链滴

后端架构学习（3）- nginx

作者：ccran

原文链接：https://ld246.com/article/1590213283729

来源网站：链滴

后端架构学习（3）- nginx

之前记录了在后端架构中加入了redis的支持，进一步提高了我们后端能承受的并发量。

现在的项目，基本上都是前后端分离了。一方面，是为了前后端分工更为明确，各自发挥所长，项目加模块化和专业化；另一方面，是因为假如后端挂了，起码用户还能看到前端的页面，只不过后端接调用不了，更为**user friendly**。

为了让后端不会轻易挂掉，一般会开多个实例，并且，通过本期的主角：nginx，通过反向代理来做载均衡，保证后端的稳定。

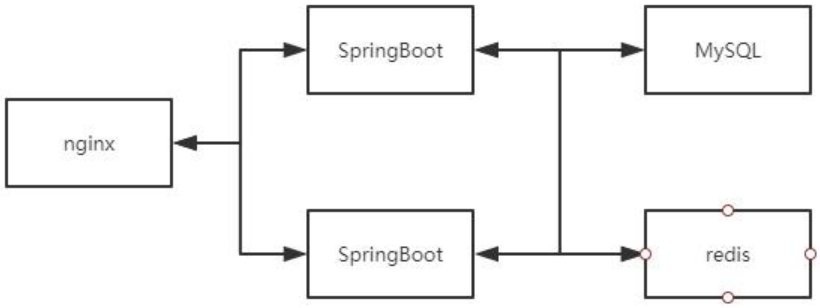
V3架构

1. 技术栈

- SpringBoot
- MySQL
- docker
- redis
- nginx

在v2的基础上加入了nginx来反向代理多实例的后端进行负载均衡。

2. 架构



3. 部署

- CentOS7（IP为 **192.168.56.101**）
- MobaXterm（SSH工具）

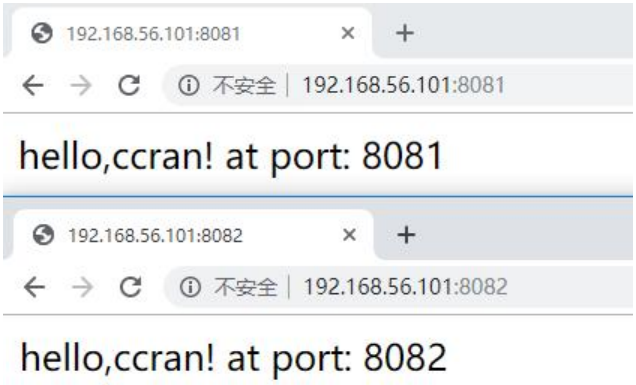
3.1 SpringBoot

本次还是使用后端架构学习（1）中的SpringBoot镜像

分别在8081端口和8082端口运行SpringBoot容器

```
docker run -d -p 8081:8081 -e JAVA_OPTS="-Dserver.port=8081" --name demo1 demo:1.0
docker run -d -p 8082:8082 -e JAVA_OPTS="-Dserver.port=8082" --name demo2 demo:1.0
```

然后通过IP+端口的方式访问，一切正常



3.2 nginx

首先是拉取nginx镜像

```
docker pull nginx
```

然后，运行nginx容器看下效果

```
docker run -d -p 80:80 --rm --name nginx nginx
```

接下来，在用户主目录下创建nginx文件夹，用于映射nginx容器内部的配置以及数据。

- conf文件夹用于映射配置文件
- www文件夹用于映射网页文件（也就是我们上图看到的那个页面）
- logs文件夹用于映射日志文件

```
[ccran@localhost ~]$ ls
demo.jar  Dockerfile
[ccran@localhost ~]$ pwd
/home/ccran
[ccran@localhost ~]$ mkdir ${PWD}/nginx
[ccran@localhost ~]$ mkdir ${PWD}/nginx/conf
[ccran@localhost ~]$ mkdir ${PWD}/nginx/logs
[ccran@localhost ~]$ mkdir ${PWD}/nginx/www
[ccran@localhost ~]$ ll nginx/
总用量 0
drwxrwxr-x. 2 ccran ccran 6 11月 30 20:55 conf
drwxrwxr-x. 2 ccran ccran 6 11月 30 20:55 logs
drwxrwxr-x. 2 ccran ccran 6 11月 30 20:55 www
```

