Grid y Computación de Altas Prestaciones

GRyCAP

Sesión Práctica 2: Introducción a Computación con Contenedores

Computación de Altas Prestaciones - Master Universitario en Ingeniería Informática

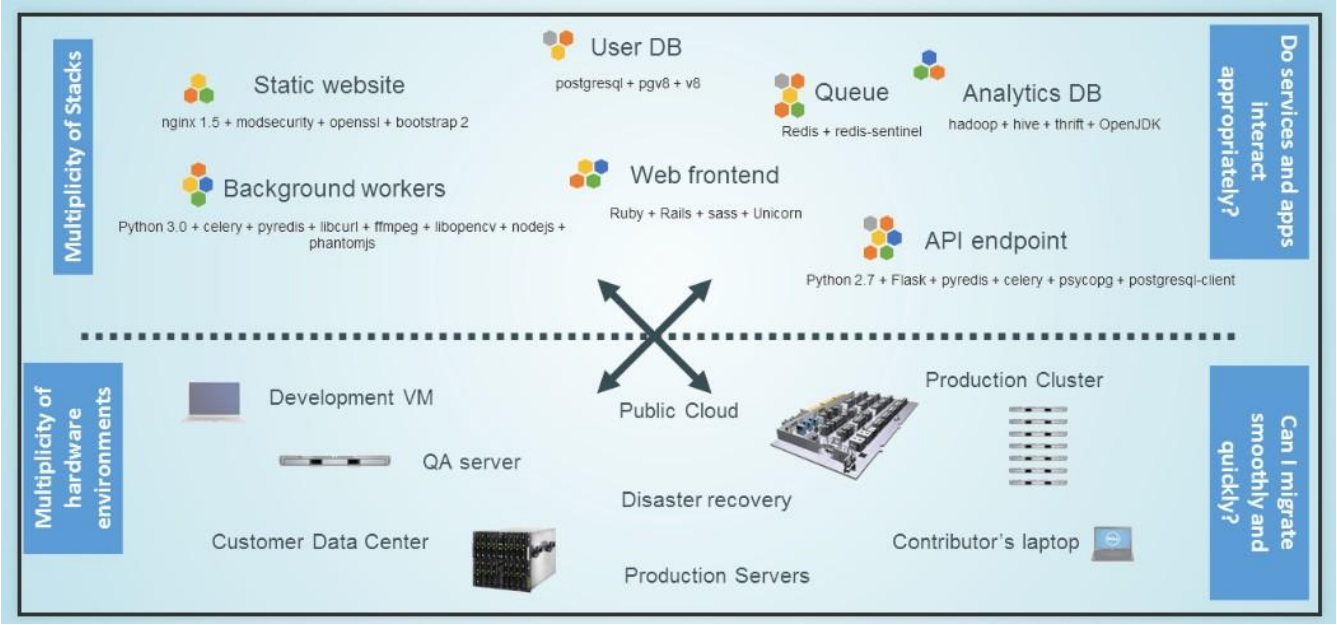
Resultados de Aprendizaje

Grid y Computación de Altas Prestaciones GRyCAP

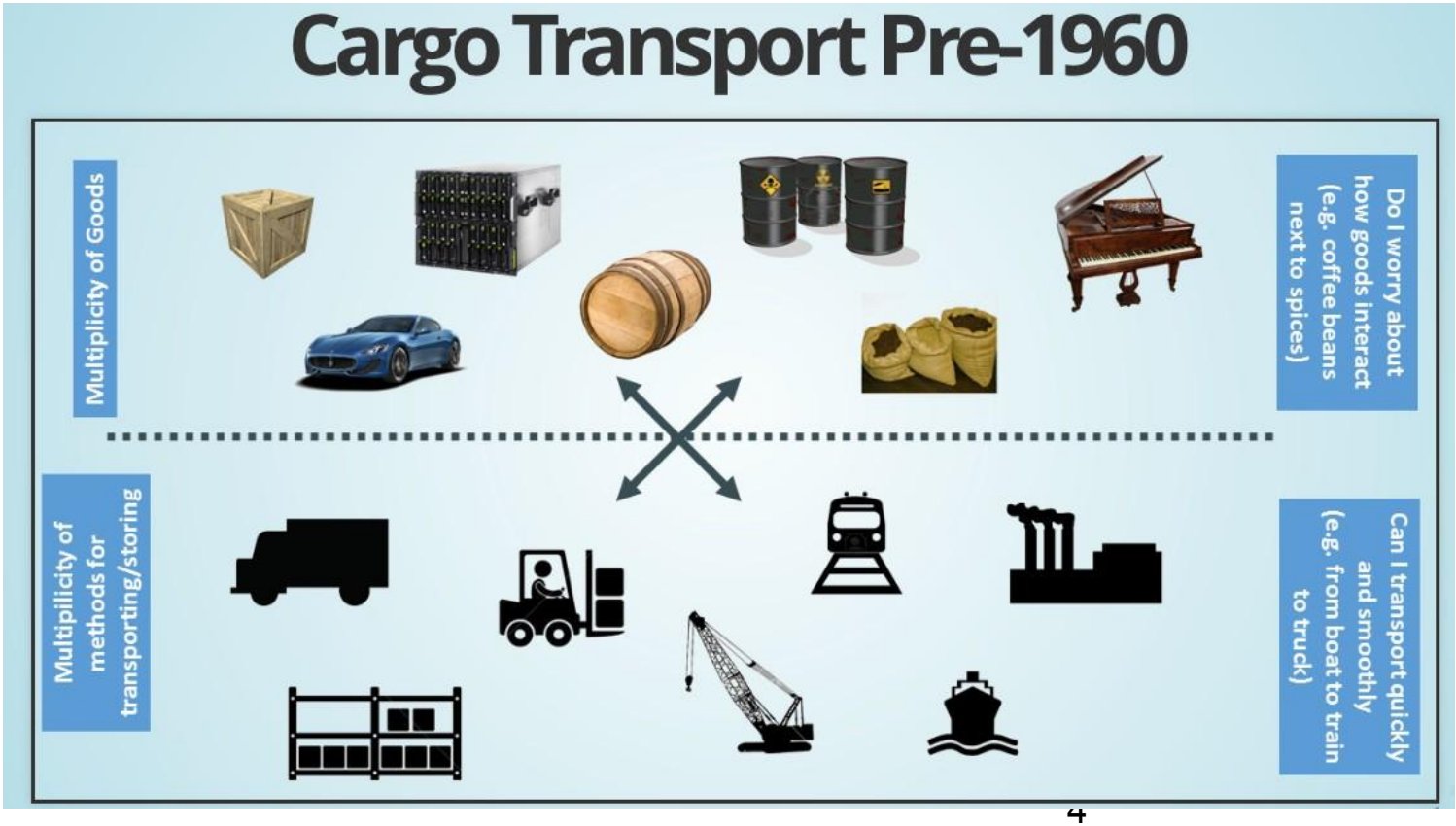
- Se espera que una vez acabes esta presentación seas capaz de:
  - Conocer las diferencias de las tecnologías de contenedores con respecto a la de máquinas virtuales.
  - Conocer la importancia de Docker como herramienta de encapsulación de aplicaciones en contenedores.
  - Comprender los principales conceptos relacionados con Docker.
  - Obtener una experiencia práctica con el entorno Docker.
  - Conocer algunas de las herramientas del amplio ecosistema de Docker.

Problemática

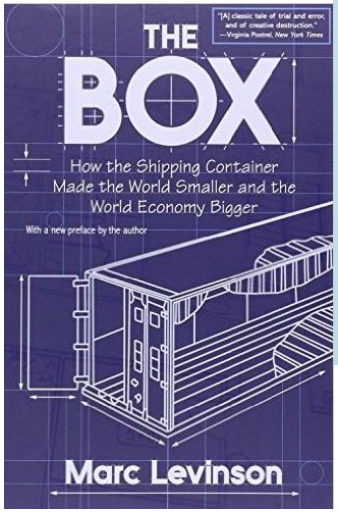
- Desarrollar aplicaciones distribuidas requiere diferentes SO, lenguajes de programación, entornos de ejecución, librerías, etc. y pueden desplegarse sobre múltiples plataformas.



