




# Как не стать рабом облака?

Paas 2.0 с Docker

<sup>В</sup>  
**infobox**  cloud

**Юрий Трухин**

cloud computing expert | InfoboxCloud  
ytrukhin@infobox.ru

<http://infoboxcloud.ru>

# О чем поговорим?

- Зачем все это???
- Контейнеры
- Как работает Docker
- Как использовать Docker
- Dockerfile
- Новое в Docker 1.9
- Docker Compose
- Docker Swarm
- Как и где установить и попробовать
- UI для Docker – надо ли?



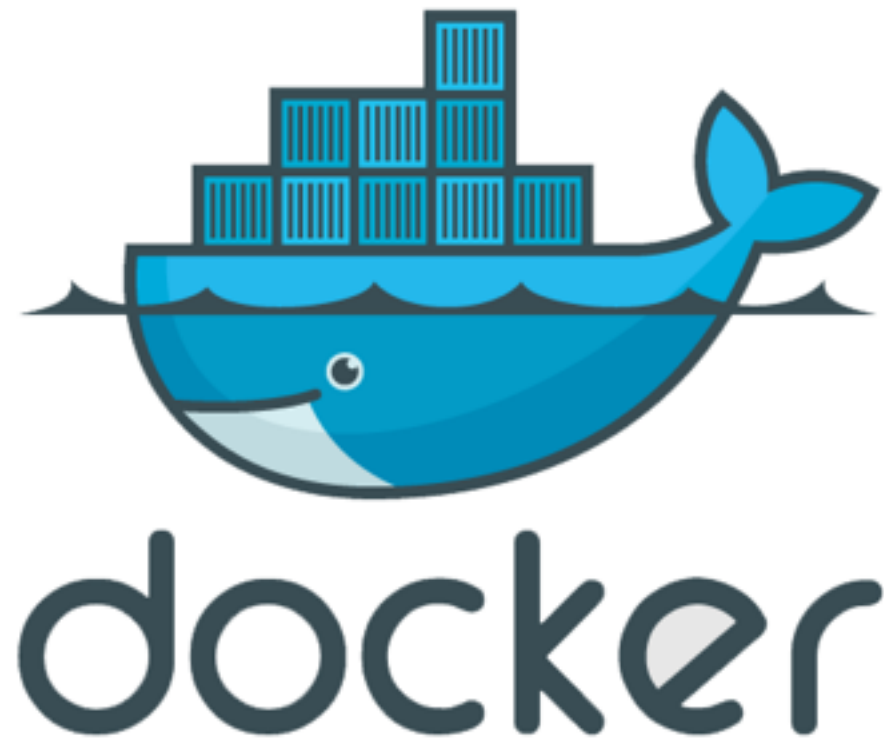
# Контейнеры

До 1960х годов большинство грузов перевозилось вперемешку

Что будет, если наковальни положить на мешки с бананами?

