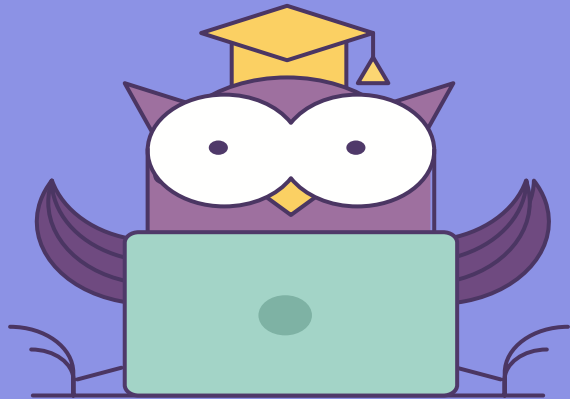





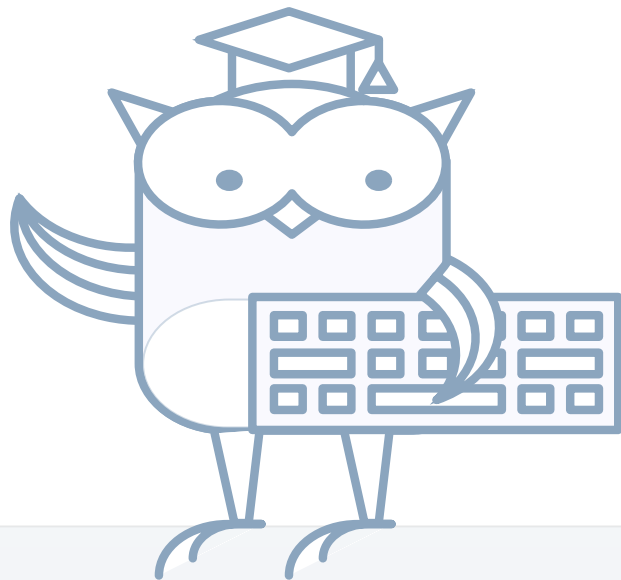
ОНЛАЙН-ОБРАЗОВАНИЕ

Меня хорошо слышно && видно?



Напишите в чат, если есть проблемы!
Ставьте  если все хорошо

Docker Dockerfile



- Контейнеризация, краткий обзор
- Компоненты docker
 - engine, cli, registry
- Сборка контейнеров, dockerfile
- Практика
 - build, run, up, down, pull, push

Существовало достаточно давно

Не было широкого распространения

В определенных случаях была заменена аппаратная виртуализация

Не столько про контейнеры (как технология)

Особенности:

- Абстракция от host-системы

- Легковесные изолированные окружения

- Общие слои файловой системы

- Компоновка и предсказуемость

- Простое управления зависимостями

- Дистрибуцию и тиражируемость

- Стандартизация описания окружения, сборки, деплоя

- 100% консистентная среда приложения

- Воспроизводимость

Docker - это не виртуальная машина!

Это приложение и его зависимости упакованные в окружение.

Принцип работы:

Namespaces

Cgroups

UnionFS

RunC

Изоляция окружения

Индивидуальный namespace для каждого контейнера

Pid, net, mount, etc.

