

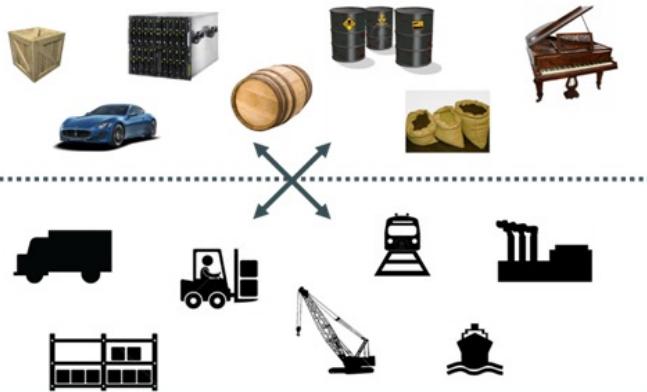
# My Experience with Docker

Premanand Achuthan

# Docker Introduction

## Cargo Transport Pre-1960

Multiplicity of methods for transporting/storing



Do I worry about how goods interact (e.g., coffee beans next to spices)  
Can I transport quickly and smoothly (e.g., from boat to truck)



## Docker is a shipping container system for code

Multiplicity of stacks

Static website User DB Web frontend Queue Analytics DB

An engine that enables any payload to be encapsulated as a lightweight, portable, self-sufficient container...

Multiplicity of hardware environments

Development VM

QA server

Customer Data Center

Public Cloud

Production Cluster

Contributor's laptop

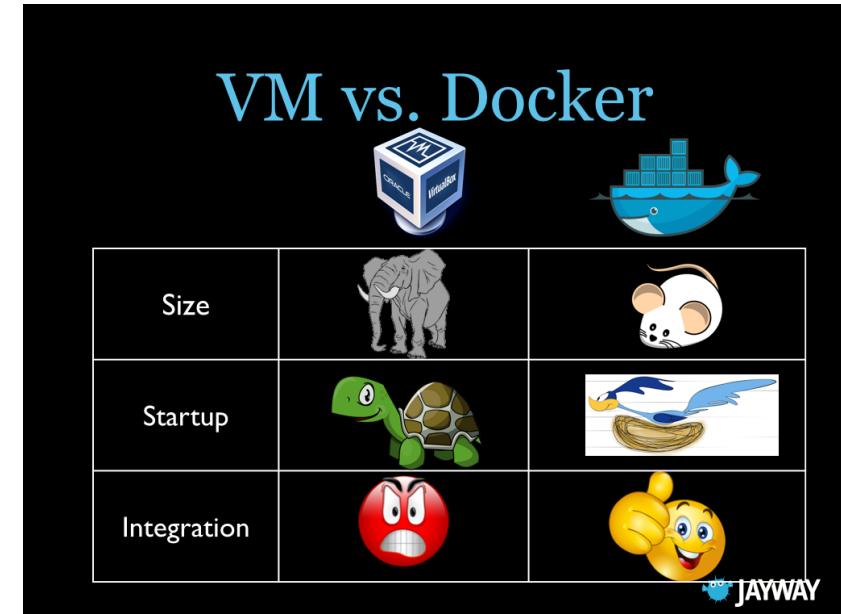
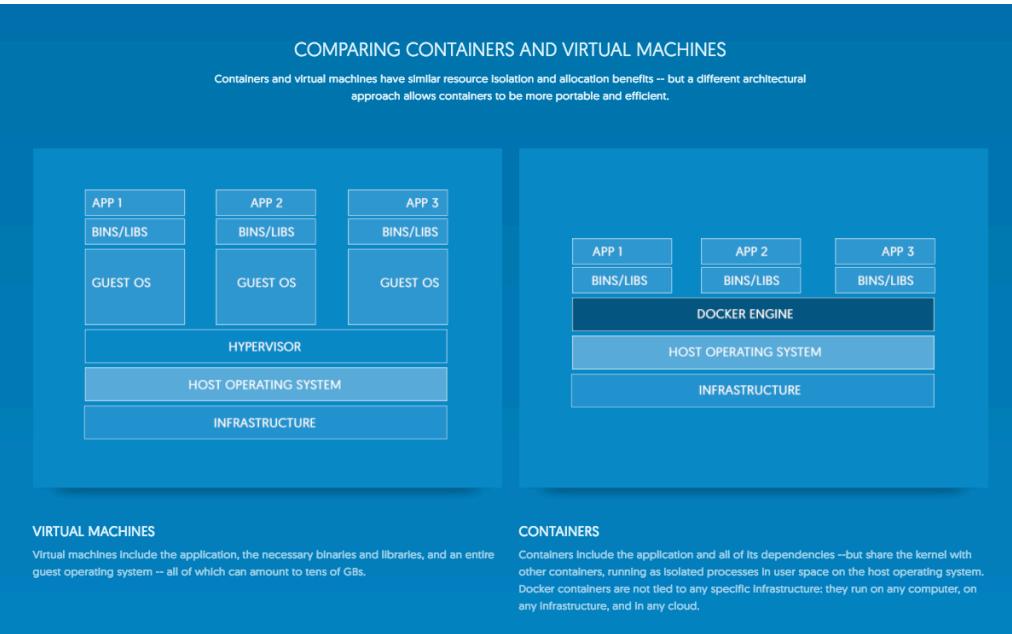