1/2 времени перевозки - погрузка, разгрузка, перегрузка

Сегодня 18 миллионов контейнеров - 90% мировой торговли



Контейнеры внутри компьютера

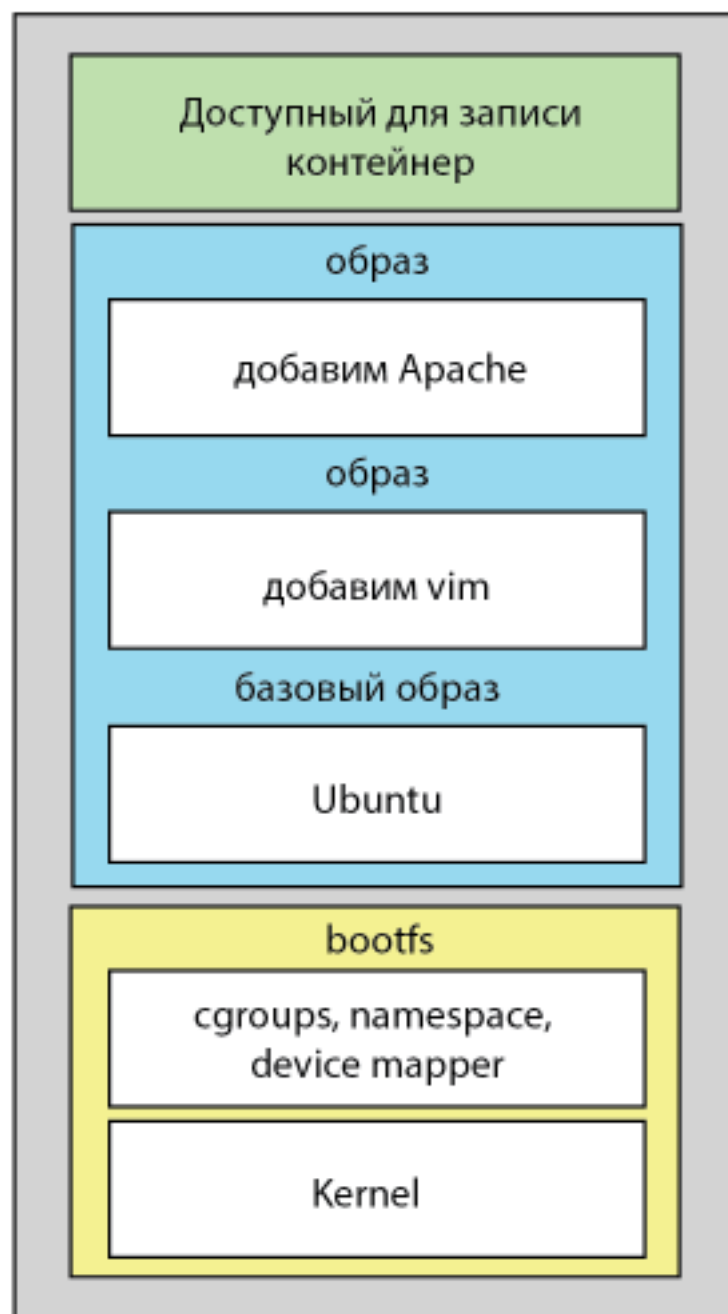
Практически любое приложение может быть упаковано в контейнер

**Приложение работает у пользователя так же, как и у разработчика**

Требуется Linux Kernel 3.8 и выше



# Как работает Docker



Клиент-серверное приложение. docker cli и Rest API  
bootfs -> rootfs read only -> fs на запись  
реестры: публичные ([hub.docker.com](https://hub.docker.com)) и приватные  
контейнеры: при старте слой записи пуст  
# docker info

```
trukhinyuri — root@dc: ~ — ssh — 73x12
Your InfoboxCloud hosted server greets you!
Don't hesitate to drop us a message should any issues or concerns arise.
We are here to help at help@infobox.ru
root@dc:~# docker info
Containers: 0
Images: 0
Storage Driver: aufs
  Root Dir: /var/lib/docker/aufs
  Dirs: 0
Execution Driver: native-0.2
Kernel Version: 3.13.0-35-generic
root@dc:~#
```

# Работа с контейнерами

```
sudo docker run -i -t ubuntu /bin/bash
```

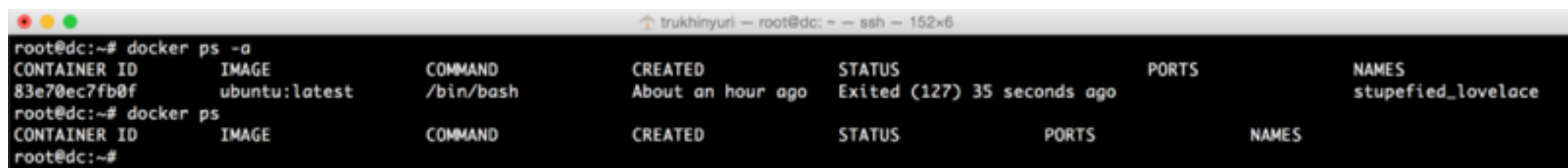
Флаг `-i` оставляет STDIN открытым, даже, когда вы не присоединены к контейнеру. Флаг `-t` назначает псевдо-`tty` контейнеру. Таким образом создается интерактивный интерфейс к контейнеру. Так же мы указываем название образа (`ubuntu` — базовый образ) и шелл `/bin/bash`.

