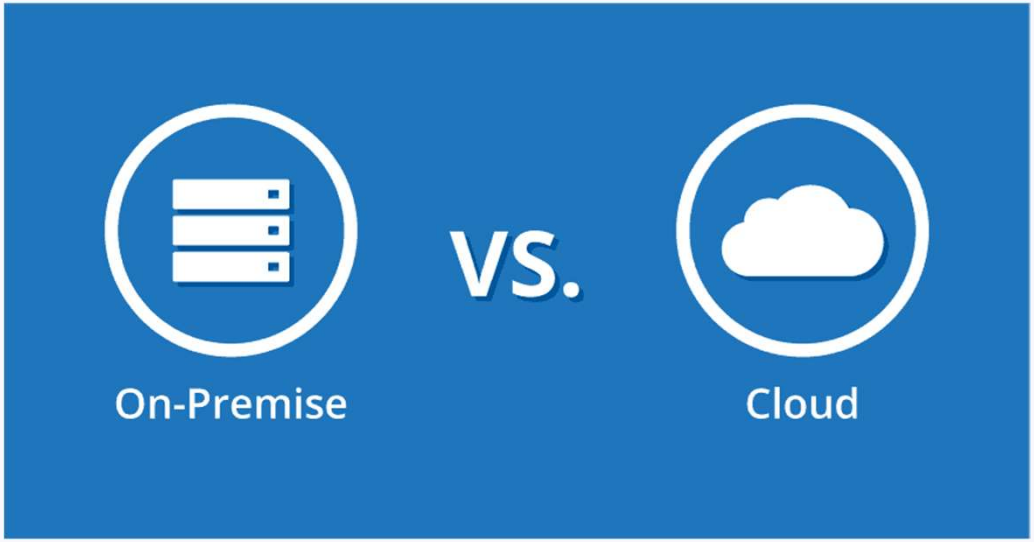
Инфраструктура систем обработки больших данных

Облачная инфраструктура и инфраструктура on-premise

On-Premise vs. Cloud

On-premise дословно переводится как “на предприятии”, что означает использование собственных ресурсов для размещения программного обеспечения. Иногда вместо on-premise встречается сокращенная версия понятия – on-prem.



Развёртывание

On-premise: вся работа должна быть выполнена силами компании

Cloud: все затраты на развёртывание и поддержание берёт на себя провайдер облачных услуг

Масштабируемость

On-premise: сложнее и дольше, даже если в компании есть уже отработанные best practises

Cloud: как правило, услуга масштабируемости идёт “из коробки” в большинстве облачных провайдеров

Управление

On-premise: компания обеспечивает полный контроль на всеми ресурсами

Cloud: по сути компания и облачный провайдер совместно владеют данными и ресурсами

Безопасность

On-premise: данные находятся в большей безопасности, т.к. к ним имеет доступ только компания

Cloud: меньше риск недоступности ресурсов из-за различного рода сбоев

Стоимость

On-premise: выше за счёт покупки всей инфраструктуры, затрат на её ввод в эксплуатацию и поддержку

Cloud: дешевле как на начальном этапе, так и в дальнейшем

Cloud Software

Плюсы:

- доступность
- отсутствие затрат на поддержку
- минимальные усилия на развертывание
- масштабирование

Минусы:

- необходимость иметь доступ к стабильному интернету
- плохая кастомизация

On-prem Software

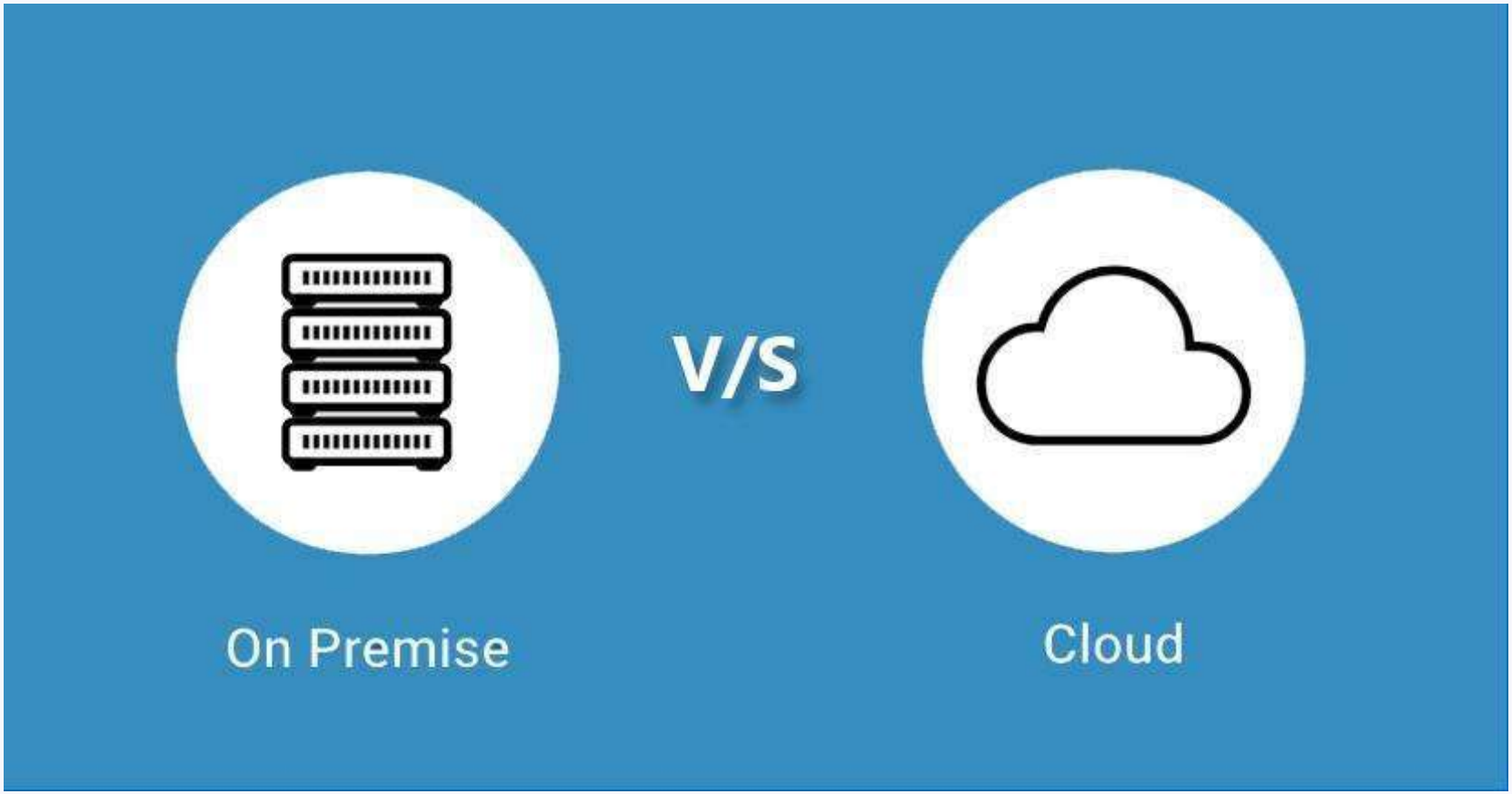
Плюсы:

- стоимость
- полный контроль над данными и ресурсами

Минусы:

- необходимость поддержки
- затраты на развёртывание

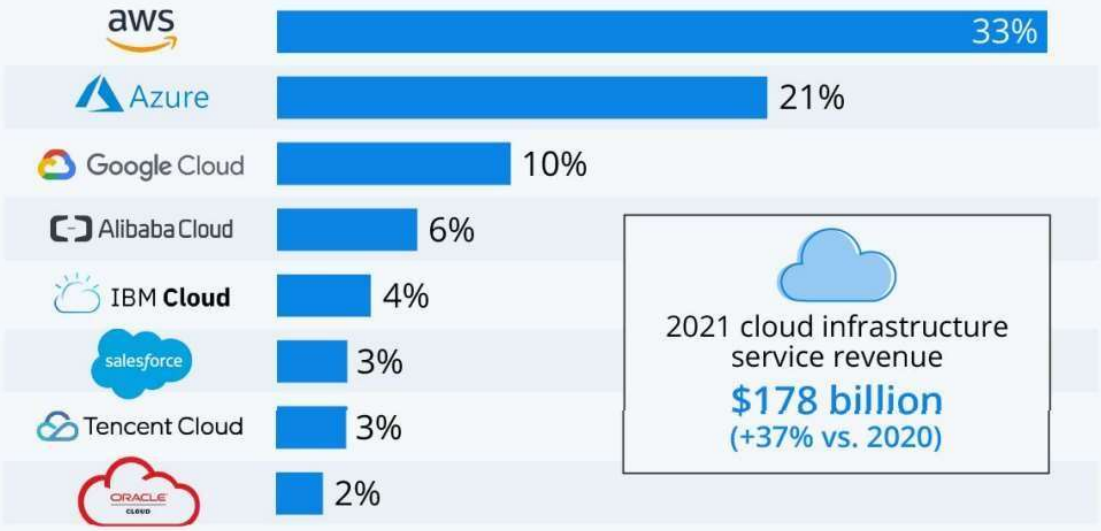
Как выбрать?



