# Nginx安装配置

## 1.源码安装

上传openssl-1.0.1s.tar.gz、pcre-8.38.tar.bz2、ngx\_cache\_purge-2.3.tar.gz、nginx-1.12.1.tar.gz，解压，切换到nginx-1.12.1所在目录：

./configure --prefix=/usr/local/nginx --with-http\_ssl\_module --with-openssl=/usr/local/src/openssl-1.0.1s --with-pcre=/usr/local/src/pcre-8.38 --with-http\_stub\_status\_module --with-http\_gzip\_static\_module --add-module=/usr/local/src/ngx\_cache\_purge-2.3

make && make install

–with-http\_ssl\_module：支持https

with-pcre：为了支持rewrite重写功能，必须制定pcre

–with-http\_stub\_status\_module：支持nginx状态查询

--with-http\_gzip\_static\_module：支持静态压缩

ngx\_cache\_purge：缓存清除

## 2.配置HTTP

user root;

worker\_processes 4;

error\_log logs/error.log notice;

pid logs/nginx.pid;

worker\_rlimit\_nofile 51200; #worker打开的最大文件数

events {

use epoll;

worker\_connections 51200;#worker的最大连接数

}

http {

include mime.types;

default\_type application/octet-stream;

server\_tokens off; #隐藏版本号

server\_names\_hash\_bucket\_size 128;

client\_header\_buffer\_size 32k;

large\_client\_header\_buffers 4 32k;

client\_max\_body\_size 10m; #上传文件最大

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;

tcp\_nodelay on;

keepalive\_timeout 60;

proxy\_set\_header Host $http\_host;

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

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

proxy\_connect\_timeout 5; #连接proxy超时

proxy\_send\_timeout 5; # proxy连接nginx超时

proxy\_read\_timeout 60;# proxy响应超时

proxy\_buffer\_size 16k;

proxy\_buffers 4 64k;

proxy\_busy\_buffers\_size 128k;

# 开启缓存

proxy\_cache\_path /usr/local/nginx/proxy\_cache levels=1:2 keys\_zone=cache\_one:200m inactive=1d max\_size=20g;

proxy\_ignore\_headers X-Accel-Expires Expires Cache-Control; #这个很关键

proxy\_hide\_header Cache-Control;

proxy\_hide\_header Pragma;

# 开启压缩

gzip on;

gzip\_min\_length 1k;

gzip\_buffers 4 16k;

gzip\_http\_version 1.1;

gzip\_comp\_level 2;

gzip\_vary off;

gzip\_disable "MSIE [1-6]\.";

gzip\_types text/plain application/x-javascript text/css application/xml text/javascript application/x-httpd-php image/jpeg image/gif image/png;

upstream server\_pool{

server localhost:8080 weight=1 max\_fails=2 fail\_timeout=30s;

}

server {

listen 80;

server\_name localhost;

location /{

proxy\_pass http://server\_pool;

}

# 状态监控

location /nginx\_status {

# Turn on nginx stats

stub\_status on;

access\_log off;

allow 127.0.0.1;

deny all;

}

#用于清除缓存

location ~ /purge(/.\*)

{

allow 127.0.0.1;

deny all;

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

}

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

{

expires 30d;

proxy\_cache cache\_one;

proxy\_cache\_valid 200 304 12h;

proxy\_cache\_valid any 1m;

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

proxy\_pass http://server\_pool;

}

#error\_page 404 /404.html;

}

}

## 3.生成自签名证书

（1）生成服务器用的私钥文件server.key

openssl genrsa -out server.key 1024

（2）生成未签署的server.csr

openssl req -new -key server.key -out server.csr

程序提示输入一系列参数，包括国别、省名、市名。。。，按照提示输入即可

（3）签署服务器证书文件server.crt

openssl req -x509 -days 365 -key server.key -in server.csr -out server.crt

（4）然后将生成的server.key、server.csr、server.crt拷贝到/usr/local/nginx/conf/目录下

## 4.配置HTTPS

server {

listen 443 ssl;

server\_name localhost;

# 启用所有协议，禁用已废弃的不安全的SSL 2和SSL 3

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2;

