Инсталиране и конфигуриране на Nginx HTTP сървър под Linux

Добре дошли в презентацията на моя проект по системно администриране. Тук ще разгледаме процеса на инсталиране и конфигуриране на Nginx като HTTP сървър в Linux среда, използвайки силата на Docker.

- **Име:** Димитър Николов
- **ФН:** 471222022
- **Специалност:** ИСН

Цел на проекта

Основната цел на този проект е да демонстрира практически умения в настройката и управлението на уеб сървъри. Той обхваща както основни, така и разширени конфигурации на Nginx, подчертавайки неговата гъвкавост и ефективност.

Реализация на HTTP сървър

Изграждане на HTTP сървър под Linux.

Конфигуриране на Nginx

Настройка на Nginx за производителност и сигурност.

Демонстрация на функционалности

Показване на ключови функции като статично обслужване, reverse proxy и сигурност.

Практическо приложение

Прилагане на знания в реална среда, използвайки съвременни технологии.

Избрана тема

Избраната тема е изключително практична и е пряко свързана с нуждите на съвременните ИТ инфраструктури. Фокусът е върху изграждането на ефективен и мащабируем уеб сървър, който може да обслужва различни типове приложения.

1

Практическа насоченост

Решаване на реални проблеми в системното администриране.

2

Nginx като HTTP сървър

Избор на един от най-популярните и мощни уеб сървъри.

3

Docker среда

Модерна и ефективна методология за разгръщане и управление.

Използвани технологии

Проектът използва набор от индустриално-стандартни технологии, осигуряващи гъвкавост, мащабируемост и лесна поддръжка.

- **macOS (host среда)**

  Работна станция за разработка и управление на Docker контейнери.

- **Docker и Docker Compose**

  За виртуална изолация, оркестрация и управление на услугите.

- **Linux-базирани контейнери**

  Alpine linux за хостване на Nginx и API.

- **Nginx (HTTP сървър)**

  Уеб сървър и reverse proxy.

- **Backend API контейнер**

  Примерно API приложение.

Защо Docker?

1 Изолирана Linux среда

Всеки контейнер работи в своя собствена изолирана среда, предотвратявайки конфликти между зависимостите.

2 Улеснено стартиране и демонстрация

Лесно и бързо стартиране на цялата инфраструктура с една команда.

3 Повторяемост и лесна конфигурация

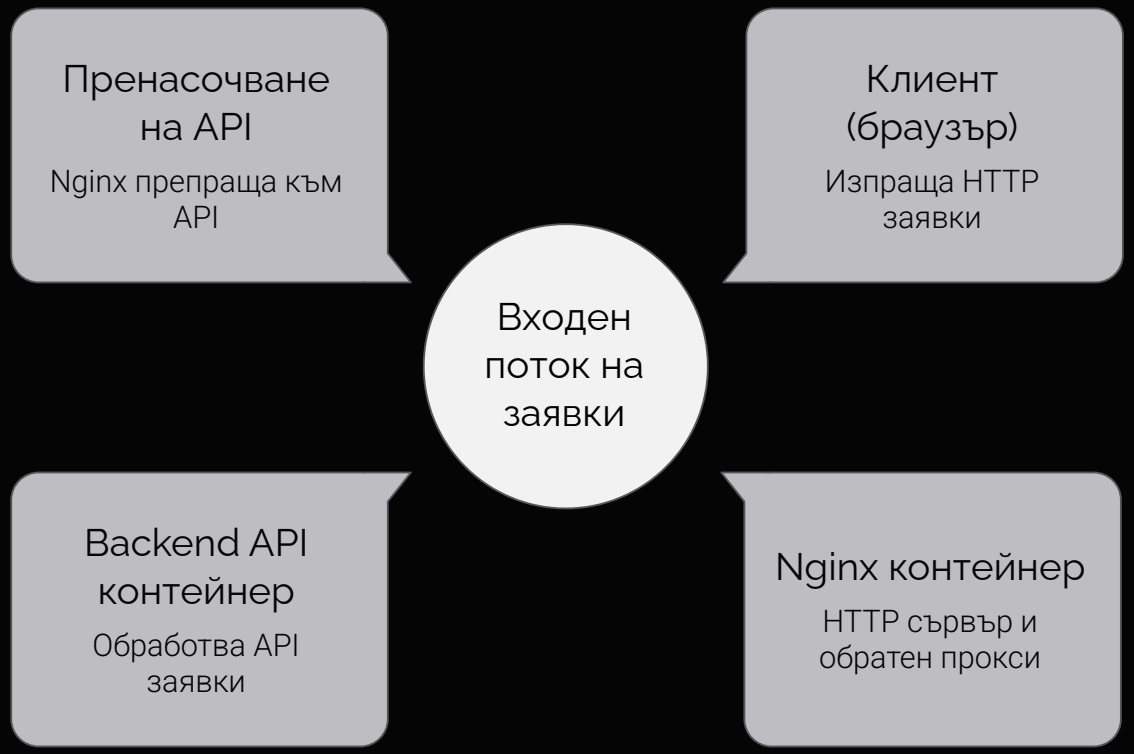
Гарантира, че средата е идентична навсякъде, което улеснява споделянето и възпроизвеждането на проекта.