Analogía con el Mundo Real

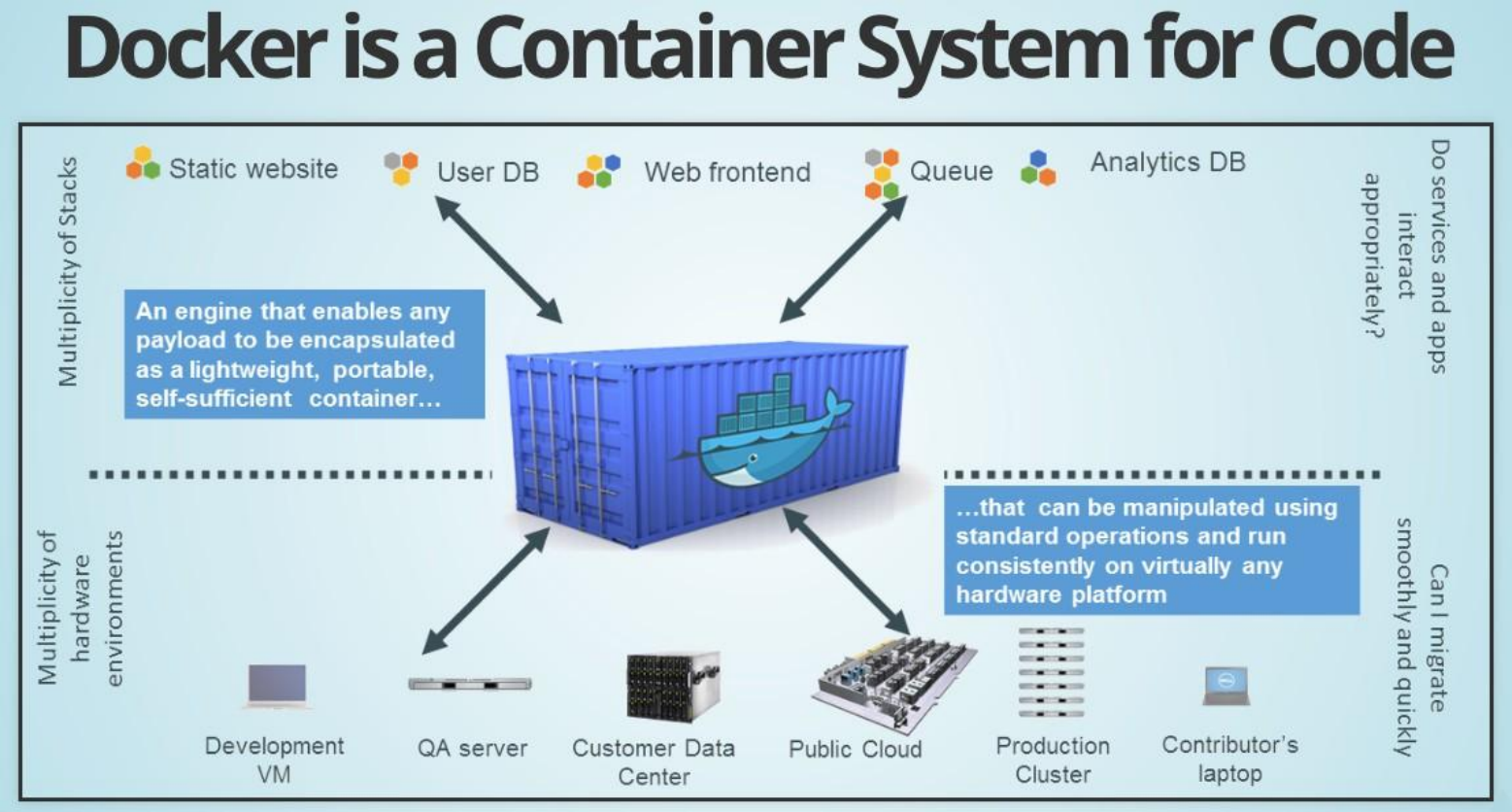


Solución en el Mundo real



http://www.amazon.com/The-Box-Shipping-Container-Smaller/dp/0691136408

Docker Containers



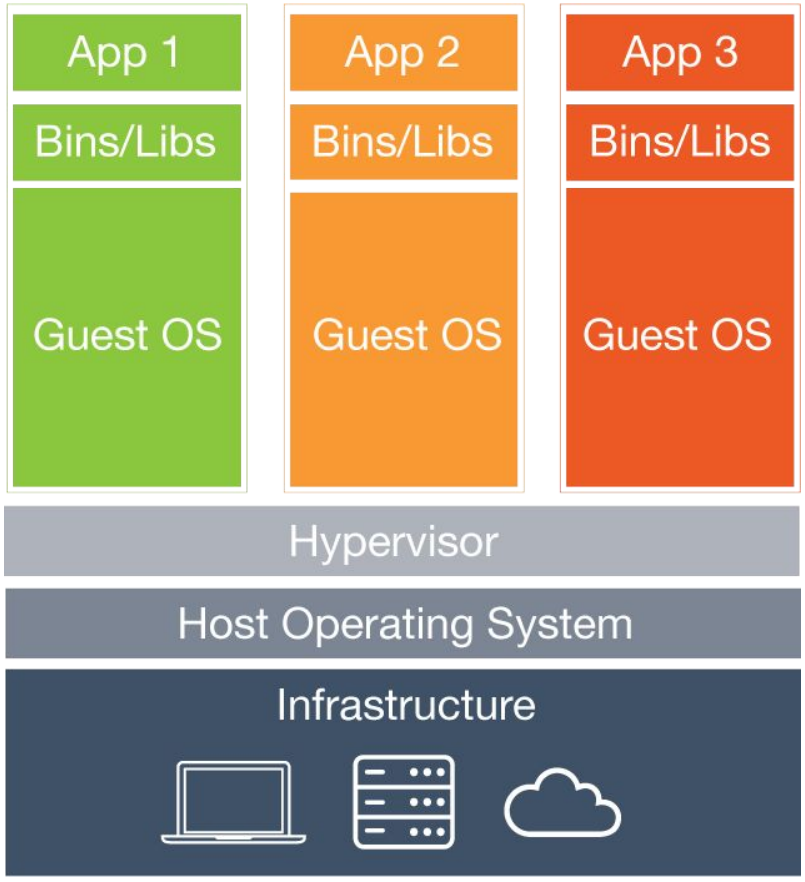
¿Qué es Docker?