# 让服务器选择要使用的算法套件

ssl\_prefer\_server\_ciphers on;

# 这个配置只启用了支持前向保密的算法，按照性能优先的顺序排列。

ssl\_ciphers "ECDHE-ECDSA-AES128-GCM-SHA256 ECDHE-ECDSA-AES256-GCM-SHA384 ECDHE-ECDSA-AES128-SHA ECDHE-ECDSA-AES256-SHA ECDHE-ECDSA-AES128-SHA256 ECDHE-ECDSA-AES256-SHA384 ECDHE-RSA-AES128-GCM-SHA256 ECDHE-RSA-AES256-GCM-SHA384 ECDHE-RSA-AES128-SHA ECDHE-RSA-AES256-SHA ECDHE-RSA-AES128-SHA256 ECDHE-RSA-AES256-SHA384 DHE-RSA-AES128-GCM-SHA256 DHE-RSA-AES256-GCM-SHA384 DHE-RSA-AES128-SHA DHE-RSA-AES256-SHA DHE-RSA-AES128-SHA256 DHE-RSA-AES256-SHA256 EDH-RSA-DES-CBC3-SHA";

# 私钥

ssl\_certificate\_key server.key; #跟nginx.conf同一级目录

# 证书：服务器证书在最前面，后面是所有必要的中间证书，不需要根证书

ssl\_certificate server.crt; #跟nginx.conf同一级目录

# 分配1MB的共享内存缓存

ssl\_session\_cache shared:ssl\_session\_cache:1M;

# 设置会话缓存过期时间为24小时

ssl\_session\_timeout 1440m;

# 禁用会话票证

ssl\_session\_tickets off;

location /{

proxy\_pass http://server\_pool;

}

}

# Nginx调优

http://nginx.org/en/docs/

## 1.状态监控

location = /nginx\_status {

stub\_status on;

access\_log off;

allow <YOURIPADDRESS>;

deny all;

}

输出结果：

Active connections: 1

server accepts handled requests

17122 17122 34873

Reading: 0 Writing: 1 Waiting: 0

Active connections：当前实时的并发连接数

accepts：收到的总连接数，

handled：处理的总连接数

requests：处理的总请求数

Reading：当前有都少个读，读取客户端的请求

Writing：当前有多少个写，向客户端输出

Waiting：当前有多少个长连接（reading + writing）

reading – nginx reads request header

writing – nginx reads request body, processes request, or writes response to a client

waiting – keep-alive connections, actually it is active - (reading + writing)

## 2.实时请求信息统计ngxtop

https://github.com/lebinh/ngxtop

(1)安装python-pip

yum install epel-release

yum install python-pip

(2)安装ngxtop

pip install ngxtop

(3)使用

指定配置文件： ngxtop -c ./conf/nginx.conf

查询状态是200： ngxtop -c ./conf/nginx.conf --filter 'status == 200'

查询那个ip访问最多？ ngxtop -c ./conf/nginx.conf --group-by remote\_addr

## 3.启用压缩

gzip on;

gzip\_disable "MSIE [1-6]\.(?!.\*SV1)";

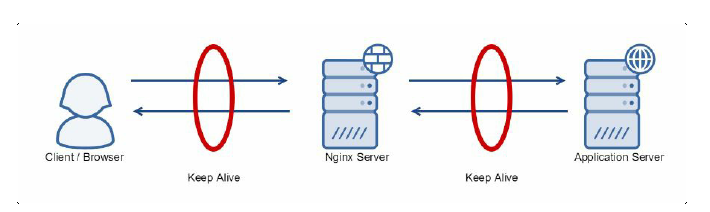
gzip\_proxied any;

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

gzip\_vary on; #Vary: Accept-Encoding

gzip\_static on; #如果有压缩好的 直接使用

## 4.Keepalive长连接



Nginx与upstream server：

upstream server\_pool{

server localhost:8080 weight=1 max\_fails=2 fail\_timeout=30s;

keepalive 300; #300个长连接

}

同时要在location中设置：

location / {

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "upgrade";

