# 前提

学习NgInx服务器要会：原理、反向代理操作，负载均衡的操作

NgInx服务器：Nginx是一款高性能的http 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器

反向代理：两个域名指向同一台nginx服务器，用户访问不同的域名显示不同的网页内容

其实就是通过nginx服务器默认端口80指向另一个服务器(tomcat)的操作

负载均衡：如果一个服务由多条服务器提供，需要把负载分配到不同的服务器处理

通俗就是：集群的服务器，让它平均的访问操作

# nginx

什么是nginx

Nginx是一款高性能的http 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器。由俄罗斯的程序设计师Igor Sysoev所开发，官方测试nginx能够支支撑5万并发链接，并且cpu、内存等资源消耗却非常低，运行非常稳定。

应用场景

1、http服务器：Nginx是一个http服务可以独立提供http服务。可以做网页静态服务器。

2、虚拟主机：可以实现在一台服务器虚拟出多个网站。例如个人网站使用的虚拟主机。

3、反向代理,负载均衡：当网站的访问量达到一定程度后，单台服务器不能满足用户的请求时，需要用多台服务器集群可以使用nginx做反向代理。并且多台服务器可以平均分担负载，不会因为某台服务器负载高宕机而某台服务器闲置的情况。

## nginx安装

### ubuntu与centos安装方式

下载nginx：官方网站：http://nginx.org/ 使用的版本是1.8.0版本。

Nginx只提供的源码。因为每个linux很多版本安装方式都不一样

要求的安装环境

1、需要安装gcc的环境。

命令 yum install gcc-c++，ubuntu：apt-get install gcc

2、第三方的开发包PCRE、zlib、openssl

PCRE(Perl Compatible Regular Expressions)是一个Perl库，包括 perl 兼容的正则表达式库。nginx的http模块使用pcre来解析正则表达式，所以需要在linux上安装pcre库。

命令 yum install -y pcre pcre-devel

ubuntu系统命令:sudo apt-get install libpcre3 libpcre3-dev

注：pcre-devel是使用pcre开发的一个二次开发库。nginx也需要此库。

zlib库提供了很多种压缩和解压缩的方式，nginx使用zlib对http包的内容进行gzip，所以需要在linux上安装zlib库。

命令 yum install -y zlib zlib-devel

ubuntu系统命令: sudo apt-get install zlib1g-dev

OpenSSL 是一个强大的安全套接字层密码库，囊括主要的密码算法、常用的密钥和证书封装管理功能及SSL协议，并提供丰富的应用程序供测试或其它目的使用。

nginx不仅支持http协议，还支持https（即在ssl协议上传输http），所以需要在linux安装openssl库。

yum install -y openssl openssl-devel

ubuntu系统命令：apt-get install openssl

Nginx安装步骤

第一步：cd到nginx源代码目录

第二步：解压缩

[root@localhost ~]# tar zxf nginx-1.8.0.tar.gz

第三步：解压完进入根目录执行./configure命令创建makeFile文件。\代表换行

./configure \

--prefix=/usr/local/nginx \ / /指定安装目录

--pid-path=/var/run/nginx/nginx.pid \

--lock-path=/var/lock/nginx.lock \

--error-log-path=/var/log/nginx/error.log \

--http-log-path=/var/log/nginx/access.log \

--with-http\_gzip\_static\_module \

--http-client-body-temp-path=/var/temp/nginx/client \

--http-proxy-temp-path=/var/temp/nginx/proxy \

--http-fastcgi-temp-path=/var/temp/nginx/fastcgi \

--http-uwsgi-temp-path=/var/temp/nginx/uwsgi \

--http-scgi-temp-path=/var/temp/nginx/scgi

第四步 创建目录 必须要自己先创建 不然报错

注意：启动nginx之前，上边将临时文件目录指定为/var/temp/nginx，需要在/var下创建temp及nginx目录

[root@localhost sbin]# mkdir /var/temp/nginx/client -p

第五步：执行命令：make

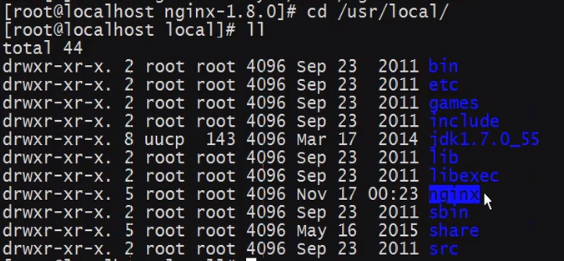
进行编译生成的makeFile文件

第六步 make install

这里就不用指定安装目录了，因为在创建makeFile文件时已经指定安装目录了

查看安装好的目录

cd /usr/local/ ls –lh



### mac安装方式

前言：nginx的强大不必多言，所以很自然很应当的我也来玩一玩。但是网上许多安装和配置都太那啥了，在这里真的想说一句带有引战性质但我真的无意引战的言论：现在的好多程序员的文笔实在是烂，写的教程让人着实难懂啊，该不会语文都是体育老师教的吧。

