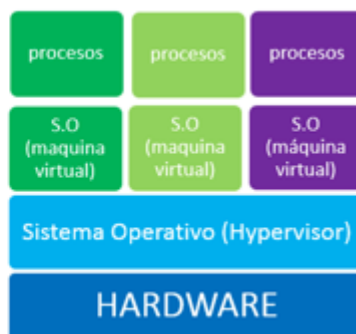


# contenedores

## Contenedores

ocultar/mostrar



Containers VS Virtualización

### Contenedores (I)

- virtualización ligera o de SO
- todos los sistemas comparten núcleo (kernel)
- usados desde hace mucho en Solaris (Containers y Zones) y FreeBSD (Jails)
  - hosting compartido
- Limitado en Linux: vservers, OpenVZ, chroot, ...
  - contenedor más sencillo existente: chroot

### Contenedores (II)

- Homogeneización del centro de datos
  - «El Software libre ganó»
  - SO que llenan los centros de datos son Linux
  - facilita el uso de un sistema de contenedores, evita el uso de virtualización completa
  - herramientas integradas en kernel que me garantizan seguridad, control recursos, etc..
- Linux cgroups (control groups)
  - acceso controlado a recursos de sistema
- Linux namespaces
- LXC

### LXC

- Integrado en el kernel de Linux
- se apoya en cgroups y namespaces
  - cgroups: grupos de control, gestión a recursos
  - namespaces : aislamiento
- los sistemas se instalan desde plantillas
- orientado al uso en sistemas
  - docker le da una vuelta de tuerca al uso de los contenedores

### Docker

- Muy similar a LXC «por abajo»
  - se aprovecha del trabajo de LXC y añade capas
- orientado al despliegue en microservicios

- mover aplicaciones
- proporciona versionado
- proporciona capas (aufs o overlayfs)
  - parto de cierta imagen y los cambios se almacenan como capas
- proporciona imágenes preconfiguradas
  - dockerhub
- muy... «Infraestructure as a Code»
- muy... «devops»
- ¿de verdad usas bien docker?
  - implica cambio de paradigma... orientado a microservicios
  - sin sentido para aplicaciones monolíticas

## rkt

- alternativa a docker
- desarrollado por CoreOS

## LXD

- hipervisor de contenedores
- muy relacionado con LXC
- ¿LXC 2.0?
- basado en el uso de imágenes
- bien integrado con Ubuntu
- Canonical

# Sistemas Operativos Ligeros

ocultar/mostrar

- más allá en el mundo de los contenedores:
  - instalo SO
  - todo lo que monto después está conteneizado
  - ¿me hace falta un sistema de gestión de paquetes? ¿necesito «instalar»?
  - monto un container, que se comporta como un Linux «normal», donde monto la aplicación que quiero
  - CoreOS: «rolling release» → siempre usando la versión actualizada

## Sistemas ligeros

- Container Linux (CoreOS)
- Project Atomic (RedHat)
- Ubuntu Core (Canonical)
  - standar de aplicaciones → inapi
- RancherOS
- Photon (VMWare)
- DCOS (Mesosphere) = Data Center Operating System
- para desplegar aplicaciones «contenerizadas»
- sin paquetes de aplicaciones
- aplicaciones sobre contenedores (snaps en Ubuntu)
- suelen utilizarse para clusters
- RancherOS: el propio sistema «contenerizado»