Do services and apps interact appropriately?

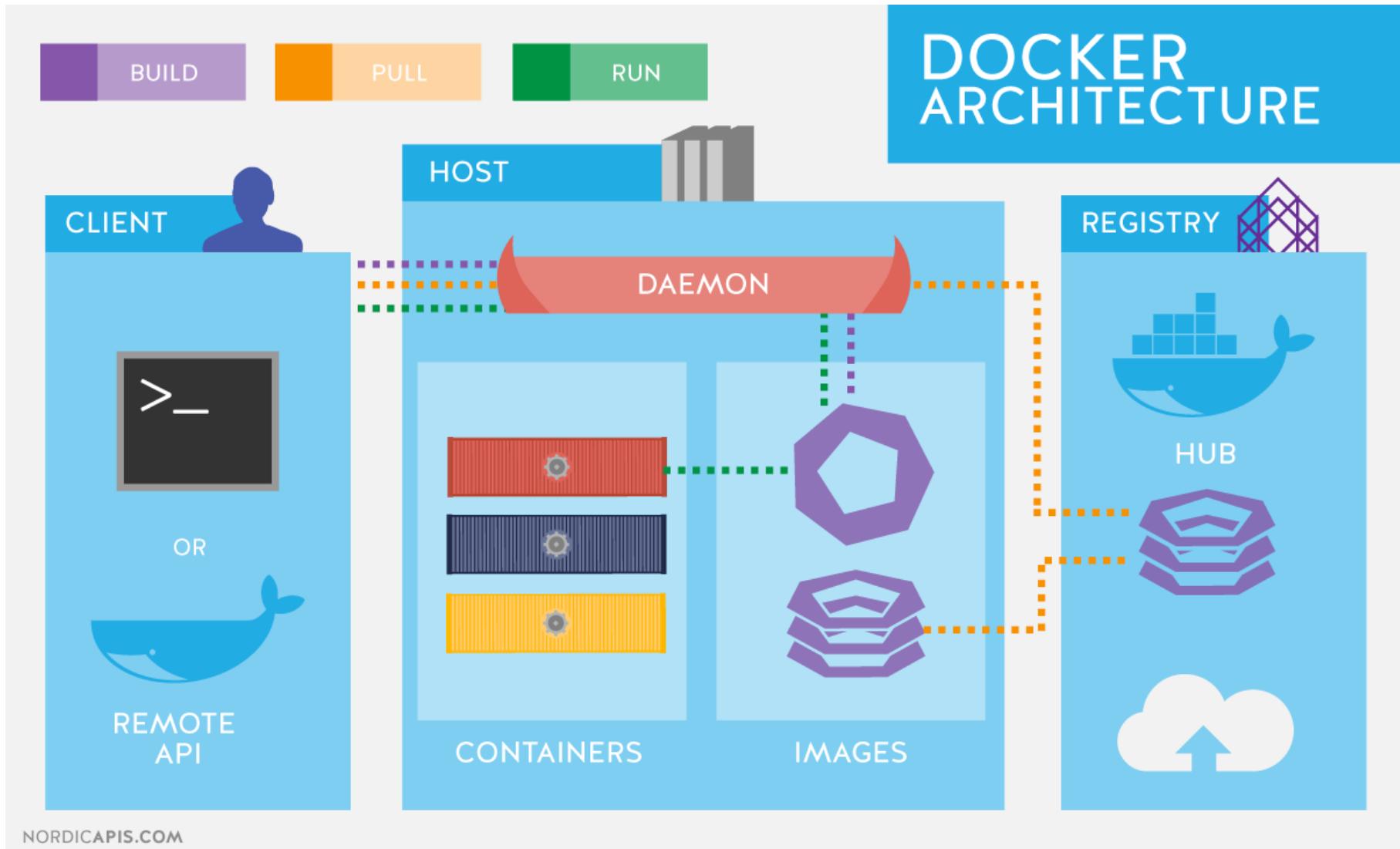
Can I migrate smoothly and quickly



# VM vs Docker



An *image* is a filesystem and parameters to use at runtime.  
A *container* is a running instance of an image.



