



ARQUITECTURA DEL SOFTWARE

2025-26

Jose Emilio Labra Gayo

Pablo González

Diego Martín

Celia Melendi



Escuela de
Ingeniería
Informática



Universidad de Oviedo

Laboratorio 3

Conceptos de despliegue



Despliegue

Desplegar una aplicación consiste en:

- Compilar código fuente

- Obtener dependencias

- Configurar entorno

- Empaquetamiento

- Envío al ordenador de destino

- Lanzamiento en entorno de ejecución

Entornos de ejecución

¿Dónde se ejecutará el software?

¿Qué dependencias tiene?

Sistema operativo

Librerías compartidas

Varias opciones

Máquinas físicas

Máquinas virtuales

Contenedores



Formas de realizar el despliegue

Manual

Puede ser más sencillo inicialmente y con despliegues esporádicos

Automático

Facilitar la re-configuración y comprobación de errores

Automático y continuo

Objetivo: Aumentar la agilidad de los equipos

¡Cuidado con el despliegue manual!

Círculo vicioso de tamaño y riesgo de despliegue



Despliegue continuo

"Si duele, hazlo más a menudo"

En el límite: "Hazlo continuamente"

Ejecutar el canal de despliegue en cada *commit*

Fase final: despliegue en producción

Posibilidades

- Confirmación por algún humano antes de ir a producción

- Despliegue automático en producción

- Despliegue en producción marcado por alguna etiqueta

Compromiso

- Coste de avanzar más rápido vs coste de errores en producción

Máquinas virtuales

Ejecución de aplicaciones en máquinas virtuales

- Requieren sistema operativo + librerías

- Permiten aislar las aplicaciones del hardware concreto

- Proveedores Cloud de máquinas virtuales: Azure, AWS, Google, Alibaba, ...

Ejemplo Azure

- <https://portal.azure.com/#home>

¿Qué es Docker?

Plataforma para desarrolladores y administradores de sistema

Comienza en 2011

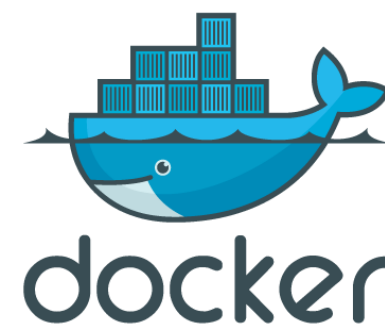
Se basa en contenedores e imágenes

Varias partes

Especificación descripciones contenedores (imágenes)

