nginx优化25条

1.1 隐藏nginx header里版本号信息

1、查看版本号

```
[root@c66-kslx ~]# curl -I 127.0.0.1

HTTP/1.1 200 OK

Server: nginx/1.6.2
```

```
06
07 Date: Sat, 14 Mar 2015 08:15:29 GMT
08
09 Content-Type: text/html
10
11 Content-Length: 25
12
13 Last-Modified: Fri, 13 Mar 2015 10:52:27 GMT
14
15 Connection: keep-alive
16
17 ETag: "5502c16b-19"
18
19 Accept-Ranges: bytes
```

2、隐藏版本号

在nginx配置文件的http标签内加入“server_tokens off;”参数，也可以放大server标签和location标签中，如下：

```
01 http {
02
03 ..........
04
05 server_tokens off;
06
07 ..........
08
09 }
10
11 /application/nginx/sbin/nginx -t
12
13 /application/nginx/sbin/nginx -s reload
```

再此查看如下，浏览器访问错误页面也就没有版本号了

```
1 [root@c66-kslx conf]# curl -I 127.0.0.1
2
3 HTTP/1.1 200 OK
4
5 Server: nginx
```

1.2 更改源码隐藏软件名称及版本号

在nginx编译安装之前，先更改，之后再编译安装

1、更改版本号，修改nginx-1.3.4/src/core/nginx.h

```
1 [root@oldboy /]# cd /home/oldboy/tools/
2
3 [root@oldboy tools]# cd nginx-1.6.2/src/core
4
5 [root@oldboy core]# sed -n '13,17p'nginx.h
6
7 #define NGINX_VERSION  "1.6.2" 修改为想要的版本号如2.4.3
8
9 #define NGINX_VER "nginx/" NGINX_VERSION 将nginx修改为想要修改的软件名称，如Apache。
```

修改后查看header结果：

```
1 [root@S1-SERVER nginx-1.3.4]# curl -I 10.0.0.182
2
3 HTTP/1.1 200 OK
4
5 Server:Apache/2.4.3
```

2、修改nginx-1.6.12/src/http/ngx_http_header_filter_module.c

需要修改的字符串

```
1 [root@M-SERVER http]# grep 'Server:nginx' ngx_http_header_filter_module.c
2
3 static char ngx_http_server_string[] ="Server:nginx" CRLF;
4
5 [root@M-SERVER http]# sed -i 's#Server:nginx#Server:Apache#g' ngx_http_header_filter_module.c
```

修改后的字符串

```
[root@M-SERVER http]# grep 'Server:Apache' ngx_http_header_filter_module.c

static char ngx_http_server_string[] ="Server:Apache" CRLF;
```

3)修改ngx_http_special_response.c

```
#vi nginx-1.3.4/src/http/ngx_http_special_response.c

static u_char ngx_http_error_full_tail[]=

"<hr><center>"NGINX_VER"</center>"CRLF

"</body>"CRLF

"</html>"CRLF

;



static u_char ngx_http_error_tail[]=

"<hr><center>nginx</center>"CRLF

"</body>"CRLF

"</html>"CRLF

;
```

修改为：

```
static u_char ngx_http_error_full_tail[]=

"<hr><center>"NGINX_VER"(http://oldboy.blog.51cto.com (http://oldboy.blog.51cto.com))
</center>"CRLF"

</body>"CRLF

"</html>"CRLF

;



static u_char ngx_http_error_tail[]=

"<hr><center>Apache</center>"CRLF

"</body>"CRLF

"</html>"CRLF

;
```

1.3　更改掉nginx默认用户及用户组（worker进程服务用户优化）

1、默认情况下，nginx服务启动，使用的用户和组默认都是nobody，查看默认配置如下：

```
[root@c66-kslx conf]# grep '#user' nginx.conf.default

#user nobody;
```

将web用户改为特殊的用户名如：nginx或更特殊点的dabaojian，但是这个用户必须是系统存在的。

2、建立nginx用户

```
useradd nginx -s /sbin/nologin -M   #禁止登陆，无家目录

id nginx
```

3、配置文件nginx.conf中修改（也可以编译安装时指定默认）

在配置文件最外层上面

```
worker_processes 1;

user nginx;
```

4、让woker进程使用普通用户运行
为master服务降权：使用非root跑nginx master

```
nginx -c config..
/home/inca/
bin,site,conf,logs
```

注意：不能用80特权端口 ，前端nginx反向代理转端口

```
[inca@nginx ~]$ /application/nginx/sbin/nginx -c /home/inca/conf/nginx.conf
nginx: [emerg] bind() to 0.0.0.0:80 failed (13: Permission denied)
```

1.4 配置nginx worker进程个数

nginx由master和worker进程组成，master进程相当于管理员，worker进程为用户提供服务

一般设置为**cpu核数**或则核数x2，用top按1查看

修改nginx.conf配置文件第一行

```
worker_processes 4;
```

1.5 根据cpu核数进行nginx进程优化

把几个进程分配在一个cup上，cup亲和力

1、不同cpu设置如下

四核cpu配置：

```
worker_processes    4;
worker_cpu_affinity 0001 0010 0100 1000;
```

八核cpu服务器参数配置：

```
worker_processes    8;

worker_cpu_affinity 00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000;
worker_cpu_affinity 0001 0010 0100 1000 0001 0010 0100 1000;
```

2、官方文档说明

```
worker_processes    4;
worker_cpu_affinity 0001 0010 0100 1000;
binds each worker process to a separate CPU, while
worker_processes    2;   4核2进程
worker_cpu_affinity 0101 1010;
```

1.6 nginx事件处理模型优化

nginx的连接处理机制在不同的操作系统上采用不用的IO模型，在linux下，nginx使用epoll的IO多路复用模型，在freebsd使用kqueue的IO多路复用模型，在solaris使用/dev/pool方式的IO多路复用模型，在windows使用的icop等等。

