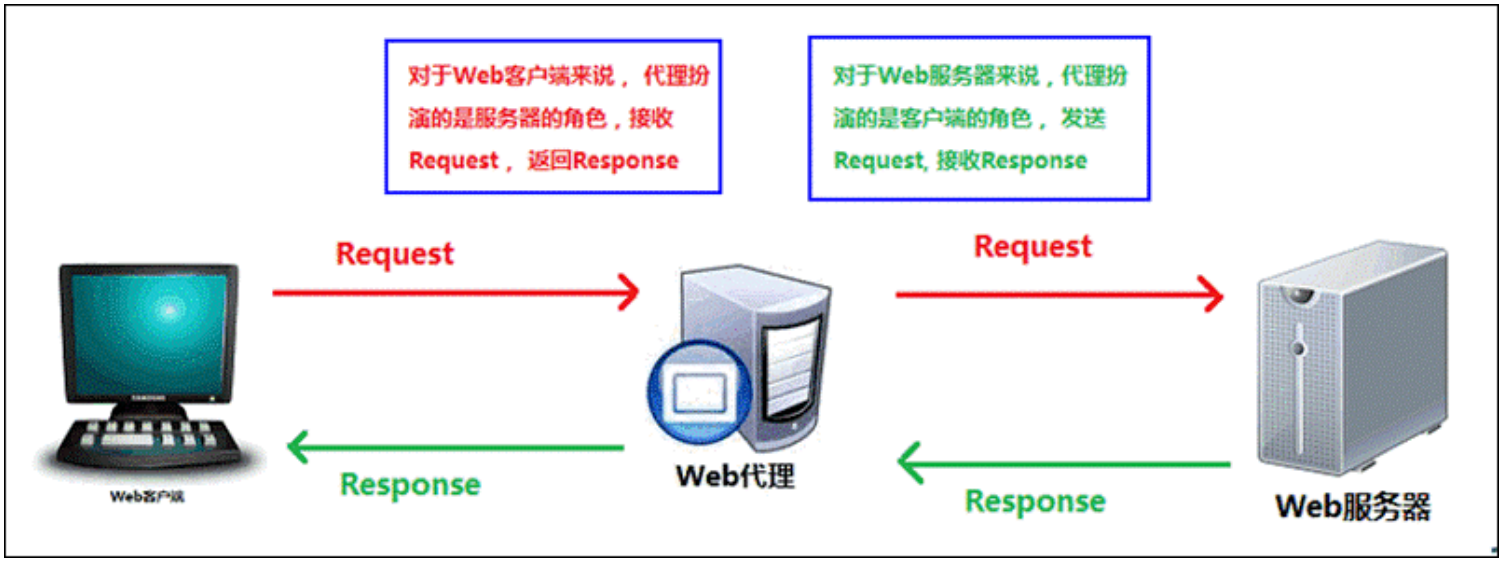
1. Nginx简介

一、 概念

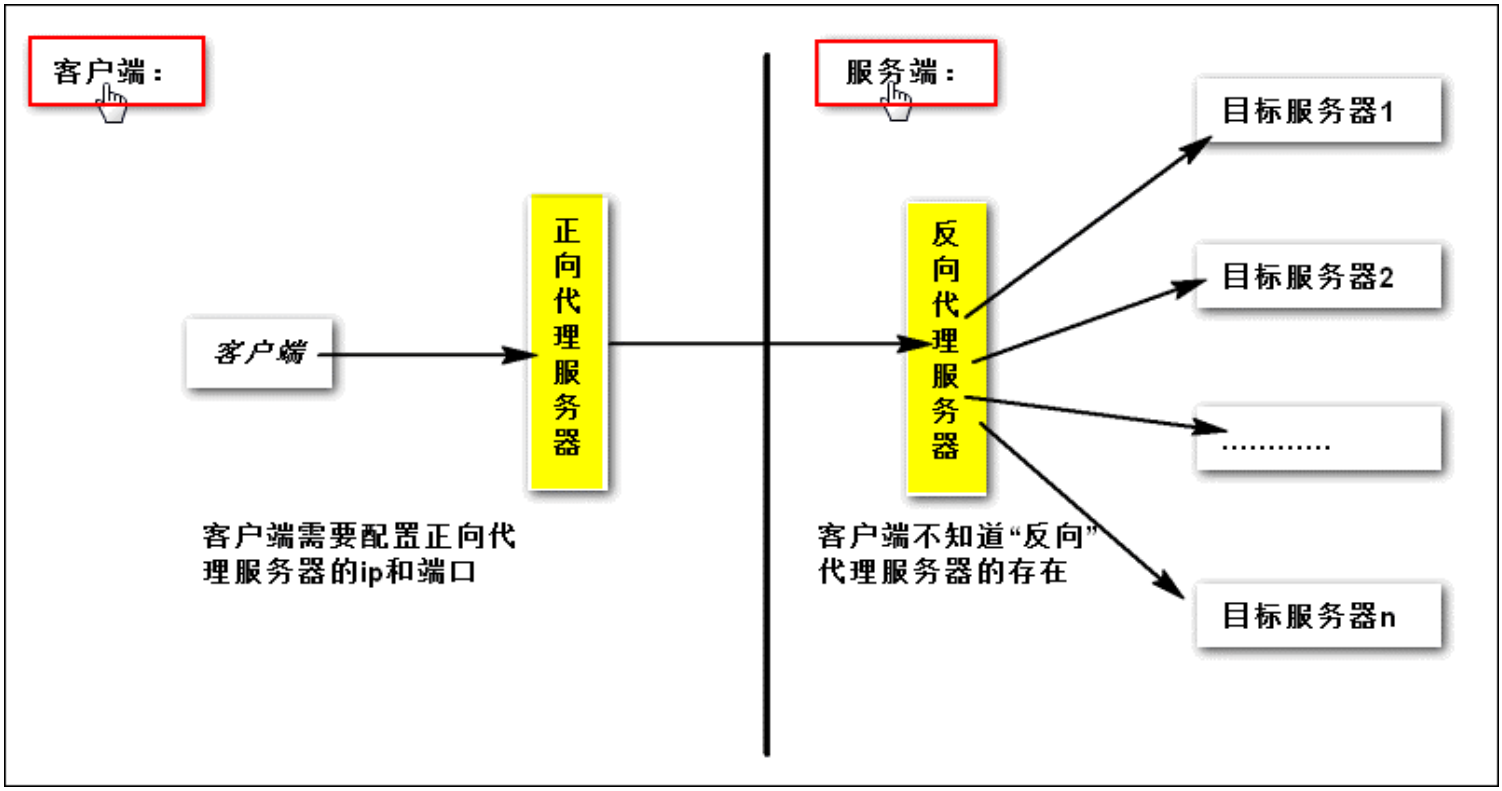
Nginx——Ngine X，是一款高性能的反向代理服务器；也是一个IMAP、POP3、SMTP代理服务器；也是一个Http服务器。也就是说Nginx本身就可以托管网站，进行Http服务处理，也可以作为反向代理服务器使用。