Grid y Computación de Altas Prestaciones GRyCAP

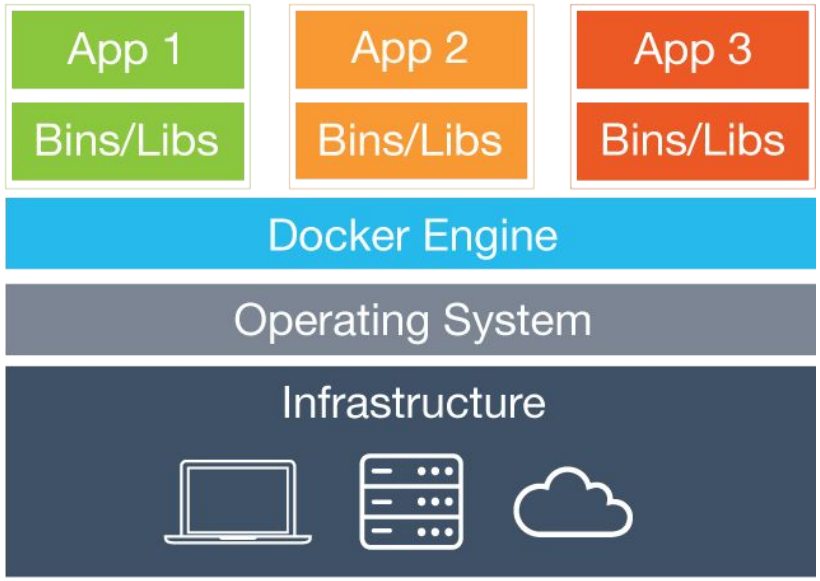
- Docker es una plataforma abierta para desarrolladores y administradores de sistemas para construir, enviar y ejecutar aplicaciones distribuidas.

- Permite empaquetar una aplicación con todas sus dependencias (SO, librerías, aplicaciones, etc.) para ser ejecutada en diferentes plataformas.
- Permite desplegar entornos de ejecución de aplicación rápidamente y de forma repetible.
  - Continuous Integration (CI) / Continuous Delivery (CD)

Contenedores vs Máquinas Virtuales



Máquina Virtual



Contenedores

Ventajas del uso de Docker

Grid y Computación de Altas Prestaciones GRyCAP

- Las instancias se arrancan en pocos segundos.
- Es fácil de automatizar e implantar en entornos de integración continua.
- Existen multitud de imágenes que pueden descargarse y modificarse libremente.
- Consume menos recursos de hardware y estos van exclusivamente a la aplicación.
- Tanto las imágenes como las instancias suelen ocupar menos espacio que las máquinas virtuales.

Inconvenientes del uso de Docker

Grid y Computación de Altas Prestaciones GRyCAP

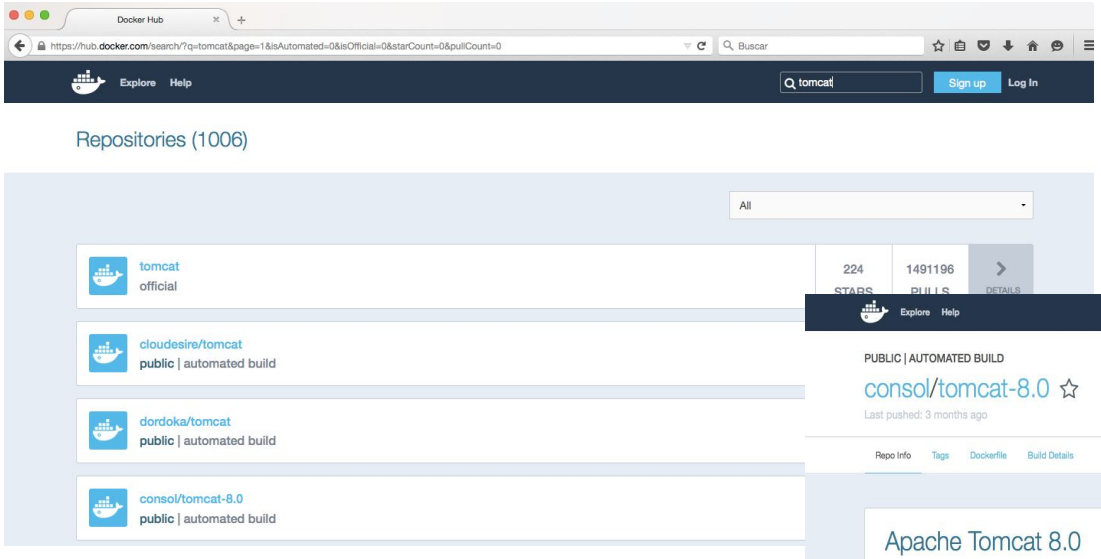
- Las imágenes sólo pueden estar basadas en versiones de Linux modernas (kernel 3.8 mínimo).
- Realmente no proporcionan el aislamiento de máquina lo que genera numerosas suspicacias sobre su seguridad.
  - Especialmente ya que el demonio funciona bajo ejecución privilegiada y en algunas operaciones necesita permisos de root.

