

Fast Reverse Proxy как альтернатива Ngrok

 habr.com/ru/post/583814



[anydasa](#) вчера в 14:11

Разработка веб-сайтов *

Tutorial

Создание общедоступного URL в сети интернет к вашему локальному проекту

Что такое Ngrok, наверное знает каждый разработчик web приложений, и многие им пользуются.

Немного предыстории...

Присоединившись к новому большому проекту, над которым работают десятки разработчиков и QA специалистов, я столкнулся с тем, что разработка ведется удаленно на специально выделенных серверах.

А т.к. я уже несколько лет разработку приложений веду исключительно в докере, я никак не мог адаптироваться к текущему подходу.

Думаю все, кто более менее освоил работу в докере, уже не мыслят как можно разрабатывать без него. Причин на это много. Конечно, как и везде у докера есть своя цена и это тоже уже много раз обсуждалось.

Итак, новый проект не похож на предыдущие. Он имеет много зависимостей с другими сервисами, как внутренними так и внешними.

Большое количество внешних интеграций порождало проблему связи локального приложения с внешним миром. И если объединить внутренние сервисы используя docker network не вызывало каких либо проблем, то необходимость связать внешний сервис уже требовал дополнительных инструментов.

Интеграции платежных систем всегда подразумевает, что будут callback (notification).

При такой необходимости, часто выбирают Ngrok. Хорошее решение, но в удобном варианте - платное. Особенно это ощущается, когда разработчиков много.

Т.к. Ngrok не подходил, первое что пришло в голову, создать виртуалку, на нее завести домен, и создавать ssh туннель с ним.

▼ `docker-compose.yml`

```

version: '3.7'

services:

  callback-tunnel:
    build:
      context: ../docker/ssh-tunnel-callback
    restart: unless-stopped
    volumes:
      - ~/.ssh:/root/ssh:ro
    environment:
      TUNNEL_HOST: ${CALLBACK_TUNNEL_HOST}
      LOCAL_PORT: ${CALLBACK_TUNNEL_LOCAL_PORT}
      LOCAL_HOST: ${CALLBACK_TUNNEL_LOCAL_HOST}
      REMOTE_PORT: ${CALLBACK_TUNNEL_REMOTE_PORT}
    networks:
      - local

  app:
    image: php
    restart: unless-stopped
    tty: true
    volumes:
      - .:/var/www/html

```

где Dockerfile callback-tunnel выглядел вот так

▼ Dockerfile

```

FROM alpine

RUN apk add --update openssh-client && rm -rf /var/cache/apk/*

CMD rm -rf /root/.ssh && mkdir /root/.ssh && cp -R /root/ssh/* /root/.ssh/ && \
  chmod -R 600 /root/.ssh/* && \
  ssh \
  -vv \
  -o StrictHostKeyChecking=no \
  -N $TUNNEL_HOST \
  -R *:$REMOTE_PORT:$LOCAL_HOST:$LOCAL_PORT \
  && while true; do sleep 30; done;

```

Данный подход решал проброс callback запросов от внешних провайдеров, но он был не надежный. Иногда при жестком разрыве соединения, порт на виртуалке оставался занятым. При совместной разработки необходимо было договариваться, кто какой порт будет использовать. Короче, решение такое себе, костыль.

Дальше - больше, при локальной разработке, бывает необходимость проверить приложение на другом устройстве, например на телефоне или планшете, или в определенной версии браузера поведение фронта не адекватное. И снова мысли об Ngrok.

Пришлось искать.