二、 正向代理和反向代理

首先，代理服务器一般指局域网内部的机器通过代理服务器发送请求到互联网上的服务器，代理服务器一般作用在客户端。例如：GoAgent翻墙软件。

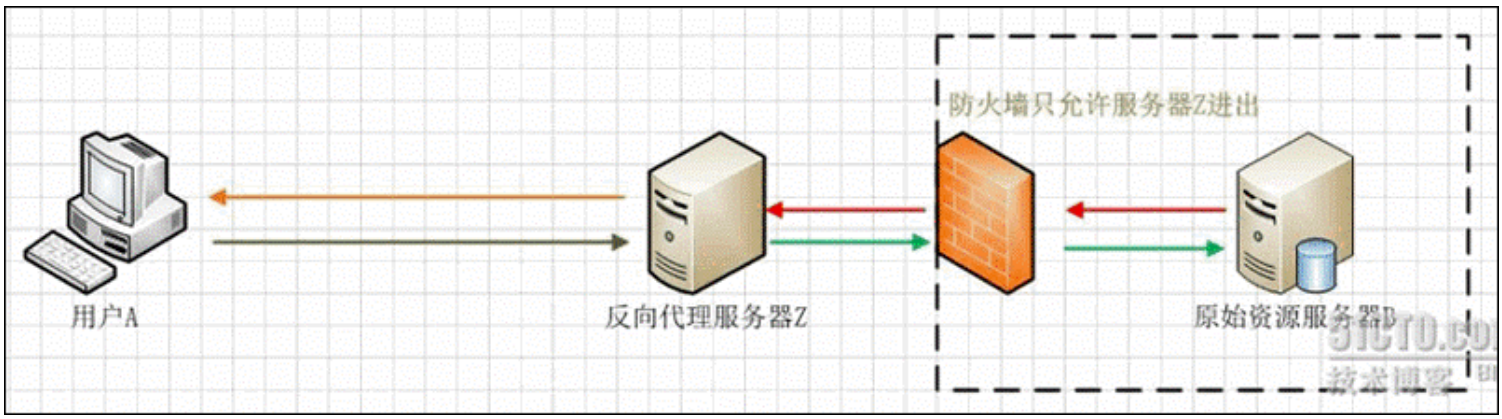


反向代理服务器作用在服务器端，它在服务器端接收客户端的请求，然后将请求分发给具体的服务器进行处理，然后再将服务器的相应结果反馈给客户端。Nginx就是一个反向代理服务器软件。



从上图可以看出：客户端必须设置正向代理服务器，当然前提是要知道正向代理服务器的IP地址，还有代理程序的端口。

反向代理正好与正向代理相反，对于客户端而言代理服务器就像是原始服务器，并且客户端不需要进行任何特别的设置。客户端向反向代理的命名空间（name-space）中的内容发送普通请求，接着反向代理将判断向何处（原始服务器）转交请求，并将获得的内容返回给客户端。



三、 特点

Ø 跨平台：可以在大多数Unix like 系统编译运行。而且也有Windows的移植版本。

Ø 配置异常简单：非常的简单，易上手。

Ø 非阻塞、高并发连接：数据复制时，磁盘I/O的第一阶段是非阻塞的。官方测试能支持5万并发连接，实际生产中能跑2~3万并发连接数（得益于Nginx采用了最新的epoll事件处理模型（消息队列）。

