MIDI TECH

Kubernetes, pour orchestrer nos containers

François Delbrayelle (@fdelbrayelle)

BIENVENUE !

- Midis techniques S03E04
- Vous aussi vous pouvez en faire ;-) !
- Présentation puis démo
- **Sondage debout : qui utilise les containers, Docker, Kubernetes… ?**

Les containers ?

LE MONDE RÉEL

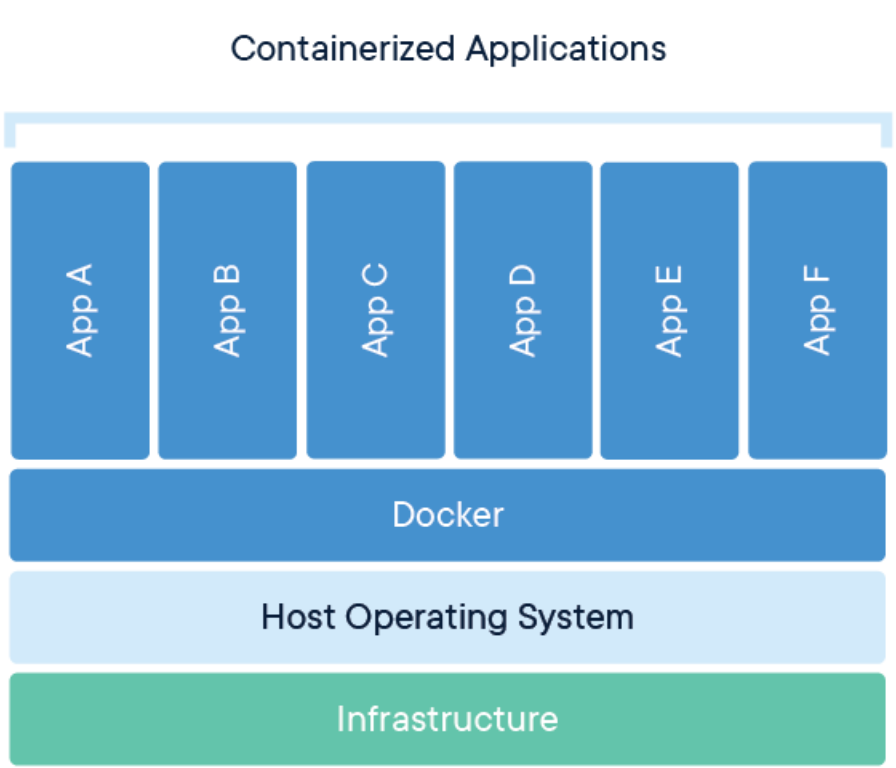
HISTORIQUE (1/2)

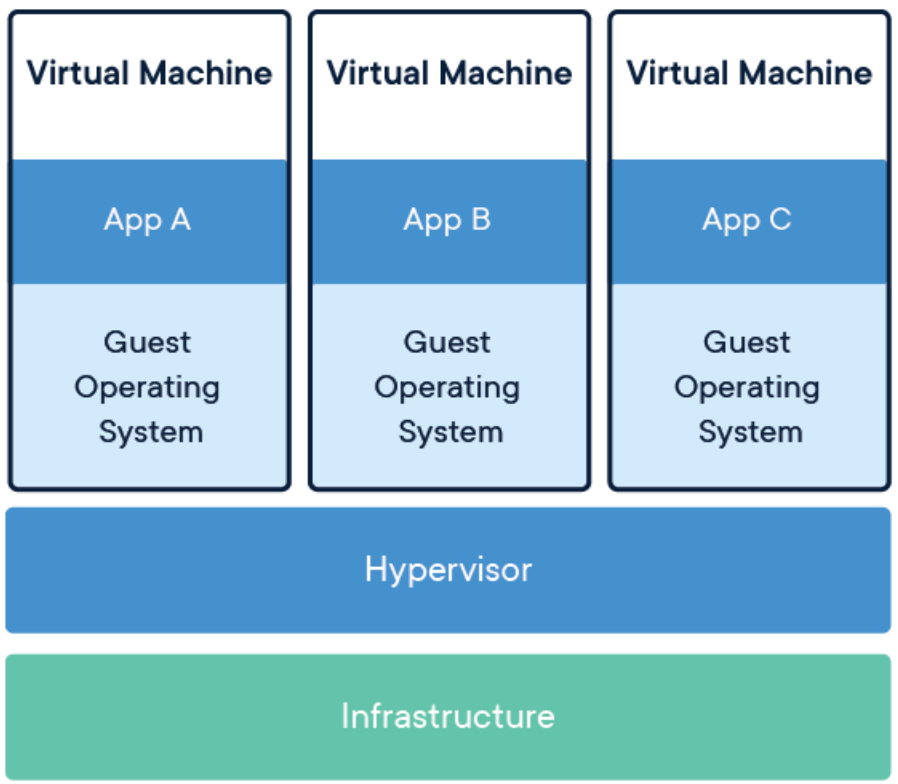
- 1979 : `chroot` (Unix V7) pour isoler le répertoire racine d'un processus et ses fils
- 1999 : les **jails** de FreeBSD avec des mini-systèmes dans le système (même kernel)
- 2001 : Linux VServer
- 2004 : Solaris Containers
- 2005 : Open VZ

