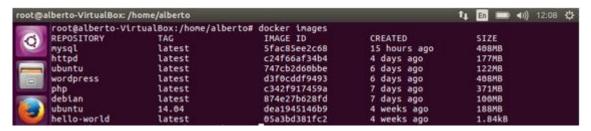
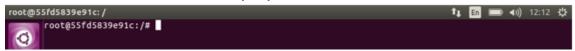
Practicando con Docker

- 1. Bajar las imágenes que serán la "plantilla" para los contenedores que usaremos posteriormente (usaremos **docker pull**)
- 2. Obtener la lista de imágenes disponibles



3. Creamos un contenedor partiendo de una imagen (la imagen ubuntu:14.04). Podéis probar con otras imágenes.

docker run -it --name u1 ubuntu:14.04 /bin/bash



Al arrancar el contenedor obtendremos el prompt de administrador para un sistema Ubuntu 14.04 que vemos arriba

Ejecutar las siguientes ordenes dentro de ese contenedor:

```
root@55fd5839e91c:/

root@55fd5839e91c:/# uname -a
Linux 55fd5839e91c 4.10.0-28-generic #32-16.04.2-Ubuntu SMP Thu Jul 20 10:19:48 UTC 2017 x86_64 x86_64 x80_fd4 x80_
```

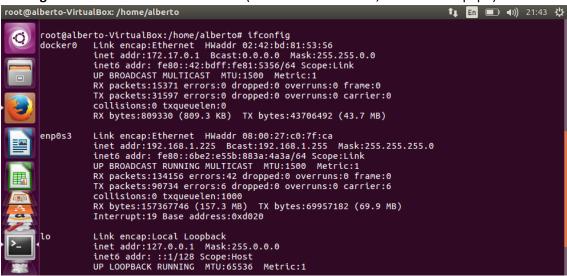
```
Get:24 http://archive.ubuntu.com trusty/multiverse amd64 Packages [169 kB]
Fetched 21.1 MB in 23s (910 kB/s)

Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
spell
The following NEW packages will be installed:
nano
0 upgraded, 1 newly installed, 0 to remove and 5 not upgraded.
Need to get 194 kB of archives.
After this operation, 614 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu/ trusty/main nano amd64 2.2.6-1ubuntu1 [194 kB]
Fetched 194 kB in 1s (112 kB/s)
Selecting previously unselected package nano.
(Reading database ... 11569 files and directories currently installed.)
Preparing to unpack .../nano 2.2.6-1ubuntu1_amd64.deb ...
Unpacking nano (2.2.6-1ubuntu1) ...
Setting up nano (2.2.6-1ubuntu1) ...
update-alternatives: using /bin/nano to provide /usr/bin/editor (editor) in auto mode root@55fd5839e91c:/# ifconfig eth0 Link encap:Ethernet HWaddr 02:42:ac:11:00:02
```

ifconfig (comprobar el tipo de ip que se le ha concedido al contenedor)

ping 8.8.8.8 desde el contenedor (podrás comprobar que funciona)

ifconfig en otra terminal del host docker (fuera del contenedor, en nuestro equipo)



ping a la ip de tu host docker

```
root@55fd5839e91c:/# ping 8.8.8.8

root@55fd5839e91c:/# ping seq=1 ttl=54 time=21.1 ms

d4 bytes from 8.8.8.8: icmp_seq=2 ttl=54 time=17.9 ms

d4 bytes from 8.8.8.8: icmp_seq=3 ttl=54 time=17.6 ms

root@55fd5839e91c:/# ping 172.17.0.1

root@55fd5839e91c:/# ping 10.0.2.15

root@55fd5839e91c:/# ping 10.0.2.15
```

Hemos salido del contenedor y el contenedor, por cómo está definido y por como lo hemos arrancado dejará de ejecutarse.