当前系统环境：10.14测试没问题

安装工具:homebrew(Ubuntu的安装软件命令apt-get 和centos的yum) 看mac系统学习

步骤：

1、打开终端，习惯性命令：

brew update

//结果：Already up-to-date.

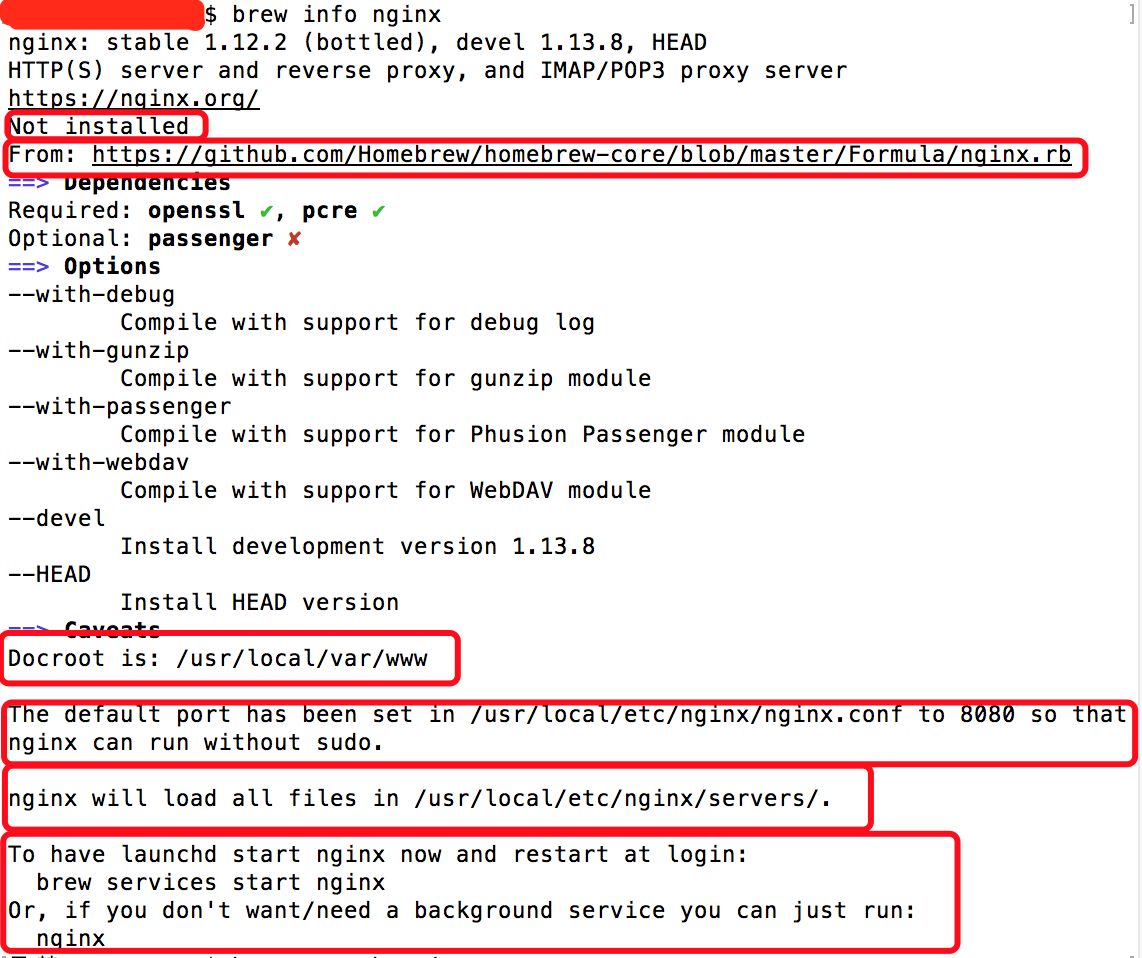
2、终端继续执行命令：

brew search nginx //查询要安装的软件是否存在

3、这里我们多执行一步“废”命令，不过有利于我们后面的配置：

brew info nginx

