# nginx反向代理  负载均衡

<http://nginx.org/en/docs/http/ngx_http_upstream_module.html> (负载均衡官方方案)

软件负载均衡  lvs  nginx  haproxy

硬件负载均衡  f5  netscalar

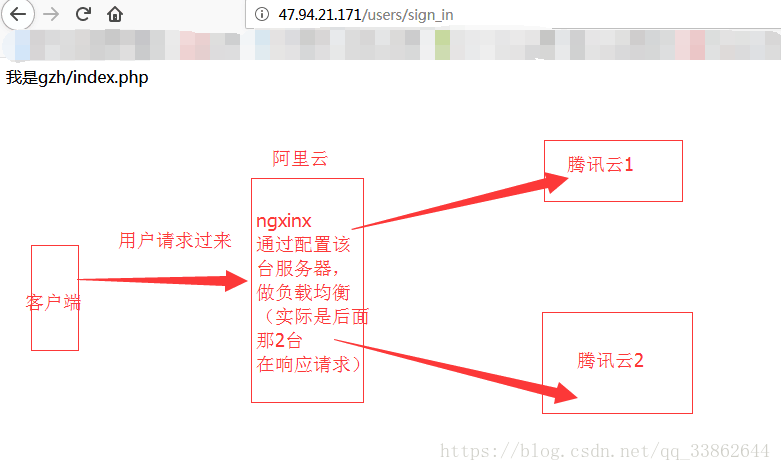
总结：1台就是反向代理，多台就是负载均衡

1、反向代理就是后端服务不直接对外暴露,请求首先发送到nginx,然后nginx将请求转发到后端服务器,比如tomcat php等.如果后端服务只有一台服务器,nginx在这里只有一个作用就是起到了代理后端服务接收请求的作用.称之为反向代理.

2、可是在现实的应用场景中,一台后端服务器出现单点故障的概率很大或者单台机器的吞吐量有限,无法承担过多请求.这时候就需要在nginx后端配置多台服务器,利用nginx内置的规则讲请求转发到后端不同的机器上.这时候就起到了负载均衡的作用.

# 原理、效果、配置文件

注意：访问的test.xiyang66.top域名指向的ip是47.94.21.171，所以用域名也行，用ip也行





**配置文件这样写**

//写在 http中，upstream backend和server是平级

upstream backend {

     server ip:81;

     server ip:82;

}

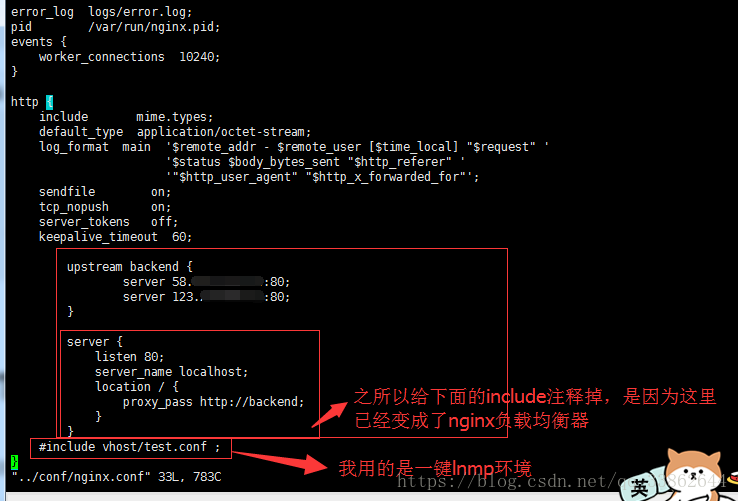
server {

    location / {

            proxy\_pass http://backend;  //这的backend是upstream后面的

    }

}



修改完别忘了重启nginx

## ****upstream目前支持 4 种方式的分配****

1)、轮询（也叫RR，是默认的策略）

　　每个请求按时间顺序逐一分配到不同的后端服务器，如果后端服务器down掉，能自动剔除

        upstream backend {

                server 58.87.xxx.xxx:80;

                server 123.xxx.xxx.xxx:80;

        }

        server {

            listen 80;

            server\_name localhost;

            location / {

                proxy\_pass http://backend;

            }

        }

**当访问calohost:81时,如果localhost:8081 挂了,也不会影响正常访问**

2)、weight 权重

　　指定轮询几率，weight和访问比率成正比，用于后端服务器性能不均的情况。

upstream test {

     server localhost:8080 weight=9;

     server localhost:8081 weight=1;

}

**10次一般只会有1次会访问到8081，而有9次会访问到8080**

3)、ip\_hash

　　每个请求按访问ip的hash结果分配，这样每个访客固定访问一个后端服务器，可以解决session的问题。

PR 和权重 在有session 进行访问时,由于随机访问,导致session无效,需要使用ip\_hash

upstream test {

    ip\_hash;

    server localhost:8080;

    server localhost:8081;

}

**4.1)、fair（第三方）**

按后端服务器的响应时间来分配请求，响应时间短的优先分配。

按后端服务器的响应时间来分配请求，响应时间短的优先分配。

upstream backend {

     fair;

     server localhost:8080;    //基于多端口

     server localhost:8081;

}

4.2)、url\_hash（第三方）

按访问url的hash结果来分配请求，使每个url定向到同一个后端服务器，后端服务器为缓存时比较有效。 在upstream中加入hash语句，server语句中不能写入weight等其他的参数，hash\_method是使用的hash算法

upstream backend {

    hash $request\_uri;

    hash\_method crc32;

    server localhost:8080;    //基于多端口

    server localhost:8081;

}

# 指令-热启动

Nginx重新读取配置的命令

nginx -s reload