4. Listar los contenedores que están en ejecución



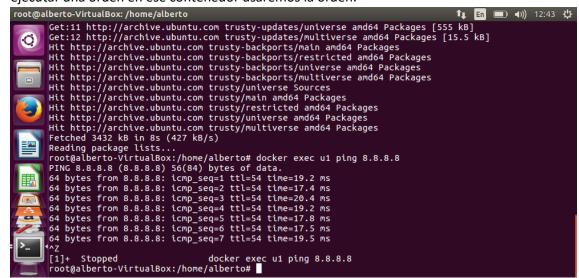
Podemos ver que el estado de nuestro contenedor es Up, es decir, está funcionando.

5. Parar (docker stop) y rearrancar (docker start) contenedores



6. Ejecutar órdenes en un contenedor que está ejecutándose pero si necesidad de tener que ejecutar un Shell

Si el contenedor está funcionando (comprueba que está UP con **docker ps**) para ejecutar una orden en ese contenedor usaremos la orden:



Y si queremos lanzar un bash en un contenedor funcionado:

Arrancando contenedor con servicios

1. Arrancar contenedores con servicios



2. Arrancar un contenedor de la imagen de MySQL

*No se puede realizar en la máquina virtual Ubuntu 16.04 de pruebas debido a problemas con la arquitectura del SO Windows, máquina real sobre la que se ejecuta el software de emulación. Se le adjunta la captura de un compañero en su defecto y a su petición

```
root@ricaberto-GA-78LMT-S2P:/home/ricaberto# docker run -d --name m1 -e MYSQL_ROOT_PASSWORD=root -p 3316:3306 my
sql
6c584b1a03e5331a6f49d2b5da91b3980e2a1bde533bd<u>9</u>47ff4155e5cf8997ba
```

```
root@ricaberto-GA-78LMT-S2P:/home/ricaberto# docker exec -it m1 mysql -u root -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.7.20 MySQL Community Server (GPL)

Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

mysql -u root -h X.X.X.X (tu ip) -P 3316 -p

3. Compartiendo directorios entre el host y el contenedor

```
froot@alberto-VirtualBox:/home/alberto# docker run -it -v /home/alberto:/datos --name d3 debian /bin/bashroot@717cffeeb178:/#
```

4. Usando una carpeta propia como directorio raíz de un servidor web.

