目录

[第一章:nginx的使用场景和安装 2](#_Toc534152816)

[1、nginx使用在那些场景 2](#_Toc534152817)

[2、安装nginx 2](#_Toc534152818)

[2.1、nginx的安装 2](#_Toc534152819)

[2.2、tengine安装 3](#_Toc534152820)

[第二章：nginx.conf配置文件 3](#_Toc534152821)

[2.1、nginx的http配置参考 4](#_Toc534152822)

[2.2、nginx访问禁止允许规则 7](#_Toc534152823)

[2.3、tengine健康检查 8](#_Toc534152824)

[2.4、nginx内置缓存 8](#_Toc534152825)

[第三章：nginx的session一致性问题 10](#_Toc534152826)

[3.1、利用tomcat本身复制session 10](#_Toc534152827)

[3.2、利用memcached共享session 10](#_Toc534152828)

[3.3、利用redis共享session 11](#_Toc534152829)

[3.4、session一致性的其他方案 11](#_Toc534152830)

[第四章：nginx高可用方案 12](#_Toc534152831)

[1、keepalived+nginx 12](#_Toc534152832)

# 第一章:nginx的使用场景和安装

## 1、nginx使用在那些场景

a) 高可用/高并发；b)优化并发，动静态分离；c)负载均衡；

**nginx和apache的优缺点：**

nginx优点：比apache占用更少的内存及资源；抗并发

apache优点：比nginx的rewrite强大，模块多

区别：处理请求nginx是异步非阻塞/apache阻塞型的 apache同步多进程

## 2、安装nginx

### 2.1、nginx的安装

**安装pcre**

获取pcre编译安装包，在<http://www.pcre.org/>上可以获取当前最新的版本

解压缩pcre-xx.tar.gz包。

进入解压缩目录，执行./configure。

make & make install

**安装openssl**

获取openssl编译安装包，在<http://www.openssl.org/source/>上可以获取当前最新的版本。

解压缩openssl-xx.tar.gz包。

进入解压缩目录，执行./config。

make & make install

**安装zlib**

获取zlib编译安装包，在<http://www.zlib.net/>上可以获取当前最新的版本。

解压缩zlib-xx.tar.gz包。

进入解压缩目录，执行./configure。

make & make install

**安装nginx**

为了安全考虑，修改源码隐藏版本号。

a) vim src/http/ngx\_http\_header\_filter\_module.c

将其中的这一行

static char ngx\_http\_server\_string[] = "Server: nginx" CRLF;

改成了

static char ngx\_http\_server\_string[] = "Server: fan" CRLF;

b) vim src/core/nginx.h

修改成

define NGINX\_VERSION "1"

define NGINX\_VER "fan/" NGINX\_VERSION

进入解压缩目录，执行

./configure \

--with-http\_ssl\_module \

--with-http\_flv\_module \

--with-http\_mp4\_module \

--with-http\_realip\_module \

--with-http\_stub\_status\_module \

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

--with-openssl=/opt/tools/openssl-1.0.2d \

--with-pcre=/opt/tools/pcre-8.36 \

--with-zlib=/opt/tools/zlib-1.2.8 \

--with-pcre

make && make install

### 2.2、tengine安装

tengine是nginx的加强版，封装版，淘宝开源：网站：<http://tengine.taobao.org/documentation_cn.html>

yum install openssl-devel pcre-devel zlib-devel -y 或者和上面nginx一样编译安装这3个必要的插件，

tengine隐藏版本号的方式必须在配置文件配置，在nginx.conf配置文件http{ }里添加如下3个参数。

server\_tag off;

server\_info off;

server\_tokens off;