## Подключение к контейнеру

```
docker exec -it container_name bash
```

## Список запущенных контейнеров

```
docker ps
```



```
root@dc:~# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS   NAMES
83e70ec7fb0f   ubuntu:latest  /bin/bash               About an hour ago  Exited (127) 35 seconds ago           stupefied_lovelace
root@dc:~# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS   NAMES
root@dc:~#
```

## Контейнер-демон

```
docker run --name city -d ubuntu /bin/bash -c "while true; do echo hello world; sleep 1; done"
```

## Перемещение данных между контейнером и хостом

```
docker cp <путь к данным на хосте> <имя контейнера>:<путь>
```

## Монтируем диски

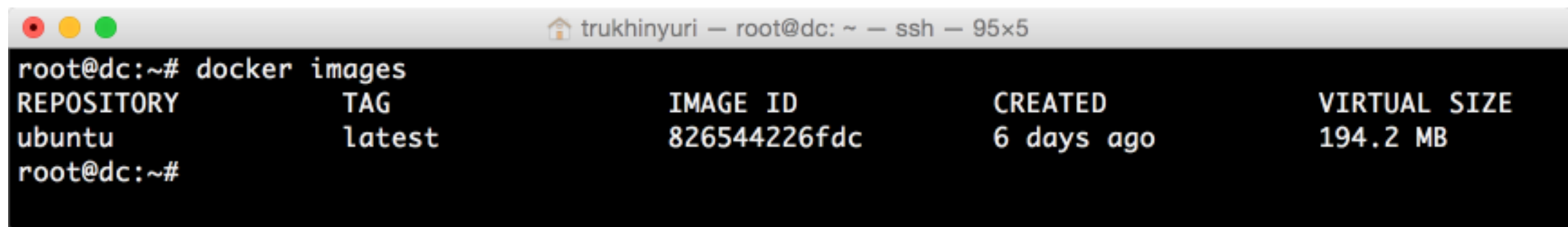
теперь... и в обратную сторону!

```
docker run -v /tmp:/root -t -i <имя образа>
```

# Переезд контейнеров

## Список образов на хосте

docker images



REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL SIZE
ubuntu	latest	826544226fdc	6 days ago	194.2 MB

## Коммит изменений в образ

docker commit <id контейнера> <имя образа>

## Сохраним образ в файл

docker save имя\_образа > ~/transfer.tar

## Восстановим на другом хосте

docker load < /tmp/transfer.tar

## Отправим образ на hub.docker.com

docker push trukhinyuri/nginx

# Dockerfile

DSL с инструкциями для построения образов Docker

```
# Version: 1.0
FROM gitlab/gitlab-ce:latest
MAINTAINER Yuri Trukhin <yuri@trukhin.com>
ENV REFRESHED_AT 2015.11.08.004
ENV GITLAB_SHELL_SSH_PORT 8005
RUN apt-get update
RUN apt-get -y upgrade
COPY dotssh/config ~/.ssh/config
EXPOSE 80
EXPOSE 443
EXPOSE 22
```

`docker build -t trukhinyuri/nginx ~/static_web`

## Можно прямо из Git

`docker build -t trukhinyuri/nginx \ git@github.com:trukhinyuri/docker-static_web`