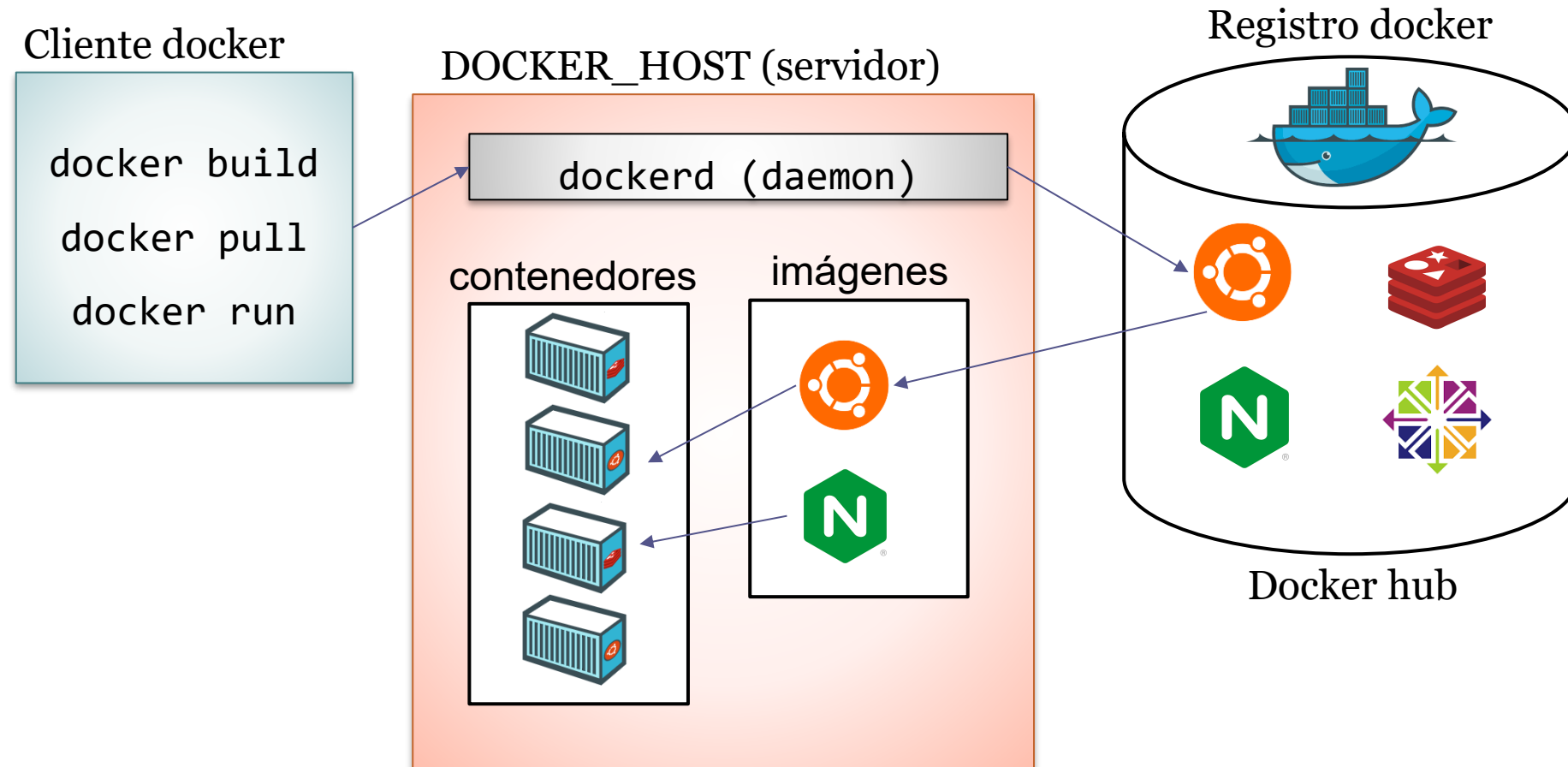
Plataforma ejecución contenedores

Registro contenedores (Docker-hub)



Arquitectura alto nivel docker

Arquitectura cliente-servidor



¿Qué es una imagen?

Un fichero que puede ser usado para crear un paquete ejecutable

Lenguaje que describe lo necesario para ejecutar una aplicación:

Código

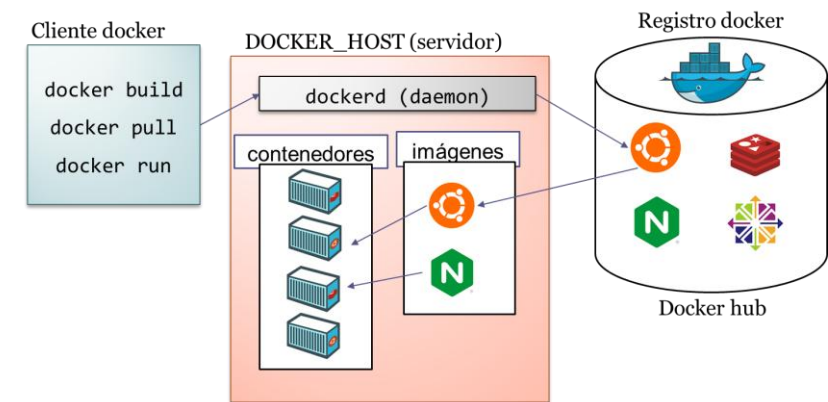
Sistema ejecutable

Librerías

Variables de sistema

Ficheros de configuración

No tiene estado y no cambia



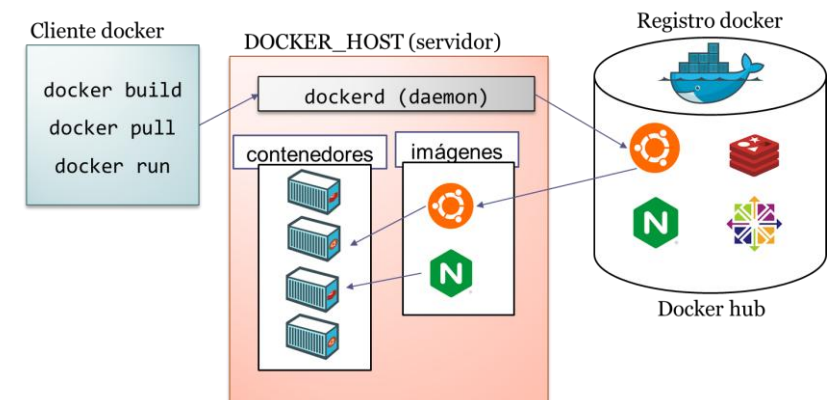
¿Qué es un contenedor?