# Docker eco-system



# Hello world



```
$ docker run ubuntu /bin/echo 'Hello world'
```

```
Hello world
```

**Hello world ubuntu container**

```
$ docker run -t -i ubuntu /bin/bash  
root@af8bae53bdd3:/#
```

**Interactive container**

```
root@af8bae53bdd3:/# pwd  
  
/  
  
root@af8bae53bdd3:/# ls  
  
bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var
```

```
$ docker run -d ubuntu /bin/sh -c "while true; do echo hello world; sleep 1; done"  
1e5535038e285177d5214659a068137486f96ee5c2e85a4ac52dc83f2ebe4147
```

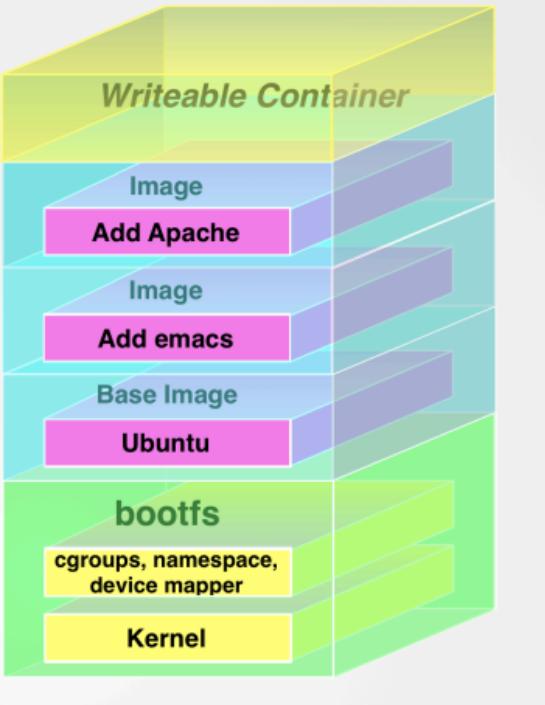
**Demonized container**

# Hello world

An *image* is a filesystem and parameters to use at runtime.  
A *container* is a runtime instance of an image.

```
prem-ml:docker_trials prem$ docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
prem-ml:docker_trials prem$ docker ps -a
CONTAINER ID    IMAGE          COMMAND       CREATED        STATUS        PORTS     NAMES
prem-ml:docker_trials prem$ docker run ubuntu /bin/echo 'Hello world'
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu

aed15891ba52: Pull complete
773ae8583d14: Pull complete
d1d48771f782: Pull complete
cd3d6cd6c0cf: Pull complete
8ff6f8a9120c: Pull complete
Digest: sha256:35bc48a1ca97c3971611dc4662d08d131869daa692acb281c7e9e052924e38b1
Status: Downloaded newer image for ubuntu:latest
Hello world
prem-ml:docker_trials prem$ docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
ubuntu          latest        e4415b714b62   7 days ago    128.1 MB
prem-ml:docker_trials prem$ docker ps -a
CONTAINER ID    IMAGE          COMMAND       CREATED        STATUS        PORTS     NAMES
1c19e1f5129a    ubuntu         "/bin/echo 'Hello wor"   37 seconds ago   Exited (0) 36 seconds ago
prem-ml:docker_trials prem$ docker run ubuntu /bin/echo 'Hello world'
Hello world
prem-ml:docker_trials prem$ docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
ubuntu          latest        e4415b714b62   7 days ago    128.1 MB
prem-ml:docker_trials prem$ docker run ubuntu /bin/echo 'Hello world'
Hello world
prem-ml:docker_trials prem$ docker run ubuntu /bin/echo 'Hello world'
Hello world
prem-ml:docker_trials prem$ docker run ubuntu /bin/echo 'Hello world'
Hello world
prem-ml:docker_trials prem$ docker ps -a
CONTAINER ID    IMAGE          COMMAND       CREATED        STATUS        PORTS     NAMES
72baaa18f8e     ubuntu         "/bin/echo 'Hello wor"   5 seconds ago   Exited (0) 4 seconds ago
34c742d78f14    ubuntu         "/bin/echo 'Hello wor"   9 seconds ago   Exited (0) 8 seconds ago
58cb4db5fc88    ubuntu         "/bin/echo 'Hello wor"   16 seconds ago  Exited (0) 15 seconds ago
e34d66353af7    ubuntu         "/bin/echo 'Hello wor"   3 minutes ago  Exited (0) 3 minutes ago
1c19e1f5129a    ubuntu         "/bin/echo 'Hello wor"   4 minutes ago  Exited (0) 4 minutes ago
prem-ml:docker_trials prem$ docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
ubuntu          latest        e4415b714b62   7 days ago    128.1 MB
prem-ml:docker_trials prem$
```