Рынок Cloud Native

Amazon Leads $180-Billion Cloud Market

Worldwide market share of leading cloud infrastructure service providers in Q4 2021*



* includes platform as a service (PaaS) and infrastructure as a service (IaaS) as well as hosted private cloud services

Source: Synergy Research Group

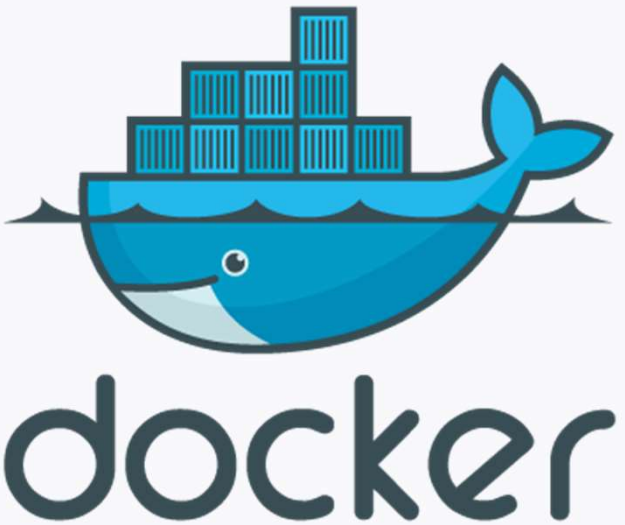
Итоги - тезисы

1. Для большинства компаний подойдёт Cloud решение, в отдельных случаях – on-prem

2. ETL vs ELT

Docker

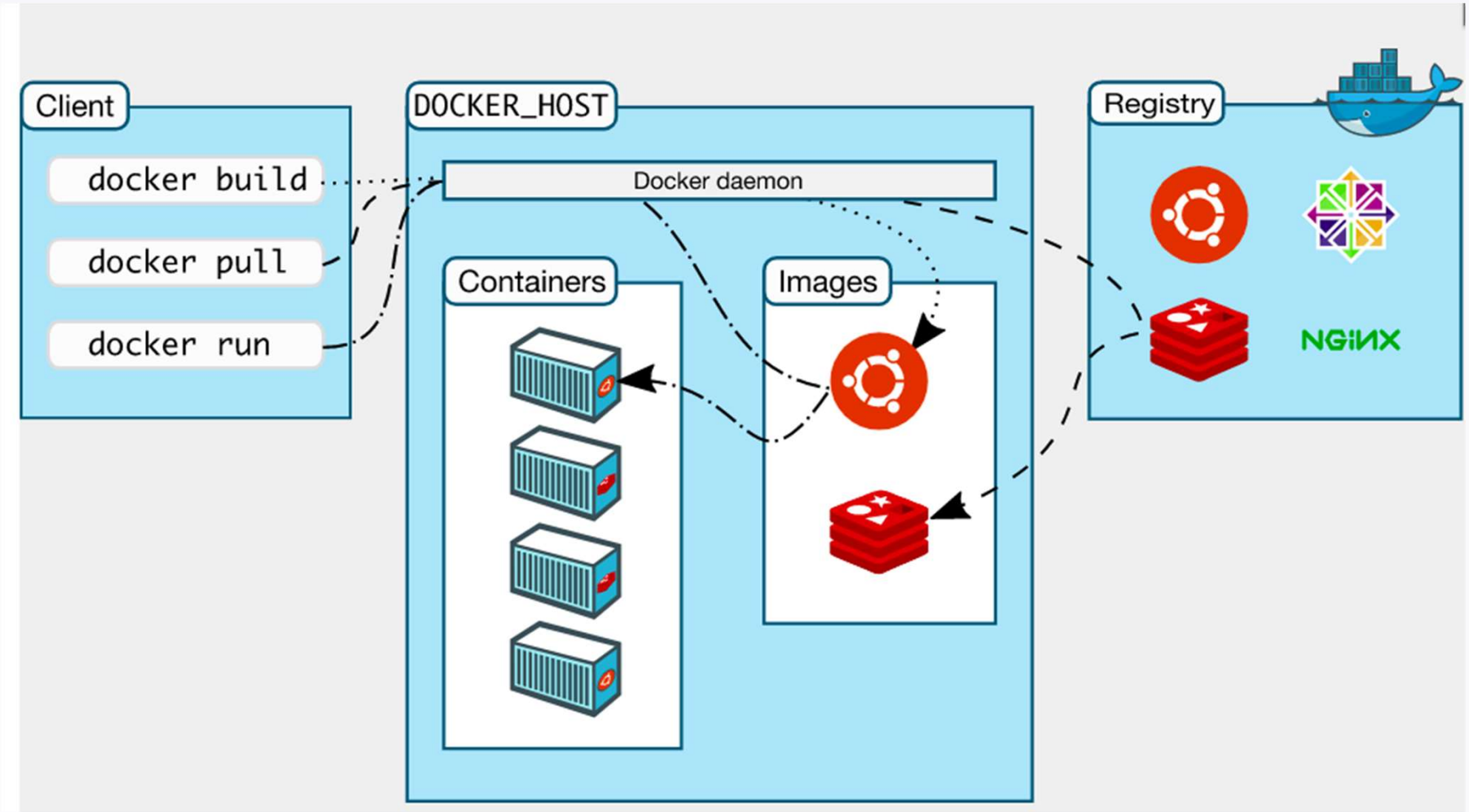
Docker — программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации, контейнеризатор приложений.