运行结果：



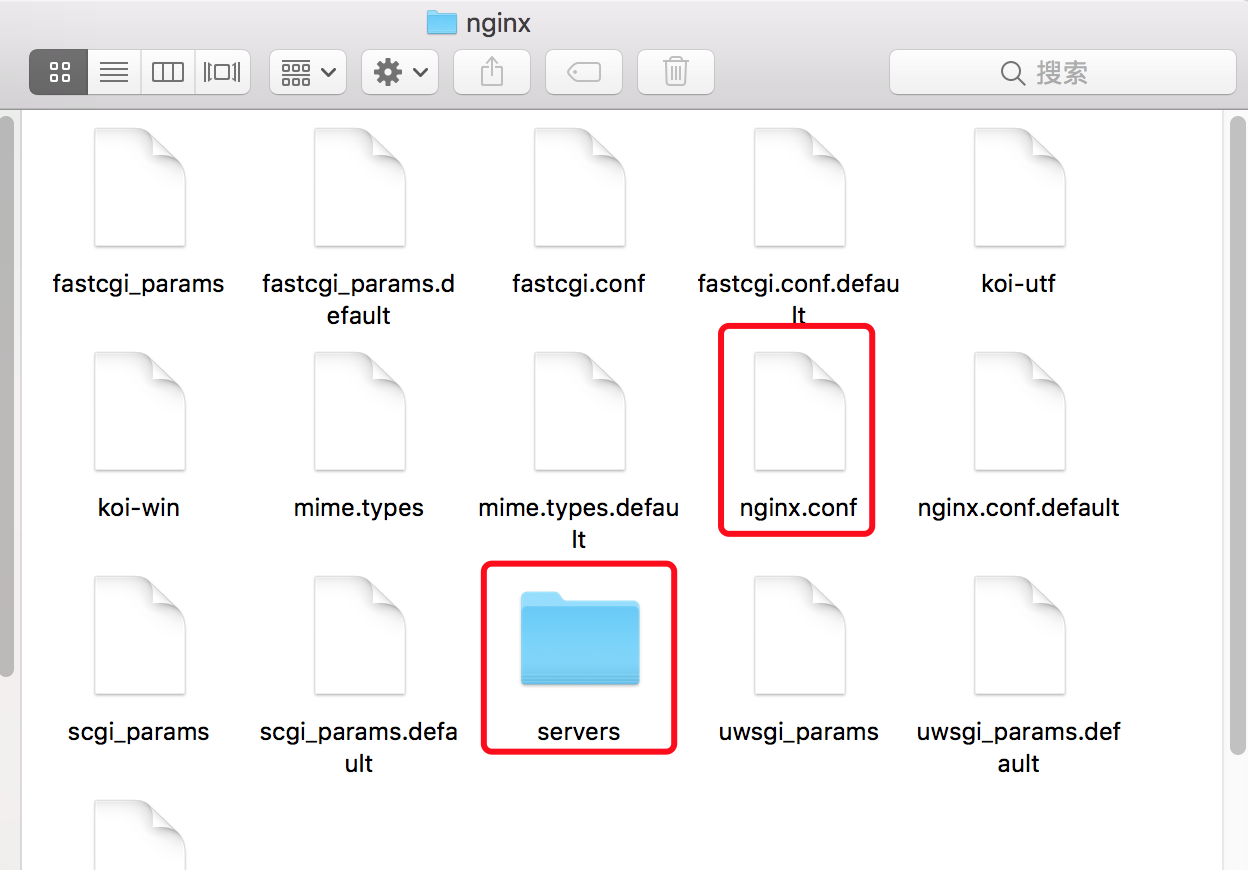
我们可以看到，nginx在本地还未安装（Not installed），nginx的来源（From），Docroot默认为/usr/local/var/www，在/usr/local/etc/nginx/nginx.conf配置文件中默认端口被配置为8080从而使nginx运行时不需要加sudo，nginx将在/usr/local/etc/nginx/servers/目录中加载所有文件，以及我们可以通过最简单的命令 ‘nginx’ 来启动nginx。