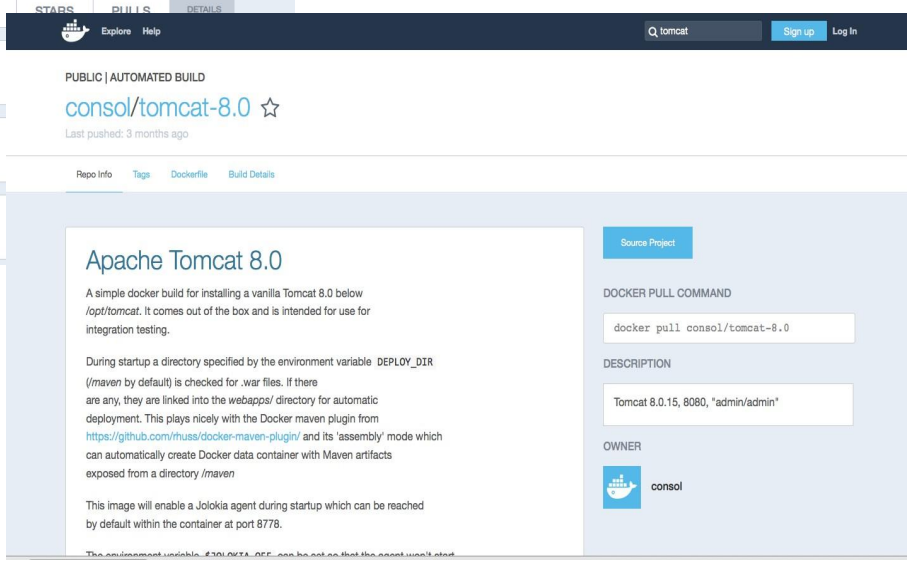
Conceptos de Docker

Grid y Computación de Altas Prestaciones GRyCAP

- Docker Hub / Docker Registry
  - Catálogo y repositorio de imágenes, accesible mediante CLI, interfaz web y REST API.
    - Docker Hub un catálogo público y el docker Registry es un catálogo propio.
- Docker Host
  - Es la máquina (o máquinas) que ejecutan contenedores Docker.
- Docker Client
  - Máquina desde la que se solicita el despliegue de contenedores Docker (puede coincidir con el Docker Host). También se corresponde con la herramienta cliente para interactuar con Docker.

Docker Hub





- Accesible mediante Web GUI, Docker Client, REST API

- Repositorios que contienen imágenes de contenedores Docker
- Automated Builds from GitHub

Conceptos de Docker

- Imagen
  - Contiene una distribución de SO (e.g Ubuntu 20.04) y una determinada configuración de paquetes / aplicaciones / datos determinada por el creador de la imagen.
  - Se almacenan las diferencias del sistema de fichero de forma incremental, de forma que se reduce el espacio en disco.
  - Siguen una estructura de tres elementos
    - organizacion/nombre_imagen:version
- Contenedor
  - Es una instancia de una imagen concreta ejecutada como un proceso aislado en una máquina concreta.
- Dockerfile
  - Descripción de las acciones de instalación y configuración que construyen una determinada imagen.
  - Las imágenes pueden generarse de forma incrementativa o directamente con el Dockerfile.