Ø Nginx代理和后端Web服务器间无需长连接；

Ø Nginx接收用户请求是异步的，即先将用户请求全部接收下来，再一次性发送到后端Web服务器，极大减轻后端Web服务器的压力。

Ø 发送响应报文时，是边接收来自后端Web服务器的数据，边发送给客户端。

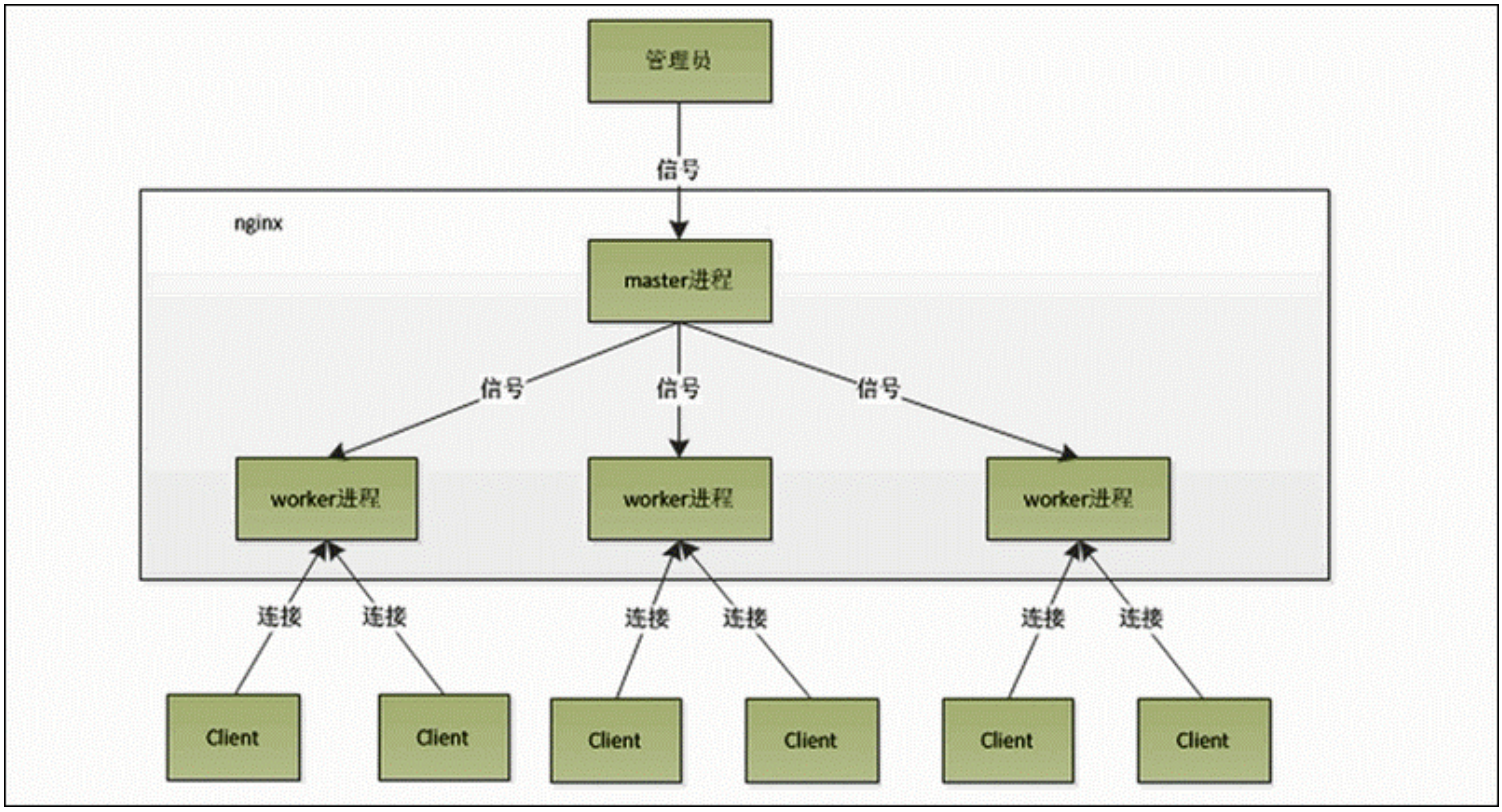
Ø 网络依赖性低，理论上只要能够ping通就可以实施负载均衡，而且可以有效区分内网、外网流量。

Ø 支持内置服务器检测。Nginx能够根据应用服务器处理页面返回的状态码、超时信息等检测服务器是否出现故障，并及时返回错误的请求重新提交到其它节点上。

Ø 采用Master/worker多进程工作模式

Ø 此外还有内存消耗小、成本低廉（比F5硬件负载均衡器廉价太多）、节省带宽、稳定性高等特点。

四、 内部进程模型



Nginx是以多进程的方式来工作的，当然Nginx也是支持多线程的方式的,只是我们主流的方式还是多进程的方式，也是Nginx的默认方式。Nginx采用多进程的方式有诸多好处。

Nginx在启动后，会有一个master进程和多个worker进程。master进程主要用来管理worker进程，包含：接收来自外界的信号，向各worker进程发送信号，监控 worker进程的运行状态,当worker进程退出后(异常情况下)，会自动重新启动新的worker进程。而基本的网络事件，则是放在worker进程中来处理了。多个worker进程之间是对等的，他们同等竞争来自客户端的请求，各进程互相之间是独立的。一个请求，只可能在一个worker进程中处理，一个worker进程，不可能处理其它进程的请求。worker进程的个数是可以设置的，一般我们会设置与机器CPU核数一致，这里面的原因与Nginx的进程模型以及事件处理模型是分不开的。

五、 处理请求

首先，Nginx在启动时，会解析配置文件，得到需要监听的端口与IP地址，然后在Nginx的master进程里面，先初始化好这个监控的socket(创建socket，设置addrreuse等选项，绑定到指定的IP地址端口，再listen)，然后再fork(一个现有进程可以调用fork函数创建一个新进程。由fork创建的新进程被称为子进程)出多个子进程出来，然后子进程会竞争accept新的连接。

此时，客户端就可以向Nginx发起连接了。当客户端与Nginx进行三次握手，与Nginx建立好一个

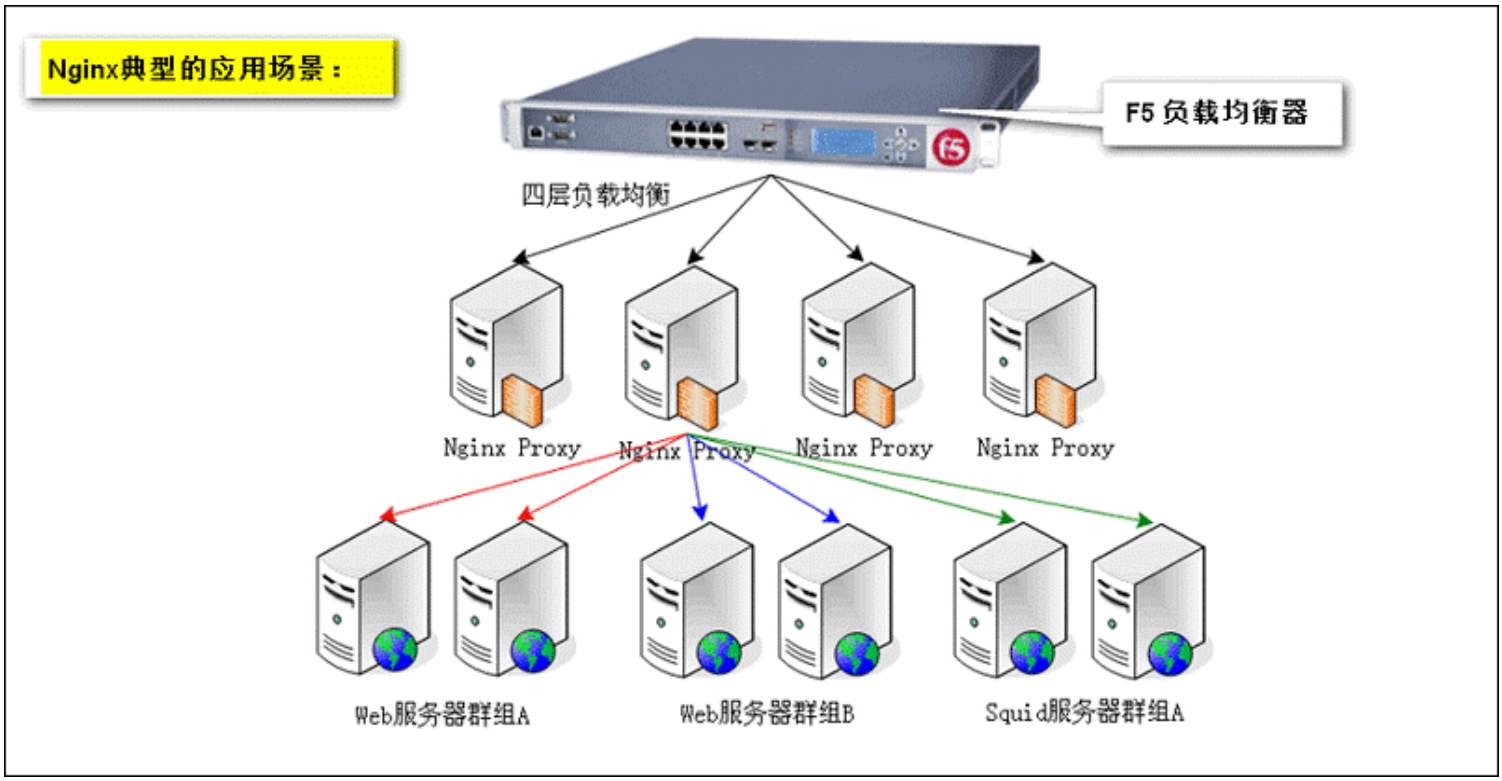
连接后，某一个子进程会accept成功，得到这个建立好的连接的socket，然后创建Nginx对连接的封装，即ngx_connection_t结构体。

