CROSS PLATFORM ENABLEMENT FOR THE YOCTO PROJECT WITH CONTAINERS

ELC 2017 – Randy Witt – Intel Open Source Technology Center

MY PERSONAL PROBLEMS

Why'd I even do this?

THE MULTIPLE DISTRO PROBLEM

- Yocto Project QA (Autobuilder) builds on multiple distros
- On failure what are the options?
  - ssh in, clone repo, set up build dir
  - Create a virtual machine and try there
- Way too much overhead for me

BITBAKE SOMETIMES LEAKS PROCESSES

- Ctrl-c may not clean up all processes
- Must manually find the processes and kill them
- May not even know the processes are there

CONTAINERS AS A SOLUTION

Quick overview

CONTAINERS AREN'T MAGIC

- Leverage Linux® kernel features
  - namespaces for isolation (pid, network, mount)
    - Running a container can be as simple as using "unshare"
  - cgroups for process encapsulation and resource management
    - Restrict number of cores, amount of memory, …
    - All processes for the container run in a cgroup so can kill at cgroup level
  - Most things that run containers, leverage these kernel features
    - docker, lxc, …

PID NAMESPACE EXAMPLE

Inside container

```
pokyuser@6325b7c8feaf:~$ sleep 5000 &
[1] 32
pokyuser@6325b7c8feaf:~$ ps -C sleep -o pid,start,args
  PID  STARTED COMMAND
   32 15:24:03 sleep 5000
```

Outside container

```
~% ps -C sleep -o pid,start,args
  PID  STARTED COMMAND
 8257 15:24:03 sleep 5000
```

DOCKER

- Use a Dockerfile to create an image
  - Dockerfile used to install software to the image and configure it
- A container uses a temporary instance of the image
  - Modifications to the filesystem instance aren't preserved

SAMPLE DOCKERFILE

```
FROM ubuntu-16.04

RUN apt-get install python

CMD echo “Hello from inside the container!”
```

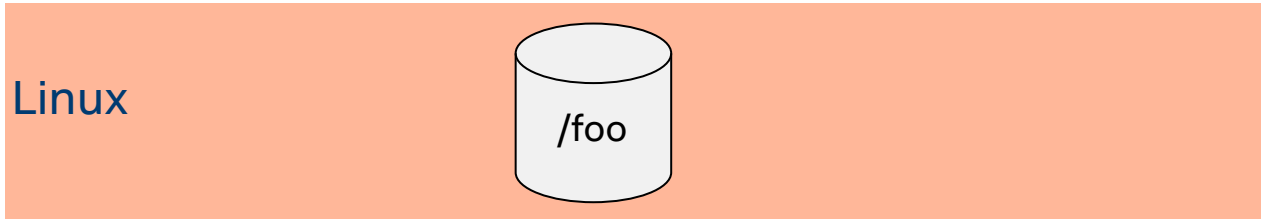
DOCKER RUN

```
docker run --rm -it -v /foo:/bar c1
```

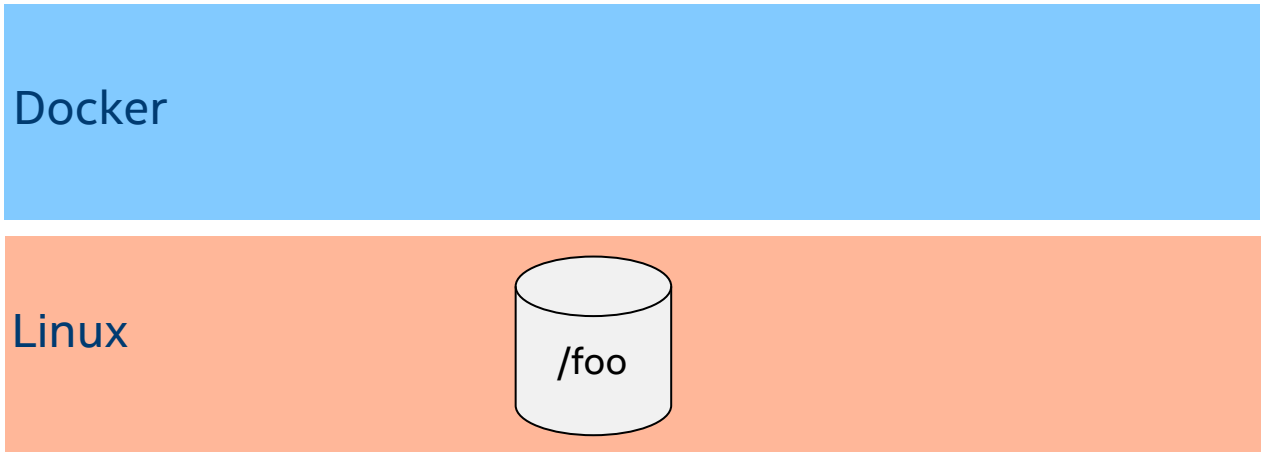
- ***--rm:*** Remove the container after it exits
- ***-it:*** Interactive terminal with a tty
- ***-v***: bind mount /foo to the container as /bar
- ***c1:*** Name of the image to run