Es una instancia de una imagen

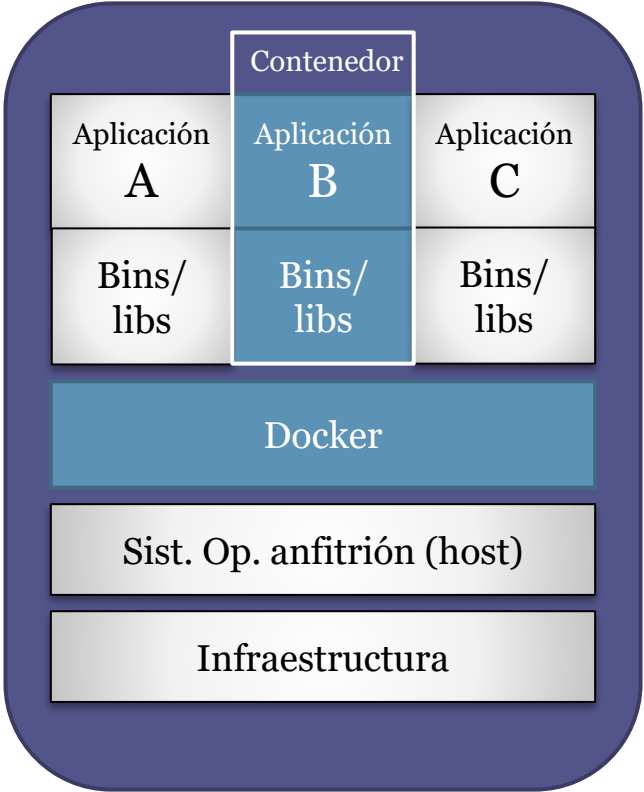
Docker está basada en contenedores que contienen aplicaciones

Docker permite orquestar varios contenedores

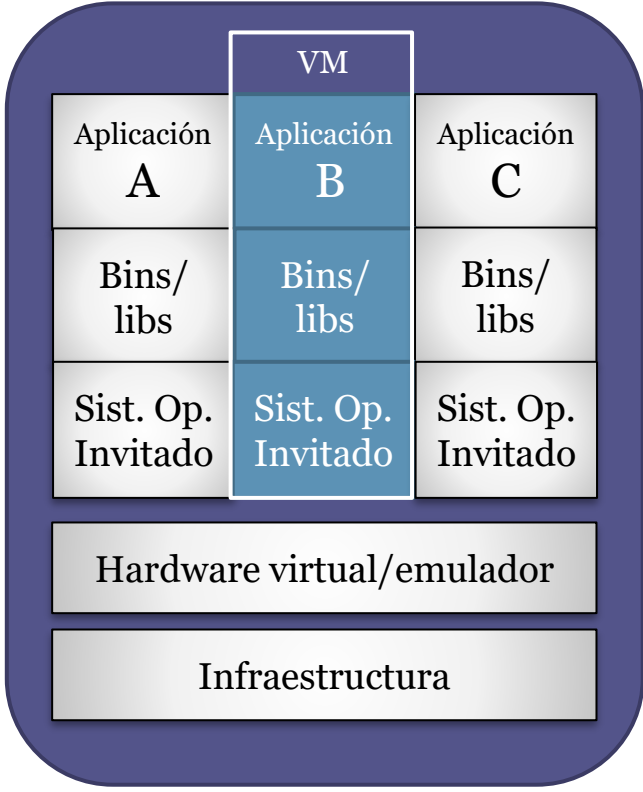
Enlazando varios contenedores podemos crear arquitecturas complejas



Contenedores vs máquinas virtuales



Contenedores



Máquinas virtuales

Source: <https://docs.docker.com/get-started/#containers-and-virtual-machines>
<https://stackoverflow.com/questions/16047306/how-is-docker-different-from-a-virtual-machine>

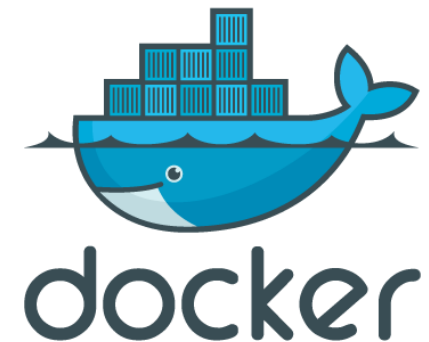
<https://docs.docker.com/get-started/#containers-and-virtual-machines>

Instalando docker

<https://www.docker.com>

Disponible para GNU/Linux, windows and Mac. Dos versiones:

- Docker desktop (Windows/Mac)
- Docker ToolBox [faq#issue3](#)



Registros de contenedores

Docker hub (<https://hub.docker.com/>)

