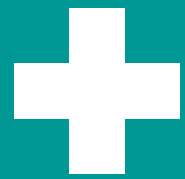
femto-st SCIENCES & TECHNOLOGIES

# Introduction to Kubernetes

*Frédéric Lassabe*



cnrs UBFC UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ UNIVERSITÉ DE FRANCHE-COMTÉ SUP MICRO TECH ENSMM utbm université de technologie Belfort-Montbéliard

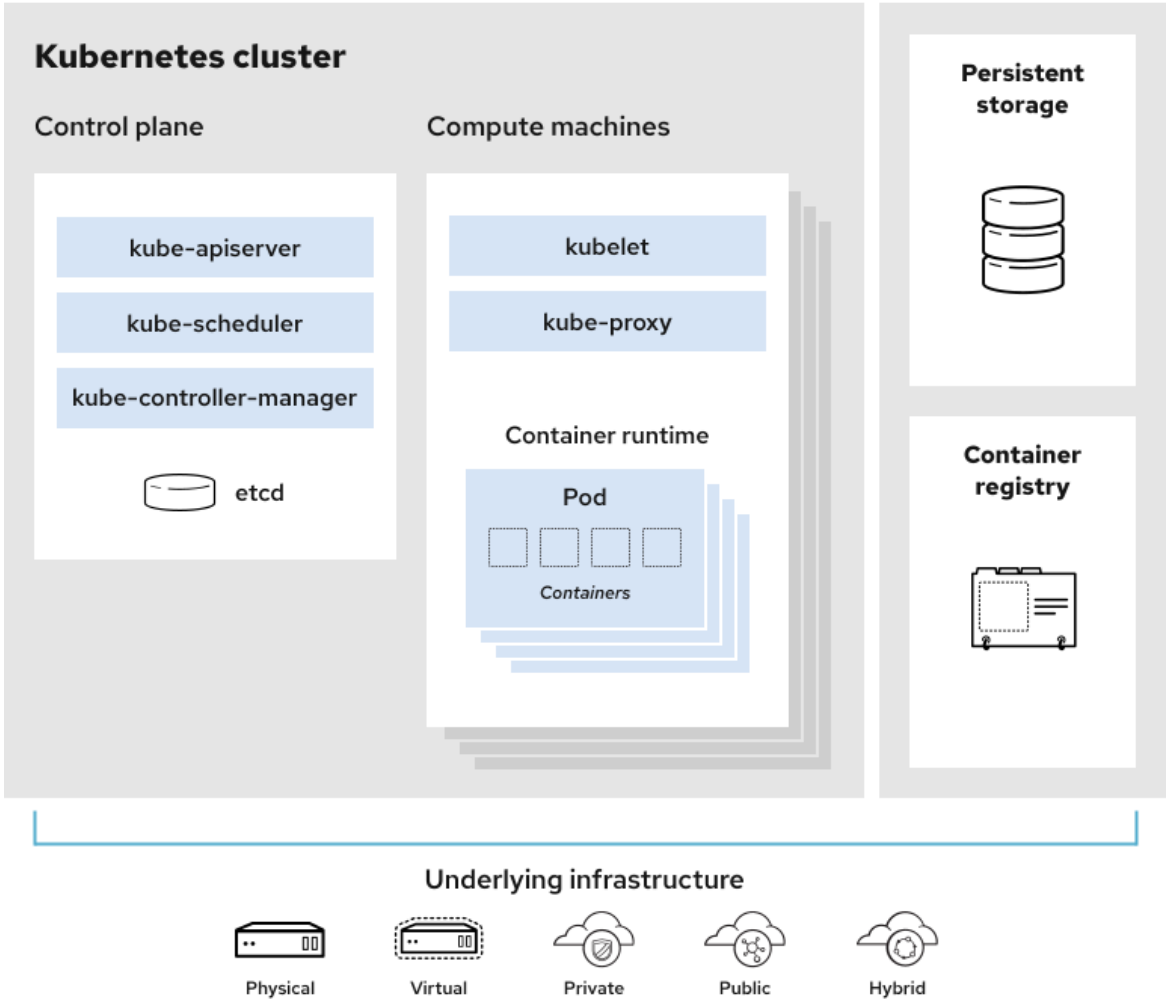
# What is kubernetes ?