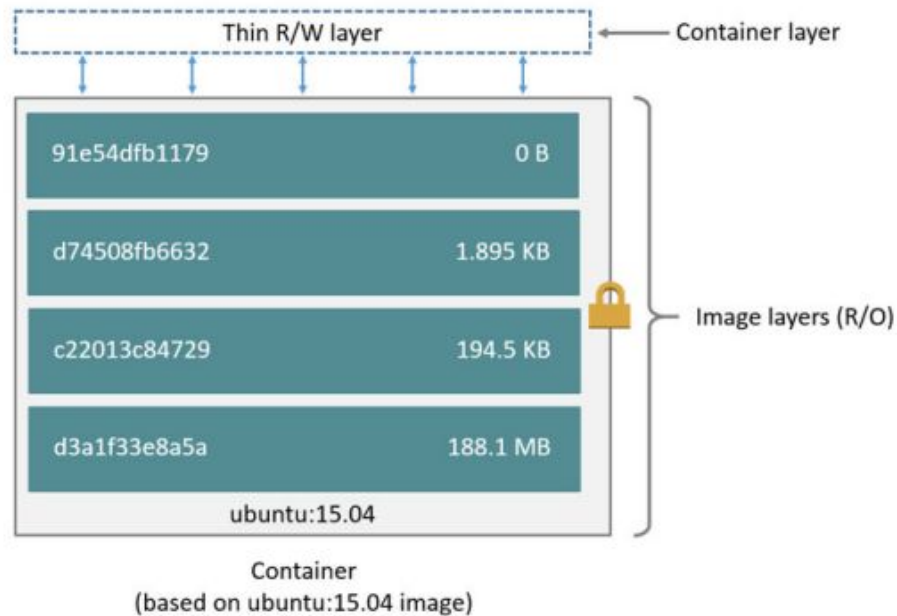
Namespace прекращает свое существование после окончания работы PID1

Использование контейнерами общих ресурсов

Ограничение ресурсов

CPU, memory, IO, etc.

Разделение по слоям
Переиспользование слоев

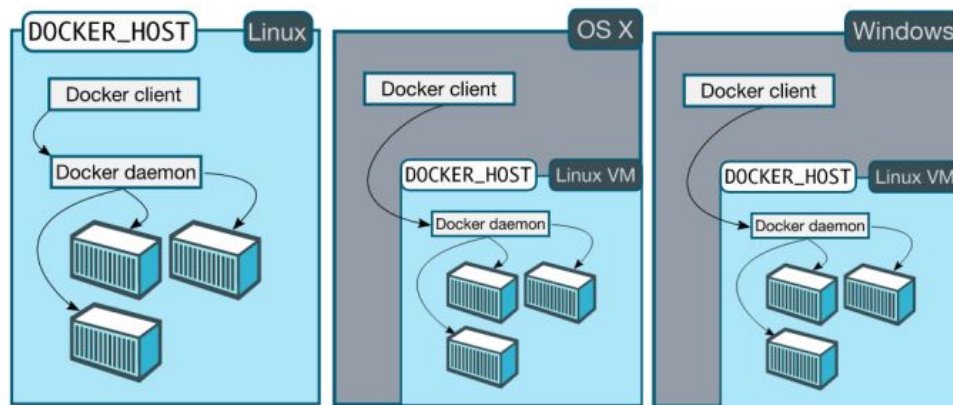


Daemon:

- предоставляет api
- управляет объектами
- взаимодействует с другими daemon`ми

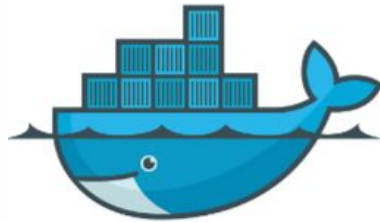
cli:

- принимает команды от пользователя
- взаимодействует с api docker daemon



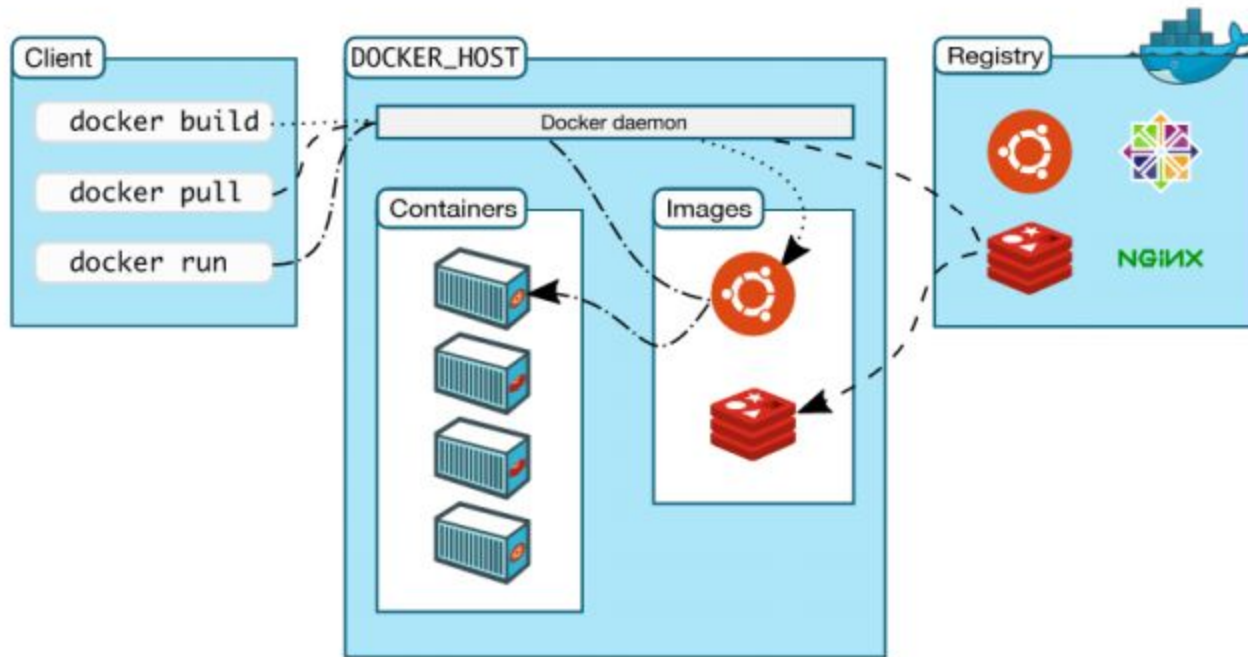
Так называемый приватный “репозиторий” для хранения image.

Docker hub, Private registry, Docker Store



GitLab

Docker engine



```
FROM ubuntu:16.04
RUN apt-get upgrade -y
RUN apt-get install nginx -y
RUN apt-get install wget -y
RUN apt-get install curl -y
RUN apt-get install openssl -y
RUN apt-get install apt-transport-https -y
RUN apt-get clean -y
RUN echo "daemon off;" >> /etc/nginx/nginx.conf
EXPOSE 80 443
CMD [ "nginx" ]
```

Так хорошо?)

```
FROM ubuntu:16.04
RUN apt-get update -y && apt-get upgrade -y \
    && apt-get install nginx wget openssl apt-transport-https curl -y \
    && apt-get clean -y
RUN echo "daemon off;" >> /etc/nginx/nginx.conf
EXPOSE 443
CMD [ "nginx" ]
```

Вот так хорошо :=)

```
docker pull nginx
docker run -d -p 80:80 nginx
docker run -it -p 80:80 nginx
docker stop $(docker ps -aq)
docker stop cont_hash
docker build -t id/cont-name:ver .
docker push id/cont-name:ver
```


Подключаемые модули для управления сетью контейнеров

Native (встроенные в Docker)

Remote (сторонние)

Встроенные модули

1. None
2. Host
3. Bridge
4. Macvlan
5. Overlay

Для контейнера создается свой network namespace
У контейнера есть только loopback интерфейс
Сеть контейнера полностью изолирована

Контейнер использует network namespace хоста

Сеть не управляется самим Docker

Два сервиса в разных контейнерах не могут слушать один и тот же порт

Производительность сети контейнера равна
производительности сети хоста

Распознает строки с ключевым словом EXPOSE в Dockerfile и флаг --expose при запуске контейнера