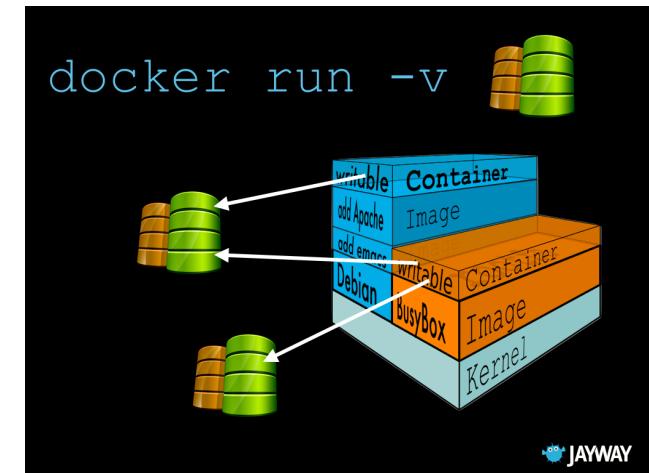


# Docker Storage

- Each image is stored as read-only layer
- Each layer of the FS is mounted on top of prior layers
- The first layer is the base image
- The top layer is the only modifiable layer – it's termed the container
- Image and containers are managed by the Docker storage driver. Driver types (e.g. aufs, overlay, vfs, zfs)
- Containers are ephemeral

## Persistence achieved through volume:

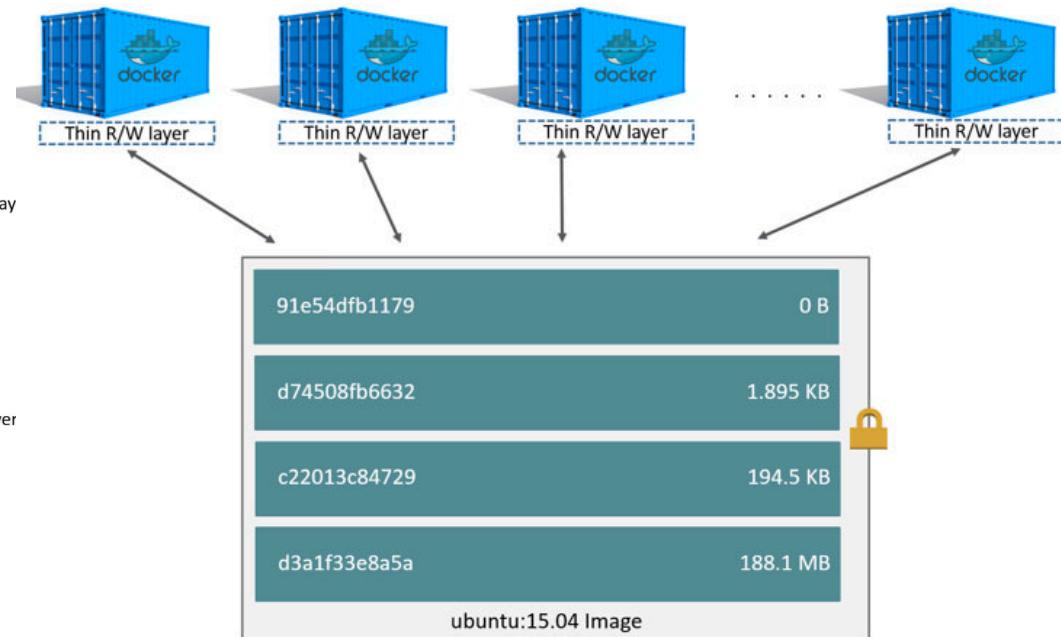
A volume is simply a directory on the Docker host that exists outside of the directory structure managed by the storage driver.