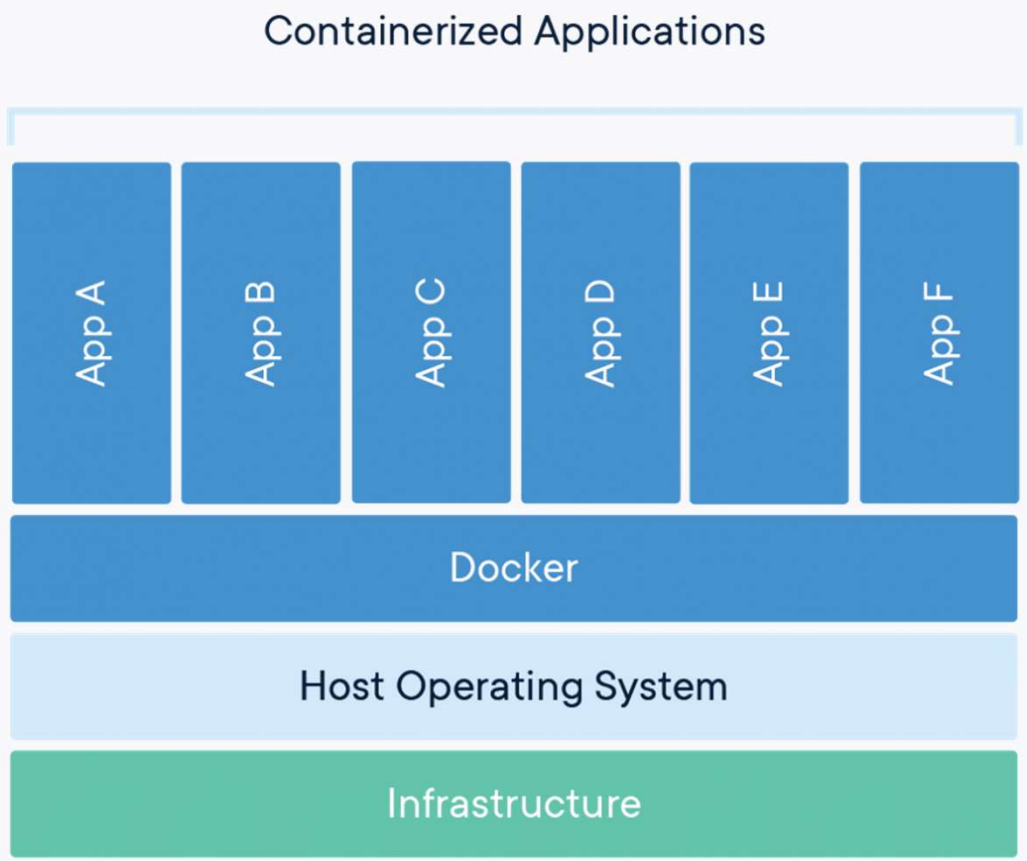
Для чего нужен (или плюсы):