Registro de imágenes

Ejemplo: Necesitamos un servidor web

```
docker pull nginx
```

```
docker pull httpd
```

Github Container Registry (<https://ghcr.io/>)

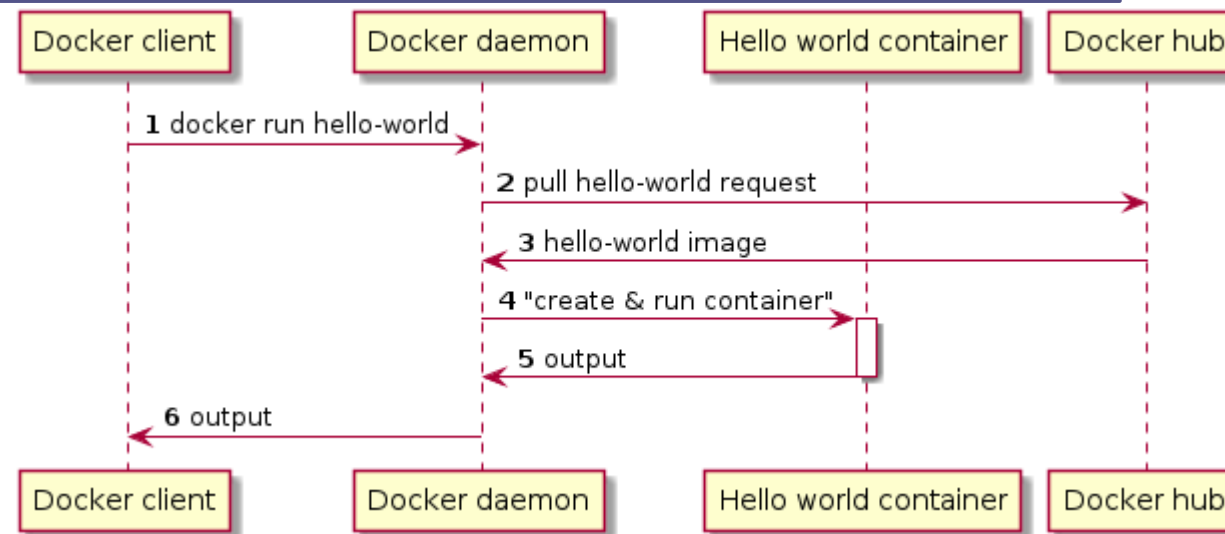
Registro de contenedores de Github (antes llamado github packages)

Docker paso a paso

Comprobar Instalación Docker Run “Hello World”

```
$ docker -v
```

```
$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:f9dfddf63636d84ef479d645ab5885156ae030f...
Status: Downloaded newer image for hello-world:latest
```



Docker para ejecutar Linux

Ejecutar Ubuntu

```
$ docker container run -it ubuntu:latest /bin/bash
. . .
root@813cb77cebb2:/# ls -la
total 72
drwxr-xr-x  1 root root 4096 Mar 30 05:46 .
drwxr-xr-x  1 root root 4096 Mar 30 05:46 ..
-rwxr-xr-x  1 root root    0 Mar 30 05:46 .dockerenv
drwxr-xr-x  2 root root 4096 Mar 11 21:05 bin
drwxr-xr-x  2 root root 4096 Apr 24  2018 boot
drwxr-xr-x  5 root root  360 Mar 30 05:47 dev
drwxr-xr-x  1 root root 4096 Mar 30 05:46 etc
. . .
drwxr-xr-x  1 root root 4096 Mar 11 21:03 usr
drwxr-xr-x  1 root root 4096 Mar 11 21:05 var
root@813cb77cebb2:/#
```


Chequear estado de docker

Commando para comprobar estados

```
λ docker image ls
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
hello-world	latest	fce289e99eb9	14 months ago	1.84kB

```
λ docker container ls --all
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
8b6518da11db	hello-world	"/hello"	9 minutes ago	Exited (0) 9 minutes ago

https://github.com/pglez82/docker_cheatsheet

Sencillo servidor web con Docker

Correr un servidor web con Docker

Ejecutado en background

publish:expose port

```
$ docker run --detach --publish=80:80 --name=webserver nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
68ced04f60ab: Pull complete
28252775b295: Pull complete
a616aa3b0bf2: Pull complete
Digest: sha256:2539d4344dd18e1df02be842ffc435f8e1f699cfc55516e2cf2cb16b7a9aea0b
Status: Downloaded newer image for nginx:latest
b7e9213eb3367cd465b29701a7e6441a7210a46d439196d30e76ddc9c72ee280
```

localhost

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

Otros comandos

```
docker info
```

```
docker ps
```

```
docker image ls
```

```
docker container ls -all
```

```
docker pull
```

```
docker run
```

```
docker stop
```

```
docker rm
```

Cómo construir una imagen

DSL para construir imágenes

Necesitamos crear un fichero que se llama **Dockerfile**

Contiene los comandos necesarios para crear la imagen

Keywords: FROM, RUN, ADD, COPY, ENV, EXPOSE, CMD...