# Docker Storage - Copy-on-write (COW)

```

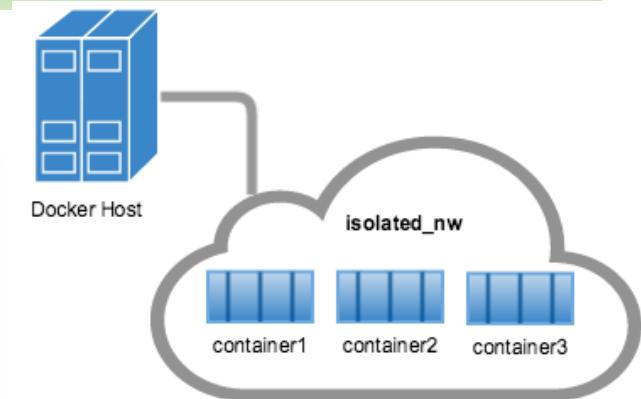
prem-m1:docker_trials prem$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
prem-m1:docker_trials prem$ docker ps -a
CONTAINER ID IMAGE COMMAND STATUS PORTS NAMES
c1d36cc6e8cf: Pull complete
8f1f6f8a8120c: Pull complete
Digest: sha256:65bcb4841ca97c3971611dc4662d88d131869da9692acb281c7e9e052924e38b1
Status: Up 36 seconds ago
Hello world
prem-m1:docker_trials prem$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
ubuntu latest e4415b714b62 7 days ago 128.1 MB
prem-m1:docker_trials prem$ docker ps -a
CONTAINER ID IMAGE COMMAND STATUS PORTS NAMES
1fb909559503: Pull complete
prem-m1:docker_trials prem$ docker run ubuntu /bin/echo 'Hello world'
Hello world
prem-m1:docker_trials prem$ docker run ubuntu /bin/echo 'Hello world'
Hello world
prem-m1:docker_trials prem$ docker run ubuntu /bin/echo 'Hello world'
Hello world
prem-m1:docker_trials prem$ docker run ubuntu /bin/echo 'Hello world'
Hello world
prem-m1:docker_trials prem$ docker ps -a
CONTAINER ID IMAGE COMMAND STATUS PORTS NAMES
773ae853d314: Pull complete
34c742ed5114: Pull complete
5bcb4d05fc08: Pull complete
e34de6633a7f: Pull complete
1fb909559503: Pull complete
prem-m1:docker_trials prem$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
ubuntu latest e4415b714b62 7 days ago 128.1 MB
prem-m1:docker_trials prem$ 
  
```



# Docker Networking

```
docker run --network=isolated_nw -itd --name=container3 busybox  
docker run -d -p 80:5000 training/webapp
```

```
trackhub-registry --- root@e87446fa0609: / --- bash --- 158:45  
  
prem-ml:trackhub-registry prem$ docker network ls  
NETWORK ID      NAME      DRIVER      SCOPE  
bd465f32c065   bridge    bridge      local  
1cbece81dd6   dockerelk_docker_elk  bridge      local  
6da10f79d6d3   dockerthr_default  bridge      local  
4c81f446bcaa   host      host      local  
245d284d93d9   none     null      local  
ff8939ac7a24   thrdocker_default  bridge      local  
prem-ml:trackhub-registry prem$ docker network inspect bridge  
[  
 {  
   "Name": "bridge",  
   "Id": "bd465f32c0659190e1e04f4a451bf0ef5fe3fea99e19cc47de58faee443e442",  
   "Scope": "local",  
   "Driver": "bridge",  
   "EnableIPv6": false,  
   "IPAM": {  
     "Driver": "default",  
     "Options": null,  
     "Config": [  
       {  
         "Subnet": "172.17.0.0/16",  
         "Gateway": "172.17.0.1"  
       }  
     ]  
   },  
   "Internal": false,  
   "Containers": {},  
   "Options": {  
     "com.docker.network.bridge.default_bridge": "true",  
     "com.docker.network.bridge.enable_icc": "true",  
     "com.docker.network.bridge.enable_ip_masquerade": "true",  
     "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",  
     "com.docker.network.bridge.name": "docker0",  
     "com.docker.network.driver.mtu": "1500"  
   },  
   "Labels": {}  
 }  
]  
prem-ml:trackhub-registry prem$
```



# Dockerizing THR

The Track Hub Registry   Submit data   Documentation • About   Help   Enter the search terms:   Sign up   Login

## The Track Hub Registry

A global centralised collection of publicly accessible track hubs

The goal of the Track Hub Registry is to allow third parties to advertise [track hubs](#), and to make it easier for researchers around the world to discover and use track hubs containing different types of genomic research data.

Search by keywords: hg19, epigenomics, mouse ...

**① Submit Data**

I want maximum visibility for my track hubs.

External track hub providers can register and submit their track databases to the registry. [Registration](#) is web-based and done on this site; submission happens programmatically via our RESTful API. Once submitted and successfully validated, the track dbs become available for search by other users worldwide, allowing for automatic and rapid integration into a genome browser.

[How to Submit](#)

**Access Data**

How do I find omics tracks for an assembly of my favourite organism?

Track hubs can be searched based on metadata information. Free text [search](#) is provided from the search box in the header of all track hub Registry web pages and in the middle of this page. Advanced search options are available for more specific and customised searches.

