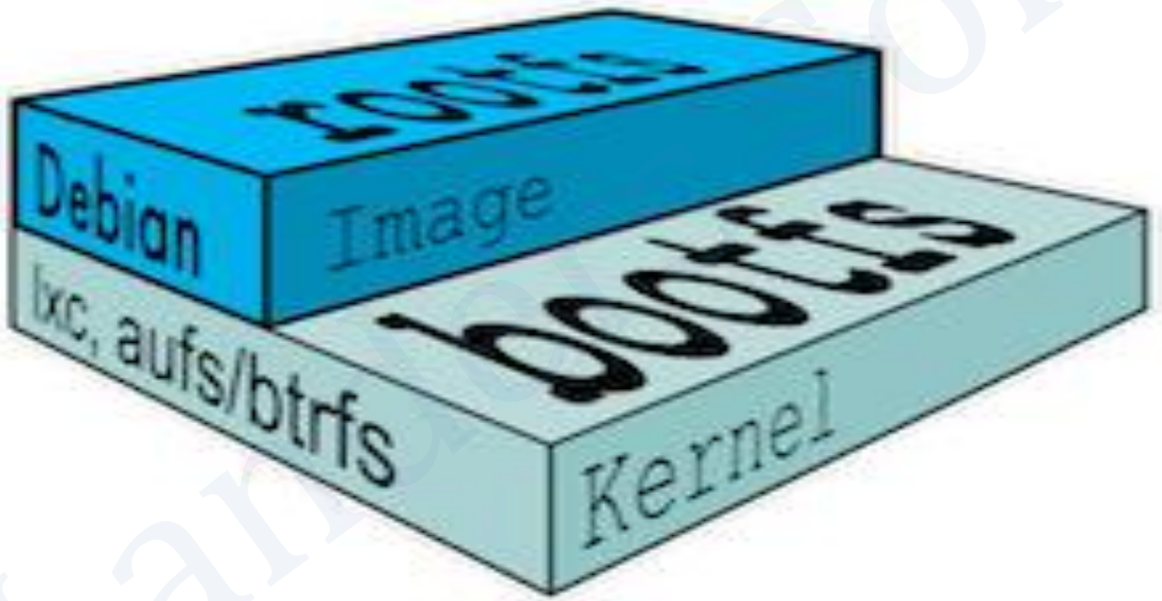
Docker Ports and Volumes

- But how to bound the container with a particulat port:
- docker run -d -p 8080:80 nginx
- Even you can bound multiple ports:
- docker run -d -p 8080:80,8081:443 nginx
- Once done stop and remove the container