**编译安装：**

./configure \

--with-pcre \

--with-http\_ssl\_module \

--with-http\_flv\_module \

--with-http\_stub\_status\_module \

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

--with-pcre=/opt/pcre-8.42 \

--with-zlib=/opt/zlib-1.2.11 \

--with-openssl=/opt/openssl-1.0.2q \

--add-module=/opt/nginx\_upstream\_hash-0.3.1 \

--add-module=/opt/sxt/ngx\_cache\_purge-2.3 \

--add-module=/opt/nginx-module-vts

make && make install

**将nginx或tengine添加到开机启动，需要写个shell脚本如下：**

****

将脚本放在/etc/init.d目录下

chmod +X nginx #赋权

chkconfig –add nginx #添加到服务,可以用service命令执行了

chkconfig –list nginx #列出nginx开启自启动服务列表

# 第二章：nginx.conf配置文件



参考：https://www.cnblogs.com/liang-wei/p/5849771.html

nginx进程数(worker\_processes)，建议设置为等于CPU总核心数

一个nginx进程打开的最多文件描述符数(worker\_rlimit\_nofile 65535)，理论值应该是最多打开文件数（系统的值ulimit -n）与nginx进程数相除，但是nginx分配请求并不均匀，所以建议与ulimit -n的值保持一致。65535;

单个进程最大连接数(worker\_connections 65535

最大连接数=连接数\*进程数,就是：worker\_processes 和 worker\_connections 的乘积

设置了反向代理的情况下，并发总数：max\_clients = worker\_processes \*worker\_connections / 4

## 2.1、nginx的http配置参考

**设定http服务器的参数介绍**

http

{

include mime.types; #文件扩展名与文件类型映射表

default\_type application/octet-stream; #默认文件类型

#charset utf-8; #默认编码

server\_names\_hash\_bucket\_size 128; #服务器名字的hash表大小

client\_header\_buffer\_size 32k; #上传文件大小限制

large\_client\_header\_buffers 4 64k; #设定请求缓

client\_max\_body\_size 8m; #设定请求缓

sendfile on; #开启高效文件传输模式，sendfile指令指定nginx是否调用sendfile函数来输出文件，对于普通应用设为 on，如果用来进行下载等应用磁盘IO重负载应用，可设置为off，以平衡磁盘与网络I/O处理速度，降低系统的负载。注意：如果图片显示不正常把这个改 成off。

autoindex on; #开启目录列表访问，合适下载服务器，默认关闭。

tcp\_nopush on; #防止网络阻塞

tcp\_nodelay on; #防止网络阻塞

keepalive\_timeout 120; #长连接超时时间，单位是秒

#FastCGI相关参数是为了改善网站的性能：减少资源占用，提高访问速度。下面参数看字面意思都能理解。

fastcgi\_connect\_timeout 300;

fastcgi\_send\_timeout 300;

fastcgi\_read\_timeout 300;

fastcgi\_buffer\_size 64k;

fastcgi\_buffers 4 64k;

fastcgi\_busy\_buffers\_size 128k;

fastcgi\_temp\_file\_write\_size 128k;

#gzip模块设置

gzip on; #开启gzip压缩输出

gzip\_min\_length 1k; #最小压缩文件大小

gzip\_buffers 4 16k; #压缩缓冲区

gzip\_http\_version 1.0; #压缩版本（默认1.1，前端如果是squid2.5请使用1.0）

gzip\_comp\_level 2; #压缩等级

gzip\_types text/plain application/x-javascript text/css application/xml;

#压缩类型，默认就已经包含text/html，所以下面就不用再写了，写上去也不会有问题，但是会有一个warn。

gzip\_vary on;

#limit\_zone crawler $binary\_remote\_addr 10m; #开启限制IP连接数的时候需要使用

upstream blog.ha97.com {

#upstream的负载均衡，weight是权重，可以根据机器配置定义权重。weigth参数表示权值，权值越高被分配到的几率越大。

server 192.168.80.121:80 weight=3;

server 192.168.80.122:80 weight=2;

server 192.168.80.123:80 weight=3;

}

**虚拟主机的配置**

server

{

#监听端口

listen 80;

#域名可以有多个，用空格隔开

server\_name [www.ha97.com](http://www.ha97.com/) ha97.com;

index index.html index.htm index.php;

root /data/www/ha97;

location ~ .\*.(php|php5)?$

{

fastcgi\_pass 127.0.0.1:9000;

fastcgi\_index index.php;

include fastcgi.conf;

