**前置知识**

Linux基本使用（ubuntu）

Docker基本使用

理论知识（非必须）：微服务，领域驱动设计

注：以上的教程都可以在github教程的各个目录找到

**基本环境**

系统：ubtunu 20

Docker：20.10.11

**Docker安装Kong**

使用如下命令拉去Kong

$ sudo docker pull kong:2.0

**创建Docker网络**

创建一个docker网络，我们的服务都放在这个网络中

$ sudo docker network create -d bridge kongnet

**Kong数据库安装部署**

Kong支持PostgreSQL、Cassandra、DB-less（无数据库）3种数据库模式，我们可以选择其中一种模式运行

1. **选择PostgreSQL**

1）拉取容器：$ sudo docker pull postgres:9.6

2）创建数据库存储目录：$ sudo mkdir -p /opt/postgres/data

3）运行容器：

执行如下命令运行postgres：

$ sudo docker run -d --name postgres \

--network kongnet \

--restart=always \

-e POSTGRES\_PASSWORD=kong123 \

-p 5432:5432 \

-v /home/software/postgres/data:/var/lib/postgresql/data \

postgres:9.6

命令说明：

--name postgres：为容器实例名称

--network kongnet：为指定容器要部署在kongnet网络

-e POSTGRES\_PASSWORD=password：为密码

-p 5432:543：为端口映射

-v \*\*\*\*\*\*：为磁盘映射

postgres:9.6：为选择的容器

1. **选择Cassandra**

用不到，懒得写

1. **选择DB-less**

选择该模式，只需将/etc/kong/kong.conf配置文件的database指令设置为off即可

注：这里我们就选择PostgreSQL作为数据库

**创建数据库**

1. 执行如下命令进入容器

$ sudo docker exec -it postgres /bin/bash

1. 执行如下命令进入postgres（-U 为用户名选项，postgres默认用户名为postgres，密码为空）

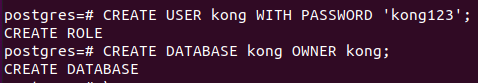
$ psql -U postgres

1. 执行如下2个命令创建名为kong的数据库（别忘了后面的 ; ）

$ CREATE USER kong WITH PASSWORD 'kong123';

$ CREATE DATABASE kong OWNER kong;