Session: 6

Docker - Images

Docker Images



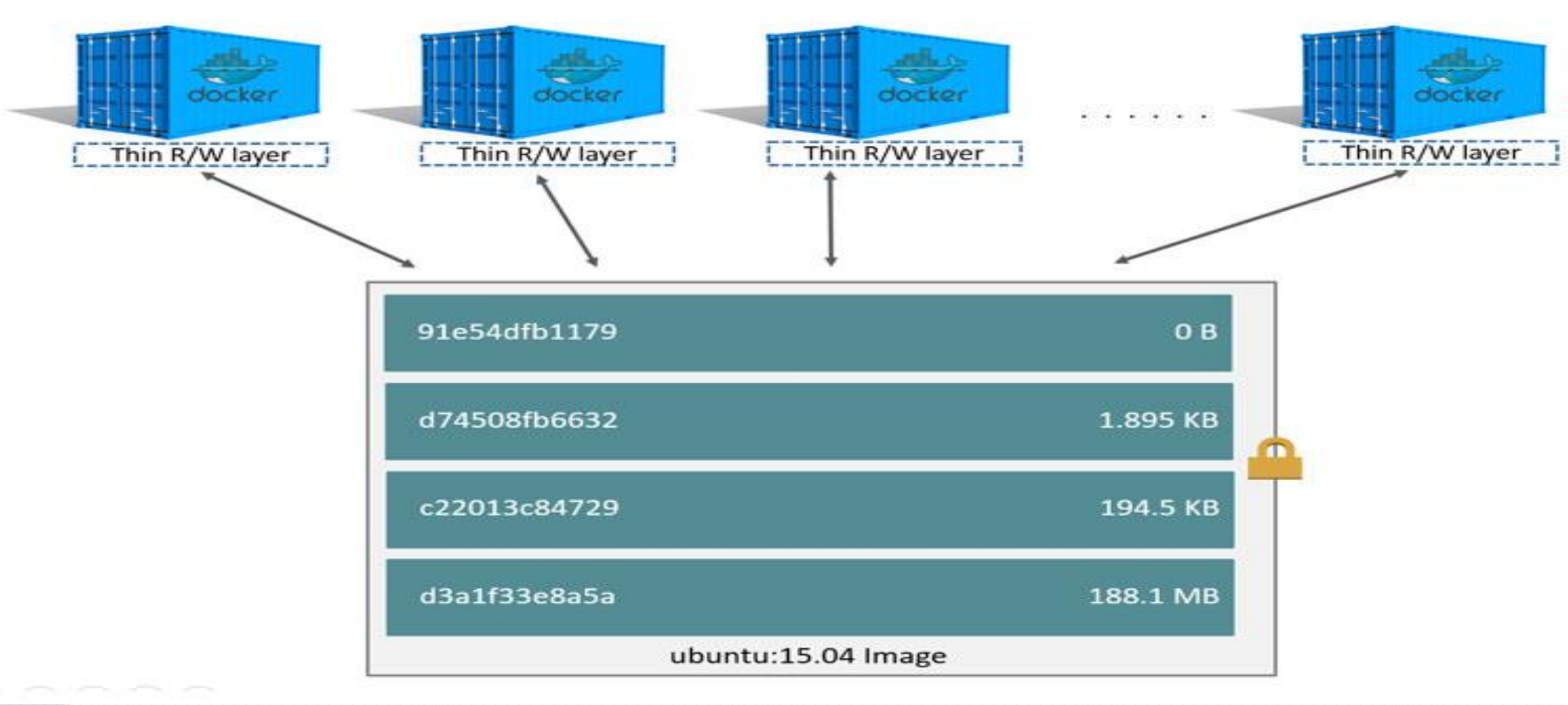
What is an image?

- An image is a collection of files + some meta data.
- Images are made of *layers, conceptually stacked on top of each other.*
- Each layer can add, change, and remove files.
- Images can share layers to optimize disk usage, transfer times, and memory use.

Container vs Image

- An image is a read-only filesystem.
- A container is an encapsulated set of processes running in a read-write copy of that filesystem.
- docker run starts a container from a given image.

Container vs Image



Store & manage images

- Images can be stored:
  - On your Docker host.
  - In a Docker registry.
- You can use the Docker client to download (pull) or upload (push) images.
- To be more accurate: you can use the Docker client to tell a Docker server to push and pull images to and from a registry.
- Lets explore docker public registry called “docker hub”

Showing current images

- Let's look at what images are on our host now.

  docker images

  docker image list

Searching for images

- We cannot list all images on a remote registry, but we can search for a specific keyword:

  docker search zookeeper

- "Stars" indicate the popularity of the image.

Downloading images

- There are two ways to download images.
  - Explicitly, with “docker pull”.
  - Implicitly, when executing “docker run” and the image is not found locally.

- Pulling an image.

  docker pull debian:jessie

- Images can have tags.
- Tags define image versions or variants.
- “docker pull ubuntu” will refer to “ubuntu:latest”.
- The :latest tag is generally updated often.

Docker Image LifeCycle

- To list docker images: docker images
- To remove docker images locally: docker rmi <image-name>
- Special Cases:
- Lets try to remove docker images used by current container: docker rmi <image-name>
- This time you will get an error, but the image can be removed forcefully using “-f” option.
- Docker rmi -f <image-name> (Note: It wont work with Image ID)

Docker Image Information

- Docker Image details:
- docker image list

- Detailed Infromation about docker image:
- ls –lrt /var/lib/docker/image/overlay2/imagedb/content/sha256

- High level certificate sign information (SHA):
- cat /var/lib/docker/image/overlay2/repositories.json

- All details with command line:
- docker image inspect <image-name>

- Even you can check all of the information about the docker image:
- docker history <image-id>

Docker Image and Conatiner LifeCycle

- To list docker: docker ps -a
- To list docker images: docker images
- To remove container: docker rm <container-id> or <Name>
- To remove multiple container: docker rm `docker ps –a –q`
- To remove docker images locally: docker rmi <image-name>

Docker Image and Conatiner LifeCycle

- Special Cases:
- Lets try to remove docker images used by current container: docker rmi <image-name>
- This time you will get an error, but the image can be removed forcefully using “-f” option.
- Docker rmi -f <image-name> (Note: It wont work with Image ID)
- The best part is though you have removed the image forcefully, but this will not impact the current container as the container preserves the metadata in containers folder.
- You containers are safe!.