}

#图片缓存时间设置

location ~ .\*.(gif|jpg|jpeg|png|bmp|swf)$

{

[exp](http://www.lengmo.net/)ires 10d;

}

#JS和CSS缓存时间设置

location ~ .\*.(js|css)?$

{

expires 1h;

}

**日志格式设定**

log\_format access ‘$remote\_addr – $remote\_user [$time\_local] “$request” ‘

‘$status $body\_bytes\_sent “$http\_referer” ‘

‘”$http\_user\_agent” $http\_x\_forwarded\_for’;

#定义本虚拟主机的访问日志

access\_log /var/log/nginx/ha97access.log access;

#对 “/” 启用反向代理

location / {

proxy\_pass http://127.0.0.1:88;

proxy\_redirect off;

proxy\_set\_header X-Real-IP $remote\_addr;

#后端的Web服务器可以通过X-Forwarded-For获取用户真实IP

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

#以下是一些反向代理的配置，可选。

proxy\_set\_header Host $host;

client\_max\_body\_size 10m; #允许客户端请求的最大单文件字节数

client\_body\_buffer\_size 128k; #缓冲区代理缓冲用户端请求的最大字节数，

proxy\_connect\_timeout 90; #nginx跟后端服务器连接超时时间(代理连接超时)

proxy\_send\_timeout 90; #后端服务器数据回传时间(代理发送超时)

proxy\_read\_timeout 90; #连接成功后，后端服务器响应时间(代理接收超时)

proxy\_buffer\_size 4k; #设置代理服务器（nginx）保存用户头信息的缓冲区大小

proxy\_buffers 4 32k; #proxy\_buffers缓冲区，网页平均在32k以下的设置

proxy\_busy\_buffers\_size 64k; #高负荷下缓冲大小（proxy\_buffers\*2）

proxy\_temp\_file\_write\_size 64k;

#设定缓存文件夹大小，大于这个值，将从upstream服务器传

}

**设定查看Nginx状态的地址**

location /NginxStatus {

stub\_status on;

access\_log on;

auth\_basic “NginxStatus”;

auth\_basic\_user\_file conf/htpasswd;

#htpasswd文件的内容可以用apache提供的htpasswd[工具](http://www.lengmo.net/)来产生。

}

**本地动静分离反向代理配置**

#所有jsp的页面均交由tomcat或resin处理

location ~ .(jsp|jspx|do)?$ {

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_pass http://127.0.0.1:8080;

}

#所有静态文件由nginx直接读取不经过tomcat或resin

location ~ .\*.(htm|html|gif|jpg|jpeg|png|bmp|swf|ioc|rar|zip|txt|flv|mid|doc|ppt|pdf|xls|mp3|wma)$

{ expires 15d; }

location ~ .\*.(js|css)?$

{ expires 1h; }

}

}

**动静分离：**

      location ^~ /(images|scripts|styles|upload)/  {

            root    /www/abc.com/www/htdocs;

            expires 30d;

       }

           location ~\*\.(gif|jpg|jpeg|png|css|ico|html)$ {

            root    /www/abc.com/www/htdocs;

            expires 30d;

       }

**反向代理配置nginx.conf：**

upstream 名字 ｛

server IP:PORT;

server IP:PORT;

｝

server {

location / {

proxy\_pass http://名字;

}

}

普通的location的匹配规则是最大前缀，正则location的匹配规则是顺序匹配，且只要匹配到第一个就停止后面的匹配。

**先匹配普通的location，再考虑匹配正则location**。，有2种情况下不需要匹配正则location。  
1)当普通location前面 指定了 ^~ 意思就是告诉nginx本条普通的location一旦匹配上就不需要继续匹配正则。

2)当普通location恰好严格匹配上，不是最大前缀匹配，则不再需要匹配正则

## 2.2、nginx访问禁止允许规则

location / {

deny 192.168.1.1; #禁止一个IP访问

allow 192.168.1.0/24; #允许一个网段访问

allow 10.1.1.0/16; #允许一个网段的访问

allow 2001:0db8::/32; #允许一个ipv6的IP访问

deny all; #禁止所有

}

禁止使用IP访问网站，return 返回444错误码

server {

    listen       80;

    server\_name  "";

    return       444;