```
root@alberto-VirtualBox:/home/alberto

root@alberto-VirtualBox:/home/alberto# docker run -d --name a2 -v ruta_a_mi_carpeta:/usr/local/apa che2/htdocs -p 8181:80 httpd 348e4487e013d541dea1a6428cf93f43a92ccec821a96d16309e5001171b43ba root@alberto-VirtualBox:/home/alberto#
```

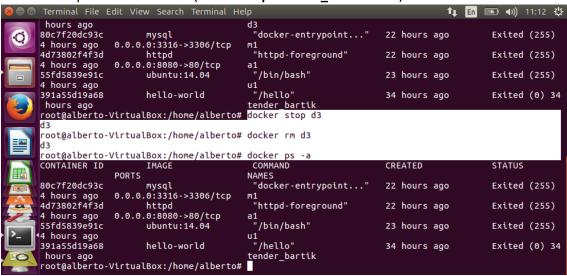
5. Viendo los puertos abiertos y compartidos por cada contenedor



Gestión de imágenes y contenedores

1. Para borrar contenedores

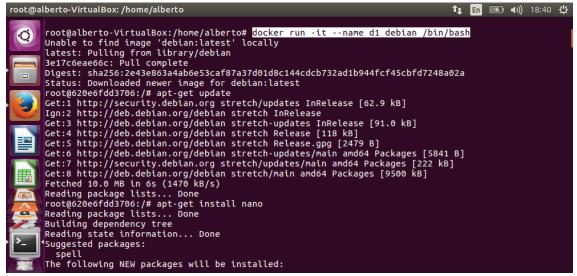
*Primero se para el contenedor (docker stop nombre_contenedor)



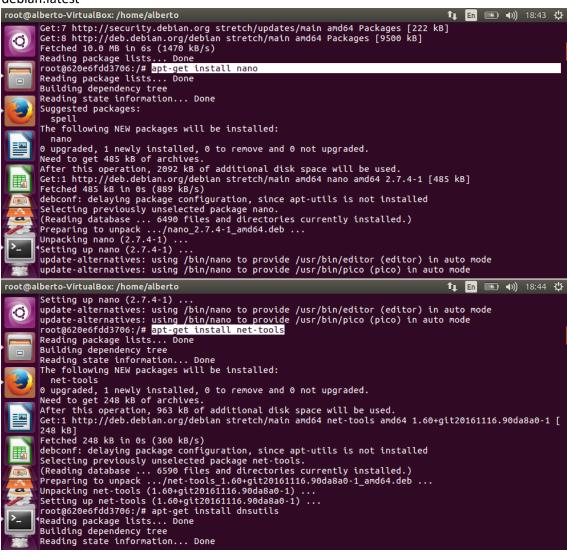
2. Para borrar imágenes

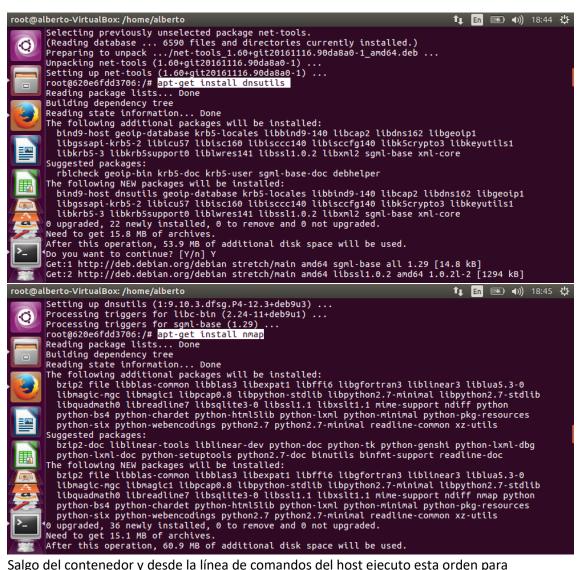


3. Creando una nueva imagen a partir de un contenedor

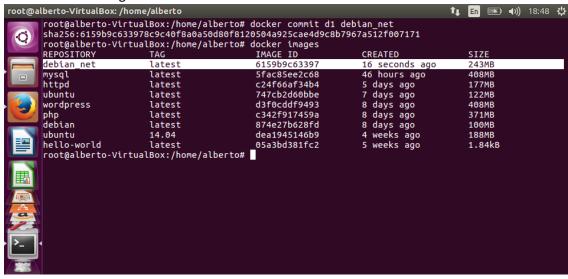


En ese contenedor vamos a ejecutar la siguiente órdenes para instalar ciertos programas para la gestión de red que por defecto no vienen instalador en la imagen de debian:latest

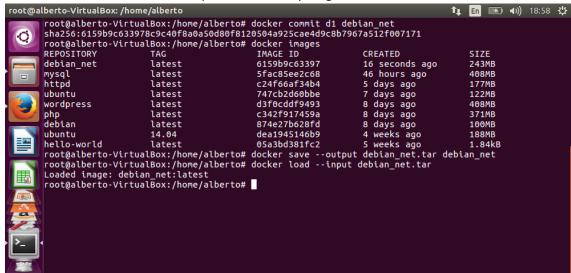




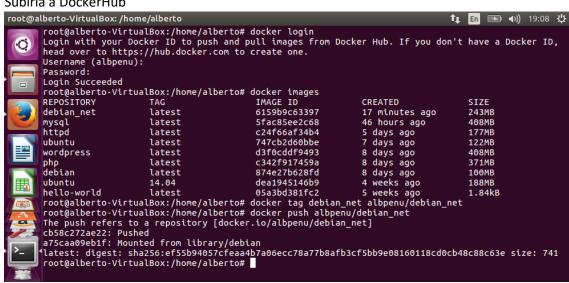
Salgo del contenedor y desde la línea de comandos del host ejecuto esta orden para crear mi nueva imagen:



Y al hacer docker images tendré esa nueva imagen en nuestra lista de imágenes. Para compartirla puedo hacer dos cosas: a) Guardarla con docker save, compartir el fichero y cargarla con docker load:



b) Subirla a DockerHub



Y podremos comprobar que esa imagen ya está pública a través de nuestra cuenta de DockerHub.