Linux

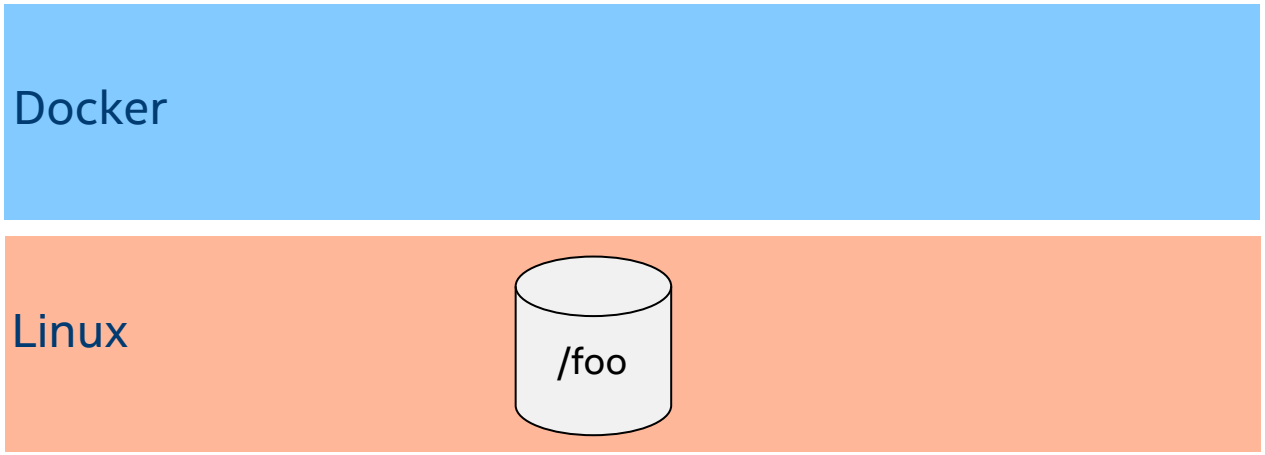
```
docker run -v /foo:/bar c1
```



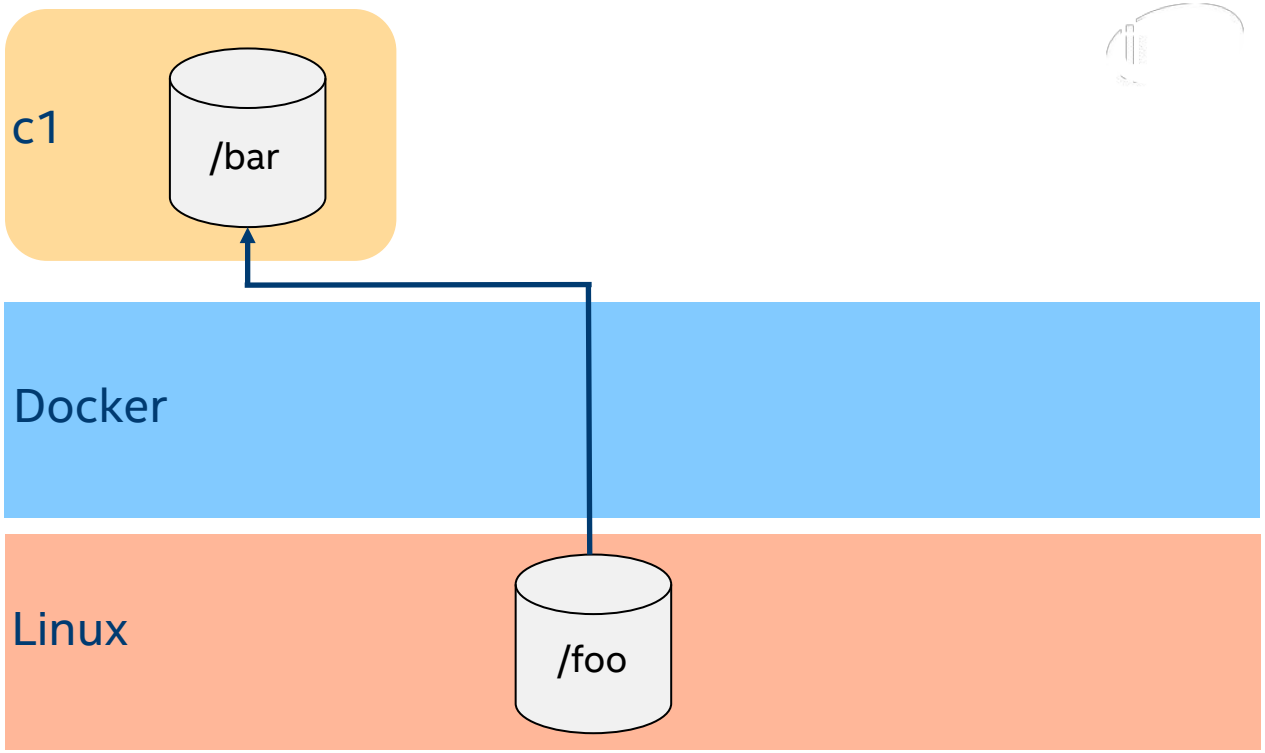
```
docker run -v /foo:/bar c1
```



```
docker run -v /foo:/bar c1
```