4 Широко използван в реални среди

Docker е стандарт в облачните и DevOps практики, което прави придобитите умения приложими в индустрията.

Архитектура на проекта

Проектът е изграден на базата на микросървизна архитектура, където Nginx играе ролята на входна точка и обратен прокси сървър.



Клиентът (уеб браузър) изпраща заявки към Nginx контейнера, който обслужва статично съдържание и препраща API заявките към Backend API контейнера, осигурявайки плавно взаимодействие.

Структура на проекта

Структурата на проекта е логична и модулна, което улеснява управлението и разширяването на функционалностите.

- `docker-compose.yml`: Дефинира услугите (Nginx, API) и тяхната конфигурация, мрежи и томове.
- `nginx/default.conf`: Основният конфигурационен файл за Nginx, указващ виртуални хостове, обслужване на статично съдържание и правила за reverse proxy.
- Директория `html/`: Съдържа статичните HTML страници, които Nginx обслужва директно.
- Директория `error_pages/`: Включва персонализирани страници за грешки (напр. 404 Not Found, 500 Internal Server Error) за по-добро потребителско изживяване.
- Директория `download/`: Зона, от която потребителите могат да изтеглят файлове, конфигурирана за директно обслужване от Nginx.

Основни функционалности

Статично обслужване

Бързо и ефективно обслужване на HTML, CSS, JavaScript и изображения.

Custom error pages

Персонализирани страници за грешки (404, 500) за по-добро потребителско изживяване.

Status monitoring

Предоставяне на информация за състоянието на сървъра чрез `stub_status` модула.

Pretty URLs

Конфигуриране на URL презаписвания за по-чисти и семантични уеб адреси.

File download зона

Специализирана директория за изтегляне на файлове с оптимизирани заглавки.

Reverse Proxy

Препращане на заявки към backend API контейнера, балансиране на натоварването.

Reverse Proxy и API

Една от ключовите роли на Nginx в тази архитектура е да функционира като reverse proxy, ефективно разделяйки frontend и backend логиката.

- **Nginx приема клиентските заявки:** Всички входящи заявки от уеб браузърите първо достигат до Nginx.
- **API заявките се препращат към backend контейнер:** Nginx интелигентно разпознава заявките, предназначени за API, и ги препраща към съответния backend API контейнер. Това се постига чрез конфигурации за местоположение (`location`) в Nginx.
- **Разделяне на frontend и backend логика:** Този подход позволява независимо мащабиране и разработка на клиентската част (статично съдържание) и сървърната логика (API).
- **Централизиран контрол чрез Nginx:** Nginx предоставя единна точка за достъп, улесняваща прилагането на политики за сигурност, кеширане и балансиране на натоварването.

Monitoring, логове и сигурност

Nginx предоставя вградени механизми за наблюдение и записване на събития, които са от решаващо значение за поддържане на здрава и сигурна сървърна среда.

1

`stub_status` endpoint

Позволява наблюдение на активните връзки, обработените заявки и други основни метрики за производителност.

2

Access и error логове

Подробни записи за всички входящи заявки и възникнали грешки, ключови за диагностика и анализ на трафика.

3

Скриване на версията на Nginx

Мярка за сигурност, предотвратяваща разкриването на информация за сървърната версия на потенциални нападатели.

4

Ограничен достъп до status endpoint

Достъпът до `stub_status` е ограничен само до доверени IP адреси за предотвратяване на неоторизиран мониторинг.

5

Контрол на достъпа до ресурси

Допълнителни правила за сигурност могат да бъдат приложени за ограничаване на достъпа до определени файлове или директории.