HISTORIQUE (2/2)

- 2006 : Process Containers (Google) -> Control Groups (cgroups) mergé dans le kernel Linux 2.6.24
- 2008 : **LXC (LinuX Containers)** avec les cgroups (ressources) et namespaces Linux isolés (même kernel) - voir `man 7 cgroups` et `man 7 namespaces`
- 2011 : RedHat OpenShift
- **2013 : Docker (basé d'abord sur LXC puis format maison : libcontainer)**

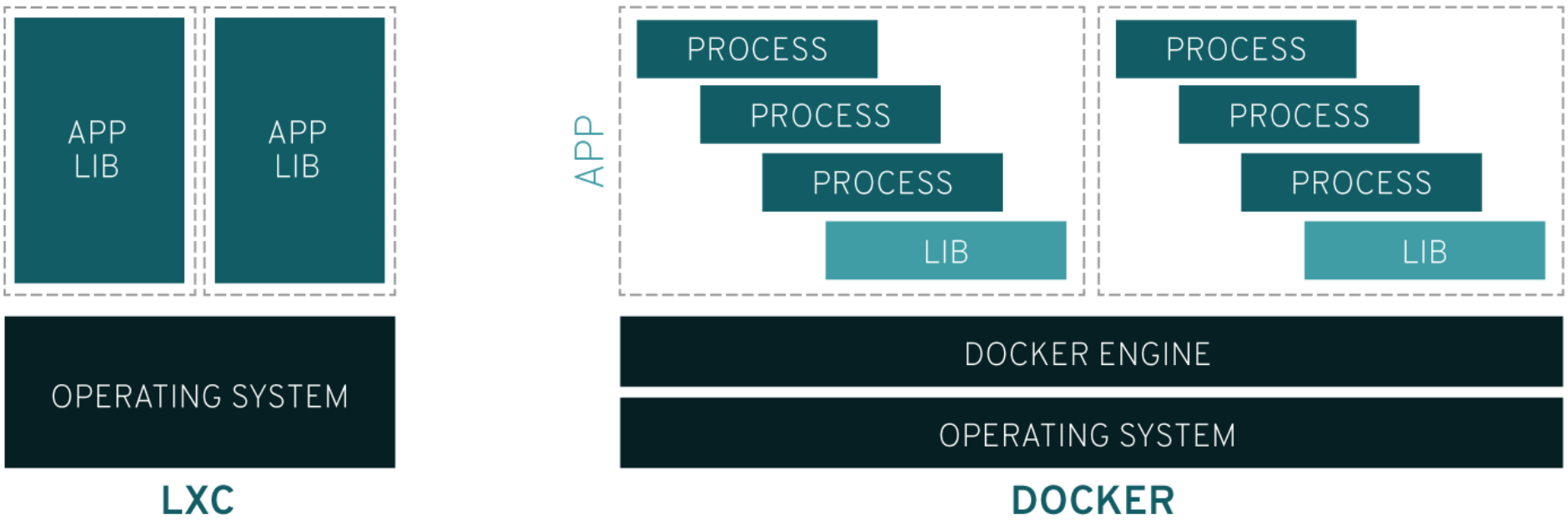
CONTAINERS VS VM





LXC VS DOCKER

Traditional Linux containers vs. Docker



CI/CD

- Légers, immutables et portables
- Provisionner/décommissionner rapidement
- Limite les erreurs (drivers, conflits...)
- Réduit les coûts entre Dev et Ops
- Élimine les restrictions (frameworks, outils...)
- Automatisation complète