- Container orchestration system
- Main features
  - Service discovery and load balancing
  - Storage orchestration
  - Deployments automation
  - Automated resource allocation
  - Self-repair
  - Configuration and secret management
  - Batch processing
  - Horizontal scaling
  - IP/IPv6 support
- a.k.a. K8s
- Open source

# Cluster

- A k8s cluster is
  - a working deployment
  - a group of hosts
    - physical
    - virtual
  - running linux containers
  - Two types of hosts :
    - Control plan
    - Nodes (workers)

# Architecture



# Workers

- also named Nodes
- have their own Linux environment
- run pods
- pods made of containers

# Control plane

- maintaining the cluster state
  - applications running
  - container images used
  - resources allocated
- takes commands from users (usually admins)
- orchestrates tasks on nodes

# Cloud-native development

- Requirement : build and maintain
- Fast deployment
- Microservices
  - Scale better
  - Easier to modify
  - Not everything can scale

# Production

- Production
  - Applications on multiple containers
  - Across multiple hosts
- Kubernetes
  - Orchestration
  - Management
  - To deploy
  - At scale

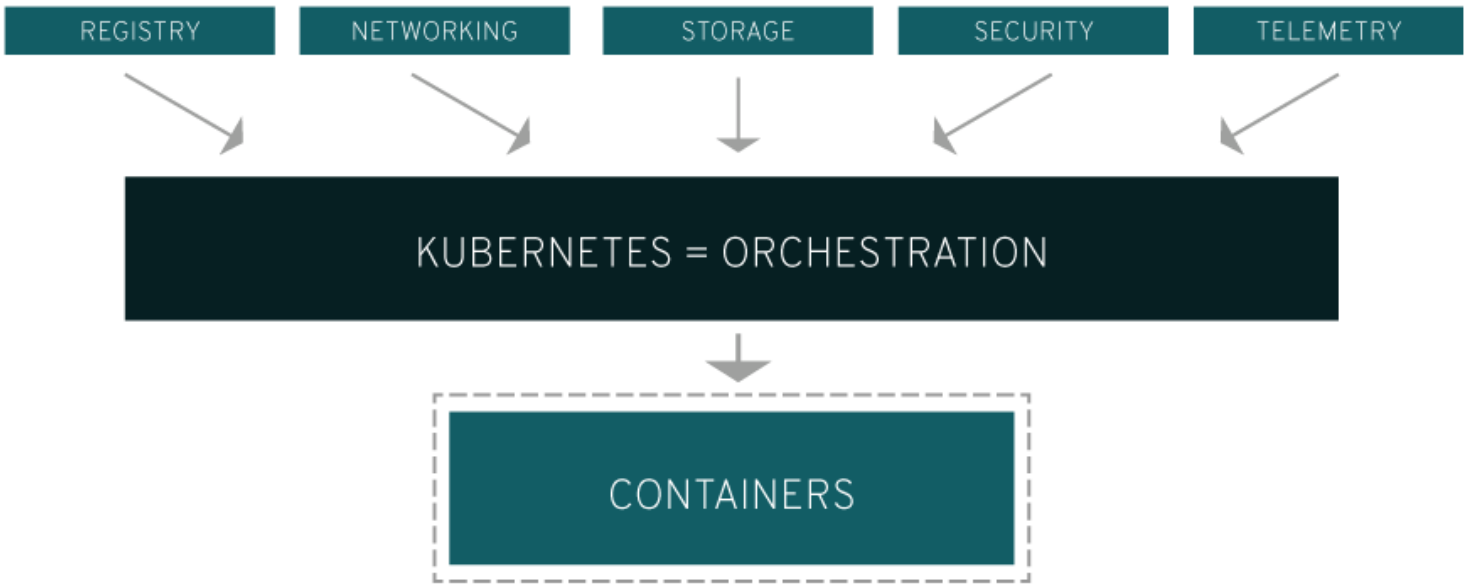
# With kubernetes

- Build application services
- Spanning multiple containers
- Scale containers
- Monitor containers' health
- Integrates
  - Network
  - Storage
  - Security
  - Telemetry
  - And other features