示例图片：



1. 执行 \q 退出postgres，执行 exit退出docker

**运行Kong**

1. **初始化数据库**

运行前我们需要初始化kong的数据库，执行如下命令初始化

$ sudo docker run --rm \

--network kongnet \

-e "KONG\_DATABASE=postgres" \

-e "KONG\_PG\_HOST=postgres" \

-e "KONG\_PG\_USER=kong" \

-e "KONG\_PG\_PORT=5432" \

-e "KONG\_PG\_PASSWORD=kong123" \

-e "KONG\_PG\_DATABASE=kong" \

kong:2.0 kong migrations bootstrap

命令说明：

-e "KONG\_PG\_HOST=postgres"：postgres为我们之前部署postgresql的容器名称

1. **运行Kong**

执行如下命令：

$ sudo docker run -d --name kong \

--network kongnet \

--restart=always \

-e "KONG\_DATABASE=postgres" \

-e "KONG\_PG\_HOST=postgres" \

-e "KONG\_PG\_USER=kong" \

-e "KONG\_PG\_PORT=5432" \

-e "KONG\_PG\_PASSWORD=kong123" \

-e "KONG\_PG\_DATABASE=kong" \

-e "KONG\_PROXY\_ACCESS\_LOG=/dev/stdout" \

-e "KONG\_ADMIN\_ACCESS\_LOG=/dev/stdout" \

-e "KONG\_PROXY\_ERROR\_LOG=/dev/stderr" \

-e "KONG\_ADMIN\_ERROR\_LOG=/dev/stderr" \

-e "KONG\_ADMIN\_LISTEN=0.0.0.0:8001" \

-e "KONG\_ADMIN\_LISTEN\_SSL=0.0.0.0:8444" \

-p 80:8000 \

-p 443:8443 \

-p 8001:8001 \

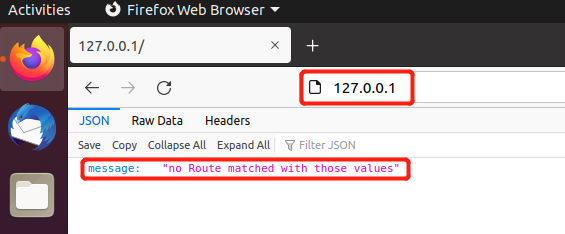
-p 8444:8444 \

kong:2.0

命令说明：还说明呢

查看运行：

访问127.0.0.1可看到如下效果



1. **停止Kong**

执行如下命令停止kong

# docker stop kong

如想重新运行kong则执行如下命令

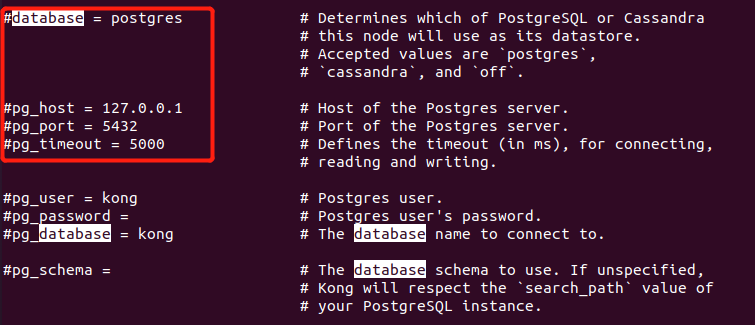
# docker start kong

**Kong配置**

**配置文件**

Kong有一份模板配置文件，为 /etc/kong/kong.conf.default（你需要先进入容器里面），由于我们使用docker容器直接运行，所以没有应用到配置文件

如下可以看到模板的一下配置示例：



**端口**

Kong默认端口有

8000：监听来自客户端的http，并转发给上游服务器

8443：监听来自客户端的https请求，并转发给上游服务器

8001：监听来自Admin API的http请求

8444：监听来自Admin API的https请求

**概念**

客户端请求：即kong要转发的请求，来自我们的客户端（如浏览器，App等请求）

Admin API请求：kong的管理也是通过http api来进行管理的，所以发送到8001、8444是用于管理kong的api请求

**Kong基础对象**

Kong由如下对象组成

1. **路由**

路由定义了客户端请求与服务的匹配规则，一个路由至少需要一个匹配规则和协议（HTTP、HTTPS、TCP等）

1. **服务**

服务将客户端请求发送给对应的上游服务器，主要属性为protocol（协议，如HTTPS）、host（上游服务器的名称或域名，如www.xxx.com）、port（端口，如80）、path（要附加的路径，可选，如 /api）