4、正式开始安装：只要这个命令就安装好了

brew install nginx

5、查看nginx安装目录（是否如info所说）：

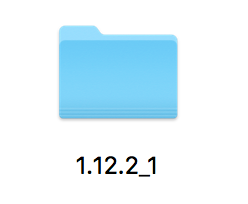
open /usr/local/etc/nginx/



成功打开nginx目录，也可以看到如info所说servers目录以及nginx.conf的配置文件（后面会用到这个配置文件）。但我们并没有找到nginx被安装到了哪里。

终端继续执行：

open /usr/local/Cellar/nginx //其实这个才是nginx被安装到的目录



会看到一个以当前安装的nginx的版本号为名称的文件夹，这个就是我们安装的nginx根目录啦。进入1.12.2\_1/bin 目录，会看到nginx的可执行启动文件。

　　同样的，我们在1.12.2\_1/目录下还可以看到一个名字为html的快捷方式文件夹（暂且就这么叫吧），进入该目录我们会发现其实它指向的就是/usr/local/var/www目录，这个在上面我们查看的info信息中有提到（Dcroot）

6、启动服务nginx，终端输入如下命令：

nginx //没有报错即为启动成功。

7、访问验证：

打开浏览器访问localhost:8080,这里跟网上的一些教程会有些不一样，正常情况下到这一步就会能看到nginx的欢迎界面啦，然而博主却遇到了坑爹的情况（如果你能正常看到nginx欢迎界面，可以直接跳过这一步了）

　　想必这个错误大家都很熟悉了，我就不再多做解释，接下来直接说原因（该原因仅为猜测，因为无从验证）：安装的nginx会默认在html（也就是/usr/local/var/www）目录下生成一个欢迎页面文件，而出现上面这种情况就是欢迎页面文件没有生成（至于为什么没有生成就不得而知了，一切都是猜测）。

那在解决这个问题之前，我们先来了解一下nginx的配置文件吧（nginx.conf）:

cat /usr/local/etc/nginx/nginx.conf

显示配置文件的代码为：

复制代码

#user nobody;

worker\_processes 1;

#error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.log info;

#pid logs/nginx.pid;

events {

worker\_connections 1024;

}

http {

include mime.types;

default\_type application/octet-stream;

#log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

# '$status $body\_bytes\_sent "$http\_referer" '

# '"$http\_user\_agent" "$http\_x\_forwarded\_for"';

#access\_log logs/access.log main;

sendfile on;

#tcp\_nopush on;

#keepalive\_timeout 0;

keepalive\_timeout 65;

#gzip on;

server {

　　　　 #侦听8080端口

listen 8080;

　　　　 #定义使用 localhost访问

server\_name localhost;

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

　　　　　　　#定义服务器的默认网站根目录位置

root html;

　　　　　　 #定义首页索引文件的名称

index index.html index.htm;

}

　　　　　...

　　　　　...

　　　　　... (注释代码太多，就不全部贴出来了)

include servers/\*;

}

　　通过配置文件我们可以看到其默认的网站根目录为html（即/usr/local/var/www），而默认的索引文件为index.html 和 index.htm，这下就找到原因了，原来我们的根目录少了首页索引文件，那就来手动创建一个吧：

cd /usr/local/var/www/ //进入到www目录下

touch index.html //创建一个新的index.html文件

vim index.html //编辑该文件

　　将如下代码写入index.html文件中：

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>title</title>

</head>

<body>

<div>

<h1>我的nginx欢迎页面</h1>

</div>

</body>

</html>

按esc键，输入:wq推出编辑并保存（这个相信大家都会，但还是强迫症的写上了）。

## Nginx命令

进入 /usr/local/nginx/sbin 目录

启动nginx服务 注意mac系统在终端命令：直接nainx(应该是自动配置了环境变量)