docker

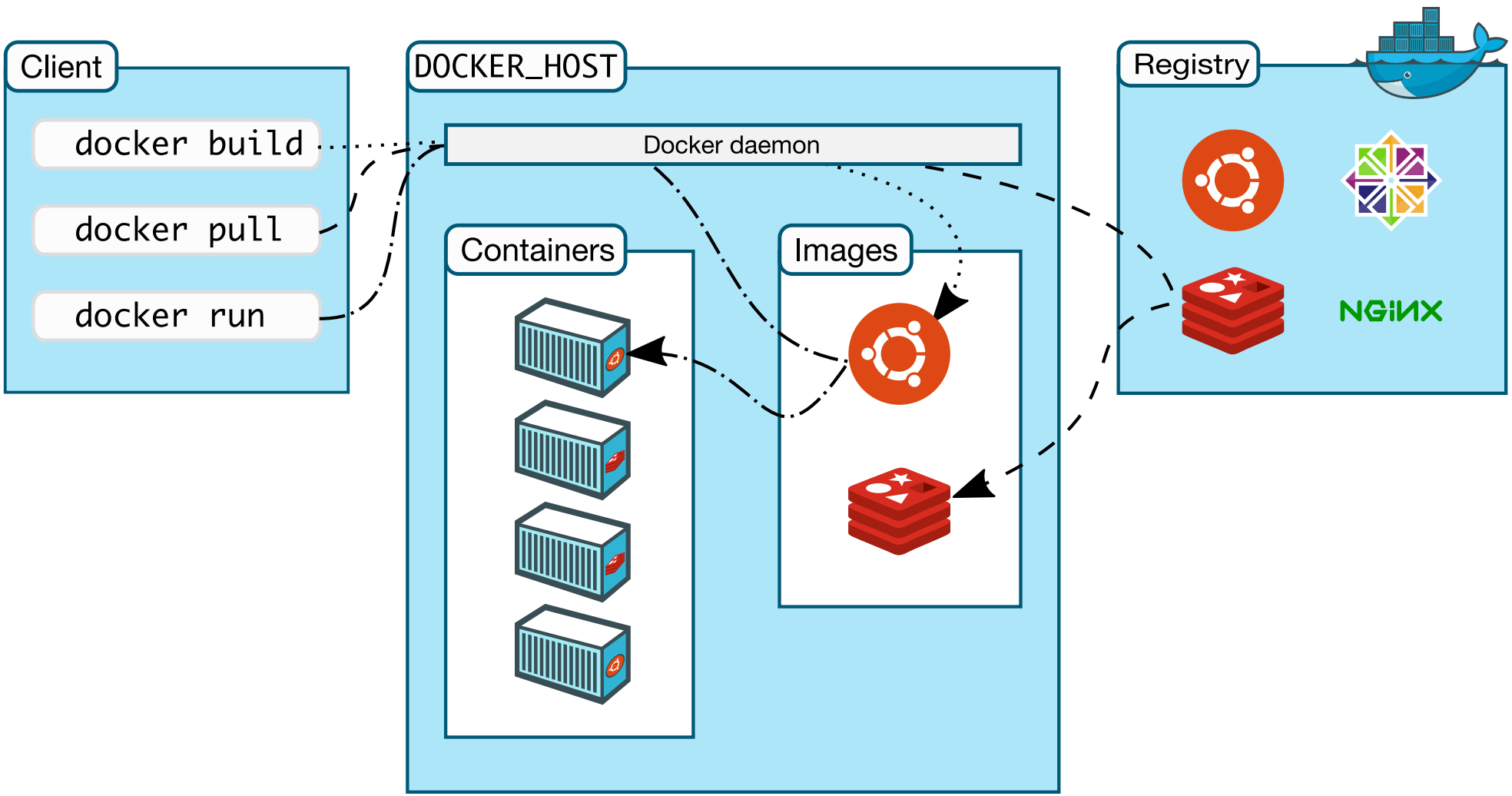
C'EST QUOI ?