1. **上游**

上游是一个节点，可以对多个目标进行负载均衡和健康检查，一个上游可以包含多个目标节点

1. **目标**

一个目标节点代表一个真实的物理服务，通常为IP地址和端口的组合

1. **消费者**

客户端请求的发起者

1. **插件**

Kong请求/响应的中间件，可以在路由和服务上添加（如，身份验证、限频等插件）

目前kong内置的插件已提供身份验证、安全防护、流量控制、日志记录等

1. **证书**

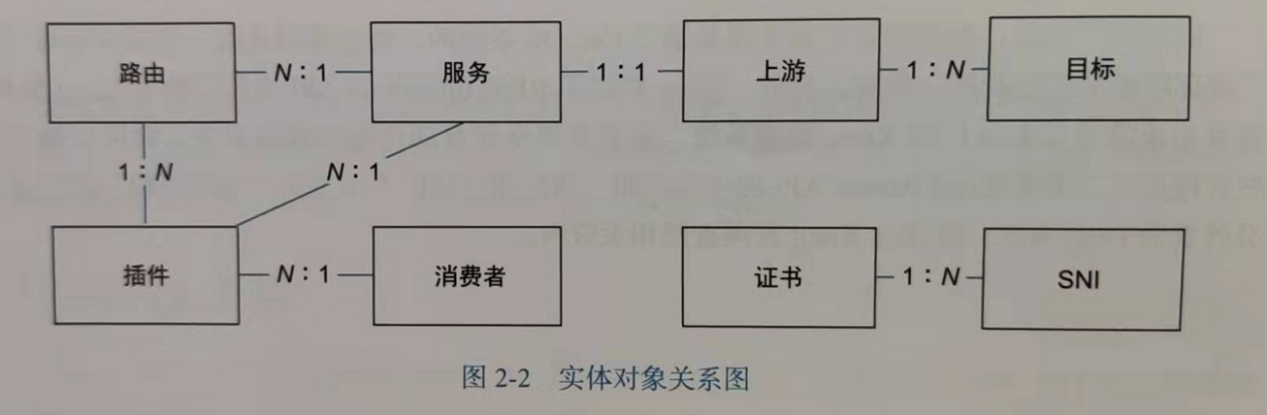
额。。。就是我们所说的SSL证书

1. **SNI**

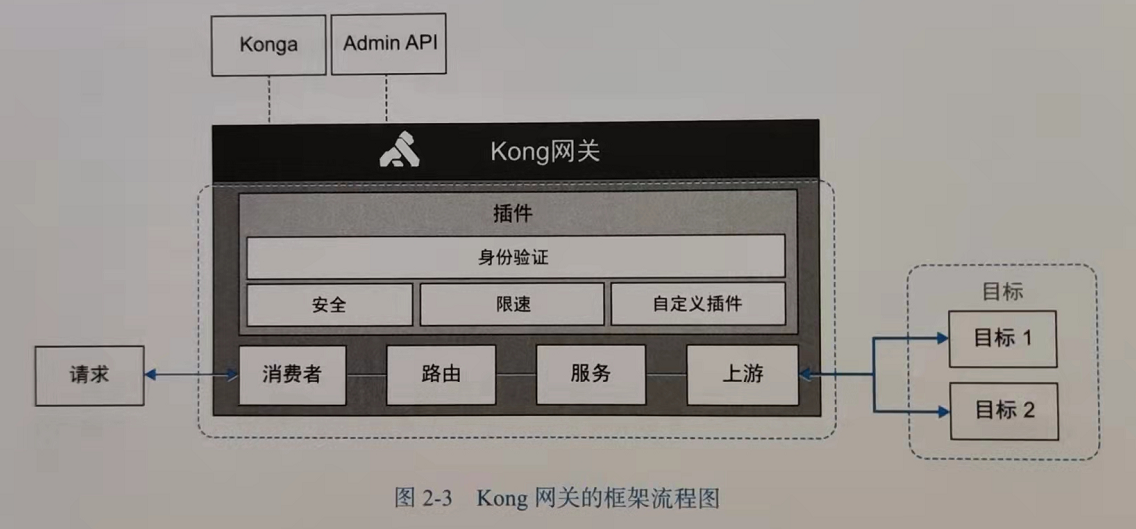
用于将多个域名映射到一个证书上

**对象之间的关系**

关系图



流程图



（哎，找不到pdf，只能拍书籍了）

**踩坑日志（不看白不看）**

1. **虚拟机挂起又开启后服务访问不了了**

话说忙碌了几天了，终于把kong运行起来了，然后我挂了虚拟机睡觉去了，第二天开启的时候，发现服务访问不了了，搞不懂，容器删了又装，装了又删，服务还是访问不了，最后无奈只能重启虚拟机，然后重启容器（docker restart xxxx）,果然重启能够解决90%的问题，这一重启，它又好了，但每次都重启容器好麻烦，最后我做了一个违背祖宗的决定，放弃ubuntu桌面版，使用ubuntu服务版，但是，然并卵。

最后，解决方案：在运行容器时加上 --restart=always 让容器自动重启，这样我们只需要重启虚拟机容器即可自动重启