}

**用户认证访问**

模块ngx\_http\_auth\_basic\_module 允许使用“HTTP基本认证”协议验证用户名和密码来限制对资源的访问。

location / {

auth\_basic "closed site";

auth\_basic\_user\_file /var/users;

}

Apache发行包中的htpasswd命令来创建user\_file 文件

htpasswd -c -m /var/users username

注：需要安装httpd才可以使用上面命令

## 2.3、tengine健康检查

tengine比nginx多了橙色字的地方，可以看到后端服务器的检查状态

upstream cluster2{

server 192.168.0.3:80;

server 192.168.0.4:80;

check interval=3000 rise=2 fall=5 timeout=1000 type=http;

check\_http\_send "HEAD / HTTP/1.0\r\n\r\n";

check\_http\_expect\_alive http\_2xx http\_3xx;

}

server {

listen 80;

location /status {

check\_status;

access\_log off;

allow 192.168.10.0/24;

deny all;

}

}

## 2.4、nginx内置缓存

需要添加缓存模块：ngx\_cache\_purge-2.3

注意缓存的基本上都是静态的东西，动态的插了java代码之类的是缓存不了的,在http的范围里面需要配置的就两个东西：proxy\_temp\_path和proxy\_cache\_path以下是配置意义

proxy\_temp\_path代理临时目录：proxy\_temp\_path /var/temp/nginx/proxy;

#proxy\_cache\_path代理缓存目录，和proxy\_temp\_path必须在同一个分区

#/var/temp/nginx/proxy\_cache\_dir缓存目录

#levels指定该缓存空间有两层hash目录，第一层目录名是1个字母或数字长度，第二层目录名为2个字母或数字长度

#keys\_zone=cache\_one:50m缓存区名称为cache\_one，在内存中的空间是50M，inactive=1d表示1天清空一次缓存 ，max\_size指定磁盘空间大小为500M

proxy\_cache\_path /var/temp/nginx/proxy\_cache\_dir levels=1:2 keys\_zone=cache\_one:50m inactive=1d max\_size=500m;

然后是配置缓存的东西和缓存的条件这里最重要的是proxy\_cache cache\_one;（注：这里的proxy\_cache一定是前面的keys\_zone）和proxy\_cache\_key $host$uri$is\_args$args，以及后面缓存没有的时候的proxy\_pass

location ~ /item(/.\*){

#指定缓存区域名称

proxy\_cache cache\_one;

#以域名、URI、参数组合成Web缓存的Key值，Nginx根据Key值哈希

proxy\_cache\_key $host$uri$is\_args$args;

#请求头中添加请求的主机名

proxy\_set\_header Host $host;

#请求头中添加真实的客户端ip，通过X-Forwarded-For可获取

proxy\_set\_header X-Forwarded-For $remote\_addr;

#代理访问后端tomcat

#请求item从缓存中找，如果缓存中没有则向tomcat请求

#$1表示取出正则表达式(/.\*)所匹配的内容，使用$1的效果例如请求http://ccc.test.com/item/100010.html则请求tomcat服务器 http://ip:port/100010.html

#如果不使用$1则会将/item/...加在tomcat服务地址之后访问，即http://ip:port/item/100010.html

proxy\_pass http://tomcat\_server\_pool$1;

}

然后就是在匹配之前配置一个purge，最重要的匹配的上和proxy\_cache\_purge cache\_one $host$1$is\_args$args

location ~ /purge(/.\*)

{

#安全设置，指定请求客户端的IP或IP段才可以清除URL缓存，这里为了方便测试设置为all

#allow 127.0.0.1;

allow all;

#指定清空缓存的区域名称cache\_one(要和上边proxy\_cache\_path缓存配置中指定的缓存区域名称一致)

#指定缓存的key规则$host$1$is\_args$args，要和下边设置缓存的key一致$host$uri$is\_args$args

#注意$host$1$is\_args$args中的$1表示当前请求的uri，$host$1$is\_args$args=$host$uri$is\_args$args

proxy\_cache\_purge cache\_one $host$1$is\_args$args;