Dockerfile

```
FROM ubuntu
```

```
CMD echo "Hi Software architecture students"
```

Ejemplo de ejecución de una imagen

1. Crear un directorio para el proyecto
2. Editar un Dockerfile (sin extensión)
3. `docker build -t nombre_imagen.`
4. `docker images` (lista de imagenes)
5. `docker run nombre_imagen`

Dockerfile

```
FROM ubuntu
CMD echo "Hi ASW students"
```

```
λ docker build -t "example1" .
Sending build context to Docker daemon 2.048kB
Step 1/2 : FROM ubuntu
latest: Pulling from library/ubuntu
5bed26d33875: Pull complete
...
Digest: sha256:bec5a2727be7fff3d308193cfde3491f8fba1a2...
Status: Downloaded newer image for ubuntu:latest
---> 4e5021d210f6
Step 2/2 : CMD echo "Hi Software architecture students"
---> Running in 9d5516995c2b
Removing intermediate container 9d5516995c2b
---> 41784c740df4
Successfully built 41784c740df4
Successfully tagged example1:latest
```

```
λ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
example1 latest 41784c740 32 seconds ago 64.2MB
```

```
λ docker run example1
Hi ASW students
```

Combinar diferentes contenedores de docker

Docker compose permite modularizar una aplicación

Los servicios definidos pueden comunicarse entre ellos

Cada servicio se encuentra en un contenedor separado

Archivo: `docker-compose.yml`



Ejecutar Docker compose

Configuración

- Podemos configurar múltiples servicios
- Los servicios pueden depender de otros
- Por defecto, todos los servicios comparten la misma red y son accesibles a través del nombre del contenedor.

Running

- Para ejecutar (o detener) un archivo docker-compose :
docker compose (up|down)

Despliegue de nuestra aplicación



Despliegue automático con Github actions

Permite ejecutar *workflows* automáticamente

A partir de ciertas acciones

Para cada commit, cada release,...

Configuración: Ficheros YAML en `.github/workflows`

Ejemplo:

https://github.com/Arquisoft/yovi_0/tree/master/.github/workflows

Requisitos

Máquina virtual de destino creada

Opciones: Azure, Oracle Cloud, AWS, ...

IP pública para la máquina virtual

Acceso a través de ssh

Puertos abiertos: 80, 3000, 4000, 9090 y 9091

Docker instalado en máquina virtual

Configurar variables de entorno de repo de github

Cómo acceder a la máquina desde github actions

Ejemplo: Máquina virtual de Azure

Subscripción: Azure for students (100\$/month)

Location: France Central

Type: Linux Ubuntu

Size: Standard B2ats v2 (2 vcpus, 1 GiB memory)

Public IP (se requiere para aplicación web)

Acceso SSH (Importante: guardar clave privada)

Crear reglas para puertos de entrada

Instalar docker en la máquina virtual

Seguir:

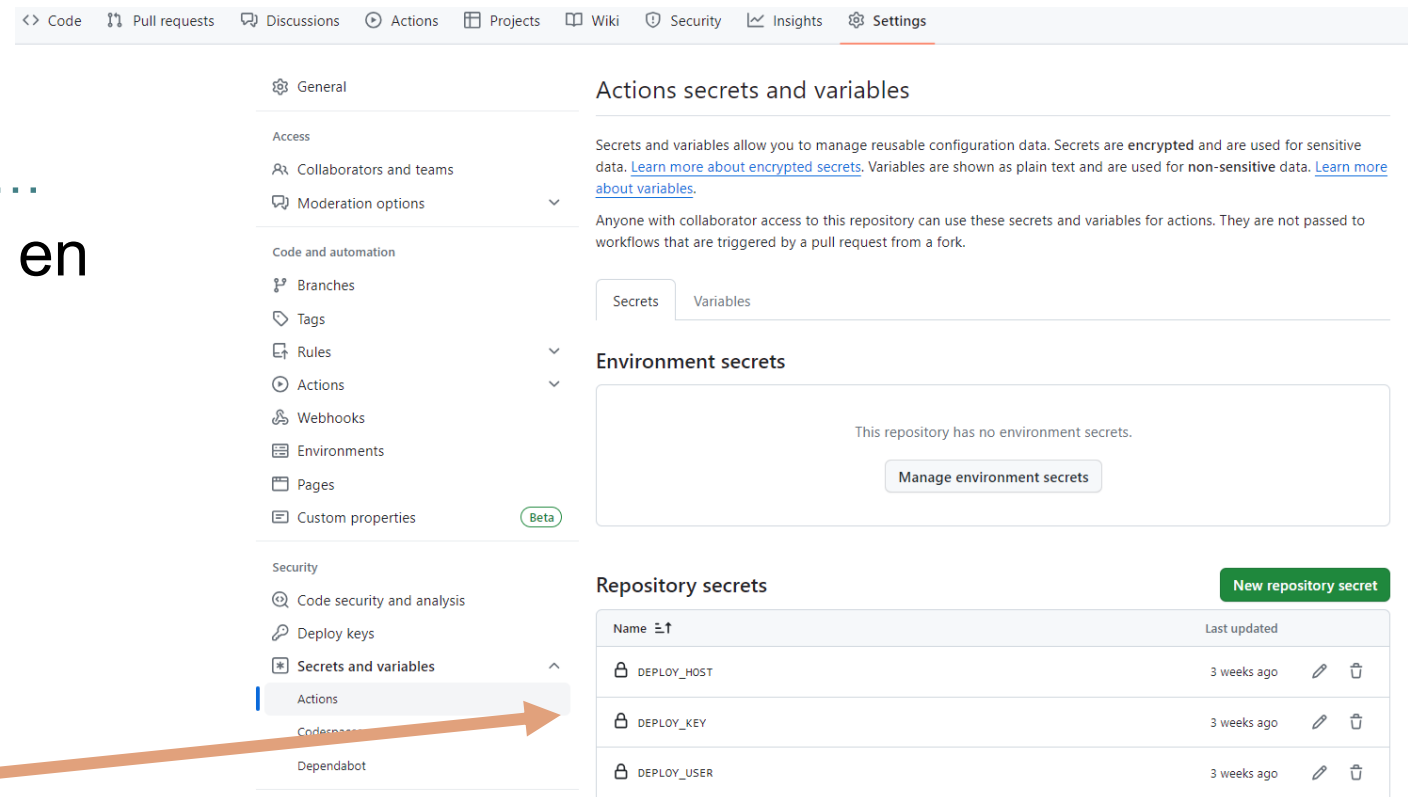
<https://docs.docker.com/engine/install/ubuntu/#install-using-the-repository>

Variables de entorno de ejecución

Parametrizar el despliegue

Definir credenciales, identificadores,...

Github permite definir SECRETS en cada repositorio



The screenshot shows the GitHub repository settings page. The left sidebar has a menu with items like General, Access, Code and automation, Security, and Secrets and variables. The 'Secrets and variables' item is highlighted. The main content area shows 'Actions secrets and variables' and 'Environment secrets'. The 'Repository secrets' table lists three secrets: DEPLOY_HOST, DEPLOY_KEY, and DEPLOY_USER, all updated 3 weeks ago.

Name	Last updated
DEPLOY_HOST	3 weeks ago
DEPLOY_KEY	3 weeks ago
DEPLOY_USER	3 weeks ago

En yovi_XXX:

DEPLOY_HOST: IP de la máquina virtual

DEPLOY_USER: Usuario con permisos para acceder por ssh a máquina virtual

DEPLOY_KEY: Clave privada de dicho usuario

Despliegue continuo a partir de release

Pasos

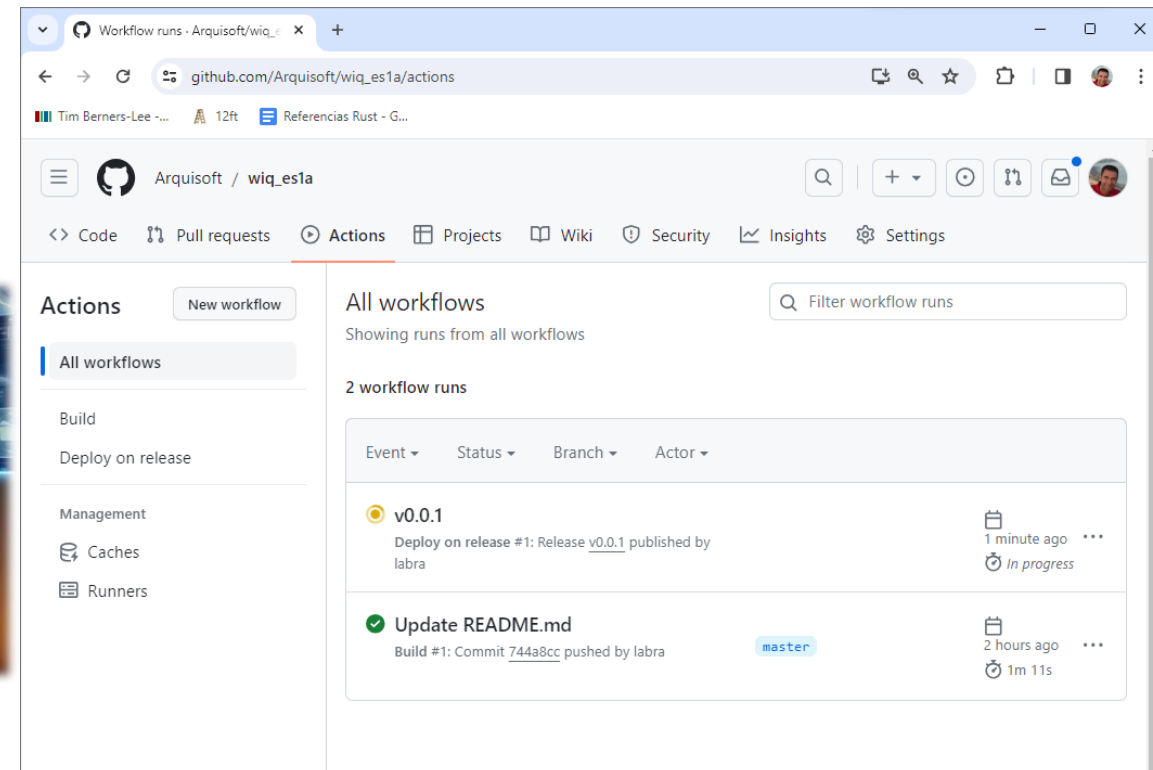
- Crear una etiqueta y enviarla a github

```
git tag -a v0.0.1 -m "v0.0.1"
```