接着，设置读写事件处理函数并添加读写事件来与客户端进行数据的交换。最后，Nginx或客户端来主动关掉连接，到此，一个连接就寿终正寝了。

六、 实际应用

由于Nginx是由俄罗斯人写的，所以，Nginx 已经在俄罗斯最大的门户网站——Rambler Media（www.rambler.ru）上运行了3年时间，同时俄罗斯超过20%的虚拟主机平台采用Nginx作为反向代理服务器。

在国内，已经有淘宝、新浪博客、新浪播客、网易新闻、六间房、56.com、Discuz!、水木社区、豆瓣、YUPOO、海内、迅雷在线等多家网站使用 Nginx 作为Web服务器或反向代理服务器。



2. Nginx+Tomcat搭建高性能负载均衡集群

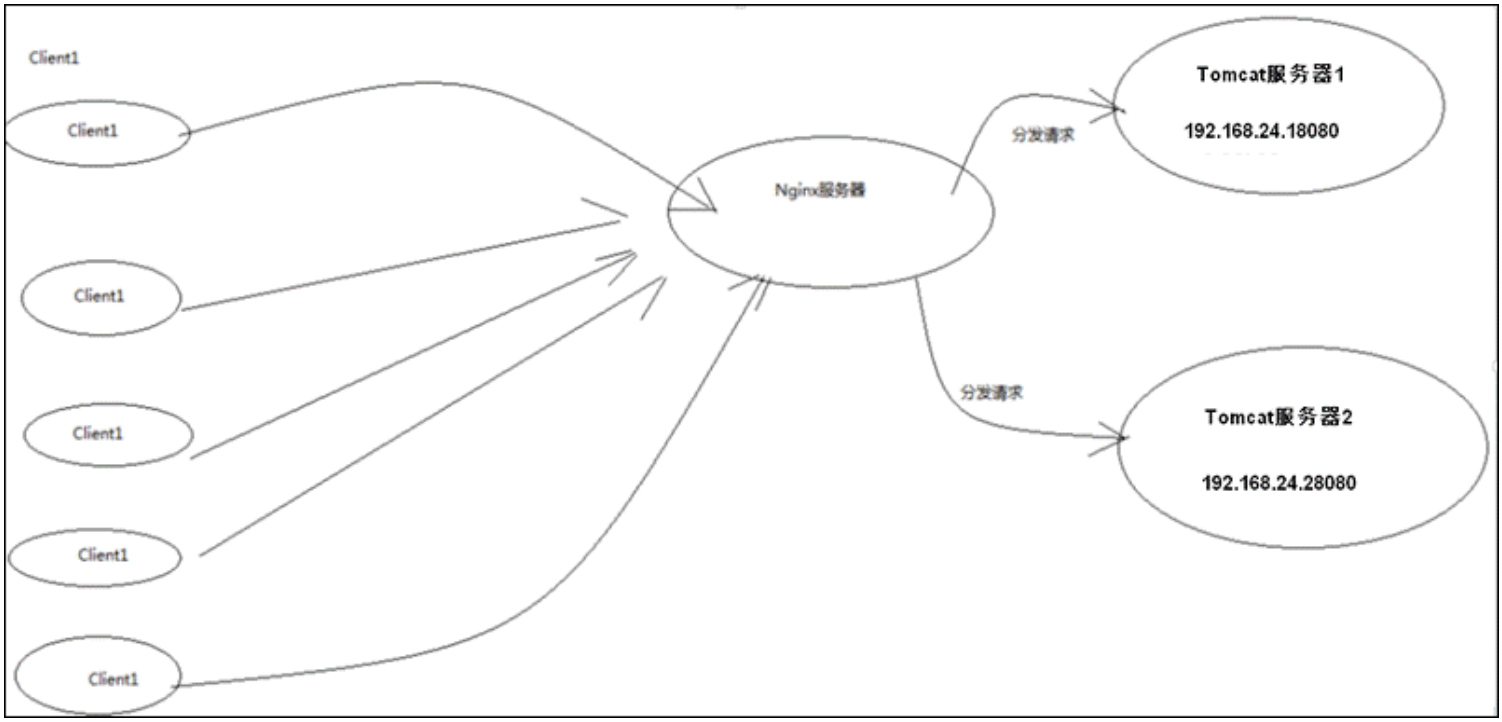
一、 工具

nginx-1.8.0

apache-tomcat-6.0.33

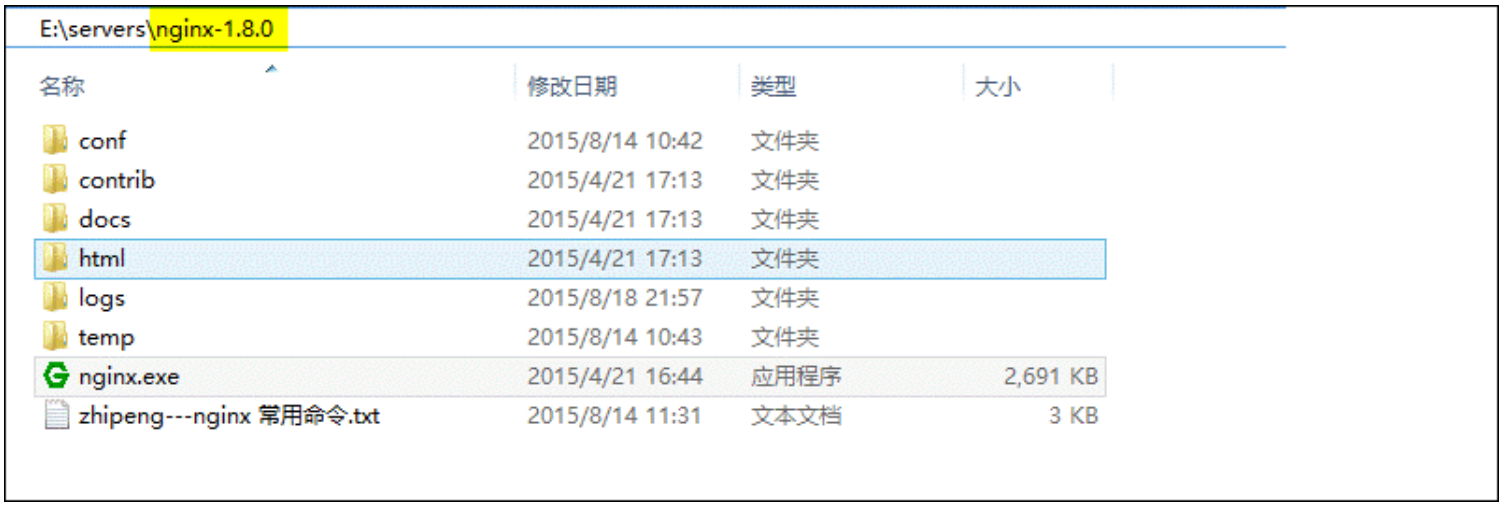
二、 目标

实现高性能负载均衡的Tomcat集群：



三、 步骤

1、首先下载Nginx，要下载稳定版：



2、然后解压两个Tomcat，分别命名为apache-tomcat-6.0.33-1和apache-tomcat-6.0.33-2：