然后，将nginx容器内部默认的配置拷贝到我们刚创建的文件夹下

```
docker cp nginx:/etc/nginx/nginx.conf /home/ccran/nginx/conf/nginx.conf
docker cp nginx:/etc/nginx/conf.d /home/ccran/nginx/conf/conf.d
```

接下来，关闭我们的nginx容器，并创建新的nginx容器

```
docker run -d -p 80:80 -p 8080:8080 --name nginx1 --link=demo1:demo1 --link=demo2:dem
2 \
-v /home/ccran/nginx/conf/nginx.conf:/etc/nginx/nginx.conf \
-v /home/ccran/nginx/conf/conf.d:/etc/nginx/conf.d \
-v /home/ccran/nginx/www:/usr/share/nginx/html \
-v /home/ccran/nginx/logs:/var/log/nginx nginx
```

- --link的作用是为了链接到我们的demo1容器和demo2容器
- -v的作用就是映射我们外部的文件到容器内部，方便我们进行配置或数据更改等操作。

demo1、demo2、nginx1都是容器，不像我们平时可以直接通过IP访问，因此需要通过--link链接来。

其实，进入nginx1容器内，看下/etc/hosts文件内容可以发现，每个容器是有各自的虚拟IP的，--lin

的作用就是将demo的域名和虚拟IP进行映射。

```
[ccran@localhost www]$ docker exec -it nginx1 bash
root@502d40032f55:/# more /etc/hosts
127.0.0.1       localhost
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
172.19.0.2      demo1 97b53917a666 # demo1的虚拟IP
172.19.0.3      demo2 6301c26b9b42 # demo2的虚拟IP
172.19.0.4      502d40032f55
```

好了，接下来就是改nginx配置文件就行了，进入conf文件夹，然后在http中添加如下信息。

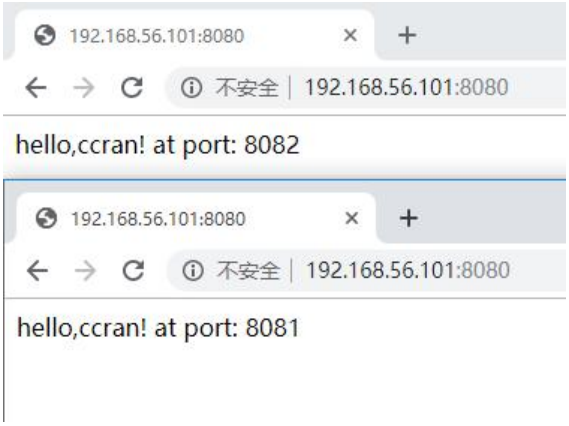
```
http {
    upstream demo{
        server demo1:8081;
        server demo2:8082;
    }

    server{
        listen 8080;
        location / {
                proxy_pass http://demo;
        }
    }
    ...
}
```

- nginx侦听8080端口
- 反向代理demo1、demo2

添加完成后，重启nginx镜像

```
docker restart nginx1
```

此时，访问8080端口，多次刷新访问，发现会轮询我们的demo1和demo2。



当然，我们也可以更改负载均衡的策略，设置权重来访问；每3次访问，一次访问demo1，两次访问dmo2

```
upstream demo{
        server demo1:8081 weight=1;
        server demo2:8082 weight=2;
    }
```

或者根据ip来进行访问，同一个ip访问固定的后端。

```
upstream demo{
        server demo1:8081;
        server demo2:8082;
        ip_hash;
  }
```

更多好玩的有待探索。smile

4. 总结

1. 通过nginx进行负载均衡提高后端的可用性（当然也会有一些坑，比如session）

2. 新的问题又来了，虽然后端可用性提高了，但是如果nginx挂了，整个后端就挂了。
mile