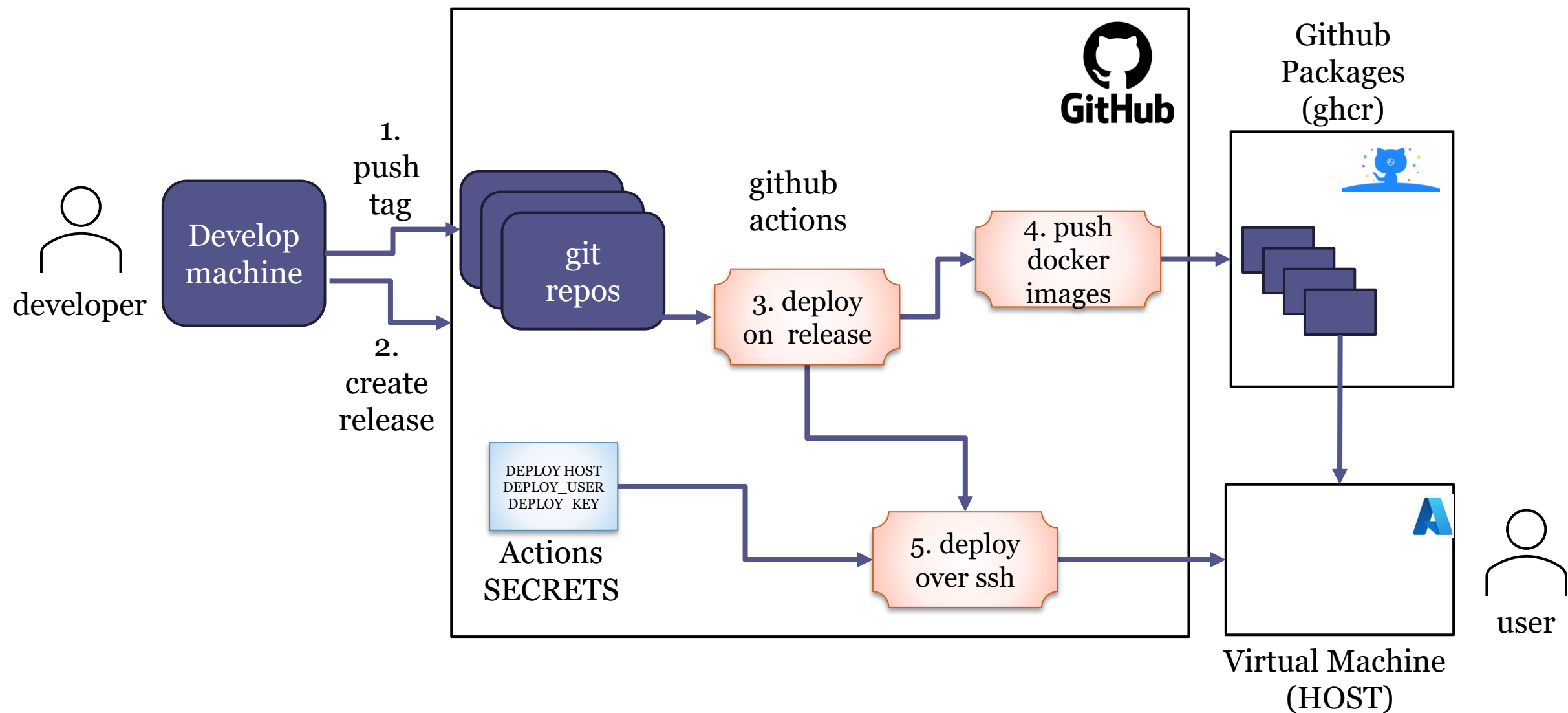
```
git push origin v0.0.1
```

- Crear una release desde github

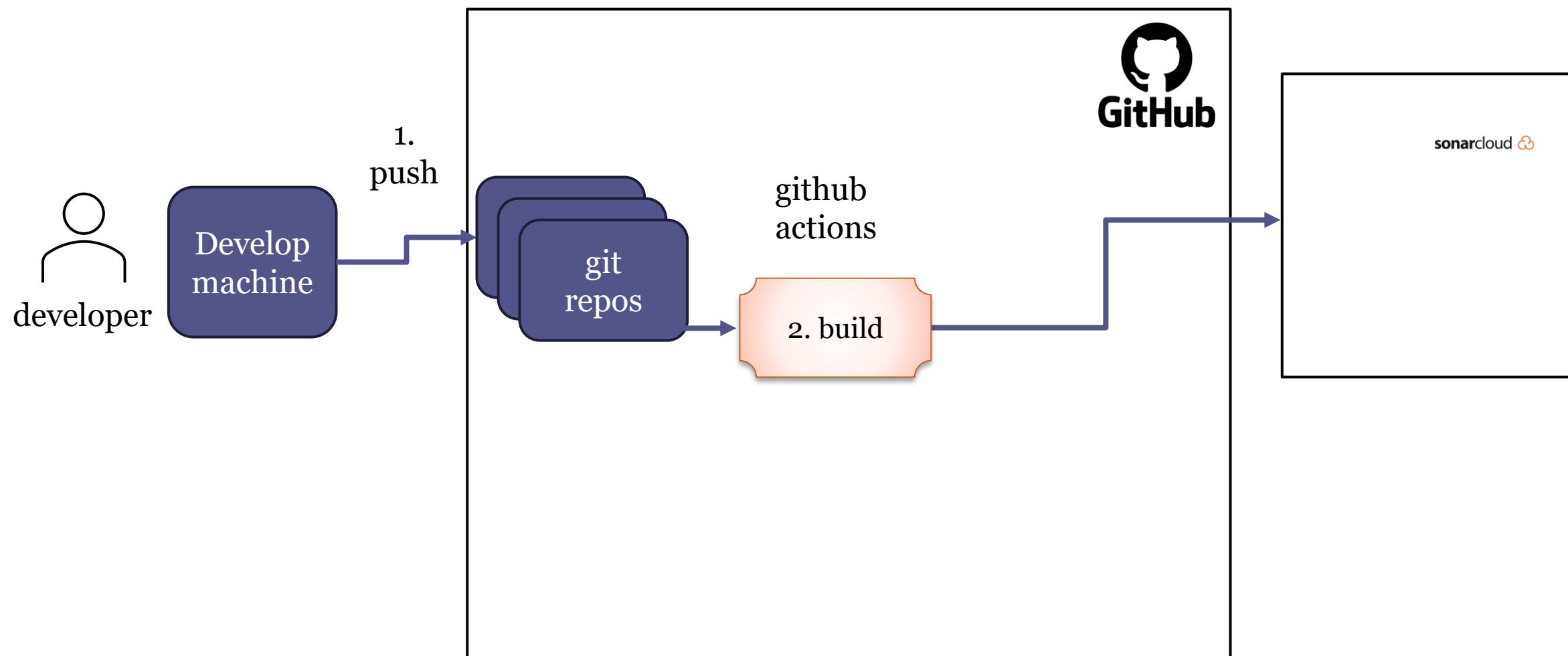
Ver cómo se despliega automáticamente



Canal de despliegue en yovi_xxx



Canal de construcción en yovi_xxx



Información adicional

Pequeño repositorio con comandos básicos docker:

https://github.com/pglez82/docker_cheatsheet