3、然后修改这两个Tomcat的启动端口，分别为18080和28080，下面以修改第一台Tomcat为例

，打开Tomcat的conf目录下的server.xml：

共需修改3处端口：

```
    <Connector port="18080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />
```

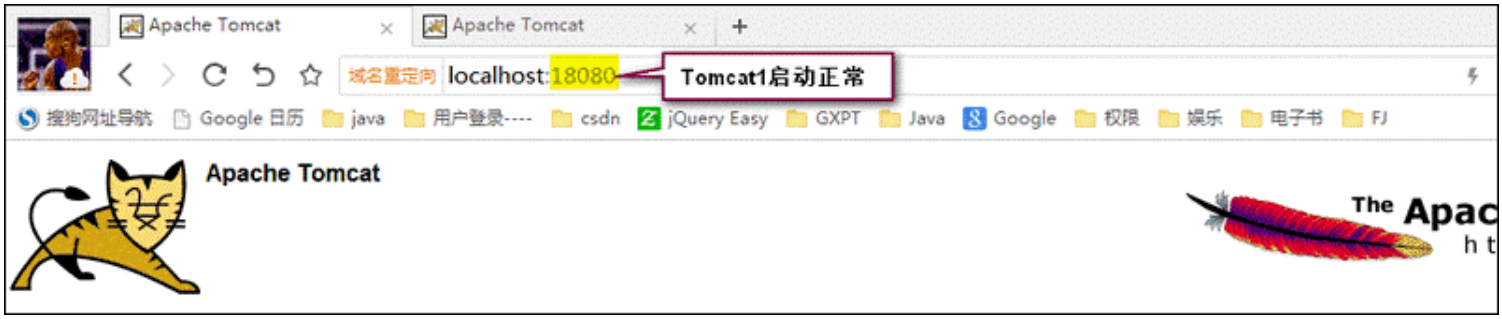
```
<Connector port="18080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

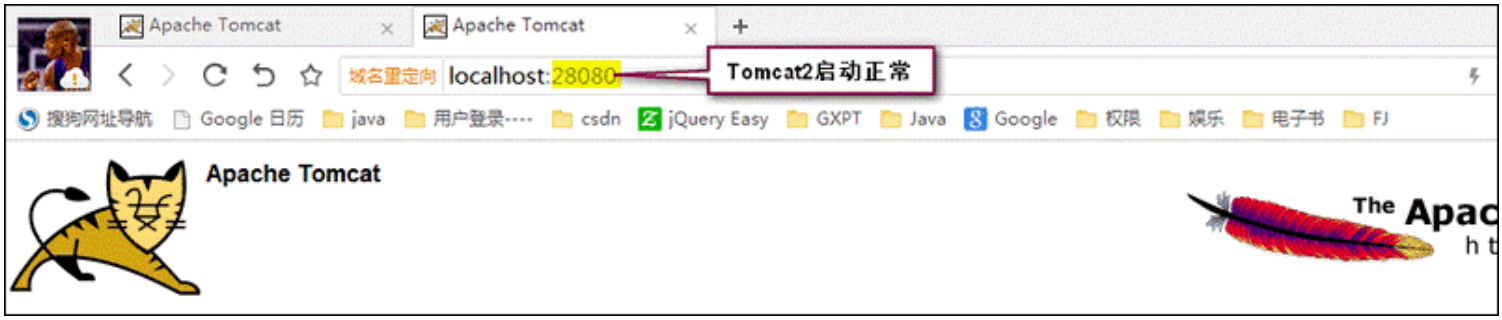
```
<!-- Define an AJP 1.3 Connector on port 8009 -->
<Connector port="18009" protocol="AJP/1.3" redirectPort="8443" />
```

当然第二台Tomcat也一样，如下图：

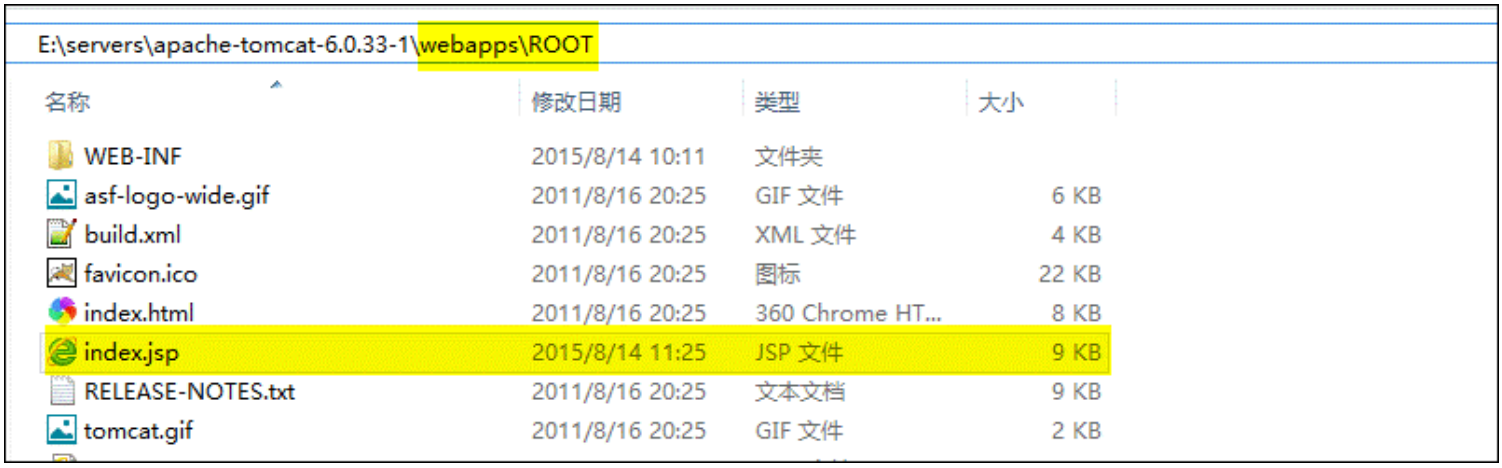
```
<Server port="28005" shutdown="SHUTDOWN">

