Контейнеризация

8 октября 2025 г.

Что мы имеем после прошлых лекций

Наш CI/CD pipeline работает:

- Git push → автоматический деплой через GitHub Actions
- Ansible настраивает nginx на сервере
- deploy.sh копирует файлы приложения

Но есть проблема:

Тестовый и основной сайт находятся на одном сервере и управляются одним nginx.

Почему это проблема?

Наша текущая ситуация:

- **Продакшн** — app.prafdin.course.prafdin.ru → /var/www/demo
- **Тест** — app-test.prafdin.course.prafdin.ru → /var/www/demo-test
- Два сервер блока в одном nginx.conf на порту 8181
- Один процесс nginx обслуживает оба окружения

Проблемы отсутствия изоляции:

- **Перезапуск nginx** — ребут теста прерывает продакшн
- **Версия nginx** — нельзя тестировать обновления безопасно

Вопрос: как изолировать приложения друг от друга?

Возможные решения

Как можно решить проблему изоляции?

- **Отдельные машины** — избыточно для нашего случая
- **Разные процессы nginx** — сложно настраивать
- **Контейнеризация** — индустриальный стандарт

Что такое контейнеризация?

Контейнеризация — технология изоляции приложений с их зависимостями в отдельные исполняемые единицы.

Контейнер — процесс или группа процессов, запущенные в изолированном окружении.

Изоляция на уровне:

- **Процессов** — у каждого контейнера своё дерево процессов
- **Сети** — у каждого контейнера свои сетевые интерфейсы, настройки маршрутизации
- **Ресурсов** — cpu, ram, IO
- **Файловой системы** — у каждого контейнера своя файловая система
- **...** — а также много других механизмов ядра

Контейнеры vs Виртуальные машины

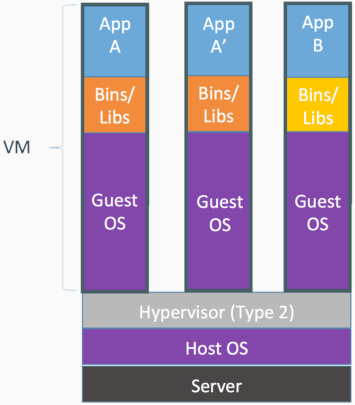
Виртуальные машины

- Полная ОС для каждого приложения
- Большой overhead
- Медленный старт
- Изоляция на уровне железа

Контейнеры

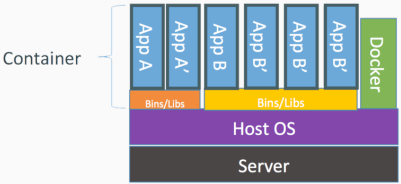
- Общее ядро ОС
- Минимальный overhead
- Быстрый старт (секунды)
- Изоляция на уровне процессов

Контейнеры vs Виртуальные машины



Containers are isolated, but share OS and, where appropriate, bins/libraries

...result is significantly faster deployment, much less overhead, easier migration, faster restart



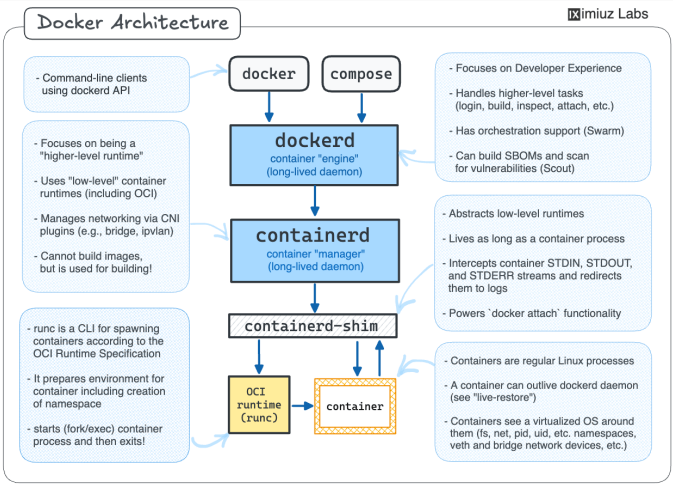
https://dzone.com/articles/evolution-of-linux-containers-future

Способы работы с контейнерами

Существует несколько технологий контейнеризации:

- **Docker** — самая популярная платформа контейнеризации
- **Podman** — альтернатива Docker без демона
- **LXC/LXD** — системные контейнеры