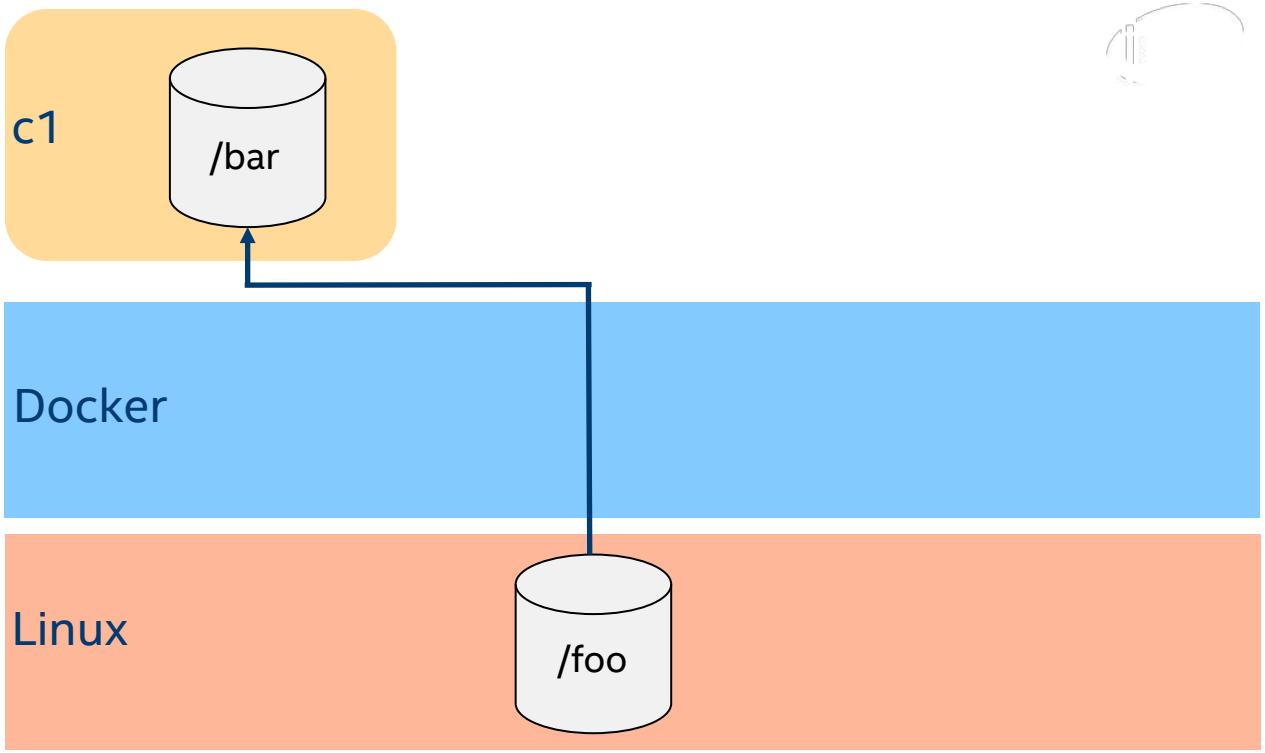


```
docker run -v /foo:/bar c1
```



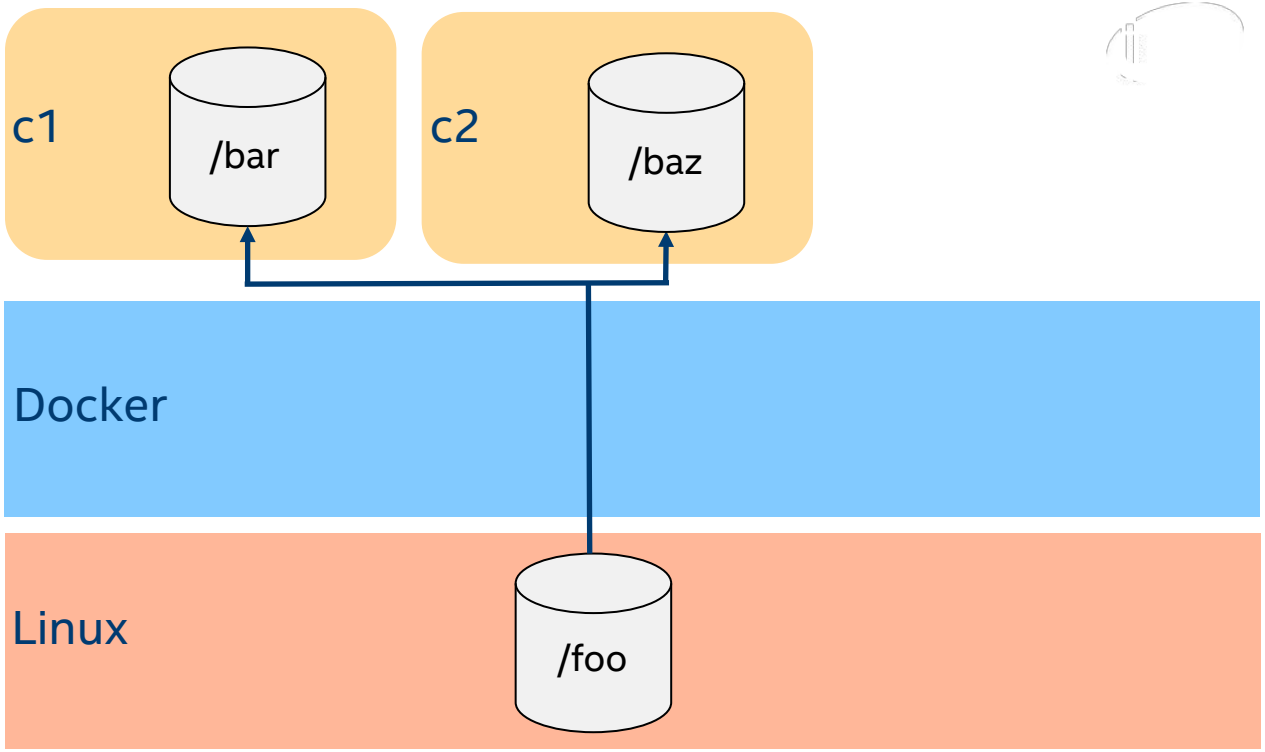
```
docker run -v /foo:/bar c1
```

```
docker run -v /foo:/baz c2
```



```
docker run -v /foo:/bar c1
```

```
docker run -v /foo:/baz c2
```



YOCTO PROJECT CONTAINERS

What's available?