# Orquestación con contenedores

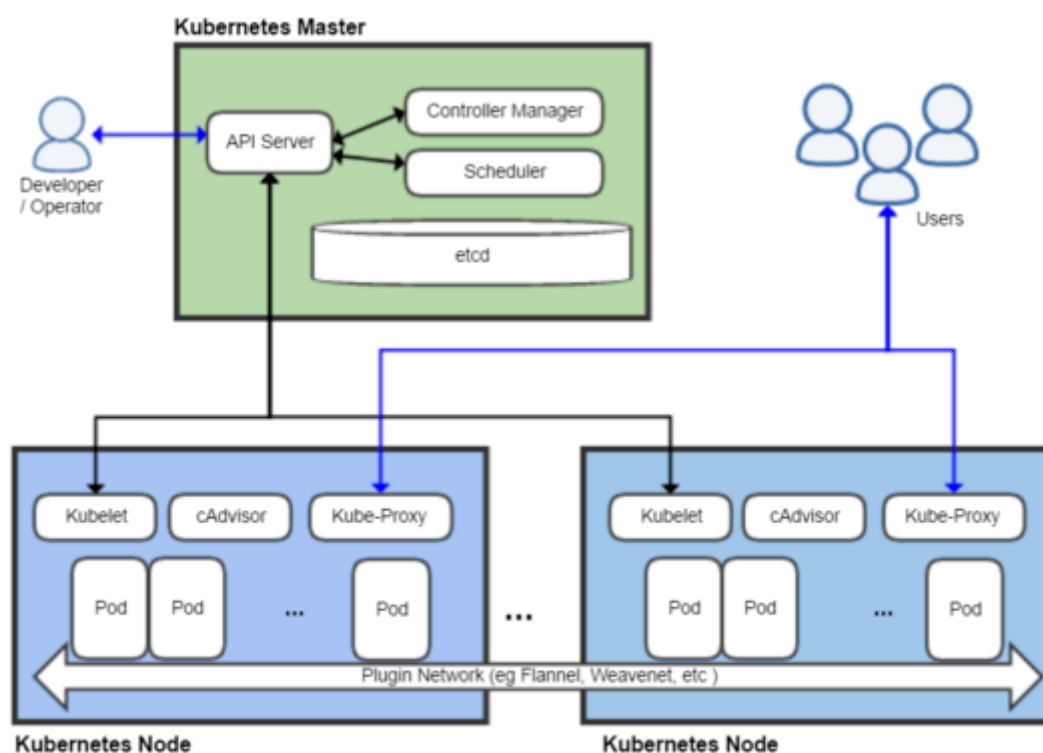
ocultar/mostrar

## orquestadores de contenedores

- gestionan clusters de equipos
- aplicaciones «contenerizadas»
- aportan funcionalidad adicional
  - alta disponibilidad
  - balanceo de carga
  - comunicación con el exterior
  - despliegues en producción

## kubernetes

- desarrollo inicial de Google
- liberado con licencia Apache
  - no es el modus operandi habitual
  - hace uso intensivo de herramientas de software libre
- Donado a la CNCF = Cloud Native Computing Foundation
  - fundación, luego Google se aparta
  - pretende estandarización y aportación por parte de otros
- Kubelet, pods
  - kubelet: agente?
  - pods: «contenedores»
  - intercomunicación entre los diferentes contenedores → compleja
  - kube-proxy : acceso a/desde el exterior
- Docker, rkt,...
- software «nuevo», métodos de instalación «mejorables» → no soportada en distros



## Docker swarm

- integrado en docker desde la v1.12
- aplicaciones definidas en docker compose
- se gestiona con comandos docker

## Mesos / Marathon

- soporte para docker y rkt
- otras cargas de trabajo: big data
- utilizado en producción por twitter o airbnb
- Mesosphere proporciona soporte comercial

# Contenedores IaaS

ocultar/mostrar

## Contenedores, clusters o "as a Service"

- despliegue desde plantillas: heat, cfn...
- ¿bare-metal?
  - servidores físicos
  - automatización arranque y configuración de servidor físico (aka VMs) → packet
- contenedores sobre MVs
  - elasticidad
- containers as a Service
- Facilidad para desplegar contenedores
- imágenes o SO preconfigurados
- registros propios para imágenes de contenedores

## Contenedores en AWS

- Amazon EC2 Container Service
- contenedores docker sobre instancias EC2
- cluster de instancias EC2
  - no implementa kubernetes

## Contenedores en Google

- Google Containers Engine
- Integración completa con kubernetes
- también docker, LXD o rkt directamente

## Contenedores Azure

- Azure Container Service
- Swarm, kubernetes o DCOS

## Contenedores en OpenStack

- contenedores sobre instancias
- despliegues de kubernetes, swarm, etc... con OpenStack heat
- OpenStack Magnum : Containers as a Service
  - integración entre MVs y Contenedores a nivel de red

# Contenedores PaaS

ocultar/mostrar

## Contenedores en PaaS

- contenedores como aplicación nativa cloud
- están sustituyendo a muchos PaaS
- nuevo enfoque: una capa más sobre orquestador
- aportan funcionalidad y facilidad de uso
- se centran en el desarrollo, no en la herramienta de orquestación

## ejemplo: OpenShift v3

- desarrollo de RedHat
  - v2 - en local o en nube (redhat en AWS)
- como servicio o en privado
- funciona sobre kubernetes
- añade funcionalidad
  - espacios de nombres
    - facilitar gestión de kubernetes por varios desarrolladores
  - RBAC
  - plantillas
  - integración con herramientas de desarrollo
    - control versiones, CI, pruebas automáticas...

From:

<https://matebcn.eu/> - miguel angel torres egea

Permanent link:

<https://matebcn.eu/info:cursos:openwebinars:intro-cloud-computing:contenedores>

Last update: **28/06/2018 11:40**