[Help on Advanced Search](#)



# Docker Compose - Single Host - multiple services

```
version: '2'  
services:  
  elasticsearch:  
    image: elasticsearch:latest  
    ports:  
      - "9201:9200"  
    volumes:  
      - ./elasticsearch/data:/usr/share/elasticsearch/data
```

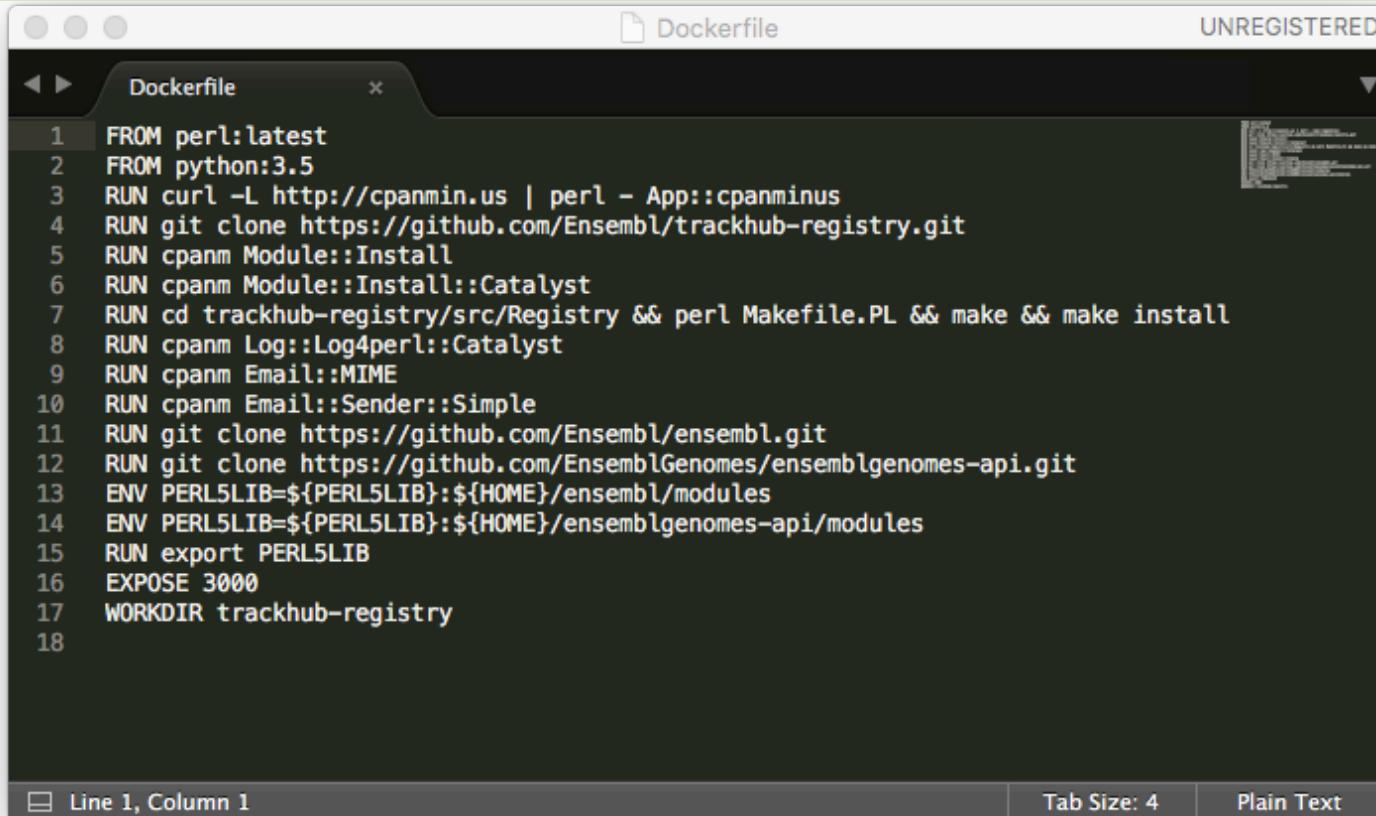


```
[prem-ml:thr_docker prem$ docker-compose up  
Starting thrdocker_elasticsearch_1  
Attaching to thrdocker_elasticsearch_1  
elasticsearch_1 | [2016-11-25T14:23:58,297][INFO ] [o.e.n.Node] [ ] initializing ...  
elasticsearch_1 | [2016-11-25T14:23:58,582][INFO ] [o.e.e.NodeEnvironment] [vRQqMjq] using [1] data paths, mounts [[/usr/share/elasticsearch/data (osxfs)]]  
, net usable_space [103gb], net total_space [232.6gb], spins? [possibly], types [fuse.osxfs]  
elasticsearch_1 | [2016-11-25T14:23:58,583][INFO ] [o.e.e.NodeEnvironment] [vRQqMjq] heap size [1.9gb], compressed ordinary object pointers [true]
```

```
[prem-ml:docker_trials prem$ curl -XGET 'http://prem-ml:9201/_cat/indices?v'  
health status index      uuid                                pri rep docs.count docs.deleted store.size pri.store.size  
yellow open   test      mZlBP5jqQT0gFTcBfrrbKA  5   1       15          1   259.8kb     259.8kb  
yellow open trackhubs_v1 J0Q4W7vqQna08heSiO_oTQ  5   1      3515         508   69mb       69mb  
yellow open  users_v1   FUz1bnbfSUahCgzpheAf3A  5   1       34          0   50.7kb     50.7kb  
prem-ml:docker_trials prem$ docker ps -a  
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS          NAMES  
a8f2b165d10b        elasticsearch:latest "/docker-entrypoint.s" 55 minutes ago   Up 22 minutes   9300/tcp, 0.0.0.0:9201->9200/tcp   thrdocker_elasticsearch_1  
e87446fa0609        ubuntu              "/bin/bash"        2 hours ago     Exited (0) 2 hours ago   sleepy_lalande  
726baaa18f8e        ubuntu              "/bin/echo 'Hello wor" 21 hours ago    Exited (0) 2 hours ago   hopeful_hawking  
34c742d78f14        ubuntu              "/bin/echo 'Hello wor" 21 hours ago    Exited (0) 21 hours ago  awesome_goldwasser  
58cb4db5fcbb        ubuntu              "/bin/echo 'Hello wor" 21 hours ago    Exited (0) 21 hours ago  desperate_einstein  
e34d66353af7        ubuntu              "/bin/echo 'Hello wor" 21 hours ago    Exited (0) 21 hours ago  infallible_blackwell  
1c19e1f5129a        ubuntu              "/bin/echo 'Hello wor" 21 hours ago    Exited (0) 21 hours ago  thirsty_leavitt  
prem-ml:docker_trials prem$ docker exec -it thrdocker_elasticsearch_1 bash  
[root@a8f2b165d10b:/usr/share/elasticsearch# curl -XGET 'http://localhost:9200/_cat/indices?v'  
health status index      uuid                                pri rep docs.count docs.deleted store.size pri.store.size  
yellow open   test      mZlBP5jqQT0gFTcBfrrbKA  5   1       15          1   259.8kb     259.8kb  
yellow open trackhubs_v1 J0Q4W7vqQna08heSiO_oTQ  5   1      3515         508   69mb       69mb  
yellow open  users_v1   FUz1bnbfSUahCgzpheAf3A  5   1       34          0   50.7kb     50.7kb
```