- SaaS/PaaS de virtualisation niveau OS
- Crée par Solomon Hykes
- Licence Apache 2.0 (OSS) depuis 2013
- Développé en Go, v18.09.7 (27/06/2019)
- Isolation et légereté
- Peut contenir de tout (Java, JavaScript, TypeScript, base de données, ...)

C'EST QUOI ?

- Client `docker` (CLI)
- API REST entre le client et le serveur
- **Docker Engine : dockerd** pour le runtime
+ containerd pour les cgroups/namespaces...
- dockerd gère des objets (images, containers, ...)
- Objets stockés dans `/var/lib/docker`

ARCHITECTURE



IMAGES

- Template en lecture seule avec instructions
- Build once, run everywhere
- Stratégie copy-on-write
- Notion d'image de base
- Stockée localement
- `docker image COMMAND`

DOCKER HUB

- Registry central d'images
- Possibilité de download/upload (tags)
- https://hub.docker.com
- Registries locaux possibles
- **Attention aux images non officielles**

DOCKERFILE

- Fichier texte utilisé par `docker build -t IMAGE PATH|URL`
- Contexte important (`PATH` : chemin relatif, `URL` : emplacement sur un dépôt git)
- Liste de commandes pour fabriquer une image : `INSTRUCTION arguments`
- `RUN`, `COPY` et `ADD` = nouvelle couche
- Autres instructions = images intermédiaires

DOCKERFILE

- `FROM image[:tag] [AS name]` **(obligatoire)** pour indiquer l'image de base (première instruction)
- `COPY` pour copier des fichiers
- `ADD` pour copier des fichiers (URL autorisées, archives également = dézippées)

DOCKERFILE

- `RUN` pour exécuter une commande (commit dans une nouvelle image temporaire)
- `CMD` ou `ENTRYPOINT` **(obligatoire)** pour définir une commande à exécuter au lancement du container

DOCKERFILE

```
FROM alpine
COPY /home/francois/dev/java/projet /app
RUN cd /app && mvn clean package
CMD java -jar /app/target/projet.jar
```

- Récupère une image de base "alpine" (5 Mo)
- Copie un projet (hôte) dans /app (container)
- Lance une compilation maven
- Définit la commande d'entrée du container

BONNES PRATIQUES

- Comprendre le **contexte**
- Utiliser un **.dockerignore**
- Utiliser le build **multi-stage**
- Minimiser le **nombre de couches**

BONNES PRATIQUES

- Exploiter le **cache**
- Construire une image avec stdin
- Meilleur support **à partir de Java 10** : https://blog.docker.com/2018/04/improved-docker-container-integration-with-java-10/

CONTAINERS

- Instance d'une image (runtime)
- Analogie avec classe/objet en POO
- Partage le noyau de la machine hôte
- Lister les containers lancés : `docker ps`
- Lancer un container en interactif : `docker container run -it IMAGE`

DOCKER COMPOSE

- Fichier YAML pour définir, lancer et scaler des **services**
- Comportement des containers en prod
- Gestion des **ressources** (CPU, RAM)
- Pull d'autres images (BDD par exemple)
- Ex : `docker-compose -f sonar.yml up`
- **Utile pour les microservices**

MULTI-STAGE

- Build en plusieurs étapes
- Plusieurs instructions `FROM`
- Le dernier `FROM` définit la dernière image et donc celle qui portera le tag
- Réduit la taille de l'image finale
- **Utile pour compiler le code dans une image intermédiaire par exemple**

#DEVSECOPS

- Images de base minimales (alpine)
- Utilisateur le moins privilégié (`USER`)
- Vérifier/signer les images (/!\ MIDM)
- Trouver/corriger les vulnérabilités (Snyk)
- Pas d'infos sensibles dans le Dockerfile

#DEVSECOPS

- Utiliser des tags fixes et élevés (pas latest)
- Utiliser `COPY` plutôt que `ADD` (risques : URL distantes, archives corrompues)
- Ajouter des métadonnées (`LABEL`)
- Utiliser le multi-stage
- Utiliser un linter (hadolint)

ET SUR WINDOWS ?