[root@localhost sbin]# ./nginx //没有任何提示证明成功

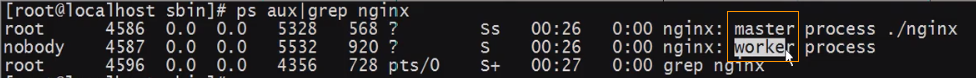
//如果出现

../../../屏幕快照%202019-03-02%20上午11.40.06.png

证明没有此目录：需要自己创建

查看进程

ps aux |grep nginx 会有两个进程一个master worker



必须要有两个才好使

访问nginx

ubuntu和contos默认是80端口。Mac系统默认是8080

注意：linux是否关闭防火墙。

直接在另一台主机浏览器中访问：http://192.168.0.104/ 即可或本机 localhost



关闭nginx服务：

[root@localhost sbin]# ./nginx -s stop

推荐使用：

[root@localhost sbin]# ./nginx -s quit

重启nginx：

1、先关闭后启动。

不重启nginx进行刷新配置文件：

[root@localhost sbin]# ./nginx -s reload

## 配置虚拟主机

要明白的就是nginx就是一台服务器：启动多个网站。

如何区分不同的网站：如淘宝 、天猫：1、域名不同，2、端口不同

两种方式配置虚拟主机

第一种 端口 演示通过端口区分不同虚拟机，(不怎么用端口配置,访问nginx一般都用80端口)

第一步 Nginx的配置文件：进行配置（就跟配置tomcat差不多，配置端口）路径文件：/usr/local/nginx/conf/nginx.conf

#从这里开始配置------配置第一个端口号80、要访问的目录、缺省文件名

server { #配置一个server可以说就是一个虚拟主机

listen 80; #这个是端口号，tomcat不能同时配置两个端口号

server\_name localhost; #这个是ip地址 ：可以写域名或ip

#charset koi8-r;

#access\_log logs/host.access.log main;

location / { #访问的路径

root html; #文件目录，这个路径是相对路径：相对于nginx安装根目录

index index.html index.htm; #首页的配置

}

}

#error\_page 500 502 503 504 /50x.html;

location = /50x.html { #发送504错误要访问的路径 这个可以不配

root html;

}

#可以配置第二个端口号81、要访问的目录、缺省文件名

server {

listen 81;

server\_name localhost;

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

root html18; #要访问的静态资源目录

index index.html index.htm; #和tomcat一样默认要进入的资源文件名

}

}

}

第二步 重新加载配置文件：sbin/nginx -s reload

第三步 为了区分在nginx目录中再新建一个html文件夹命名为html18

访问操作

http://192.168.0.104/ 访问的是默认配置的80端口进入/usr/local/nginx目录下的html文件夹的index.html

http://192.168.0.104:81/ 你上面配置的第二个端口默认进入的是html18文件夹的index.html

这三步就是配置不同的虚拟机端口操作

第二种 域名 演示通过域名区分虚拟主机 ( 后面负载均衡和反向代理要用到此方式)

域名就是网站

Dns服务器：把域名解析为ip地址。保存的就是域名和ip的映射关系。

一级域名：Taobao.com二级域名 www.baidu.com三级域名Aaa.image.baidu.com

一个域名对应一个ip地址，一个ip地址可以被多个域名绑定。

第一步 修改window的hosts文件：（C:\Windows\System32\drivers\etc）

可以配置域名和ip的映射关系，如果hosts文件中配置了域名和ip的对应关系，不需要走dns服务器。

第二步 Nginx的配置和上面端口配置的差不多，不同点：端口号不变还是80

server {

listen 80; #配置两个虚拟机端口号不变

server\_name www.taobao.com; #这里配置域名一

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

root html-taobao; #/usr/local/nginx目录下的文件夹名字

index index.html index.htm;

}

}

server {

listen 80;

server\_name www.baidu.com; #这里配置域名二

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

root html-baidu;

index index.html index.htm;

}

}

}

# 1、反向代理(重要)

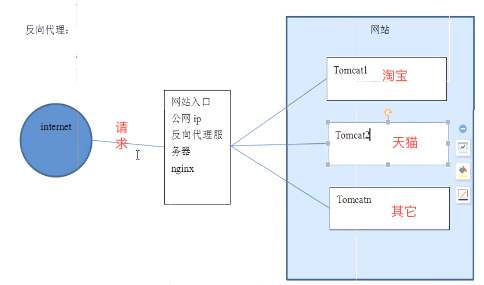
正向代理：用户角度：指的的是用户本身电脑连接到有网的然后返还结果给你

通俗话语：把请求发送到有网的电脑上：有网电脑拿到请求给对应的服务器如百度，响应回来后再响应给你

反向代理: 服务器角度：代理用户访问的地址，让此地址拿来访问哪一个服务器提供服务

两个域名指向同一台nginx服务器，用户访问不同的域名显示不同的网页内容

比喻：自己服务器有多个网站：如淘宝、天猫：我只有一个公网ip(每年要租的;一个域名只能绑定一个ip)，我淘宝、天猫两个域名如何绑定到一个公网ip上呢？这时候nginx起到一个用户请求的地址判断处理，再转发到相应的网站，这就是反向代理