# Dockerfile - THR

A Dockerfile is a text document that contains all the commands you would normally execute manually in order to build a Docker image



The screenshot shows a code editor window titled "Dockerfile". The file content is a Dockerfile with the following commands:

```
1 FROM perl:latest
2 FROM python:3.5
3 RUN curl -L http://cpanmin.us | perl - App::cpanminus
4 RUN git clone https://github.com/Ensembl/trackhub-registry.git
5 RUN cpanm Module::Install
6 RUN cpanm Module::Install::Catalyst
7 RUN cd trackhub-registry/src/Registry && perl Makefile.PL && make && make install
8 RUN cpanm Log::Log4perl::Catalyst
9 RUN cpanm Email::MIME
10 RUN cpanm Email::Sender::Simple
11 RUN git clone https://github.com/Ensembl/ensembl.git
12 RUN git clone https://github.com/EnsemblGenomes/ensemblgenomes-api.git
13 ENV PERL5LIB=${PERL5LIB}:${HOME}/ensembl/modules
14 ENV PERL5LIB=${PERL5LIB}:${HOME}/ensemblgenomes-api/modules
15 RUN export PERL5LIB
16 EXPOSE 3000
17 WORKDIR trackhub-registry
18
```

The editor has a dark theme and includes status bars at the bottom showing "Line 1, Column 1", "Tab Size: 4", and "Plain Text".

```
docker build -t prem/thr_master_docker .
```