- Docker Desktop for Windows ou Docker Toolbox avec PowerShell ou cmd ou Cmder
- Compatibilité avec Hyper-V
- Mêmes commandes que sur Linux
- **Windows Subsystem for Linux 2 (WSL) (Windows 10, preview juillet 2019)**

DOCKER SWARM

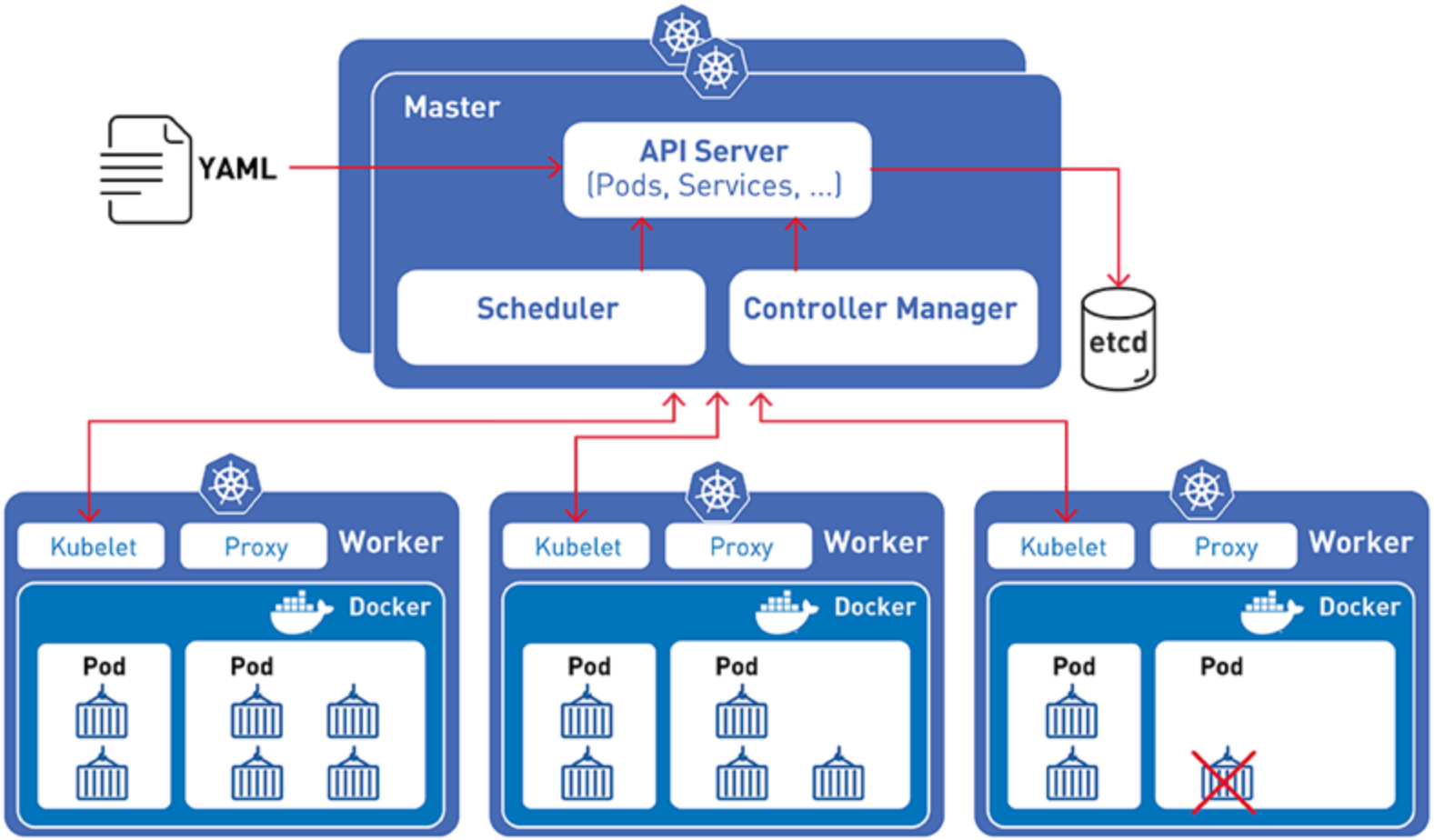
- Outil standalone de clustering pour orchestrer les containers dans un **swarm**
- Scalabilité : plusieurs machines physiques/virtuelles (et donc 1 dockerd / machine = **node**)
- Nodes **managers** et **workers**
- Load balancing : `docker swarm init` et `docker stack deploy...`
- Se base sur les fichiers **Docker Compose**