1. Изолированный запуск приложений в контейнерах.
2. Упрощение разработки, тестирования и деплоя приложений.
3. Отсутствие необходимости конфигурировать среду для запуска — она поставляется вместе с приложением — в контейнере.
4. Упрощает масштабируемость приложений и управление их работой с помощью систем оркестрации контейнеров.

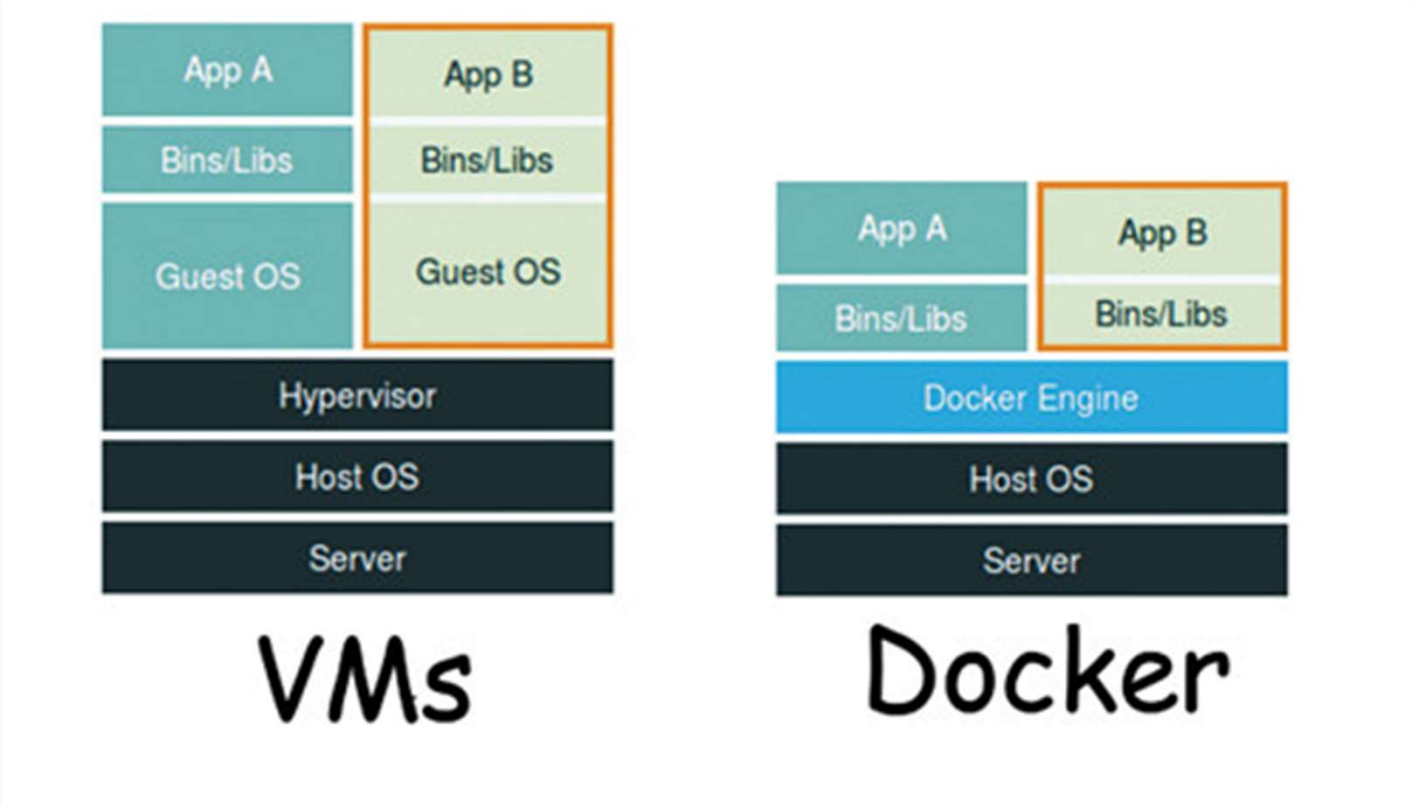
Архитектура



Контейнер

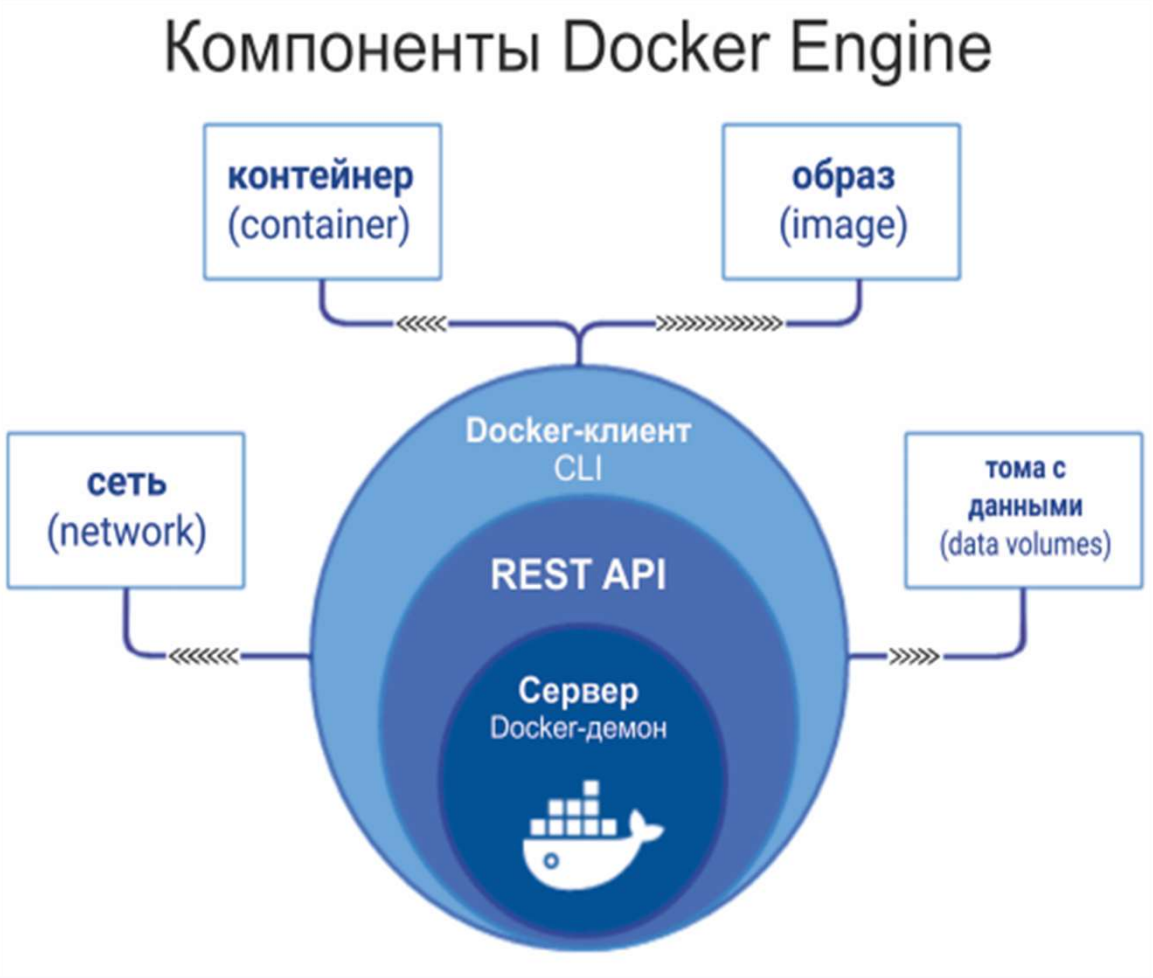

Containerized Applications
App A
App B
App C
App D
App E
App F
Docker
Host Operating System
Infrastructure