POKY CONTAINER

- Drops to a shell where you follow normal Yocto Project instructions

```
docker run --rm -it -v /home/myuser/mystuff:/workdir crops/poky
 --workdir=/workdir
```

- Default based on Ubuntu 14.04
- Can use a different distro

```
docker run --rm -it -v /home/myuser/mystuff:/workdir
 crops/poky:fedora-24 --workdir=/workdir
```

POKY CONTAINER

```
docker run --rm -it -v /home/myuser/mystuff:/workdir crops/poky
 --workdir=/workdir
```

- ***--workdir:*** The working directory when dropped to the shell

POKY CONTAINER DISTROS

```
docker run --rm -it -v /home/myuser/mystuff:/workdir
crops/poky:opensuse-42.2 --workdir=/workdir
```

- debian-8
- fedora-22
- fedora-23
- fedora-24
- fedora-25
- opensuse-13.2
- opensuse-42.1
- opensuse-42.2
- ubuntu-14.04
- ubuntu-16.04
- ubuntu-16.10

POKY CONTAINER SCREENCAST

https://www.youtube.com/watch?v=vt18U5twrgw

EXTENSIBLE SDK CONTAINER

- Downloads an extensible sdk and drops to a shell ready to run sdk commands

```
docker run --rm -it -v /home/myuser/sdkstuff:/workdir
 crops/extsdk-container
 --url http://someserver/extensible_sdk_installer.sh
```

- If the sdk has already been installed and setup, just leave off the url

```
docker run --rm -it -v /home/myuser/sdkstuff:/workdir
 crops/extsdk-container
```

EXTSDK CONTAINER SCREENCAST

https://www.youtube.com/watch?v=L-sXqUoU49Y

TOASTER CONTAINER

- Runs toaster

```
docker run -it --rm -p 127.0.0.1:18000:8000
-v /home/myuser/toasterstuff:/workdir crops/toaster
```

- ***-p:*** Forwards port 8000 in the container to 127.0.0.1:18000

TOASTER CONTAINER SCREENCAST

https://www.youtube.com/watch?v=LJ9TBsuMwFA

OTHER PLATFORMS

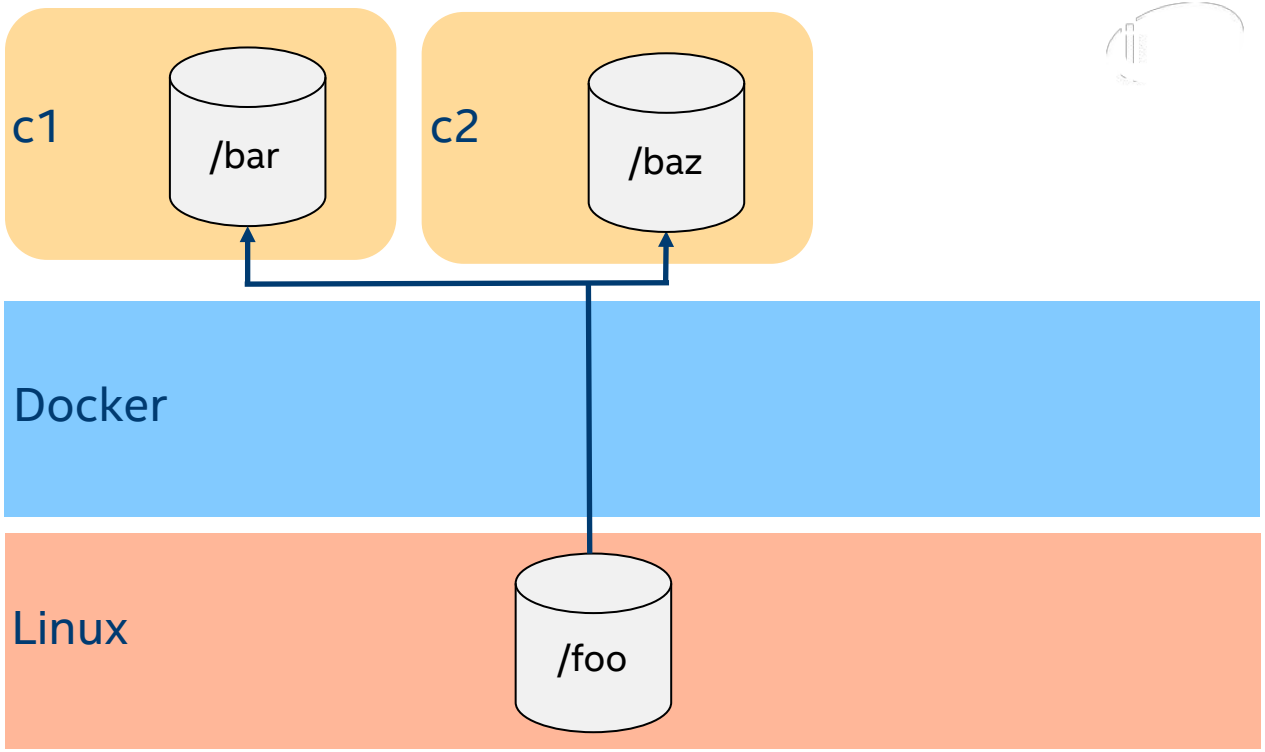
Running the containers on the macOS™ operating system

DIFFERENCES

- Setup
  - Instructions at https://github.com/crops/docker-win-mac-docs/wiki
- Runs in a hypervisor (intended to be transparent)
- Uses a Docker volume rather than bind mount