kubernetes

C'EST QUOI ?

- Système d'orchestration de containers
- Crée par Google
- Licence Apache 2.0 (OSS) depuis 2014
- Développé en Go, v1.15 (19/06/2019)
- κυβερνήτης en grec = capitaine
- k8s

ARCHITECTURE



INSTALLATION

- dockerd doit tourner
- **kubeadm** : pour créer un cluster
- **kubelet** : pour démarrer pods et containers
- **kubectl** : pour parler au cluster
- https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/#installing-kubeadm-kubelet-and-kubectl

KUBECTL

- Configuration en lignes de commandes
- Gérer les objets d'un cluster
- Déployer des applications, voir les logs...
- `kubectl [cmd] [TYPE] [NAME] [flags]`
- https://kubernetes.io/docs/tasks/tools/install-kubectl/

CLUSTERS

- Ensemble de nodes
- Gestion avec `kubectl`
- Manipulation d'objets

MASTER

- Noeud particulier, maître
- Contrôle les workers (**controller manager + scheduler**)
- Contient l'**API server** (`kubectl`, web UI)
- **etcd** : data store persistent, distribué et léger pour la configuration clé/valeur du cluster

NODES

- Workers (minions) : exécutent les tâches
- Machine physique ou virtuelle
- **Container runtime** : lance les containers
- **kubelets** : s'assure que les containers tournent dans un pod
- **kube-proxy** : implémente une IP virtuelle pour les services

PODS

- Objet REST top-level
- 1 ou n containers (Docker, rktlet...)
- Réseau et stockage partagé
- Relancé avec nouvelle IP à la destruction
- `kubectl get|delete|describe... pod(s) NAME [ARGS]`

SERVICES

- Objet REST
- Abstraction pour accès à 1 ou n pod(s)
- Différent d'un service de Docker Compose

CONFIGURATION YAML

- Définition d'un objet k8s
- Ajout de complexité possible
- Historique via contrôle de sources
- Validation sur https://kubeyaml.com/
- `kubectl apply|create|delete|replace -f obj.yaml`