}

# 第三章：nginx的session一致性问题

**session一致性解决方案：**

1、session复制：tomcat本身带有复制session的功能

2、共享session：memcached或者redis缓存服务，可以和tomcat整合，帮助tomcat共享管理session

## 3.1、利用tomcat本身复制session

## 3.2、利用memcached共享session

安装memcached步棸：1、安装libevent依赖库；2、安装memcached；3、启动memcached

libevent下载网站：<http://libevent.org/>或者<https://github.com/libevent/libevent/releases/download/release-2.1.8-stable/libevent-2.1.8-stable.tar.gz>

memcached下载网站：<http://memcached.org/downloads>

**安装libevent**

tar -zxvf libevent-2.1.8-stable.tar.gz

./configure --prefix=/usr/local/libevent

make && make install

**安装memcached**

tar -zxvf memcached-1.5.12.tar.gz

./configure --prefix=/usr/local/memcached --with-libevent=/usr/local/libevent

make && make install

**启动memcached**

cd /usr/local/memcached/bin

./memcached -d -m 128m -p 11211 -u root

**参数说明：**

-d:后台启动服务

-m:缓存大小

-p：端口

-l:IP

-P:服务器启动后的系统进程ID，存储的文件

-u:服务器启动是以哪个用户名作为管理用户

**tomcat配置**

1、将jar包拷贝到tomcat的lib下



2、在tomcat的server.xml中配置jvmRoute

<Engine name="Catalina" defaultHost="localhost" jvmRoute="tomcat2">

3、在context.xml的Context配置Manager添加如下这些：

<Manager className="de.javakaffee.web.msm.MemcachedBackupSessionManager"

memcachedNodes="n1:192.168.17.4:11211"

sticky="false"

lockingMode="auto"

sessionBackupAsync="false"

requestUriIgnorePattern=".\*\.(ico|png|gif|jpg|css|js)$"

sessionBackupTimeout="1000" transcoderFactoryClass="de.javakaffee.web.msm.serializer.kryo.KryoTranscoderFactory" />

4、验证session

在index.jsp文件里添加如下，获取session：

<%@ page language="java" contentType="text/html; charset=UTF-8"  pageEncoding="UTF-8"%>

SessionID:<%=session.getId()%>

<br/>

SessionIP:<%=request.getServerName()%>

<br/>

<h1>tomcat2 page</h1>

## 3.3、利用redis共享session

将redis的jar包放进tomcat的lib目录中



与memcached类型需要在server.xml中配置JVMRoute

在context.xml中配置

<Valve className="com.orangefunction.tomcat.redissessions.RedisSessionHandlerValve" />

<Manager className="com.orangefunction.tomcat.redissessions.RedisSessionManager"

host="192.168.17.4"

port="6379"

database="0"

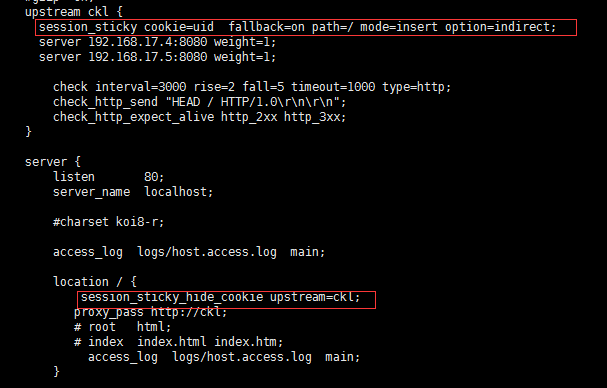
maxInactiveInterval="60" />

以上实验版本是tomcat7.0.61，lib包可以直接支持，最新版本的tomcat已经不能支持了。

## 3.4、session一致性的其他方案

session的粘滞模式：如图在upstream下添加：session\_sticky cookie=uid fallback=on mode=insert option=indirect;

在location下添加：session\_sticky\_hide\_cookie upstream=fan;



# 第四章：nginx高可用方案

## 1、keepalived+nginx