## Отключение кеша сборок

`docker build --no-cache -t trukhinyuri/nginx .`



# Отладка Dockerfile

Можно создать контейнер из предпоследнего шага где упало и посмотреть в чем дело

```
trukhinyuri — root@(none): ~/static_web — ssh — 68x21

root@(none):~/static_web# docker build -t trukhinyuri/nginx .
Sending build context to Docker daemon 2.56 kB
Sending build context to Docker daemon
Step 0 : FROM ubuntu:14.04
---> 9cbaf023786c
Step 1 : MAINTAINER Yuri Trukhin <trukhinyuri@infoboxcloud.com>
---> Using cache
---> 71bf67ac628a
Step 2 : RUN apt-get update
---> Using cache
---> 1154a8c71f01
Step 3 : RUN apt-get install -y nginx
---> Running in 066b799ea548
Reading package lists...
Building dependency tree...
Reading state information...
E: Unable to locate package nginx
2014/10/16 11:08:46 The command [/bin/sh -c apt-get install -y nginx]
returned a non-zero code: 100
root@(none):~/static_web#
```

```
docker run -i -t 066b799ea548 /bin/bash
```

```
docker logs <container_id>
```

```
docker logs --tail 10 <container_id>    Последние 10 строк
```

```
docker logs --tail 0 -f <container_id>    Что в логе сейчас
```

# Шаблонизация с помощью кеша сборок

Обновлять только нужный слой

```
FROM ubuntu:14.04
MAINTAINER Yuri Trukhin <trukhinyuri@infoboxcloud.com>
ENV REFRESHED_AT 2014-10-16
RUN apt-get -qq update
```

# Инструкции Dockerfile: запуск процесса

CMD `["/bin/bash", "-l"]`      Какую команду запустить при старте контейнера

ENTRYPOINT `["/usr/sbin/nginx"]`      То же самое, но нельзя перегружать

Параметр передается процессу в Entrypoint

```
docker run -d trukhinyuri/static_web -g "daemon off"
```

Можно использовать вместе:

ENTRYPOINT `["/usr/sbin/nginx"]`

CMD `["-h"]`

# Инструкции Dockerfile: откуда и кто

WORKDIR – откуда запускать CMD

```
WORKDIR /opt/webapp/db  
RUN bundle install  
WORKDIR /opt/webapp  
ENTRYPOINT [ "rackup" ]
```

USER – пользователь, от которого должен быть запущен образ

```
USER user  
USER user:group  
USER uid  
USER uid:gid  
USER user:gid  
USER uid:group
```



# Инструкции Dockerfile: данные

VOLUME ["/opt/project"]

**Точка монтирования тома. (пробрасываются через Union File System)**

ADD software.lic /opt/application/software.lic

ADD http://wordpress.org/latest.zip /root/wordpress.zip

ADD latest.tar.gz /var/www/wordpress/

**Добавляет файлы и папки из билд-окружения в образ. Источник - URL, файл или директория**

COPY conf.d/ /etc/apache2/

То же самое, но только для локальных данных

**Отложенные триггеры, выполнятся когда образ используется как базовый для другого образа**

```
ONBUILD ADD . /app/src
ONBUILD RUN cd /app/src && make
```

**Как смонтировать всю файловую систему хоста в папку контейнера (не стоит так делать)**

```
docker run -v /:/host ubuntu:ro ls /my_host
```

# Инструкции Dockerfile: коммуникация

## Проброс портов

```
docker run -p 127.0.0.1:80:80
```

```
docker run -p 80:80/udp
```

## Линковка

```
docker run -d -P --name web --link db:db trukhinyuri/webapp python app.py
```

## Просмотр переменных окружения

```
$ sudo docker run --rm --name web2 --link db:db training/webapp env
. . .
DB_NAME=/web2/db
DB_PORT=tcp://172.17.0.5:5432
DB_PORT_5432_TCP=tcp://172.17.0.5:5432
DB_PORT_5432_TCP_PROTO=tcp
DB_PORT_5432_TCP_PORT=5432
DB_PORT_5432_TCP_ADDR=172.17.0.5
```

# Рестарт контейнера при перезапуске хоста



Раньше мы писали скрипты для хоста, чтобы рестартовать контейнеры вручную.  
**И однажды снизошло чудо.**

```
docker run --restart=always redis
```

```
docker run --restart=on-failure:5 redis
```

# Поддержка IPv6

**С версии Docker 1.5 можно указать во флаге —ipv6 адрес IPv6**

*А еще сегодня мы запустили IPv6 для всех клиентов InfoboxCloud и VPS от Infobox в Санкт-Петербурге!*