Контейнеризация vs Виртуализация

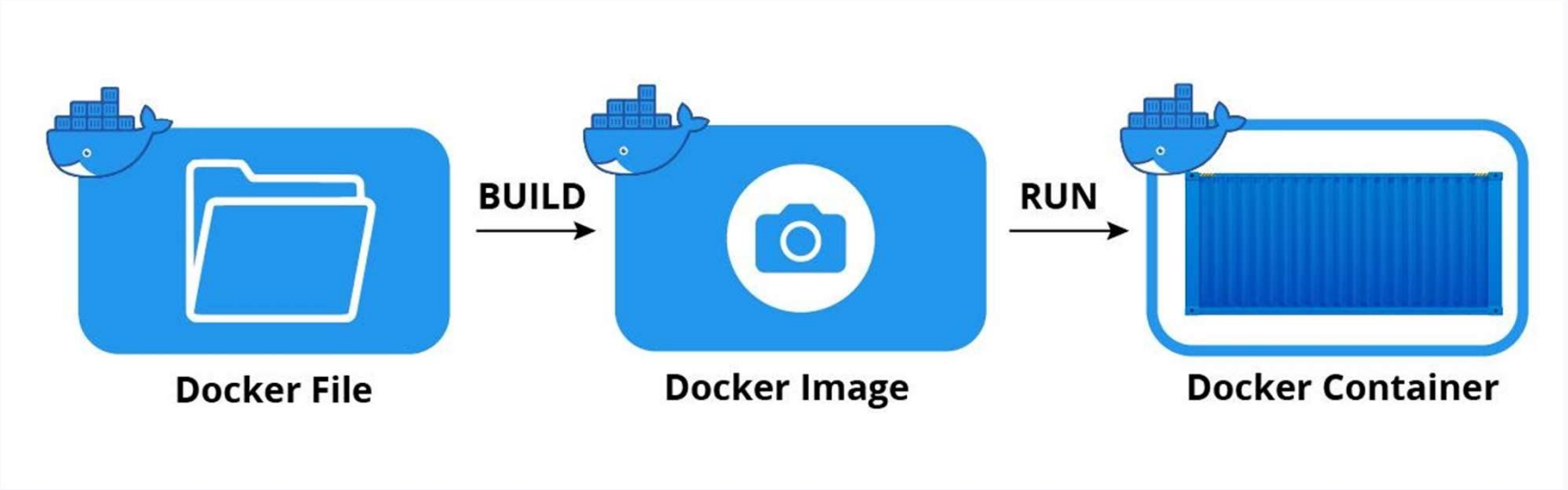


Состав контейнера

1. Образы
2. Контейнеры
3. Volumes
4. Networks



Образ

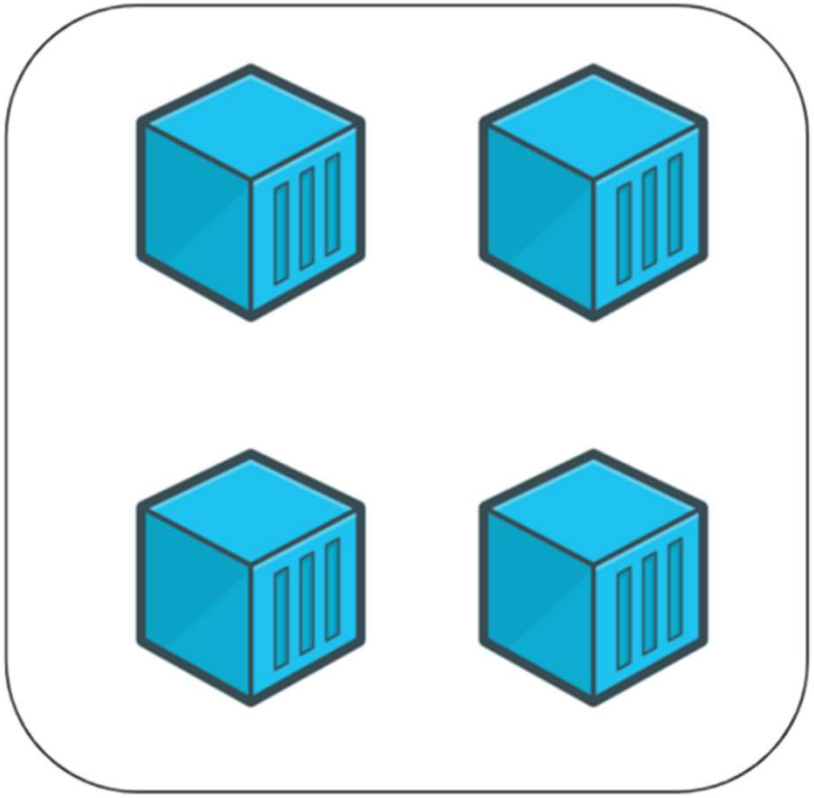


Основные команды

- **docker ps** — список запущенных контейнеров
- **docker pull** — загрузка образа
- **docker build** — собирает образ
- **docker run** — запуск контейнера
- **docker stop** — останавливает контейнер
- **docker rm** — удаляет контейнер
- **docker rmi** — удаляет образ

Docker compose

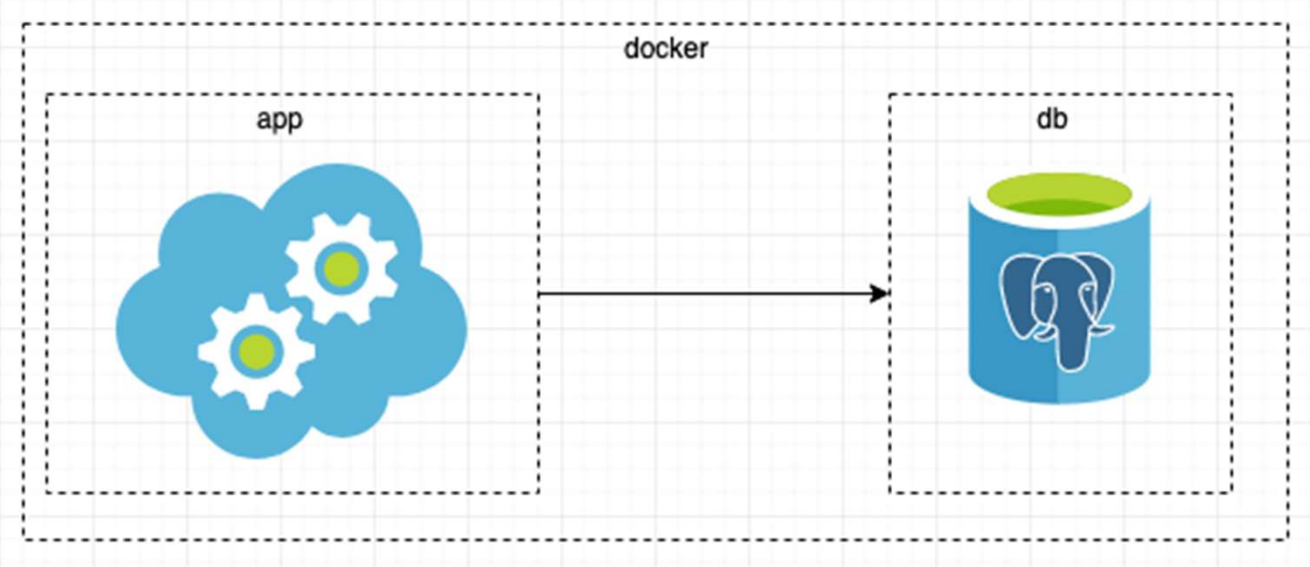
Docker



Docker-Compose

Docker compose

docker-compose - инструментальное средство, предназначенное для решения задач, связанных с развёртыванием проектов.



Docker compose

Основные команды: up

$ docker-compose ps

Создает все зависимости (сеть, volumes) для запуска сервисов, скачивает или собирает образы и запускает все сервисы docker-compose

$ docker-compose ps

Выведет список запущенных контейнеров в рамках проекта.

Docker compose

Основные команды: up

$ docker-compose down

docker-compose down = docker stop + docker rm (для каждого сервиса)

Останавливает все запущенные сервисы, удаляет контейнеры, сети, вольюмы и образы, созданные командой up.

$ docker-compose build

Собирает образы всех сервисов проекта

Docker compose

Основные команды: up

$ docker-compose run --rm [service] [command]

Запуск команды в контейнере. Аргумент rm по аналогии с docker удаляет контейнер после остановки

$ docker-compose logs -f [service]

Просмотр логов контейнера. -f выводит логи после запуска команды до остановки работы logs или контейнера.

Volumes

Рекомендуемый разработчиками Docker способ хранения данных