# Docker-compose (THR + elastic)

```
docker-compose.yml ×  
1 version: '2'  
2 services:  
3   thr_registry:  
4     image: prem/thr_master_docker:latest  
5     ports:  
6       - "3000:3000"  
7     links:  
8       - elasticsearch  
9     entrypoint: sh /usr/src/app/entrypoint.sh  
10    volumes:  
11      - /usr/src/app/thregistry  
12      - ./thregistry/entrypoint.sh:/usr/src/app/entrypoint.sh  
13      - ./thregistry/registry.conf:/trackhub-registry/src/Registry/registry.conf  
14      - ./thregistry/registry_testing.conf:/trackhub-registry/src/Registry/registry_testing.conf  
15  elasticsearch:  
16    image: elasticsearch:1.7.6  
17    command: elasticsearch -Des.network.host=0.0.0.0  
18    ports:  
19      - "9200:9200"  
20      - "9300:9300"  
21    volumes:  
22      - ./elasticsearch_1_7/data:/usr/share/elasticsearch/data  
23    container_name: es_thr_17  
24  
25  
26
```

entrypoint.sh

./src/Registry/script/registry\_server.pl

**DEMO**

# Development to Production

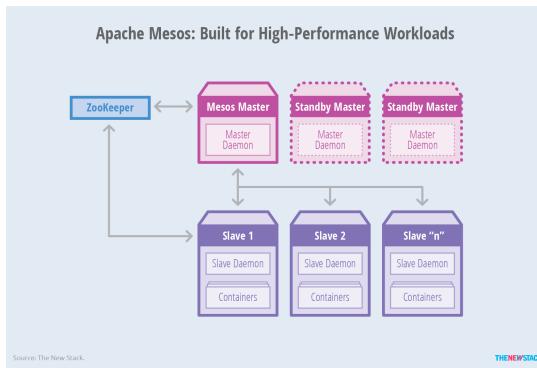
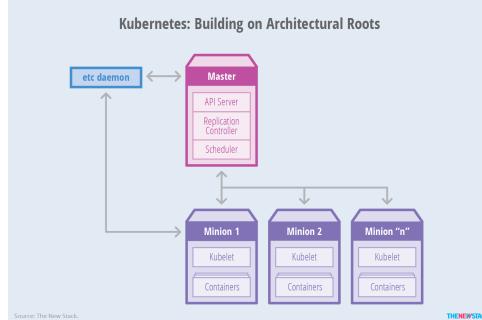
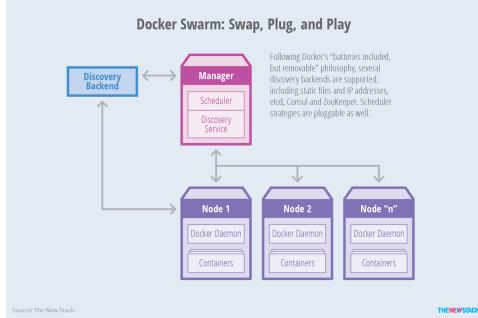
## Requirements

- Developer set up as identically to production as possible.
- A complete flow from developer laptop to staging and production.

## Challenges:

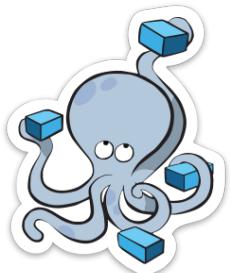
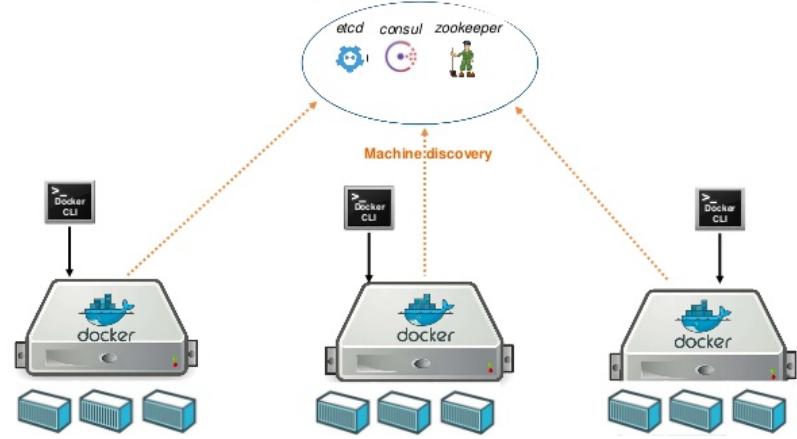
- Image management : Continuous deployment pipeline – code commit to Docker image to Docker compose
- Container's root user and Docker host's root user
- Cluster provisioning
- Service Discovery
- Network Access and Security
- Load Balancing across containers and hosts
- Managing differences in volume bindings, port binding, logs etc.,
- Container monitoring

# Orchestration



## Service Discovery

### Multi-host networking



# Conclusion

## Challenges:

Deployment into production requires some effort initially  
Container management, networking, security etc.,  
Other tools are needed for large workflow automation.

## Benefits:

Better deployment strategy  
Breakdown monolithic application to tiny single-minded applications  
Portable – Multi-cloud platform deployment  
Reusable components  
Continuous Deployment and Testing  
Public registry for sharing containers  
Rapidly growing Docker container ecosystem