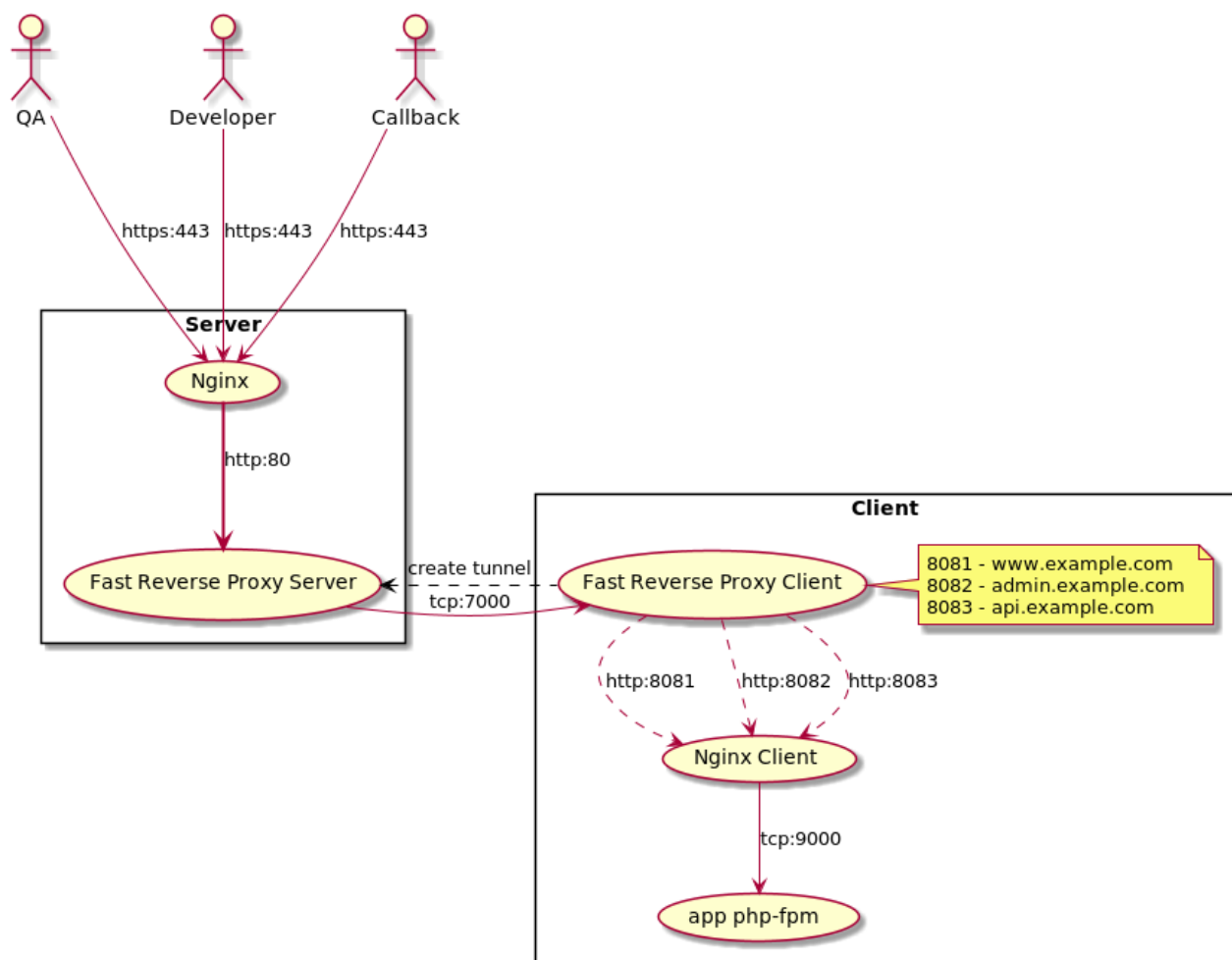
В какой то момент мне попался список бесплатных инструментов решающих данную проблему. Сам список уже не найду, но один инструмент меня сильно заинтересовал, и я решил попробовать.

Fast Reverse Proxy

<https://github.com/fatedier/frp>

Сервер написан на Go
JS, Vue для Dashboard

Схема



Репозиторий с примерами <https://github.com/anydasa/frp-example>

Как поднимал

1. Подготовил на github репозиторий
2. Купил домен
3. Создал дроплет на Digital Ocean (5\$ docker)
4. Установил туда nginx

5. Установил letsencrypt и создал Wildcard SSL Certificate по этой доке. Wildcard нужен для того чтоб public URL были https
6. Настроил nginx, он выступает как первый проху server. Можно обойтись без него, но мне так было проще
7. Склонил <https://github.com/anydasa/frp-example> и запустил server-ную часть
8. Локально,
 1. склонил <https://github.com/anydasa/frp-example>
 2. Создал .env из .env-example и прописал необходимые переменные
 3. запустил client-ский docker-compose
9. В зависимости от того какой указал REVERSE_PROXY_PERSONAL_ALIAS, будет мой URL.

В моем примере есть 3 хоста (обычно нужно для проекта), и в зависимости какой PERSONAL_ALIAS указан, будут доступны по URLs. К примеру PERSONAL_ALIAS=project, тогда

- frontend - <https://project.frp.example.com>
- backend - <https://admin-project.frp.example.com>
- api - <https://api-project.frp.example.com>

Т.к. Wildcard SSL настраивается на *.frp.example.com, то все поддомены нужно указывать без точек.

- Server Nginx
- Для удобства, поднял песок, чтоб можно было пощупать. (временно)