nginx配置文件nginx.conf超详细讲解

#nginx进程,一般设置为和cpu核数一样

worker\_processes 4;

#错误日志存放目录

error\_log /data1/logs/error.log crit;

#运行用户，默认即是nginx，可不设置

user nginx

#进程pid存放位置

pid /application/nginx/nginx.pid;

#Specifies the value for maximum file descriptors that can be opened by this process.

#最大文件打开数（连接），可设置为系统优化后的ulimit -HSn的结果

worker\_rlimit\_nofile 51200;

cpu亲和力配置，让不同的进程使用不同的cpu

worker\_cpu\_affinity 0001 0010 0100 1000 0001 00100100 1000;

#工作模式及连接数上限

events

{

use epoll; #epoll是多路复用IO(I/O Multiplexing)中的一种方式,但是仅用于linux2.6以上内核,可以大大提高nginx的性能

worker\_connections 1024; #;单个后台worker process进程的最大并发链接数

}

###################################################

http

{

include mime.types; #文件扩展名与类型映射表

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

#limit模块，可防范一定量的DDOS攻击

#用来存储session会话的状态，如下是为session分配一个名为one的10M的内存存储区，限制了每秒只接受一个ip的一次请求 1r/s

limit\_req\_zone $binary\_remote\_addr zone=one:10m rate=1r/s;

limit\_conn\_zone $binary\_remote\_addr zone=addr:10m;

include mime.types;

default\_type application/octet-stream;

#第三方模块lua防火墙

lua\_need\_request\_body on;

#lua\_shared\_dict limit 50m;

lua\_package\_path "/application/nginx/conf/waf/?.lua";

init\_by\_lua\_file "/application/nginx/conf/waf/init.lua";

access\_by\_lua\_file "/application/nginx/conf/waf/access.lua";

#设定请求缓存

server\_names\_hash\_bucket\_size 128;

client\_header\_buffer\_size 512k;

large\_client\_header\_buffers 4 512k;

client\_max\_body\_size 100m;

#隐藏响应header和错误通知中的版本号

server\_tokens off;

#开启高效传输模式

sendfile on;

-------------------------------------------------------------------------------------------------

#激活tcp\_nopush参数可以允许把httpresponse header和文件的开始放在一个文件里发布，

积极的作用是减少网络报文段的数量

tcp\_nopush on;

#激活tcp\_nodelay，内核会等待将更多的字节组成一个数据包，从而提高I/O性能

tcp\_nodelay on;

tcp\_nopush

View Code

tcp\_nodelay

View Code

-------------------------------------------------------------------------------------

#FastCGI相关参数：为了改善网站性能：减少资源占用，提高访问速度

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;

----------------------------------------------

#连接超时时间，单位是秒

keepalive\_timeout 60;

#开启gzip压缩功能

gzip on；

#设置允许压缩的页面最小字节数，页面字节数从header头的Content-Length中获取。默认值是0，表示不管页面多大都进行压缩。建议设置成大于1K。如果小于1K可能会越压越大。

gzip\_min\_length 1k;

#压缩缓冲区大小。表示申请4个单位为16K的内存作为压缩结果流缓存，默认值是申请与原始数据大小相同的内存空间来存储gzip压缩结果。

gzip\_buffers 4 16k;

#压缩版本（默认1.1，前端为squid2.5时使用1.0）用于设置识别HTTP协议版本，默认是1.1，目前大部分浏览器已经支持GZIP解压，使用默认即可。

gzip\_http\_version 1.0;

#压缩比率。用来指定GZIP压缩比，1压缩比最小，处理速度最快；9压缩比最大，传输速度快，但处理最慢，也比较消耗cpu资源。

gzip\_comp\_level 9;

#用来指定压缩的类型，“text/html”类型总是会被压缩

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

#vary header支持。该选项可以让前端的缓存服务器缓存经过GZIP压缩的页面，例如用

Squid缓存经过Nginx压缩的数据。

gzip\_vary off;

#开启ssi支持，默认是off

ssi on;

ssi\_silent\_errors on;

#设置日志模式

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

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

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

#反向代理负载均衡设定部分

#upstream表示负载服务器池，定义名字为backend\_server的服务器池

upstream backend\_server {

server 10.254.244.20:81 weight=1 max\_fails=2 fail\_timeout=30s;

server 10.254.242.40:81 weight=1 max\_fails=2 fail\_timeout=30s;

server 10.254.245.19:81 weight=1 max\_fails=2 fail\_timeout=30s;

server 10.254.243.39:81 weight=1 max\_fails=2 fail\_timeout=30s;