# The need for k8s

- Scaling application
  - Multiple containers
  - Colocated and cooperating
- Containers
  - Efficient and self-contained applications modules
- Microservices
  - Efficient for app development, deployment, maintenance
  - Significant increase of numbers of containers
  - Increased complexity
- k8s attempts to solve complexity

# Kubernetes features



# Role

- Cluster control
- Data about cluster state
- Check applications have sufficient
  - Containers
  - Resources
- Monitors nodes constantly
- You set an application
- The control plane guarantees it is executed

# kube-apiserver

- API to interact with the cluster
- Control plane's front end
- Both for requests
  - internal
  - external
- Checks request validity
- Then executes it
- Accessible from
  - REST API
  - kubectl
  - kubeadm

# kube-scheduler

- In charge of
  - Checking containers health
  - Finding resources for deploying new containers
- Uses various metrics
  - Resources requirements for a pod
  - Health of the cluster
- Schedule pods to appropriate node

# kube-controller-manager

- Controllers run the cluster
- Controller manager contains several controller features :
  - Check running pods against scheduler
    - Can restart failed containers
  - Route requests to endpoints
  - Create accounts
  - Manage API access tokens
  - etc.

# etcd

- As etc manages your linux host configuration
- etcd is your configuration repository
  - Key-value database
  - Fault tolerant
  - Distributed

# What are nodes ?

- Compute resource
- Hosts services via containers inside pods
- Pods run on nodes
- Scheduled from the control plane
- Scaling up :
  - Add more nodes
  - Join them to the cluster

# Pod

- Smallest unit in kubernetes object model
- One pod $=$ one instance of application
- One pod
  - One container
  - Or tightly coupled containers
  - Options for running the containers
- Pods may have persistent storage

# Container runtime engine

- k8s manages containers
- Requires a container runtime
  - docker (with cri-dockerd)
  - CRI-O
  - Containerd
- You shall choose a runtime for any k8s instance
- Used runtime shall support the Open Container Interface standard

# kubelet

- Each node has a kubelet
- Application to communicate with the control plane
- Ensures that containers run in pods
- Instructions from control plane executed by kubelet

# kube-proxy

- Required in all nodes
- Network proxy
- Supports Kubernetes networking services
- Handles network communications
  - Internal as well as external
  - relying either on
    - OS' packet filtering layer
    - Directly forwarded by kube-proxy

# Persistent storage

- Containers run applications
- May need persistent data
- Stateful applications
- Hardware storage mapping on nodes would tie a pod to a node
- Use of network storage
- Transparent for the pod and its container
- Persistent network storage outlives pods

# Container registry

- A storage for container images used by kubernetes
- Images for your applications
- Registry owned by the cluster's administrators
- Or a third party registry

# Underlying infrastructure

- Cloud infrastructures are not untangible : real hardware
- Kubernetes deploys either on :
  - Bare metal
  - Virtual machines
  - Public cloud providers
  - Hybrid cloud environments
- one or more hosts for *control plan*, with all its services
  - Redundancy in case of failure
  - Desirable : at least two control plan nodes
- one or more different hosts for workers
  - No colocation of control plan and worker
  - Workers count depends on :
    - available resources
    - Target load

Kubernetes overview

Kubernetes deployment

- Control plane
- Nodes

Using the cluster

# Constraints

- Deployments based on
  - 3 hosts
    - 1 control plane
    - 2 nodes
  - Ubuntu hosts
  - In VM
- Based on various documents, mainly
  - kubernetes website
  - devopscube.com
  - Red Hat website

# Steps

- Install a container runtime
- Install kubeadm, kubelet, kubectl
- Initialize
  - Control plane configuration
  - Generate the join command for nodes
- Install a network plugin

# Steps

- Join the worker to the master node
- Validate all components (check everything is working)

# And then ?

- Install a metrics server
- Use the cluster

Kubernetes overview

Kubernetes deployment

Using the cluster

# Pods

- Pods are the service-level unit
- They contain one or more containers
- Everything in a pod
  - Tightly coupled
  - Described by manifests
  - Orchestrated by kubernetes
- What goes in a pod or another depends on
  - Persistence requirements
  - Application architecture

# Best way to understand k8s

- Practice
- Details for the steps are available
  - In online documentation
  - In the lab instructions
- So technical/know-how details aren't in this doc

# Thanks !