Session: 7

Building Images

Building Images Interactively

- Let’s have a Use Case:
  - We will build an image that has httpd.
  - First, we will do it manually with docker commit.
  - Then, we will use a Dockerfile and “docker build”.

Create a new container

- Let's start from base image "centos":
  docker run -dit --name c1 centos:7

  docker attach c1
  yum update –y

  yum install –y httpd
  exit

- Inspect the changes:
  docker diff <yourContainerId>

- Commit the changes:

docker commit -m "added httpd and updated" -a "raman khanna" c1 ramancentosimage:v1

Dockerfile overview

- A Dockerfile is a build recipe for a Docker image.
- It contains a series of instructions telling Docker how an image is constructed.
- The “docker build” command builds an image from a Dockerfile.

First Dockerfile

- Create a directory to hold our Dockerfile.
  mkdir myimage

- Create a Dockerfile inside this directory.
  cd myimage
  vi Dockerfile

- Write below in our Dockerfile

```
FROM centos:7
RUN yum update -y
RUN yum -y install httpd
```

First Dockerfile

- “FROM” indicates the base image for our build.
- Each “RUN” line will be executed by Docker during the build.
- No input can be provided to Docker during the build.

First Dockerfile

- Build the Dockerfile:

  docker build -t httpd . Or docker build -t httpd /home/ubuntu/raman/myimage/

- -t indicates the tag to apply to the image.

- . indicates the location of the Directory of Dockerfile.

Run & Tag the image

- Let’s run the new images:
  docker run -it <newImageId>
  rpm –qa | grep –i httpd

Using Image & viewing history

- The history command lists all the layers composing an image.
- For each layer, it shows its creation time, size, and creation command.
- When an image was built with a Dockerfile, each layer corresponds to a line of the Dockerfile.

  docker history httpd

Dockerfile

- Dockerfile:

```
FROM centos:7
MAINTAINER  Raman Khanna raman.khanna@TechLanders.com
RUN mkdir /data
RUN yum update -y
RUN yum -y install httpd   php
RUN echo " TechLanders Solutions Deals in DevOps and Cloud" > /var/www/html/index.html
EXPOSE 80
VOLUME  /data
RUN echo "httpd" >> /root/.bashrc
CMD ["/bin/bash"]
```

- Build the image:

```
docker build -t webapp:v1 .

docker run -dit --name c1 -p 8080:80 webapp:v1

curl 172.31.84.13:8080
```

Browse in browser as well

COPY Instruction

- For Use Case, let's build a container that copy file from localhost

  echo " TechLanders Solutions Deals in DevOps and Cloud " > /home/ubuntu/image/index.html

Dockerfile content

```
FROM centos:7
RUN yum update -y
RUN yum install -y httpd
COPY ./index.html /var/www/html/index.html
EXPOSE 80
WORKDIR /var/www/html
CMD ["httpd","-D","FOREGROUND"]
```

COPY Instruction

- For Use Case, let's build a container that compiles a basic "Hello world" program in C.
- hello.c

```
[root@TechLanders yogesh]# cat hello.c
#include<stdio.h>
int main () {
      puts("Hello, TechLanders!");
      return 0;
}
```

- Dockerfile

```
[root@TechLanders yogesh]# cat Dockerfile
FROM ubuntu
RUN apt-get update
RUN apt-get install -y build-essential
COPY hello.c /
RUN make hello
CMD /hello
```

Note: Using COPY keyword we can copy the files from Docker Host to a container.

COPY Instruction

- docker build -t helloworld:v1 .

- docker run -it –name prodv4 helloworld:v1 /bin/bash

- root@519a9d815a29:/# ls -lrt /hello
- -rwxr-xr-x. 1 root root 8600 Sep 18 11:54 /hello

- root@519a9d815a29:/# ./hello
- Hello, world!