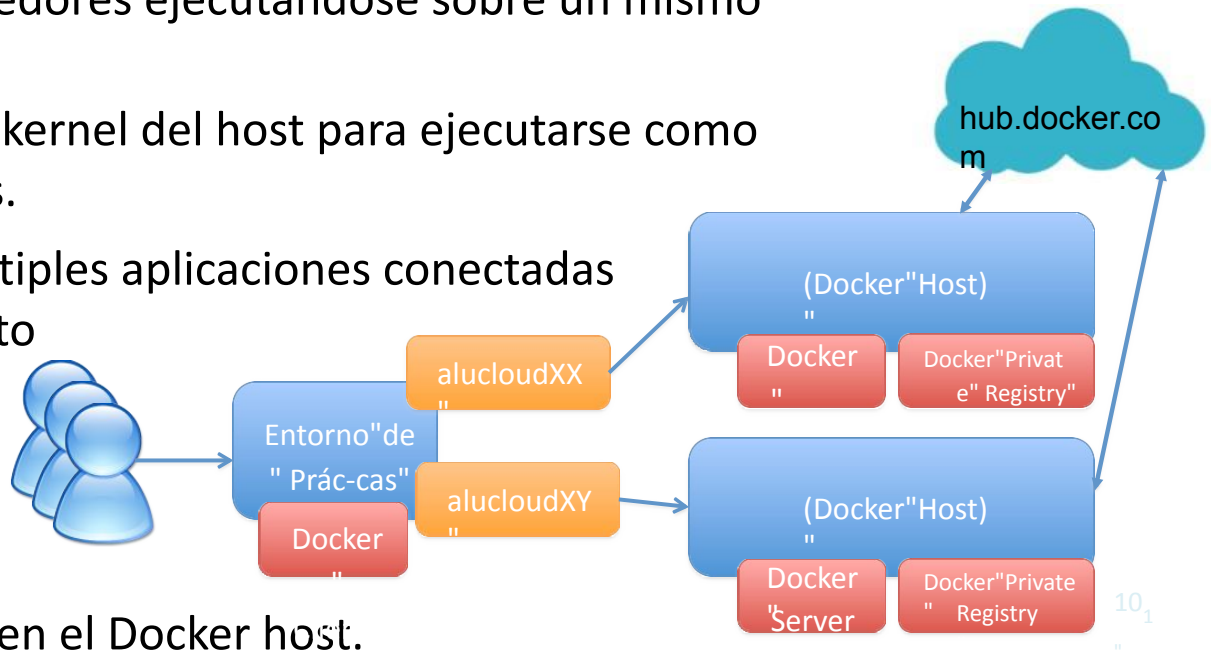
Dockerfile

- **FROM :** Imagen que se toma de base.
- **MAINTAINER:** Especifica el autor de la imagen.
- **ENV:** Variables de entorno en la imagen base.
- **RUN:** Comandos a ejecutar sobre la imagen base
  - Incluimos un enlace simbólico porque la versión actual de Ubuntu necesita que se le configure la región al instalar Apache2.
- **EXPOSE:** Exponemos el puerto 80 del contenedor para que pueda ser mapeado por el anfitrión.
- **ENTRYPOINT:** Comando que se ejecutará cuando se arranque el contenedor.

```
FROM ubuntu
RUN apt-get update
ENV TZ=Europe/Madrid
RUN ln -snf /usr/share/zoneinfo/$TZ
/etc/localtime && echo $TZ >
/etc/timezone
RUN apt-get install -y apache2
RUN echo "<h1>Apache with Docker</h1>" >
/var/www/html/index.html
EXPOSE 80
ENTRYPOINT apache2ctl -D FOREGROUND
```

Flujo de Trabajo con Docker

- Los usuarios usan el *Docker Client* para desplegar contenedores en un *Docker Host* a partir de imágenes almacenadas previamente en *Docker Hub* que pueden ser modificadas y almacenadas tanto en Docker Hub como en un *Docker Private Registry*.
  - Múltiples contenedores ejecutándose sobre un mismo Docker Host.
  - Compartiendo el kernel del host para ejecutarse como procesos aislados.
  - Puede haber múltiples aplicaciones conectadas a un mismo puerto (e.g. 80/http) en contenedores diferentes.
    Se mapean a un puerto diferente en el Docker host.



¿Qué se puede hacer con Docker ?

Grid y Computación de Altas Prestaciones GRyCAP

- Gestionar el ciclo de vida de contenedores
  - start, stop, kill, restart, etc.
- Gestionar las imágenes de contenedores
  - push, pull, tag, rmi, etc.
- Inspeccionar/acceder el contenedor
  - logs, attach
- …

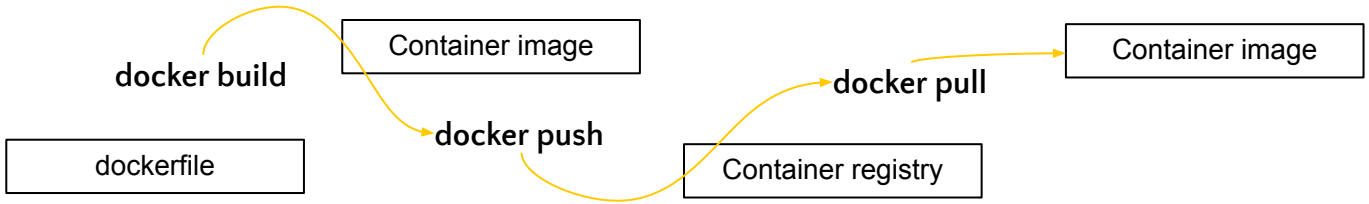
Ciclo de vida básico

Uso

- Descarga (docker pull imagen
- Ejecución (docker run)
- Reactivación (docker start)
- Ejecución (docker exec)
- Parada (docker stop)
- Borrado (docker rm)

Creación

- Creación del contenedor
  - Vía creación del Dockerfile
    - Creación de la imagen (docker build).
  - Vía actualización
    - Descarga (docker pull) y actualización.
    - docker commit.
- Publicación (docker push)