**infobox**  **cloud**  
**IPv6 адреса бесплатно!**



# Новое в Docker 1.9

## Persistent Storage!

```
$ docker volume create -d flocker --name=myvolume  
$ docker run -v myvolume:/data busybox sh -c "echo hello > /data/file.txt"  
$ docker run -v myvolume:/data busybox sh -c "cat /data/file.txt"
```

## Multi-host networking (через consul), пока настраивается грабельно

Позволяет строить приватные сети между физическими и виртуальными хостами

ALM не влияет на возможность соединения

Не обязательно создавать контейнер до указания соединения (в отличие от линковки)

```
$ docker network create frontend  
$ docker run -itd --net=frontend --name web nginx
```

Сети можно создавать отдельно от контейнеров

```
$ docker network create app  
$ docker run -itd --name myapp --net=app <my application container>  
$ docker network connect app web
```

# Docker Compose – просто запускать окружения

```
dspbbalancer:
```

```
  build: .
```

```
  ports:
```

```
    - 80:80
```

```
    - 443:443
```

```
  container_name: dspbbalancer
```

```
  restart: always
```

docker-compose build  
docker-compose up -d

```
web:
```

```
  image: trukhinyuri/apache-php
```

```
  ports:
```

```
    - "80:80"
```

```
    - "443:443"
```

```
  volumes:
```

```
    - ./apache/www:/var/www/
```

```
#    - ./apache/conf/sites-enabled:/etc/apache2/sites-enabled/
```

```
  links:
```

```
    - db
```

```
  restart: always
```

```
db:
```

```
  image: centurylink/mysql:latest
```

```
  volumes:
```

```
    - ./mysql/conf:/etc/mysql/conf.d/
```

```
    - ./mysql/data:/var/lib/mysql/
```

```
  environment:
```

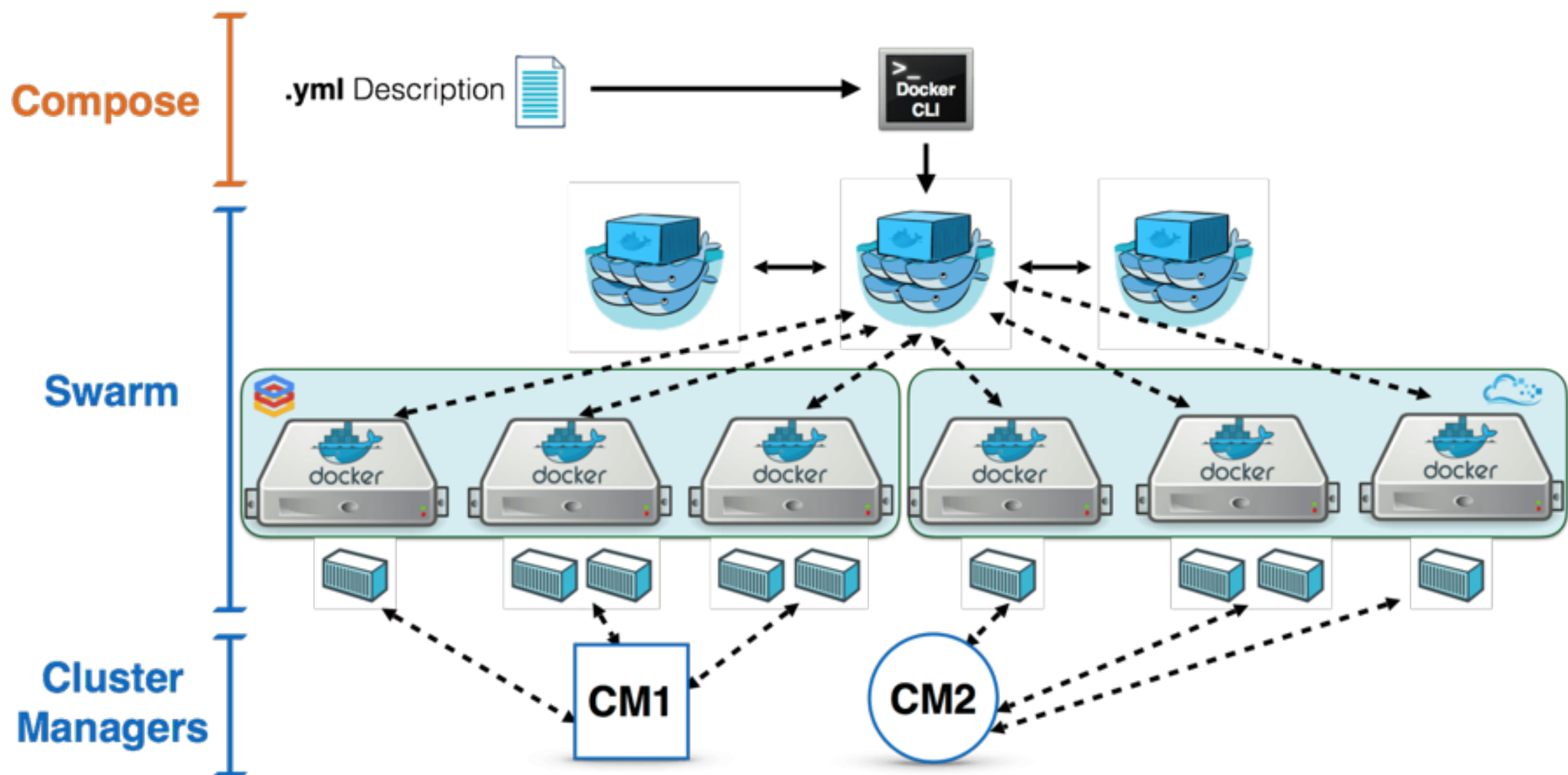
```
    MYSQL_ROOT_PASSWORD: *****
```

```
#    MYSQL_DATABASE: wordpress
```

```
  restart: always
```