#设置由 fail\_timeout 定义的时间段内连接该主机的失败次数，以此来断定 fail\_timeout 定义的时间段内该主机是否可用。默认情况下这个数值设置为 1。零值的话禁用这个数量的尝试。

设置在指定时间内连接到主机的失败次数，超过该次数该主机被认为不可用。

#这里是在30s内尝试2次失败即认为主机不可用！

}

###################

#基于域名的虚拟主机

server

{

#监听端口

listen 80;

server\_name www.abc.com abc.com;

index index.html index.htm index.php; #首页排序

root /data0/abc; #站点根目录，即网站程序存放目录

error\_page 500 502 404 /templates/kumi/phpcms/404.html; #错误页面

#伪静态 将www.abc.com/list....html的文件转发到index.php。。。

#rewrite ^/list-([0-9]+)-([0-9]+)-([0-9]+)-([0-9]+)-([0-9]+)-([0-9]+)-([0-9]+)-([0-9]+)-([0-9]+)\.html$ /index.php?m=content&c=index&a=lists&catid=$1&types=$2&country=$3&language=$4&age=$5&startDate=$6&typeLetter=$7&type=$8&page=$9 last;

#location 标签，根目录下的.svn目录禁止访问

location ~ /.svn/ {

deny all;

}

location ~ \.php$

{ #符合php扩展名的请求调度到fcgi server

fastcgi\_pass 127.0.0.1:9000; #抛给本机的9000端口

fastcgi\_index index.php; #设定动态首页

include fcgi.conf;

}

allow 219.237.222.30 ; #允许访问的ip

allow 219.237.222.31 ;

allow 219.237.222.32 ;

allow 219.237.222.33 ;

allow 219.237.222.34 ;

allow 219.237.222.35 ;

allow 219.237.222.61 ;

allow 219.237.222.28 ;

deny all; #禁止其他ip访问

}

location ~ ^/admin.php

{

location ~ \.php$

{

fastcgi\_pass 127.0.0.1:9000;

fastcgi\_index index.php;

include fcgi.conf;

}

allow 219.237.222.30 ;

allow 219.237.222.31 ;

allow 219.237.222.32 ;

allow 219.237.222.33 ;

allow 219.237.222.34 ;

allow 219.237.222.35 ;

allow 219.237.222.61;

allow 219.237.222.28;

deny all;

}

#将符合js,css文件的等设定expries缓存参数，要求浏览器缓存。

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

expires 30d; #客户端缓存上述js,css数据30天

}

##add by 20140321#######nginx防sql注入##########

###start####

if ( $query\_string ~\* ".\*[\;'\<\>].\*" ){

return 444;

}

if ($query\_string ~\* ".\*(insert|select|delete|update|count|\\*|%|master|truncate|declare|\'|\;|and|or|\(|\)|exec).\* ")

{

return 444;

}

if ($request\_uri ~\* "(cost\()|(concat\()") {

return 444;

}

if ($request\_uri ~\* "[+|(%20)]union[+|(%20)]") {

return 444;

}

if ($request\_uri ~\* "[+|(%20)]and[+|(%20)]") {

return 444;

}

if ($request\_uri ~\* "[+|(%20)]select[+|(%20)]") {

return 444;

}

set $block\_file\_injections 0;

if ($query\_string ~ "[a-zA-Z0-9\_]=(\.\.//?)+") {

set $block\_file\_injections 1;

}

if ($query\_string ~ "[a-zA-Z0-9\_]=/([a-z0-9\_.]//?)+") {

set $block\_file\_injections 1;

}

if ($block\_file\_injections = 1) {

return 448;

}

set $block\_common\_exploits 0;

if ($query\_string ~ "(<|%3C).\*script.\*(>|%3E)") {

set $block\_common\_exploits 1;

}

if ($query\_string ~ "GLOBALS(=|\[|\%[0-9A-Z]{0,2})") {

set $block\_common\_exploits 1;

}

if ($query\_string ~ "\_REQUEST(=|\[|\%[0-9A-Z]{0,2})") {

set $block\_common\_exploits 1;

}

if ($query\_string ~ "proc/self/environ") {

set $block\_common\_exploits 1;

}

if ($query\_string ~ "mosConfig\_[a-zA-Z\_]{1,21}(=|\%3D)") {

set $block\_common\_exploits 1;