Ecosistema de Docker

Grid y Computación de Altas Prestaciones GRyCAP

- Herramientas basadas en Docker para:
  - Docker as a Service
  - Networking
  - Scheduler / Orchestration
  - Monitoring
  - DevOps
  - Developer

https://www.google.es/search?q=docker+ecosystem

Conclusiones

Grid y Computación de Altas Prestaciones GRyCAP

- Docker es una plataforma para la creación y ejecución de contenedores así como la gestión y almacenamiento de imágenes de contenedores para facilitar el desarrollo y ejecución de aplicaciones en múltiples entornos.
- En escenarios donde tradicionalmente se ha virtualizado GNU/Linux sobre GNU/Linux se impone como una solución efectiva, sin sobrecargas innecesarias.
- Relación con las Máquinas Virtuales
  - Virtualización para aislamiento en entornos multi-tenancy
  - Contenedores en cada VM para particionar uso de recursos entre aplicaciones.

Instalación

- Crear una nueva máquina en Azure
  - Linux 20.04
- Instalar Docker
  - `sudo apt update`
  - `sudo apt install -y apt-transport-https ca-certificates curl software-properties-common`
  - `curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -`
  - `sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"`
  - `sudo apt update`
  - `apt-cache policy docker-ce`
  - `sudo apt install -y docker-ce`
  - `sudo groupadd docker`
  - `sudo usermod -aG docker $USER`

Instalación

- No necesario, pero para evitar utilizar sudo en todos los comandos docker, agregamos el usuario al grupo docker
  - `sudo groupadd docker`
  - `sudo gpasswd -a ${USER} docker`
  - `sudo service docker restart`
  - Log out, y volver a entrar en la sesión.

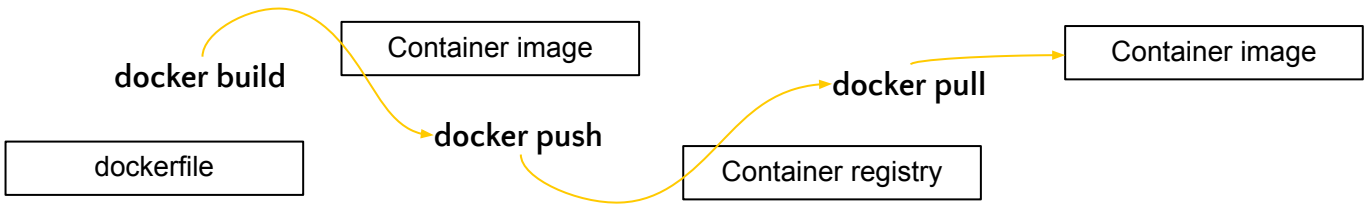
Recordatorio de comandos

Uso

- Descarga (docker pull imagen
- Ejecución (docker run)
- Reactivación (docker start)
- Ejecución (docker exec)
- Parada (docker stop)
- Borrado (docker rm)

Creación

- Creación del contenedor
  - Vía creación del Dockerfile
    - Creación de la imagen (docker build).
  - Vía actualización
    - Descarga (docker pull) y actualización.
    - docker commit.
- Publicación (docker push)



Primer ejemplo

Grid y Computación de Altas Prestaciones GRyCAP

- Crear un contenedor y ejecutar un shell
  - Descarga la imagen del contenedor "ubuntu"
    - `docker pull ubuntu`
  - Lista las imágenes descargadas
    - `docker images`
  - Ponemos el contenedor en marcha
    - `docker run -i -t --name micont ubuntu /bin/bash`
  - Comandos dentro del contenedor
    - `hostname`
    - `cat /etc/hosts`
    - `ls –l /`
    - `whoami`
  - Salimos del contenedor y comprobamos su estado
    - `docker ps -a`
  - Borramos el contenedor activo
    - `docker rm -f micont`

Otras carácterísticas

Grid y Computación de Altas Prestaciones GRyCAP

- Montaje de volúmenes
  - Es posible montar un volumen del sistema "host" sobre el contendor cuando se pone en marcha
  - Opción `-v directorio_host:directorio contenedor`
  - Ejemplo
    - `docker run -i -t --name micont -v $HOME:/mnt ubuntu /bin/bash`
    - Dentro del contenedor comprobamos el montaje
      - `ls -la /mnt`
      - `echo "dentro_del_contenedor" > /mnt/fichero`
    - Salir y comprobar la existencia del fichero ¿quién es el propietario?
- Copiado de datos host<->contenedor
  - Es posible copia datos desde un contenedor activo al host y viceversa
    - `docker cp nombre_cont:/ruta_origen ruta_destino`

Otras características

- Reglas de Firewall
  - La IP de un contenedor coincide con la de su host.
  - Docker define unas reglas de redirección de puertos que permiten redireccionar tráfico desde el exterior del host al interior del contenedor
  - Se establecen mediante la opción `-p puerto_host:puerto_contenedor`

Ejercicios

Grid y Computación de Altas Prestaciones GRyCAP

- Instalar un servidor web y acceder
  - A través de commit y como dockerfile (como el de la transparencia 14).
  - Modificar el fichero index.html y ejecutarlo en background.
  - O utilizar un directorio compartido donde modificar el html.
- Instalar octave y ejecutar una línea de proceso
  - Montar un directorio compartido, un proceso que lea de un directorio y procese cada imagen que recibe.

