# Nginx概述

nginx是由c编写的，能够支撑5万并发，CPU、内存资源占用小，运行稳定。

# Nginx基本操作

| **操作** | **命令** | **备注** |
| --- | --- | --- |
| 启动 | /usr/local/nginx/sbin/nginx |  |
| 退出 | /usr/local/nginx/sbin/nginx -s quit | 正常关闭Nginx，推荐使用 |
| 停止 | /usr/local/nginx/sbin/nginx -s stop | 相当于杀死Nginx进程，不推荐使用 |
| 重启 | /usr/local/nginx/sbin/nginx -s quit /usr/local/nginx/sbin/nginx |  |
| 重新加载配置文件 | /usr/local/nginx/sbin/nginx -s reload |  |

# Nginx原理

## nginx进程功能

启动Nginx后，其实就是在80端口启动了Socket服务进行监听，如图所示，Nginx涉及Master进程和Worker进程。

|  |
| --- |
| [root@xmm logs]# ps -ef | grep nginx | grep -v grep  root 8527 1 0 13:46 ? 00:00:00 nginx: master process ./sbin/nginx  nobody 8528 8527 0 13:46 ? 00:00:00 nginx: worker process  [root@xmm logs]# netstat -anp | grep nginx  tcp 0 0 0.0.0.0:80 0.0.0.0:\* LISTEN 8527/nginx: master |

master进程的作用：读取并验证配置文件nginx.conf；管理worker进程。

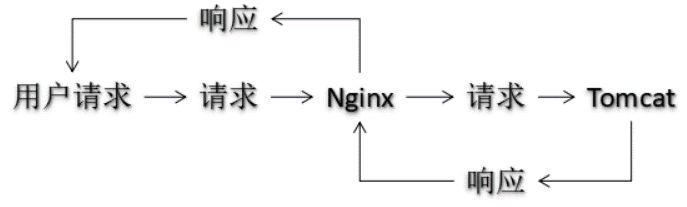
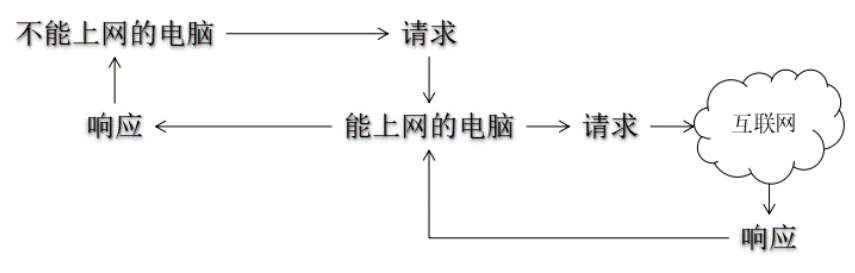
worker进程的作用：每一个Worker进程都维护一个线程（避免线程切换），处理连接和请求；注意Worker进程的个数由配置文件决定，一般和CPU个数相关（有利于进程切换），配置几个就有几个Worker进程。

## 正向代理和反向代理

正向代理“代理”的是客户端，而且客户端是知道目标的，而目标是不知道客户端是通过VPN访问的。反向代理“代理”的是服务器端，而且这一个过程对于客户端而言是透明的。

| **正向代理** | **反向代理** |
| --- | --- |
| 为浏览器进行代理 | 为服务器进行代理 |

在一个局域网中，只有一台计算机可以接入互联网。其他计算机通过这台计算机间接接入互联网。



## 热部署

所谓热部署，就是配置文件nginx.conf修改后，不需要stop Nginx，不需要中断请求，就能让配置文件生效！（nginx -s reload 重新加载/nginx -t检查配置/nginx -s stop）。

worker进程负责处理具体的请求，修改配置文件nginx.conf后，重新生成新的worker进程，当然会以新的配置进行处理请求，而且新的请求必须都交给新的worker进程，至于老的worker进程，等把那些以前的请求处理完毕后，kill掉即可。

## 高并发高效处理

Nginx采用了Linux的epoll模型，epoll模型基于事件驱动机制，它可以监控多个事件是否准备完毕，如果OK，那么放入epoll队列中，这个过程是异步的。worker只需要从epoll队列循环处理即可。

