - ott-entitlement-backoffice-ingress.yaml
    - ott-entitlement-backoffice-service.yaml
    - ott-entitlement-ingress.yaml
    - ott-entitlement-secrets.yaml
    - ott-entitlement-service-loadbalancer.yaml
    - ott-entitlement-service.yaml
    - ott-entitlement-viprules-secret.yaml
  - Chart.yaml
  - values.yaml
- ott-entitlements-provider
- ott-forge-catalog-connector
- ott-identity-adapter-adobe
- ott-identity-management
- ott-ingenico-adapter

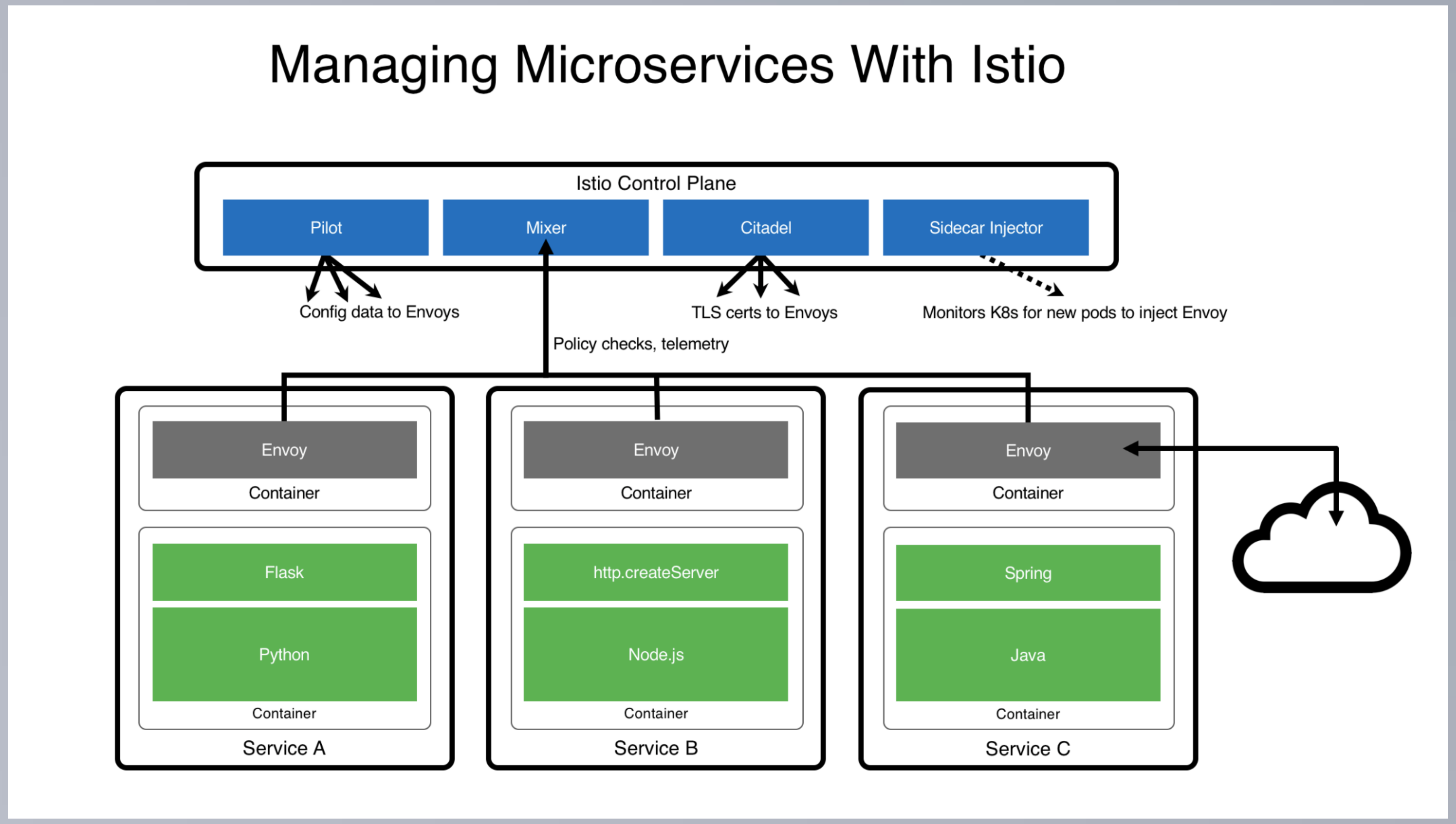
You updated terraform 4 hours ago — Create a pull request

```yaml
{{ if .Values.component.backoffice.enabled }}
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: {{ .Chart.Name }}-bo
  namespace: {{ .Values.amplify.namespace }}
  labels:
    name: {{ .Chart.Name }}-bo
spec:
  replicas: 1
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
  template:
    metadata:
      namespace: {{ .Values.amplify.namespace }}
      labels:
        name: {{ .Chart.Name }}-bo
        tier: dotnet-service
    spec:
      containers:
      - name: {{ .Chart.Name }}-bo
      {{ if regexMatch "-[a-zA-Z]+" .Chart.AppVersion }}
        image: ottregistrydev.azurecr.io/ott-entitlement-bo:{{ .Chart.AppVersion }}
      {{ else }}
        image: ottregistryprd.azurecr.io/ott-entitlement-bo:{{ .Chart.AppVersion }}
      {{ end }}
        resources:
          requests:
            memory: "{{ .Values.component.backoffice.resources.requests.memory }}"
            cpu: "{{ .Values.component.backoffice.resources.requests.cpu }}"
          limits:
            memory: "{{ .Values.component.backoffice.resources.limits.memory }}"
            cpu: "{{ .Values.component.backoffice.resources.limits.cpu }}"
        ports:
```