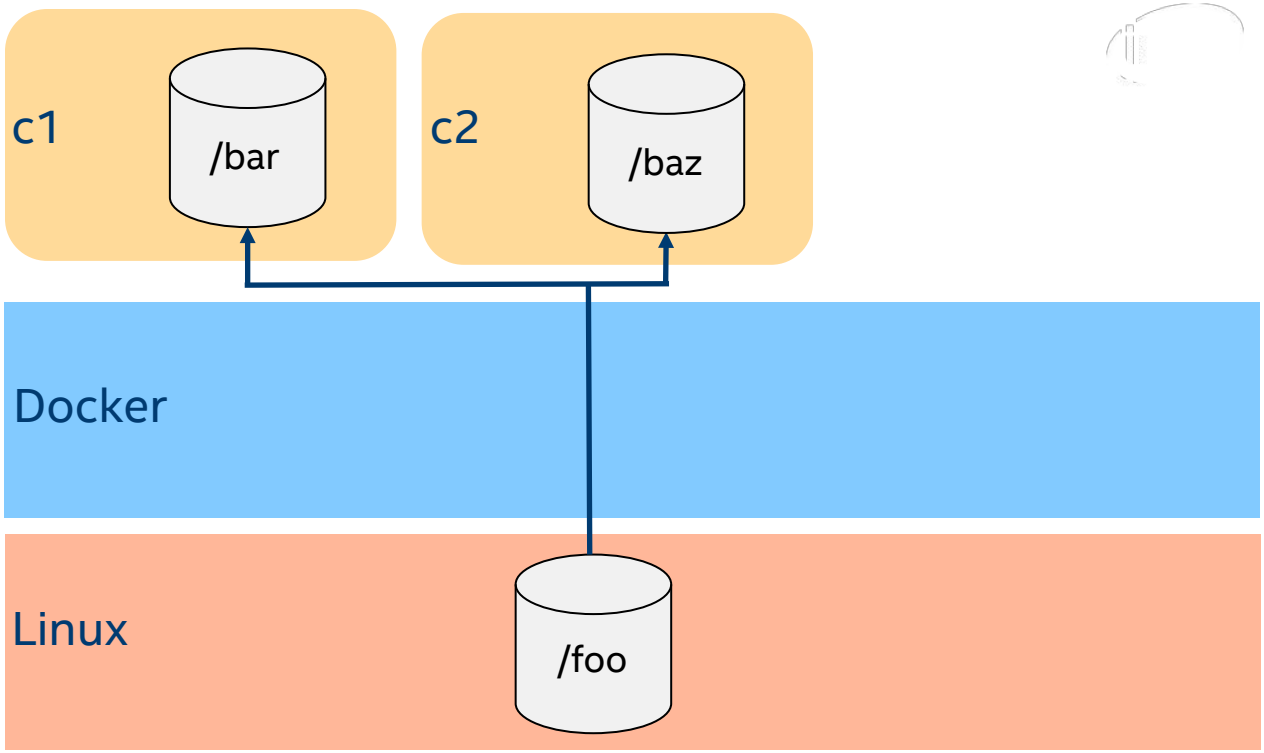
```
docker run -v /foo:/bar c1
```

```
docker run -v /foo:/baz c2
```



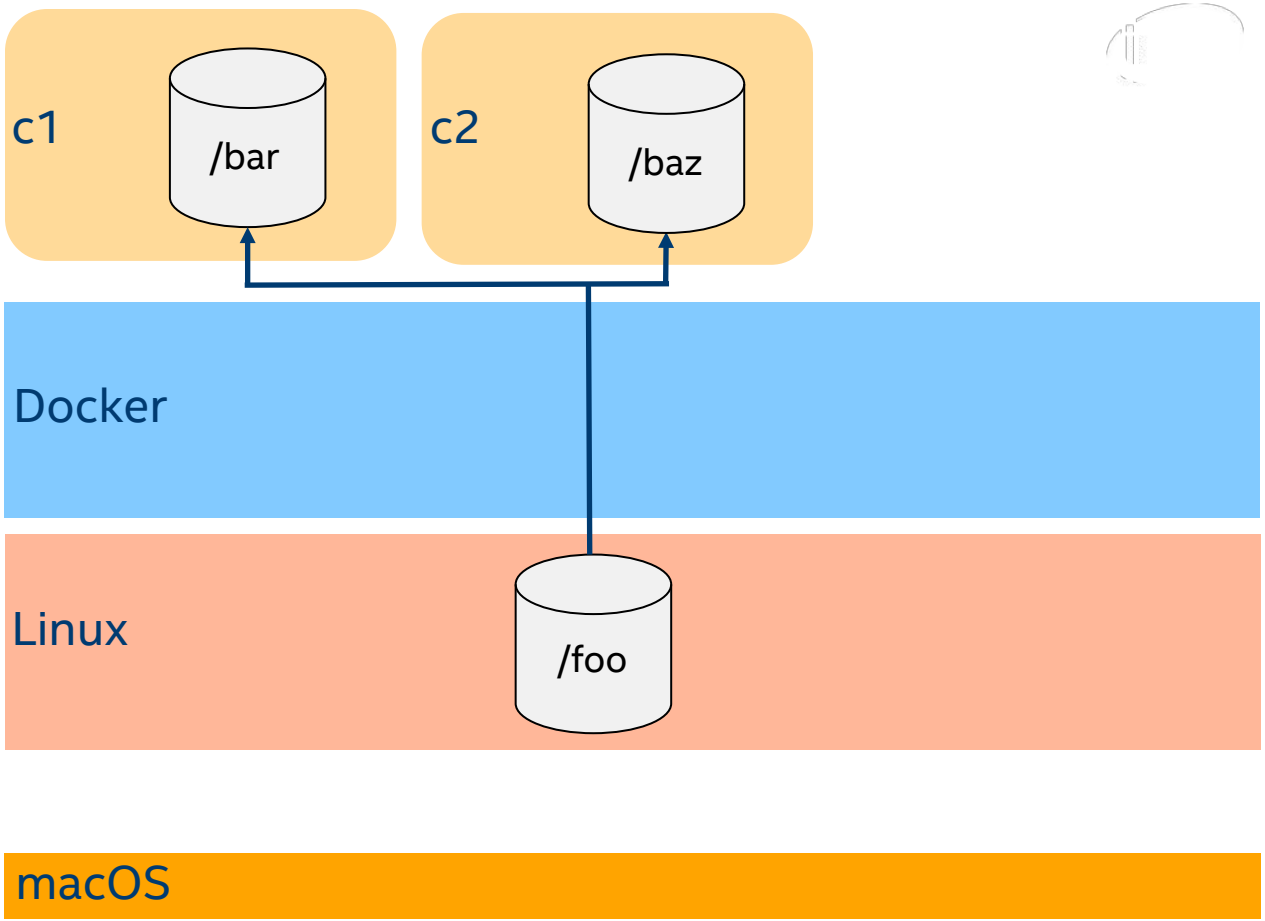
```
docker run -v vol:/bar c1
```

```
docker run -v vol:/baz c2
```