## 负载均衡原理

其实这是把Nginx作为web server来处理静态资源。

第一：location可以进行正则匹配，应该注意正则的几种形式以及优先级。（这里不展开）

第二：Nginx能够提高速度的其中一个特性就是：动静分离，就是把静态资源放到Nginx上，由Nginx管理，动态请求转发给后端。

第三：我们可以在Nginx下把静态资源、日志文件归属到不同域名下（也即是目录），这样方便管理维护。

第四：Nginx可以进行IP访问控制，有些电商平台，就可以在Nginx这一层，做一下处理，内置一个黑名单模块，那么就不必等请求通过Nginx达到后端在进行拦截，而是直接在Nginx这一层就处理掉。

### 反向代理

所谓反向代理（proxy\_pass），很简单，其实就是在location这一段配置中的root替换成proxy\_pass即可。root说明是静态资源，可以由Nginx进行返回；而proxy\_pass说明是动态请求，需要进行转发，比如代理到Tomcat上。

反向代理，上面已经说了，过程是透明的，比如说request -> Nginx -> Tomcat，那么对于Tomcat而言，请求的IP地址就是Nginx的地址，而非真实的request地址，这一点需要注意。不过好在Nginx不仅仅可以反向代理请求，还可以由用户自定义设置HTTP HEADER。

### 负载均衡

上面的反向代理中，我们通过proxy\_pass来指定Tomcat的地址，很显然我们只能指定一台Tomcat地址，那么我们如果想指定多台来达到负载均衡呢？

第一，通过upstream来定义一组Tomcat，并指定负载策略（IPHASH、加权论调、最少连接），健康检查策略（Nginx可以监控这一组Tomcat的状态）等。

第二，将proxy\_pass替换成upstream指定的值即可。

### 负载均衡问题

负载均衡所带来的明显的问题是，一个请求，可以到A server，也可以到B server，这完全不受我们的控制，当然这也不是什么问题，只是我们得注意的是：用户状态的保存问题，如Session会话信息，不能在保存到服务器上。

### 缓存

缓存，是Nginx提供的，可以加快访问速度的机制，说白了，在配置上就是一个开启，同时指定目录，让缓存可以存储到磁盘上。

# Nginx服务器安装

在安装nginx服务器前需要安装其需要的环境，执行一下语句进行安装：

yum install -y gcc-c++

yum install -y pcre pcre-devel

yum install -y zlib zlib-devel

yum install -y openssl openssl-devel

**扩展**

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。pcre-devel是使用PCRE开发的一个二次开发库。Nginx也需要这个库。

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。

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。

## 安装步骤

1、上传nginx的tar包并解压，解压前注意tar包中是否是完整的目录

2、执行下列语句生成Makefile文件

|  |
| --- |
| [root@xmm nginx-1.8.0]# ./configure \  > --prefix=/usr/local/nginx \  > --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 |

3、执行make命令进行编译操作，编译完毕后执行make install命令进行安装

4、安装完毕后进入安装目录/usr/local/nginx，执行该目录下的sbin目录下的nginx开启nginx服务器，在浏览器中输入Linux的IP，如果显示的是Welcome to nginx!则表示安装成功。

## 配置日志

首先在/usr/local/nginx目录下创建logs目录，并在logs目录下创建nginx.pid文件记录nginx服务器的pid。

|  |
| --- |
| [root@xmm logs]# ps -ef | grep nginx | grep -v grep  root 8527 1 0 13:46 ? 00:00:00 nginx: master process ./sbin/nginx  nobody 8528 8527 0 13:46 ? 00:00:00 nginx: worker process |

如上所示nginx的pid为8527，因此nginx.pid文件中输入的内容为8527

|  |
| --- |
| [root@xmm logs]# cat nginx.pid  8527 |

然后使用vim编辑器打开/usr/local/nginx/conf/nginx.conf文件进行修改配置，配置完成后重启nginx服务器。如果重启后显示找不到pid文件的错误，则可能是因为没有手动创建该文件并设置pid的值。

|  |
| --- |
| error\_log /usr/local/nginx/logs/error.log info;  pid /usr/local/nginx/logs/nginx.pid;  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 /usr/local/nginx/logs/access.log main; |