ISTIO – SERVICE MESH

- Traffic Management
  - Decouples traffic flow and infrastructure scaling, letting you specify via Pilot what rules you want traffic to follow rather than which specific pods/VMs
- Security
  - Strong identity, powerful policy, transparent TLS encryption, and authentication, authorization and audit (AAA) tools
- Policy and Telemetry
  - A flexible model to enforce authorization policies and collect telemetry for the services in a mesh
- Performance and Scalability
  - Support for Horizontal Pod Autoscaling





Oh hey! You're not looking happy. What's up?
Pfft... if only you knew...
After several years of religiously documenting everything the team developped, I decided to face the truth
I installed Google Analytics on the wiki to see which pages the guys really read
Sounds like a good idea! What happened?
I waited for 3 days, but there was no visitor data. I thought the tracker was installed wrong, I spent hours looking for the bug...
But on day 4 I finally realised....
It wasn't a problem with the tracker...!
BAAAHHHHHHHAHH HHAHAHAH

CommitStrip.com

LINKS

Processes, Containers, Virtual Machines - https://medium.com/@jessgreb01/what-is-the-difference-between-a-process-a-container-and-a-vm-f36ba0f8a8f7

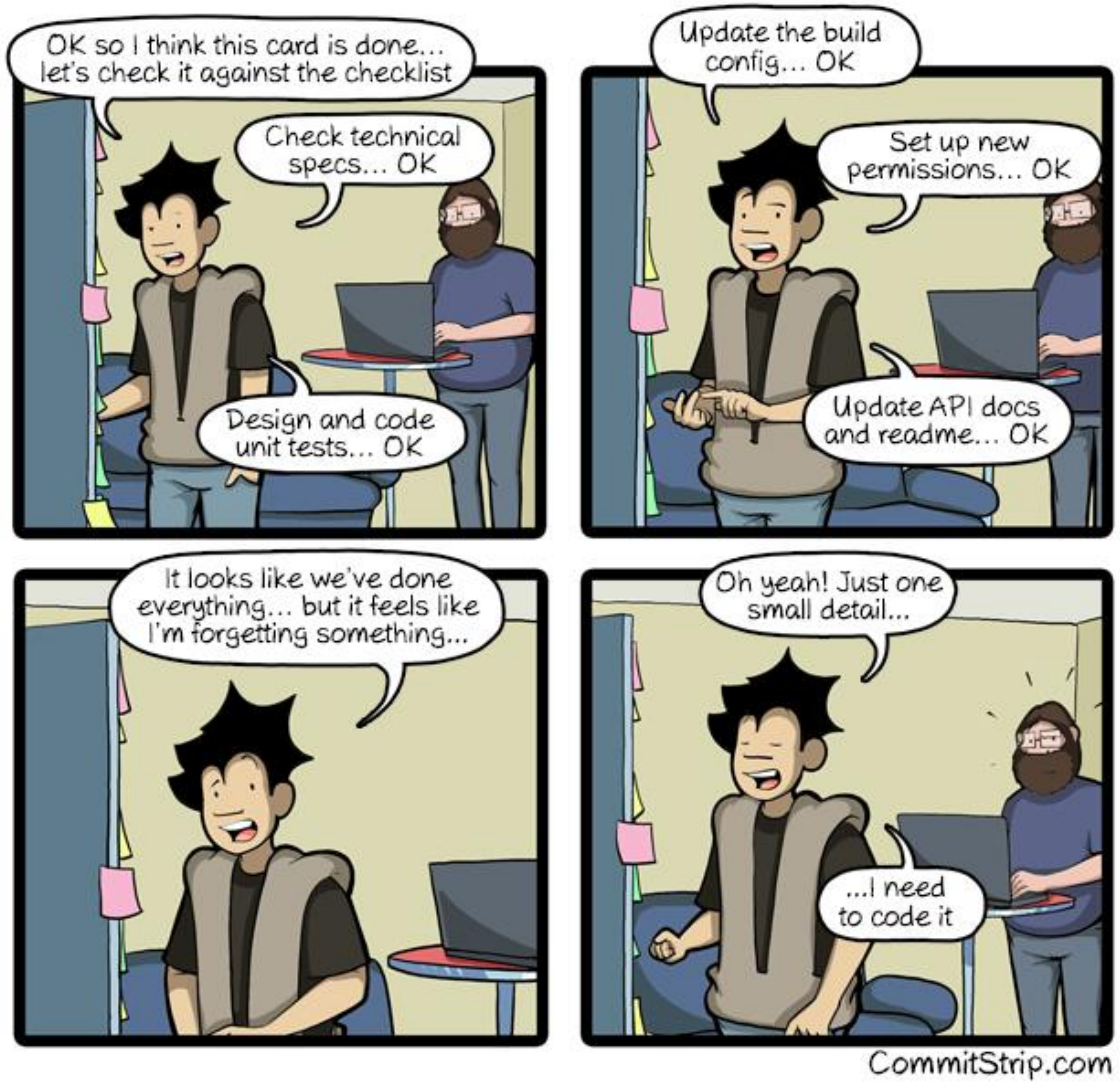
Introduction to Kubernetes for Vmware users - https://blogs.vmware.com/cloudnative/2017/10/25/kubernetes-introduction-vmware-users/

Introduction to Kubernetes Architecture - https://phoenixnap.com/kb/understanding-kubernetes-architecture-diagrams

Docker and Windows - https://techcommunity.microsoft.com/t5/windows-dev-appconsult/first-steps-with-docker-introduction/ba-p/317547

Kubernetes and Windows - https://techcommunity.microsoft.com/t5/windows-dev-appconsult/first-steps-with-docker-and-kubernetes-introduction/ba-p/357525

THE END – Q&A ?


OK so I think this card is done... let's check it against the checklist
Check technical specs... OK
Design and code unit tests... OK
Update the build config... OK
Set up new permissions... OK
Update API docs and readme... OK
It looks like we've done everything... but it feels like I'm forgetting something...
Oh yeah! Just one small detail...
...I need to code it
CommitStrip.com