}

if ($query\_string ~ "base64\_(en|de)code\(.\*\)") {

set $block\_common\_exploits 1;

}

if ($block\_common\_exploits = 1) {

return 444;

}

set $block\_spam 0;

if ($query\_string ~ "\b(ultram|unicauca|valium|viagra|vicodin|xanax|ypxaieo)\b") {

set $block\_spam 1;

}

if ($query\_string ~ "\b(erections|hoodia|huronriveracres|impotence|levitra|libido)\b") {

set $block\_spam 1;

}

if ($query\_string ~ "\b(ambien|blue\spill|cialis|cocaine|ejaculation|erectile)\b") {

set $block\_spam 1;

}

if ($query\_string ~ "\b(lipitor|phentermin|pro[sz]ac|sandyauer|tramadol|troyhamby)\b") {

set $block\_spam 1;

}

if ($block\_spam = 1) {

return 444;

}

set $block\_user\_agents 0;

if ($http\_user\_agent ~ "Wget") {

set $block\_user\_agents 1;

}

# Disable Akeeba Remote Control 2.5 and earlier

if ($http\_user\_agent ~ "Indy Library") {

set $block\_user\_agents 1;

}

# Common bandwidth hoggers and hacking tools.

if ($http\_user\_agent ~ "libwww-perl") {

set $block\_user\_agents 1;

}

if ($http\_user\_agent ~ "GetRight") {

set $block\_user\_agents 1;

}

if ($http\_user\_agent ~ "GetWeb!") {

set $block\_user\_agents 1;

}

if ($http\_user\_agent ~ "Go!Zilla") {

set $block\_user\_agents 1;

}

if ($http\_user\_agent ~ "Download Demon") {

set $block\_user\_agents 1;

}

if ($http\_user\_agent ~ "Go-Ahead-Got-It") {

set $block\_user\_agents 1;

}

if ($http\_user\_agent ~ "TurnitinBot") {

set $block\_user\_agents 1;

}

if ($http\_user\_agent ~ "GrabNet") {

set $block\_user\_agents 1;

}

if ($block\_user\_agents = 1) {

return 444;

}

###end####

location ~ ^/list {

#如果后端的服务器返回502、504、执行超时等错误，自动将请求转发到upstream负载均衡池中的另一台服务器，实现故障转移。

proxy\_next\_upstream http\_502 http\_504 error timeout invalid\_header;

proxy\_cache cache\_one;

#对不同的HTTP状态码设置不同的缓存时间

proxy\_cache\_valid 200 301 302 304 1d;

#proxy\_cache\_valid any 1d;

#以域名、URI、参数组合成Web缓存的Key值，Nginx根据Key值哈希，存储缓存内容到二级缓存目录内

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

proxy\_set\_header Host $host;

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

proxy\_ignore\_headers "Cache-Control" "Expires" "Set-Cookie";

#proxy\_ignore\_headers Set-Cookie;

#proxy\_hide\_header Set-Cookie;

proxy\_pass http://backend\_server;

add\_header Nginx-Cache "$upstream\_cache\_status from km";

expires 1d;

}

access\_log /data1/logs/abc.com.log access; #nginx访问日志

}

-----------------------ssl（https）相关------------------------------------

server {

　　listen 13820; #监听端口

　　server\_name localhost;

　　charset utf-8; #gbk,utf-8,gb2312,gb18030 可以实现多种编码识别

　　ssl on; #开启ssl

　　ssl\_certificate /ls/app/nginx/conf/mgmtxiangqiankeys/server.crt; #服务的证书

　　ssl\_certificate\_key /ls/app/nginx/conf/mgmtxiangqiankeys/server.key; #服务端key

　　ssl\_client\_certificate /ls/app/nginx/conf/mgmtxiangqiankeys/ca.crt; #客户端证书

　　ssl\_session\_timeout 5m; #session超时时间

　　ssl\_verify\_client on; # 开户客户端证书验证

　　ssl\_protocols SSLv2 SSLv3 TLSv1; #允许SSL协议

　　ssl\_ciphers ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP; #加密算法

　　ssl\_prefer\_server\_ciphers on; #启动加密算法

　　access\_log /lw/logs/nginx/dataadmin.test.com.ssl.access.log access ; #日志格式及日志存放路径

　　error\_log /lw/logs/nginx/dataadmin.test.com.ssl.error.log; #错误日志存放路径

}

-------------------------------------------------------------------------

}