## 负载均衡配置

使用vim编辑器打开/usr/local/nginx/conf/nginx.conf文件进行修改配置，在http{}中添加如下语句：

|  |
| --- |
| upstream balance{  server 192.168.235.23:8180;  server 192.168.235.23:8280;  server 192.168.235.23:8380;  }  server{  listen 80;  server\_name www.balance.com;  location / {  proxy\_pass http://balance;  index index.jsp;  }  } |

在Windows系统的C:\Windows\System32\drivers\etc目录下的host文件中添加192.168.235.23 www.balance.com完成配置，192.168.235.23表示的是Linux服务器的IP地址，www.balance.com表示的是添加了负载均衡的网址。

配置完成后在浏览器中输入www.balance.com，然后多次刷新可以看出不同tomcat服务器中的网址，即nginx的负载均衡已经配置完成。

# Tomcat搭建集群

当用户访问量非常大时，我们需要使用多台Tomcat来构成集群，由Nginx负责将负载分摊到每一台Tomcat服务器上。

用户请求

Nginx

tomcat-one

tomcat-two

tomcat-three

搭建Tomcat集群的基本思路是把Tomcat解压目录复制多份，然后让每一份Tomcat分别使用不同端口号即可。启动时分别进入自己的bin目录执行startup.sh即可。需要修改端口号的文件是server.xml。

## 具体步骤

1、在/opt目录下创建cluster-tomcat目录，然后将apache-tomcat-7.0.75.tar.gz压缩包解压到该目录下，解压时注意压缩包中是完整的目录还是许多分散的文件，如果是分散的文件则需要指定解压到目录tomcat-one

2、将解压的目录修改为tomcat-one，然后复制两份分别为tomcat-two和tomcat-three

3、将tomcat-one目录下的conf目录下的server.xml备份一份

4、使用vim编辑器修改server.xml配置文件，将其第22、71、93行的port值分别修改为8105、8180和8109

|  |
| --- |
| <Server port="8105" shutdown="SHUTDOWN">  <Connector port="8180" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8443" />  <Connector port="8109" protocol="AJP/1.3" redirectPort="8443" /> |

5、依次将tomcat-two和tomcat-three中的server.xml配置文件22、71、93行的值分别修改为[8205，8280，8209]和[8305，8380，8309]

6、使用vim编辑器编写可执行脚本batchStart.sh，使其具有批量开启tomcat服务器的功能

|  |
| --- |
| /opt/cluster-tomcat/tomcat-one/bin/startup.sh  /opt/cluster-tomcat/tomcat-two/bin/startup.sh  /opt/cluster-tomcat/tomcat-three/bin/startup.sh |

将batchStart.sh修改为可执行文件

|  |
| --- |
| [root@xmm cluster-tomcat]# chmod 755 batchStart.sh |

7、执行batchStart.sh批量开启tomcat服务器，然后使用netstat -anp | grep java命令查看是否开启成功

|  |
| --- |
| tcp6 0 0 127.0.0.1:8205 :::\* LISTEN 7445/java  tcp6 0 0 :::8109 :::\* LISTEN 7414/java  tcp6 0 0 127.0.0.1:8305 :::\* LISTEN 7476/java  tcp6 0 0 :::8209 :::\* LISTEN 7445/java  tcp6 0 0 :::8180 :::\* LISTEN 7414/java  tcp6 0 0 :::8309 :::\* LISTEN 7476/java  tcp6 0 0 :::8280 :::\* LISTEN 7445/java  tcp6 0 0 :::8380 :::\* LISTEN 7476/java  tcp6 0 0 127.0.0.1:8105 :::\* LISTEN 7414/java |

8、使用vim编辑器编写可执行脚本batchShutdown.sh，使其具有批量关闭tomcat服务器的功能

|  |
| --- |
| /opt/cluster-tomcat/tomcat-one/bin/shutdown.sh  /opt/cluster-tomcat/tomcat-two/bin/shutdown.sh  /opt/cluster-tomcat/tomcat-three/bin/shutdown.sh |

将batchStart.sh修改为可执行文件

|  |
| --- |
| [root@xmm cluster-tomcat]# chmod 755 batchShutdown.sh |