Instalar un servidor Web

Grid y Computación de Altas Prestaciones GRyCAP

- Descargar una imagen de ubuntu
  - `docker pull ubuntu`
- Ejecutar en background con un bash
  - `docker run -i -t --name micont ubuntu /bin/bash`
- Entrar en la máquina e instalar apache2
  - `apt-get update`
  - `apt-get install -y apache2`
  - `echo "<h1>Apache with Docker</h1>" > /var/www/html/index.html`
- Salir de la máquina y registrar los cambios en la imagen
  - `docker commit micont miweb`
- Eliminar y volver a crear con la redirección de puertos
  - `docker rm micont`
  - `docker run -d -it -p 8080:80 --name micont miweb apache2ctl -D FOREGROUND`
- El puerto 8080 debe ser accesible en la máquina.
  - `curl localhost:8080`
    `<h1>Apache with Docker</h1>`

Instalar un servidor Web

Grid y Computación de Altas Prestaciones GRyCAP

- Alternativamente se puede crear desde el dockerfile (Tr. 14)
- Se copia el fichero Dockerfile en un directorio (p.e. DFileDir) y con ese nombre.
- Se ejecuta el comando build especificando la etiqueta
  - `docker build –t miweb:latest DFileDir`
- Se puede poner en marcha directamente y sin tener que activar el servicio
  - `docker run -d -it -p 8080:80 --name micont miweb:latest`
- Desde fuera del contenedor
  - `curl localhost:8080`

    `<h1>Apache with Docker</h1>`

- Si quisiéramos acceder desde un navegador, deberíamos abrir el puerto 8888 en la consola de Azure.

Crear un contenedor con octave

- Queremos resolver un sistema de ecuaciones lineales que tenemos en un fichero (procesa.tgz, en PoliformaT).
- El fichero procesa.tgz contiene 3 ficheros:
  - `octave/procesa.m`
  - `octave/A`
  - `octave/b`
- El contenido de procesa.m es:

```
cd /octave
load A;
load b;
x=A\b;
save x;
err=norm(x-ones(size(x)))
```

- Cambiamos al directorio /octave
- Cargamos la matriz de coeficientes A
- Cargamos el vector de términos independientes b
- Resolvemos el sistema
- Guardamos la solución del sistema
- Calculamos el error cometido comparando con la solución esperada.

- Vamos a resolver el problema mediante un contenedor.

Crear un contenedor con octave

- Descomprimimos el fichero procesa.tgz en un directorio de la máquina host (p.e. `$HOME/myvol`)
  - Para copiar a la máquina host procesa.tgz podemos utilizar `scp`
    - `scp procesa.tgz usuario@máquina:.`
- Pondremos en marcha un contenedor con octave
  - Una imagen válida de octave es simexp/octave.
  - Podríamos instalar octave sobre un contenedor ubuntu (más costoso) mediante el comando `apt-get install -y octave`
- Lanzamos la imagen con un directorio compartido
  - `docker run -it --name mioctave -v $HOME/myvol:/myvol imagen_cont bash`

- Editamos el fichero procesa.m para que el directorio dentro del contenedor sea `/myvol/octave`
- Ejecutamos el proceso con el siguiente comando
  - `docker exec -it mioctave octave /myvol/procesa.m`

1. Docker Docs. https://docs.docker.com
2. Intro to Docker. http://pointful.github.io/docker-intro
3. Open Container Ecosystem. https://www.mindmeister.com/es/389671722/open-container-ecosystem-formerly-docker-ecosystem
4. Docker Slideshare. http://www.slideshare.net/docker
5. Containers have arrived -- and no one knows how to secure them.
   - http://www.infoworld.com/article/2923852/security/containers-have-arrived-and-no-one-knows-how-to-secure-them.html
6. Docker, Linux Containers (LXC), and security.
   - http://es.slideshare.net/jpetazzo/docker-linux-containers-lxc-and-security
7. Docker Ecosystem Survey. https://github.com/weihanwang/docker-ecosystem-survey
8. Proven Real-World Ways to Use Docker.
   - https://www.airpair.com/docker/posts/8-proven-real-world-ways-to-use-docker

- `docker pull <imagen>:<tag>`
- `docker images`
- `docker run -i -t <imagen> <comando>`
- `docker ps –a`
- `docker start <contenedor>`
- `docker rm <contenedor>`
- `docker stop <contenedor>`
- `docker exec -i -t <contenedor> <comando>`

- `docker run -it --name nombre -v dir_host:dir_contenedor imagen comando`
- `docker build -t nombre:tag directorio`