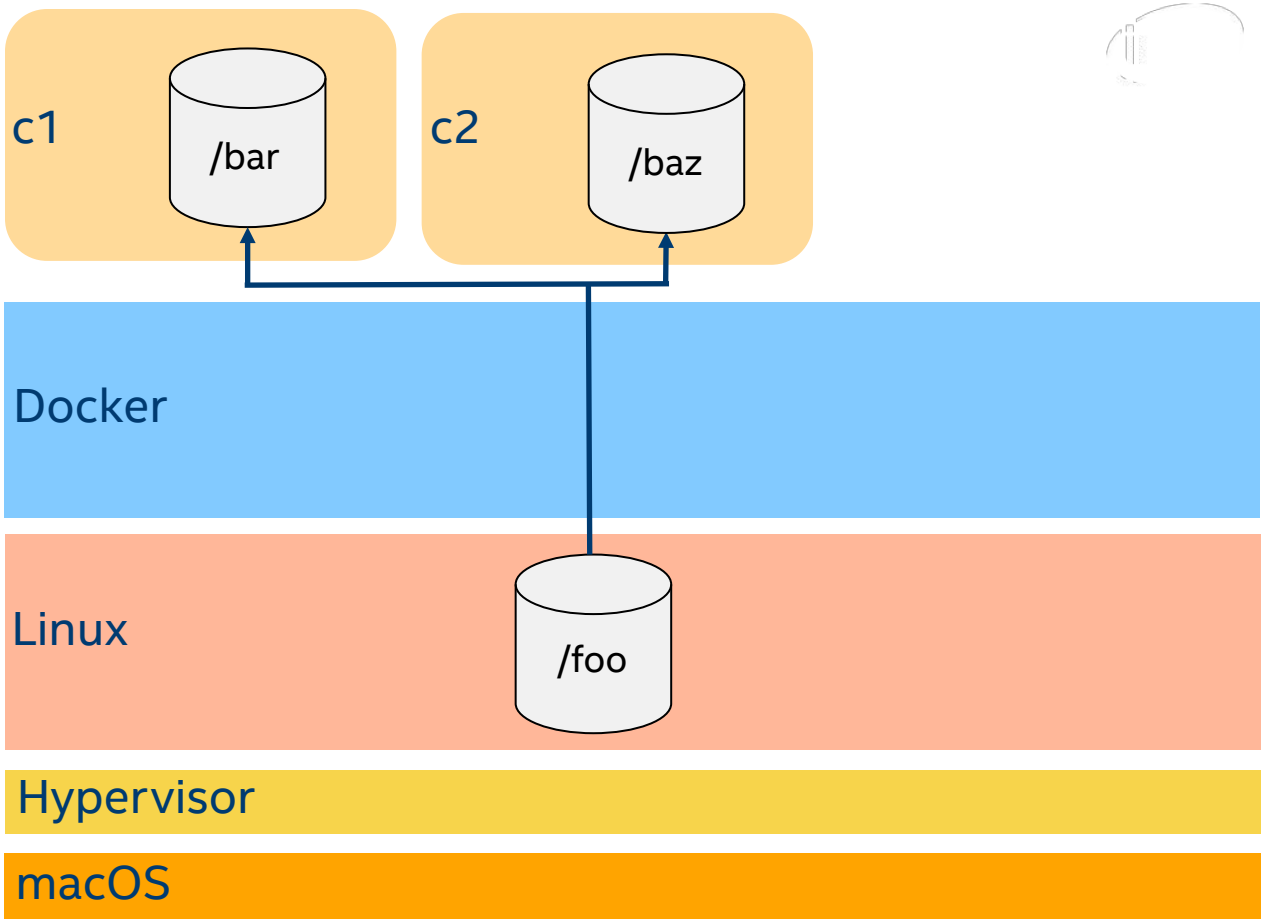
```
docker run -v vol:/bar c1
```

```
docker run -v vol:/baz c2
```