}

客户端与nginx（默认是打开的）：

keepalive\_timeout 60s; #长连接的超时时间

keepalive\_requests 100; #100个请求之后就关闭连接，可以调大

keepalive\_disable msie6; #ie6禁用

## 5.工作线程数和并发连接数

worker\_processes 4; #cpu，如果nginx单独在一台机器上

worker\_processes auto;

events {

worker\_connections 4096;#每一个进程打开的最大连接数，超出了log中会有记录

multi\_accept on; #可以一次建立多个连接

use epoll;

}

worker\_rlimit\_nofile 10240;每个进程打开的最大的文件数，受限于操作系统：

vi /etc/security/limits.conf

\* hard nofile 102400

\* soft nofile 102400

\* soft core unlimited

\* soft stack 10240

## 6.操作系统优化

配置文件/etc/sysctl.conf

sysctl -w net.ipv4.tcp\_syncookies=1#防止一个套接字在有过多试图连接到达时引起过载

**sysctl-w net.core.somaxconn=1024#默认128，连接队列**

**sysctl-w net.ipv4.tcp\_fin\_timeout=10 # timewait的超时时间**

**sysctl -w net.ipv4.tcp\_tw\_reuse=1 #os直接使用timewait的连接**

**net.ipv4.tcp\_tw\_recycle = 0 #回收禁用**

## 7. 样例

user www;

worker\_processes 4;#取决于cpu

error\_log logs/error.log;

pid logs/nginx.pid;

worker\_rlimit\_nofile 10240; #每个进程打开的最大的文件数，受限于操作系统/etc/security/limits.conf

events {

worker\_connections 10240;#每一个进程打开的最大连接数，超出了log中会有记录

multi\_accept on; #可以一次建立多个连接

use epoll;

}

http {

include mime.types;

default\_type application/octet-stream;

server\_tokens off; #隐藏版本号

client\_max\_body\_size 10m; #文件上传需要调大

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;

#默认写日志：打开文件写入关闭，max:缓存的文件描述符数量，inactive缓存时间，valid：检查时间间隔，min\_uses：在inactive时间段内使用了多少次加入缓存

open\_log\_file\_cache max=200 inactive=20s valid=1m min\_uses=2;

sendfile on;

tcp\_nopush on;

#与浏览器的长连接

keepalive\_timeout 65;#长连接超时时间

keepalive\_requests 500;#500个请求以后，关闭长连接

keepalive\_disable msie6;

# 启用压缩

gzip on;

gzip\_disable "MSIE [1-6]\.(?!.\*SV1)";

gzip\_proxied any;

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

gzip\_vary on; #Vary: Accept-Encoding

gzip\_static on; #如果有压缩好的 直接使用

#超时时间

proxy\_connect\_timeout 5; #连接proxy超时

proxy\_send\_timeout 5; # proxy连接nginx超时

proxy\_read\_timeout 60;# proxy响应超时

# 开启缓存,2级目录

proxy\_cache\_path /usr/local/nginx/proxy\_cache levels=1:2 keys\_zone=cache\_one:200m inactive=1d max\_size=20g;

proxy\_ignore\_headers X-Accel-Expires Expires Cache-Control;

proxy\_hide\_header Cache-Control;

proxy\_hide\_header Pragma;

#反向代理服务器集群

upstream server\_pool{

server localhost:8080 weight=1 max\_fails=2 fail\_timeout=30s;

server localhost:8081 weight=1 max\_fails=2 fail\_timeout=30s;

keepalive 200; # 最大的空闲的长连接数

}

server {

listen 80;

server\_name localhost 192.168.220.133;

location / {

#Tomcat获取真实用户ip

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_pass http://server\_pool;

}

# 状态监控

location /nginx\_status {

stub\_status on;

access\_log off;

allow 127.0.0.1;

allow 192.168.220.133;

deny all;

}

#用于清除缓存

location ~ /purge(/.\*)

{

allow 127.0.0.1;

allow 192.168.220.133;

deny all;

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

}

# 静态文件加缓存

location ~ .\*\.(gif|jpg|jpeg|png|bmp|swf|js|css|html|htm)?$

{

expires 1d;

proxy\_cache cache\_one;

proxy\_cache\_valid 200 304 1d;

proxy\_cache\_valid any 1m;

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

proxy\_pass http://server\_pool;

}

}

}