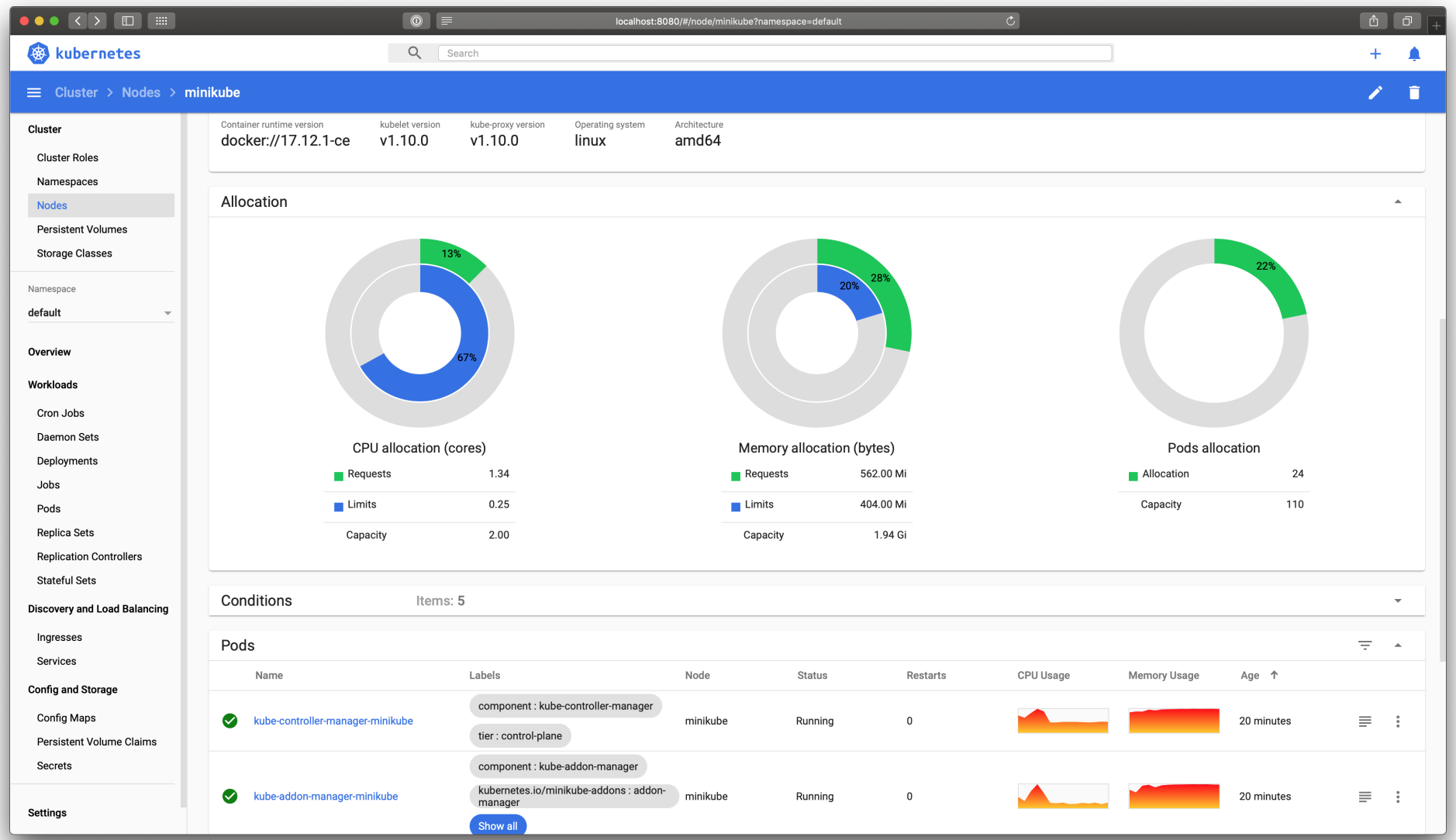
MINIKUBE

- Créer un cluster en local
- Commande `minikube [cmd]`
- Load balancing, dashboard, ...
- Alternatives : Kind, MicroK8s...
- **Nécessite un hyperviseur (KVM, VirtualBox)**

CLOUD

- Google Kubernetes Engine (GKE) avec la commande `gcloud` : https://cloud.google.com/kubernetes-engine
- Amazon Elastic container service for KuberneteS (EKS) : https://aws.amazon.com/fr/eks
- Azure Kubernetes Service (AKS) : https://docs.microsoft.com/en-us/azure/aks

WEB UI (DASHBOARD)



HELM

- Gestionnaire de packages
- Maintenu par la CNCF
- Basé sur le `Chart.yaml`
- https://helm.sh/docs/

Démo !

https://github.com/fdelbrayelle/midi-tech-k8s/demo

EN COMPLÉMENT

- Service mesh (Istio)
- Jenkins X mieux adapté
- Écosystème complexe et vaste

ET LE TURFU ?

- Serverless
- LinuxKit

LIENS UTILES

- Docker : https://docs.docker.com/get-started
- K8s : https://kubernetes.io/docs/concepts
- DevFest Lille 2019 : https://www.youtube.com/user/francegdg
- https://dzone.com/articles/docker-layers-explained
- https://dzone.com/refcardz/cicd-with-containers

SUR WINDOWS

- Docker Windows Containers : https://www.docker.com/products/windows-containers
- Docker for Windows : https://docs.docker.com/docker-for-windows/
- Docker for WSL 2 : https://engineering.docker.com/2019/06/docker-hearts-wsl-2/

Merci !

https://github.com/fdelbrayelle/midi-tech-k8s