Docker



https://labs.iximiuz.com/challenges/docker-install-on-ubuntu

Основные понятия Docker

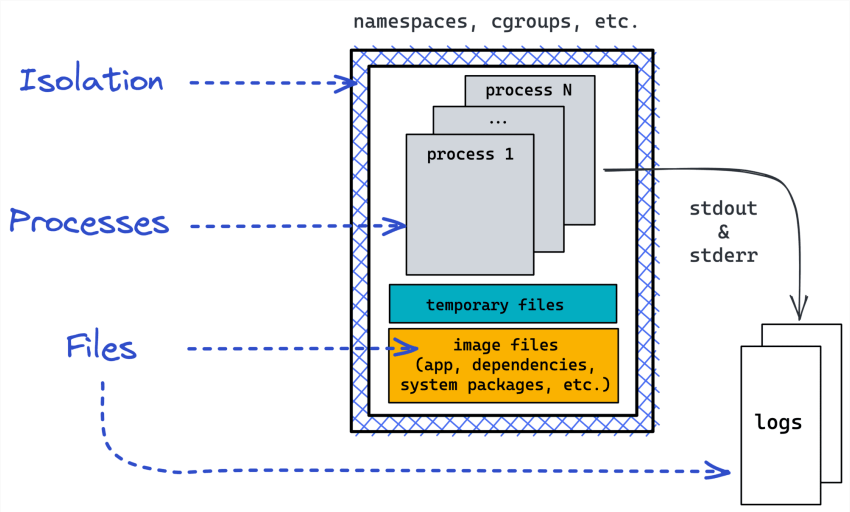
Ключевые абстракции Docker:

- **Container** — процесс или группа процессов, запущенные в изолированном окружении
- **Image** — шаблон контейнера, содержащий снимок файловой системы и метаданные
- **Dockerfile** — инструкции для сборки образа (image)

Принцип работы:

1. Описываем как собрать образ в Dockerfile
2. Собираем образ из Dockerfile
3. Запускаем контейнер из образа

Docker контейнер



https://labs.iximiuz.com/tutorials/docker-container-management-commands

Docker Registry

Docker Registry — это сервер для хранения и распространения Docker-образов.

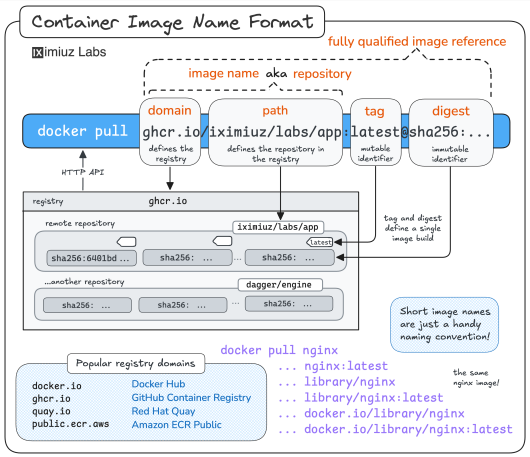
Примеры:

- Docker Hub
- GitLab Registry
- GitHub Container Registry
- Harbor

Основные операции:

- **Push** — загрузка образа в удаленный registry
- **Pull** — скачивание образа из registry

Docker Image



https://labs.iximiuz.com/challenges/build-and-publish-container-image-with-docker

Демонстрация: Docker Image, Container, Registry

Что покажем:

1. Устройство слоёв имеджей
2. Базовая работа с контейнерами через docker cli
3. Работа с docker registry

Демо-репозиторий:
https://github.com/prafdin/devops-course-demos/tree/docker

Демонстрация: Контейнеризация сайта

Что покажем:

1. Dockerfile для сайта
2. Работа с контейнерами через docker cli
3. Использование контейнера в CI/CD пайплайне

Демо-репозиторий:
https://github.com/prafdin/devops-demo-website/tree/docker-demo

Эволюция нашего pipeline

Было:

- Ansible устанавливаете nginx
- deploy.sh копирует файлы

Стало:

- Ansible устанавливает только docker
- Docker build, push в pipeline
- Запуск на сервере

Что запомнить

- **Контейнеризация** — ключевой принцип современной инфраструктуры
- **Контейнеризация решает проблемы** переносимости и конфликтов зависимостей
- **Контейнеризация стандартизирует окружение** и упрощает развертывание

Вопросы?