    <Connector port="28080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />

<Connector port="28009" protocol="AJP/1.3" redirectPort="8443" />
```

4、然后启动两个Tomcat，并访问，看是否正常：

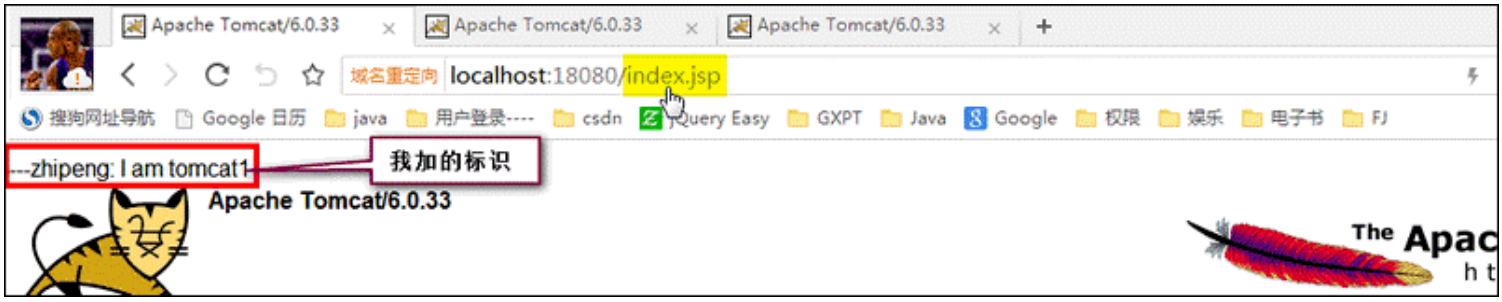




5、然后修改上面两个Tomcat的默认页面（为了区分下面到底访问的是那一台Tomcat，随便改一下即可）：



改完以后，进行访问，如下图：



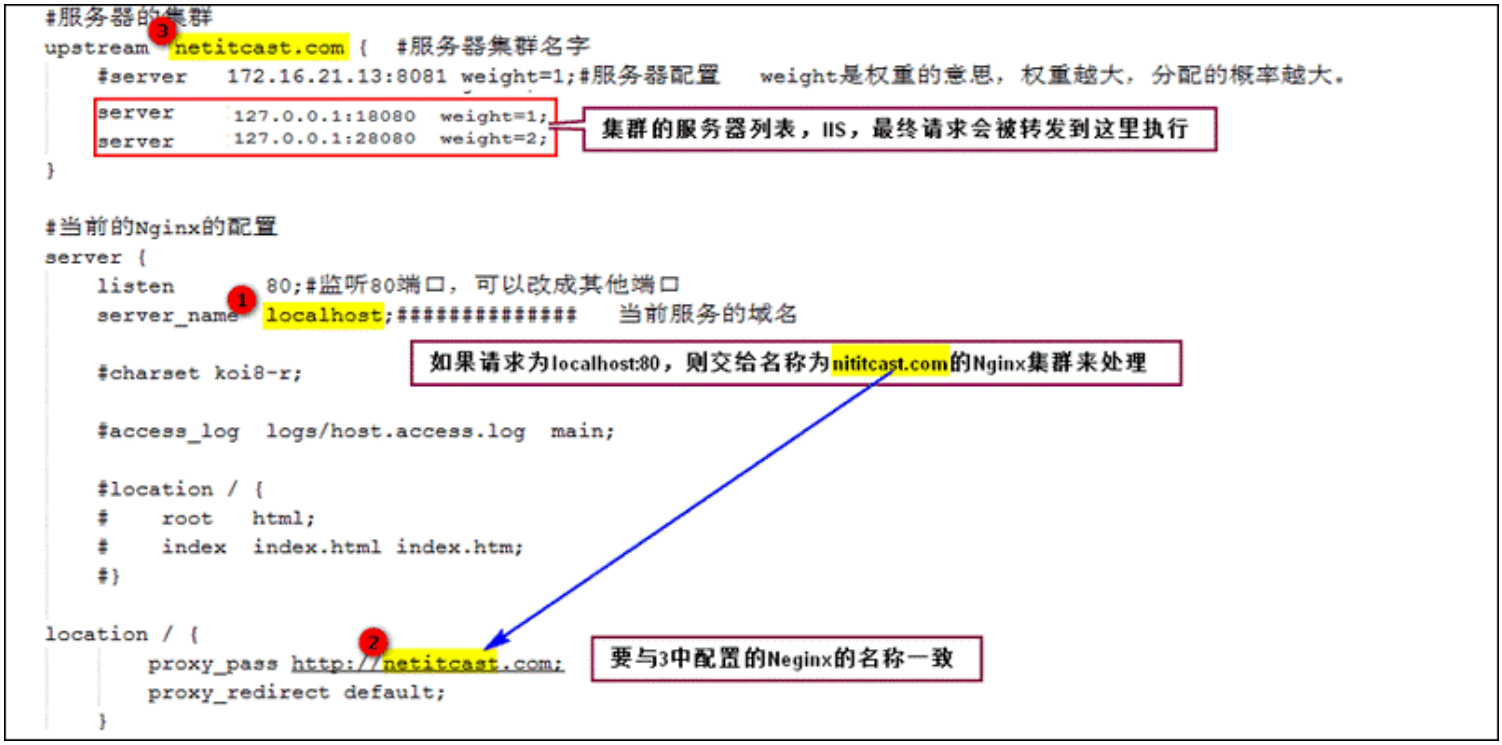
6、OK，现在我们可以开始配置Nginx来实现负载均衡了，其实非常的简单，只需要配置好Nginx的配置文件即可：

配置如下（这里只进行了简单的配置，实际生产环境可以进行更详细完善配置）：