Привязывает доступный порт на хосте из диапазона 32768 - 61000

Для избежания неявно открытых портов не рекомендуется использовать

Особенности default bridge network

Назначается по умолчанию для контейнеров

Нельзя вручную назначать IP-адреса

Нет Service Discovery

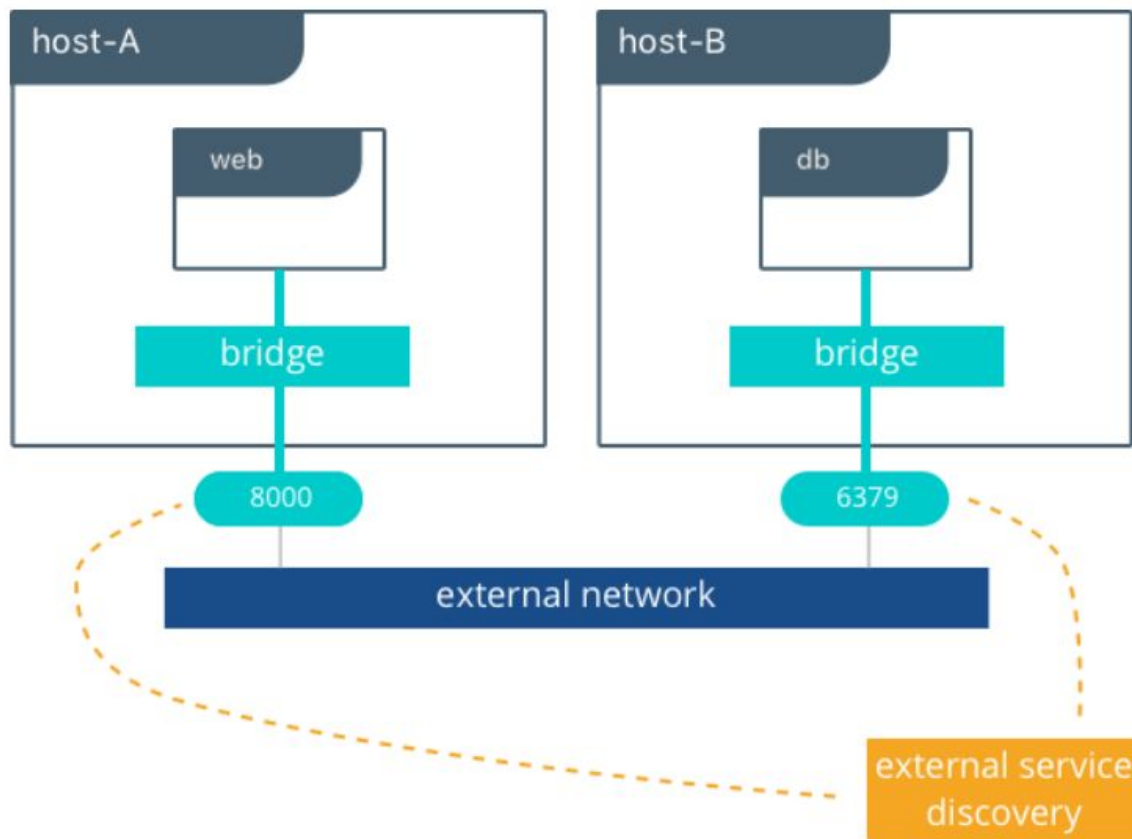
По IP-адресам

Docker Links (deprecated, только для default bridge сети)