docker-compose pull  
docker-compose up -d

# Docker Swarm – кластер из докеров



Провижнинг через docker-machine

Развертывание и масштабирование через Docker Compose

# UI для Docker – Rancher и другие

The screenshot displays the Rancher management interface. At the top, there's a navigation bar with 'APPLICATIONS' and 'INFRASTRUCTURE' tabs. Below this, 'HOSTS' and 'CONTAINERS' are visible. The 'Hosts' section shows four active hosts: 'experiments-ams', 'experiments-msk', 'experiments-spb', and 'Rancher-Management Worker'. Each host card includes its IP address, CPU (4x2.3 GHz or 4x2.5 GHz), memory (3.7 GiB), and disk space (49.1 GiB or 9.7 GiB). The 'Rancher-Management Worker' also shows a container 'root\_rancherse...' with IP 172.17.0.1. Below the hosts, the 'Registries' section shows a single registry 'DockerHub' with state 'ACTIVE', address, email, username, and creation time 'Today at 2:34 PM'.

Host	IP	CPU	Memory	Disk
experiments-ams	77.221.145.39	4x2.3 GHz	3.7 GiB	49.1 GiB
experiments-msk	109.120.186.7	4x2.5 GHz	3.7 GiB	49.1 GiB
experiments-spb	109.120.155.31	4x2.5 GHz	3.7 GiB	49.1 GiB
Rancher-Management Worker	172.17.42.1	2.5 GHz	0.97 GiB	9.7 GiB

Registry	State	Address	Email	Username	Created
DockerHub	ACTIVE				Today at 2:34 PM

Пока делают жизнь сложнее, а не упрощают ее



# Как установить и где попробовать?

<http://infoboxcloud.ru>

“Разрешить управление ядром ОС” // CentOS 7

Развертывание докера и тулов на CentOS 7 упрощено до предела

```
bash <(curl -s http://repository.sandbox.infoboxcloud.ru/scripts/docker/centos7/install.sh)
```

Не помните все команды из доклада? :)

<https://infoboxcloud.ru/community>

В поиске введите docker



<http://infoboxcloud.ru>

# **Q&A: Ваши вопросы?**

(рад на них ответить!)

**Юрий Трухин**

cloud computing expert | InfoboxCloud

[ytrukhin@infobox.ru](mailto:ytrukhin@infobox.ru)