```
worker_processes  1;#cpu
events {
    worker_connections  1024;#=*
}
http {
    include       mime.types; #
    default_type  application/octet-stream;#
    sendfile        on;#sendfilenginxsendfile onIOoffI/Ooff

 keepalive_timeout  65; #
    gzip  on;#Gizp
```

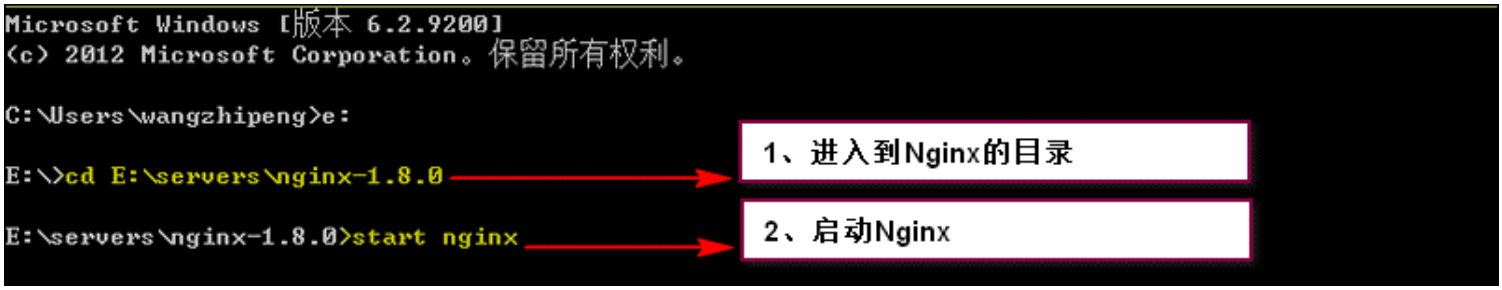
```
 #
    upstream  netitcast.com {  #
  server    127.0.0.1:18080  weight=1;#   weight
  server    127.0.0.1:28080  weight=2;
 }
 #Nginx
    server {
        listen       80;#80
        server_name  localhost;##############
 location / {
            proxy_pass http://netitcast.com;
            proxy_redirect default;
        }