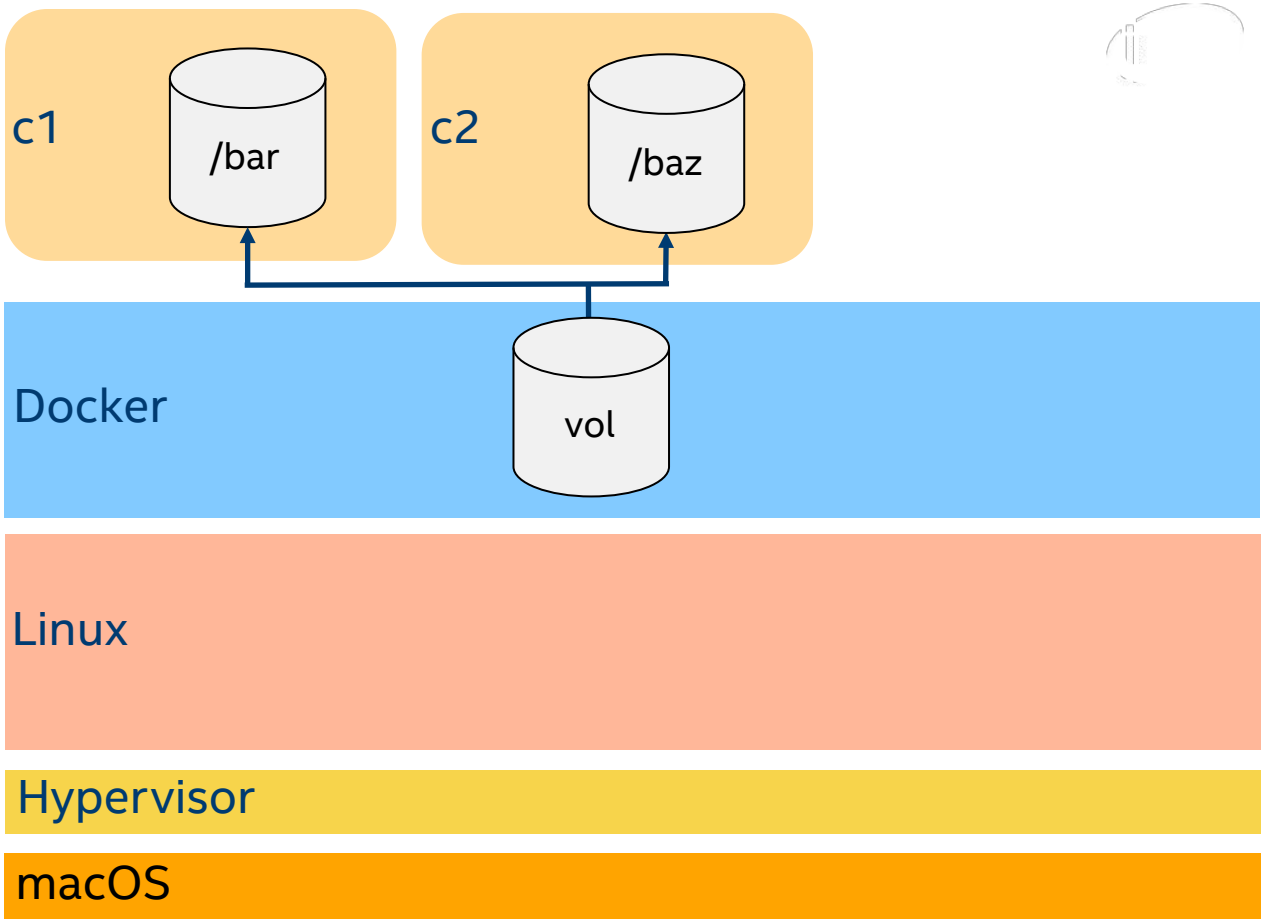
```
docker run -v vol:/bar c1
```

```
docker run -v vol:/baz c2
```



```
docker run -v vol:/bar c1
```

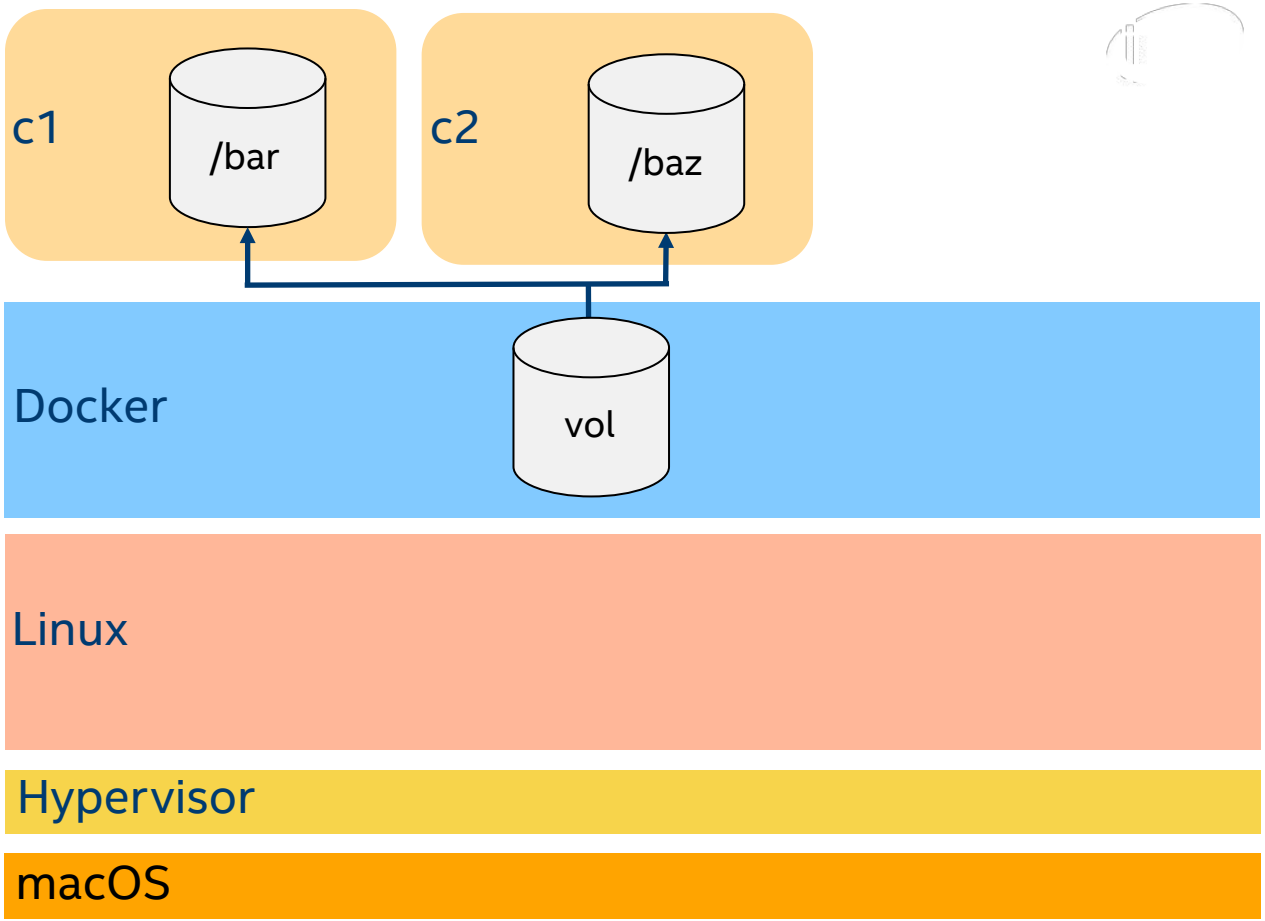
```
docker run -v vol:/baz c2
```



```
docker run -v vol:/bar c1
```

```
docker run -v vol:/baz c2
```