第一步：安装两个tomcat，分别运行在8080和8081端口。

第二步：启动两个tomcat。

第三步：反向代理服务器的配置

用root用户打开：/usr/local/nginx/conf/nginx.conf

修改的主要位置：

1、添加tomcat服务器的ip和端口upstream tomcat1 {server 192.168.25.148:8080; }

2、location / { root html; 这个文件目录 改为 proxy\_pass http://tomcat1; #指向 上面红色配好的地址和端口号

upstream tomcat1 {

server 192.168.25.148:8080; #指定tomcat服务器的ip地址和端口号

}

server {

listen 80; #nginx的端口号

server\_name www.wwl1.com; #自己配置的域名 也就是用户在网址输入此域名会指定到此server

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

proxy\_pass http://tomcat1; #指向 上面红色配好的地址和端口号

index index.html index.htm; # 要访问的网址

}

}

upstream tomcat2 {

server 192.168.25.148:8081;

}

server {

listen 80;

server\_name www.wwl2.com;

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

proxy\_pass http://tomcat2;

index index.html index.htm;

}

}

2、刷新配置文件：进入 /usr/local/nginx/sbin 目录

命令： ./nginx -s reload

第四步 修改window的hosts文件：（C:\Windows\System32\drivers\etc）

linux 和mac的都在 /etc/hosts下修改

192.168.0.104 www.wwl2.com

192.168.0.104 www.wwl1.com

第五步 访问操作

http://www.wwl2.com/ nginx设置的就是80端口，nginx就会反向代理把请求转发给tomcat2所以就会访问tomcat2的主页 index.jsp

http://www.wwl1.com/ 就会访问tomcat1的主页 index.jsp

我们请求的是nginx，由nginx转发给指定的tomcat应用服务器，这就是反向代理

这5步就是反向代理访问不同的tomcat服务的操作

# 2、负载均衡(重要)

根据上面的步骤（反向代理步骤）

如果一个网站有多个服务器服务需要把负载分配到不同的服务器处理，需要负载均衡。

下面的配置就是在访问tomcat2时（平均访问）：访问8081服务器后访问8082服务器

upstream tomcat2 {

server 192.168.25.148:8081;

server 192.168.25.148:8082;

}

可以根据服务器的实际情况调整服务器权重。权重越高分配的请求越多，权重越低，请求越少。默认是都是1，

upstream tomcat2 {

server 192.168.25.148:8081;

server 192.168.25.148:8082 weight=2;#这写法就是让此服务器不是1比1形式访问

}

测试的时候要刷新快一点，才会出现效果

# 3、Nginx高可用(了解)

了解：老师说一般都是运维操作的

就是 解决了tomcat服务器的压力和挂掉的问题，谁来解决我（Nginx）挂掉呢，我一旦挂掉就都不行了

nginx作为负载均衡器，所有请求都到了nginx，可见nginx处于非常重点的位置，如果nginx服务器宕机后端web服务将无法提供服务，影响严重。

为了屏蔽负载均衡服务器的宕机，需要建立一个备份机。主服务器和备份机上都运行高可用（High Availability）监控程序，通过传送诸如“I am alive”这样的信息来监控对方的运行状况。当备份机不能在一定的时间内收到这样的信息时，它就接管主服务器的服务IP并继续提供负载均衡服务；当备份管理器又从主管理器收到“I am alive”这样的信息时，它就释放服务IP地址，这样的主服务器就开始再次提供负载均衡服务。

使用到的技术 ：keepalived+nginx实现主备

什么是keepalived 通俗就是：两台装了此软件备用机连续访问主用机”你死了没有”死了就自己顶上，主用机复活了自动又抢回nginx的服务

keepalived是集群管理中保证集群高可用的一个服务软件，用来防止单点故障。

Keepalived的作用是检测web服务器的状态，如果有一台web服务器死机，或工作出现故障，Keepalived将检测到，并将有故障的web服务器从系统中剔除，当web服务器工作正常后Keepalived自动将web服务器加入到服务器群中，这些工作全部自动完成，不需要人工干涉，需要人工做的只是修复故障的web服务器。

高可用环境

两台nginx，一主一备：192.168.101.3和192.168.101.4

两台tomcat服务器：192.168.101.5、192.168.101.6