        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
}
```

核心配置如下：



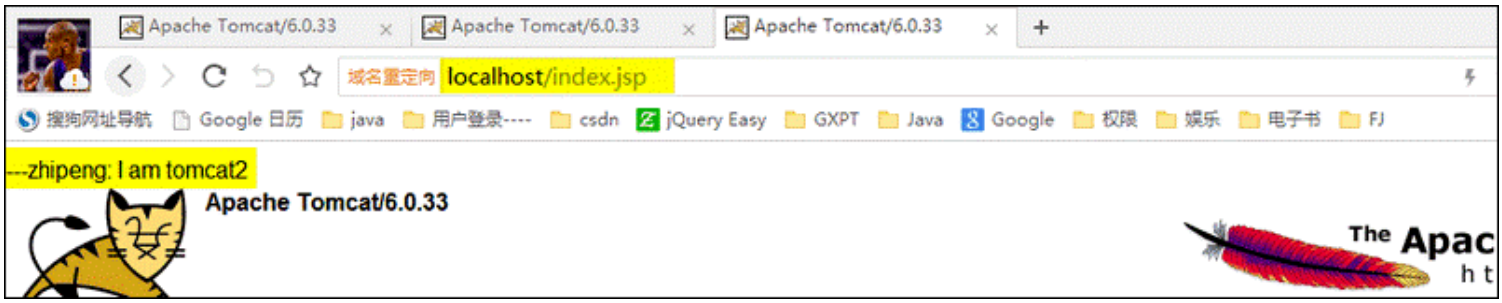
到此配置完成，下面开始演示负载均衡。

7、首先，我们启动Nginx：

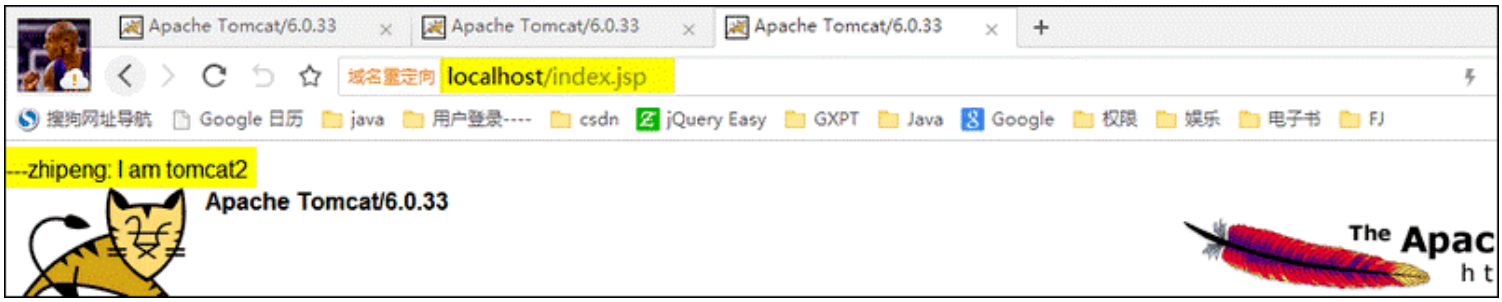


8、然后我们即可输入：localhost/index.jsp查看运行状况了

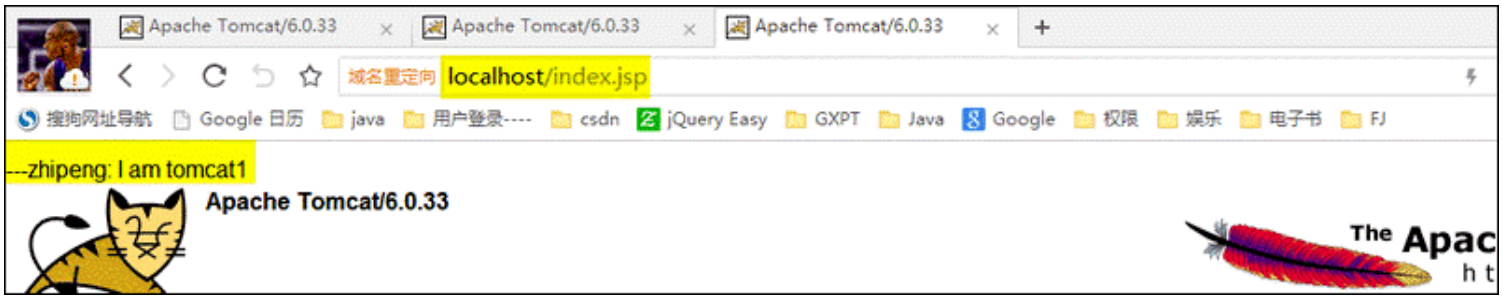
第一次访问，发现访问的是Tomcat2上的程序：



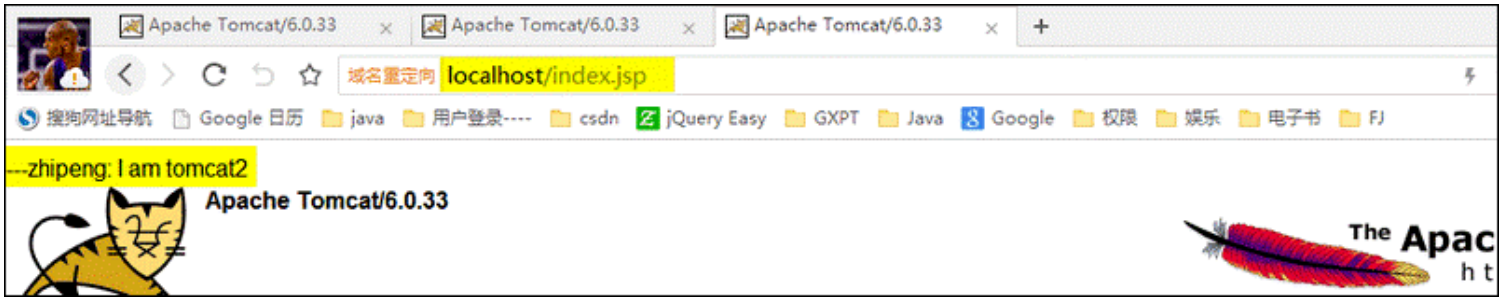
然后刷新，访问的还是Tomcat2上的程序：



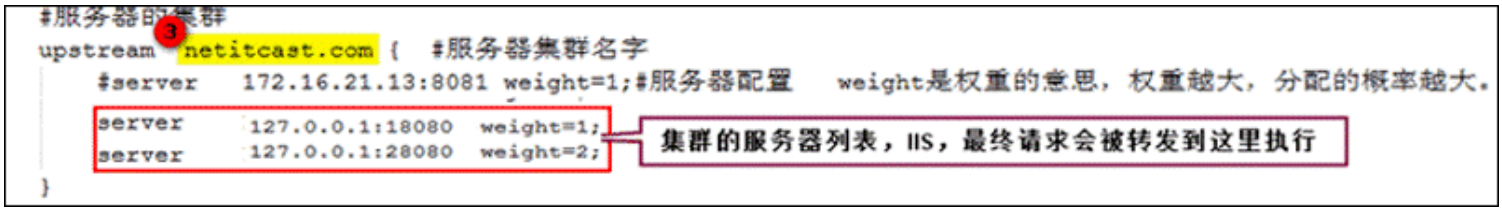
再刷新，发现变为了Tomcat1上的程序：



再刷新，发现又变为了Tomcat2上的程序:



到此，我们利用Nginx已经实现了负载均衡的Tomcat集群。我们不断的刷新，发现访问Tomcat2的概率大概是Tomcat1的2倍，这是因为我们在Nginx中配置的两台Tomcat的权重起的作用，如下图：



四、 总结

谁能想到实现一个高性能的负载均衡集群会如此简单。Nginx的功能如此强大，配置却如此简单，我们还有什么理由拒绝它呢？这比我们动不动就十多万至几十万人民币的F5 BIG-IP、NetScaler等硬件负载均衡交换机廉价了不知多少。此外，大家别忘了Nginx不仅仅是一个反向代理服务器，它本身也可以托管网站，作为Web服务器，进行Http服务处理。