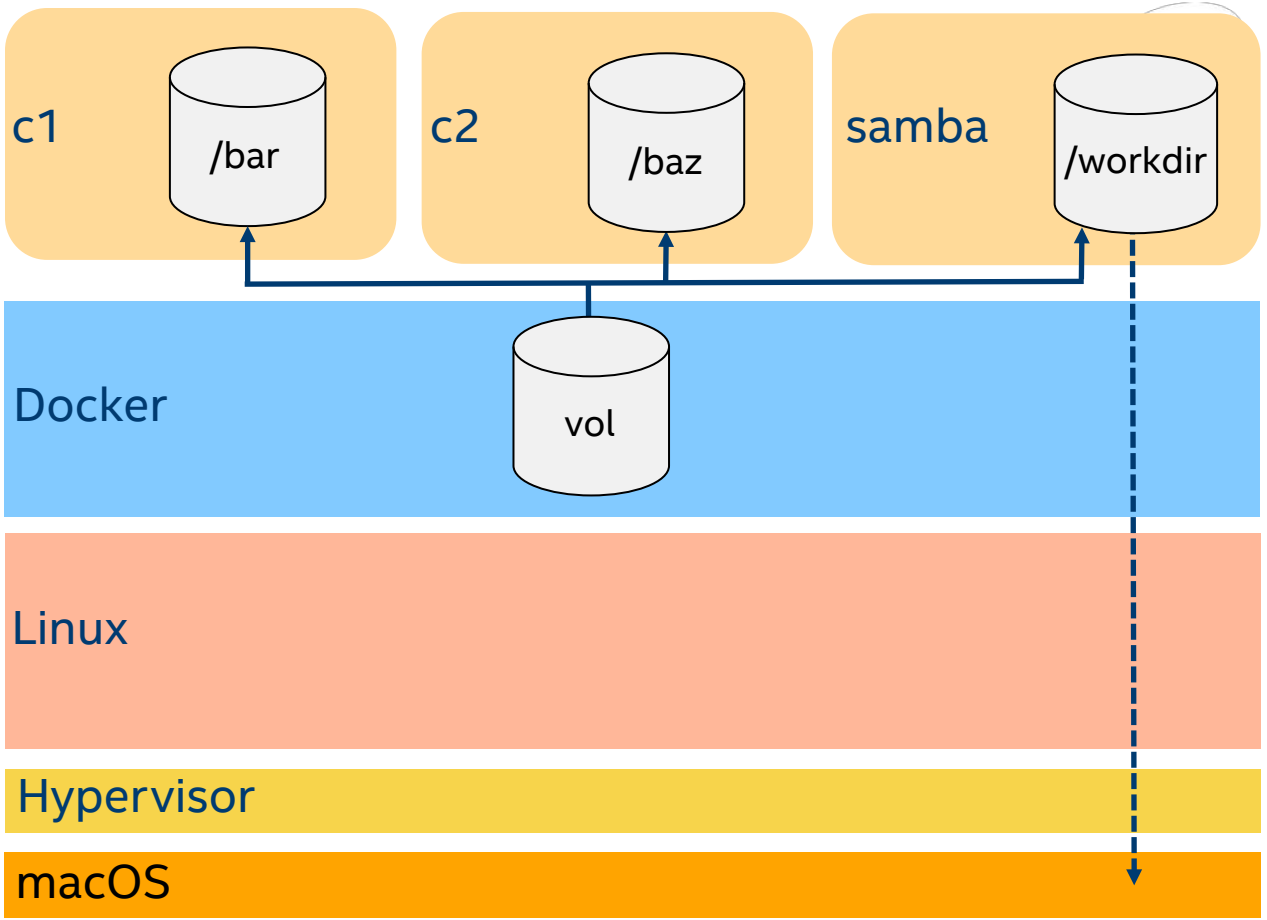
```
docker start samba
```



```
docker run -v vol:/bar c1
```

```
docker run -v vol:/baz c2
```

```
docker start samba
```



OTHER PLATFORM SCREENCAST

https://www.youtube.com/watch?v=w9_Wt6iQK3g


intel
QUESTIONS?
37

MORE INFO

- poky container
  - https://github.com/crops/poky-container
  - https://hub.docker.com/r/crops/poky/
- extsdk container
  - https://github.com/crops/extsdk-container
  - https://hub.docker.com/r/crops/extsdk-container/
- toaster container
  - https://github.com/crops/toaster-container
  - https://hub.docker.com/r/crops/toaster/

CREDITS

macOS™ are registered trademarks of Apple Inc.

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

“Docker and the Docker logo are trademarks or registered trademarks of Docker, Inc. in the United States and/or other countries. Docker, Inc. and other parties may also have trademark rights in other terms used herein.”


intel®