Встроенный в Docker DNS

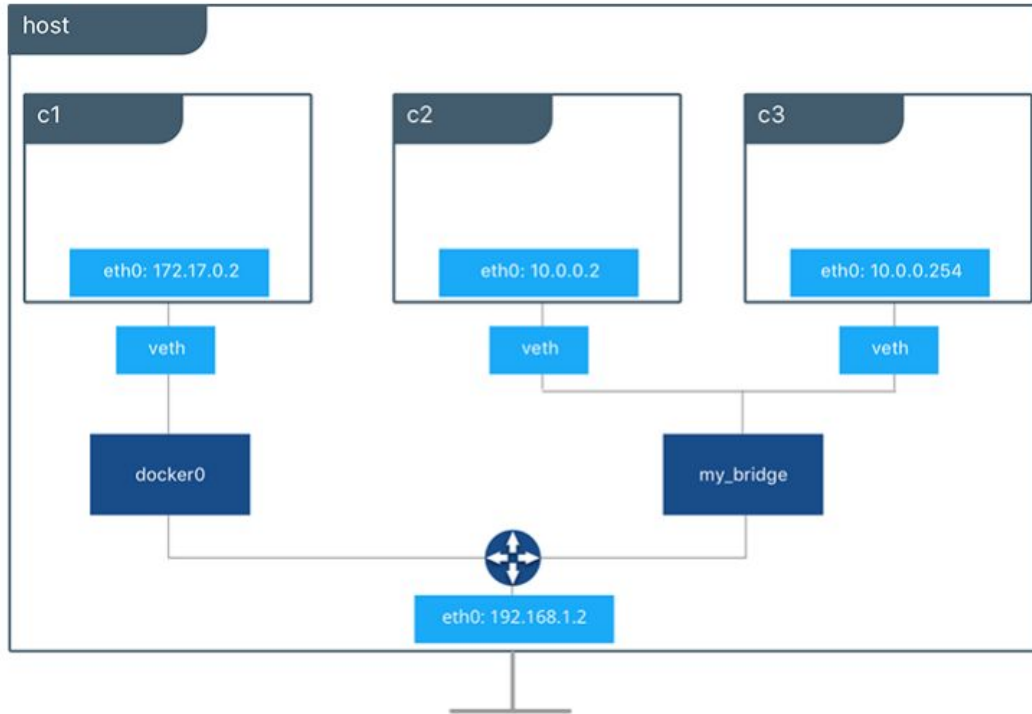
Внешний Service Discovery/DNS сервер

Docker взаимодействие контейнеров



Если нужно отделить контейнер или группу контейнеров
Контейнер может быть подключен к нескольким Bridge сетям
(без рестарта)
Работает Service Discovery
Произвольные диапазоны IP-адресов

Bridge



Работает на основе sub-interfaces Linux

Более производительный, чем bridge

Если нужно подключить контейнер к локальной сети

Поддерживается тегирование VLAN (802.1Q)

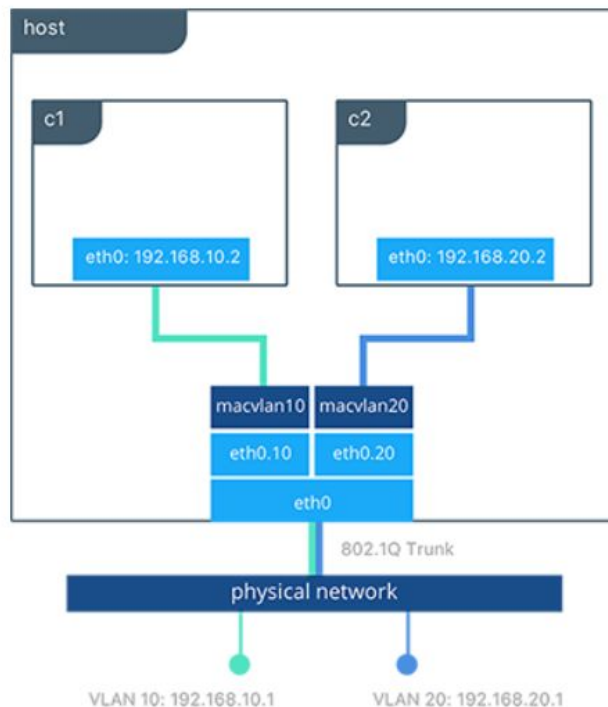
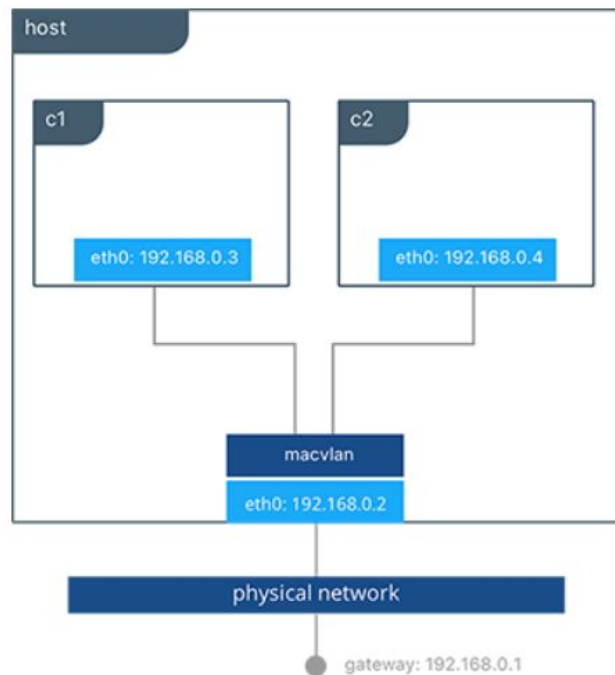
Особенности:

Легко исчерпать пул DHCP

Много MAC адресов в L2 сегменте

Сетевой интерфейс в promiscuous mode

Macvlan

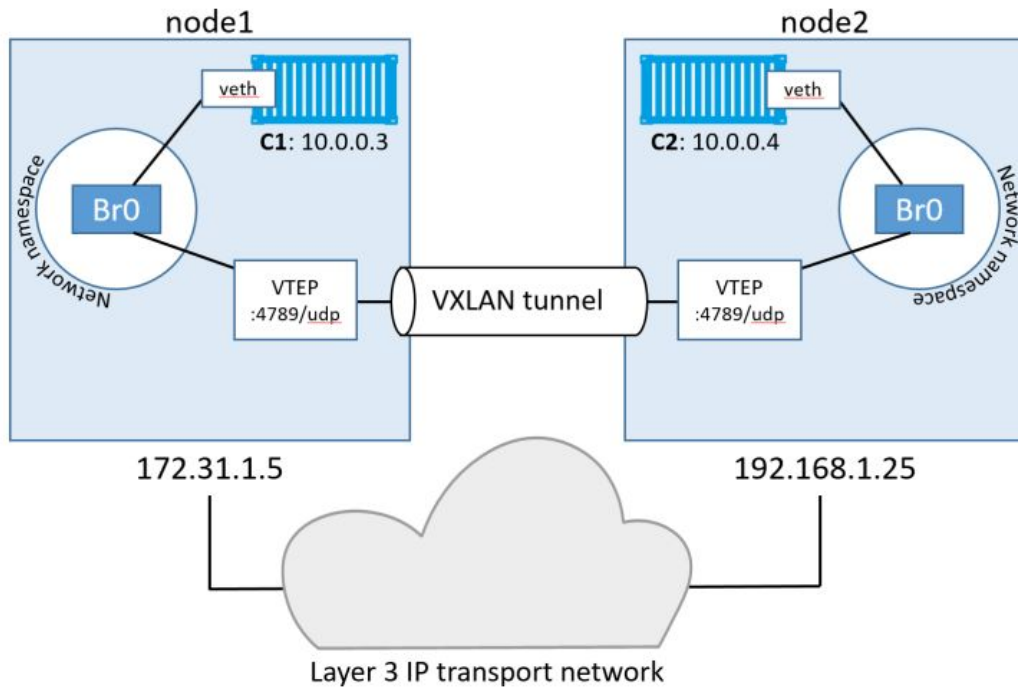


Позволяет объединить в одну сеть контейнеры нескольких Docker хостов

Работает поверх VXLAN

Необходимо хранить состояние распределенной сети

Overlay



Docker docker-compose

version: "3"

services:

 mongo_db:

 image: mongo:3.2

 volumes:

 - db:/data/db

 networks:

 - reddit

 ui:

 build: ./ui

 image: \${USERNAME}/ui:1.0

 ports:

 - 9292:9292/tcp

 networks:

 - reddit