根据系统类型不同选择不同的事务处理模型，选择有“use [kqueue | rtsig |epool |dev/pool |select |pllo];”我们使用的是Centos6.5的linux，因此将nginx的事件处理模型调整为epool模型。

1、具体参数如下在优化4下边挨着：

```
events {

use epoll;

worker_connections 1024;

}
```

1.7 调整nginx worker单个进程允许的客户端最大连接数

这个值根据服务器性能和程序的内存来指定（一个进程启动使用的内存根据程序确定）

```
events {

use epoll;

worker_connections 20480;

}
```

这个参数是单个进程的最大链接数，实际最大链接数是worker技能书乘以这个数。

Max_client=worker_processes*worker_connections

1.8 配置nginx worker进程最大打开文件数

```
worker_rlimit_nofile 65535;
```

相当于系统ulimit -HSn，应该是总的。

理念：配置参数不是越大越好，最好设为服务器承受的极限点。

1.9 开启高效的文件传输模式

在http字段设置

```
http {

include       mime.types; 媒体类型

default_type application/octet-stream; 默认媒体类型

sendfile        on;

tcp_nopush     on; 只有在sendfile开启模式下有效

..........

}
```

tcp_nopush参数可以允许把http response 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。
From: http://nginx.org/en/docs/http/ngx_http_core_module.html

1.10 设置连接超时时间

保护服务器资源，硬件CPU mem，连接数。

建立连接也是要消耗资源的，我们一般断掉那些连上的链接，但是不做事的

php网站建议短连接，PHP程序建立连接消耗的资源和时间要少。

JAVA网站建议长连接，JAVA程序建立连接消耗的资源和时间要多。

在http字段设置

```
http {

..........

keepalive_timeout 60;

###设置客户端连接保持会话的超时时间，超过这个时间，服务器会关闭该连接。

tcp_nodelay on;

####打开tcp_nodelay，在包含了keepalive参数才有效

client_header_timeout 15;

####设置客户端请求头读取超时时间，如果超过这个时间，客户端还没有发送任何数据，Nginx将返回“Request time out（408）”错误

client_body_timeout 15;

```

```
####设置客户端请求主体读取超时时间，如果超过这个时间，客户端还没有发送任何数据，Nginx将返回“Request time out（408）”错误

send_timeout 15;

####指定响应客户端的超时时间。这个超过仅限于两个连接活动之间的时间，如果超过这个时间，客户端没有任何活动，Nginx将会关闭连接。

......

}
```

1.11 上传文件大小限制（动态应用）

nginx.conf中http字段添加如下参数，具体大小根据业务做调整

即http协议原理中请求报文的请求主体，此功能在php参数中还有设置。

```
http {

..........

client_max_body_size 10m;

......

}
```

1.12 fastcgi调优（配合PHP引擎动态服务）

cache 写入缓存区

buffer 读取缓冲区

fastcgi是静态服务和动态服务之间的一个接口

1、参数详解：有的只能放在http标签

```
fastcgi_connect_timeout 300;

###指定链接到后端FastCGI的超时时间。

fastcgi_send_timeout 300;

###向FastCGI传送请求的超时时间，这个值是指已经完成两次握手后向FastCGI传送请求的超时时间。

fastcgi_read_timeout 300;

###指定接收FastCGI应答的超时时间，这个值是指已经完成两次握手后接收FastCGI应答的超时时间。

fastcgi_buffer_size 64k;

###指定读取FastCGI应答第一部分需要用多大的缓冲区，这个值表示将使用1个64KB的缓冲区读取应答的第一部分（应答头），可以设置为gastcgi_buffers选项指定的缓冲区大小。

fastcgi_buffers 4 64k;

###指定本地需要用多少和多大的缓冲区来缓冲FastCGI的应答请求，如果一个php脚本所产生的页面大小为256KB，那么会分配4个64KB的缓冲区来缓存，如果页面大小大于256KB，那么大于256KB的部分会缓存到fastcgi_temp指定的路径中，但是这并不是好方法，因为内存中的数据处理速度要快于磁盘。一般这个值应该为站点中php脚本所产生的页面大小的中间值，如果站点大部分脚本所产生的页面大小为256KB，那么可以把这个值设置为“8 16K”、“4 64k”等。

fastcgi_busy_buffers_size 128k;

###建议设置为fastcgi_buffer的两倍，繁忙时候的buffer

fastcgi_temp_file_write_size 128k;

###在写入fastcgi_temp_path时将用多大的数据库，默认值是fastcgi_buffers的两倍，设置上述数值设置小时若负载上来时可能报502 Bad Gateway

fastcgi_cache oldboy_ngnix;

###表示开启FastCGI缓存并为其指定一个名称。开启缓存非常有用，可以有效降低CPU的负载，并且防止502的错误放生，但是开启缓存也可能会引起其他问题，要很据具体情况选择

```

```
fastcgi_cache_valid 200 302 1h;

###用来指定应答代码的缓存时间，实例中的值表示将2000和302应答缓存一小时，要和fastcgi_cache配合使用

fastcgi_cache_valid 301 1d;

###将301应答缓存一天

fastcgi_cache_valid any 1m;

###将其他应答缓存为1分钟

fastcgi_cache_min_uses 1;

###请求的数量

fastcgi_cache_path

###定义缓存的路径
```

2、参数配置如下：

修改nginx.conf配置文件

在http标签中添加如下：

```
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;

#fastcgi_temp_path /data/ngx_fcgi_tmp;

fastcgi_cache_path /data/ngx_fcgi_cache levels=2:2 keys_zone=ngx_fcgi_cache:512m

inactive=1d max_size=40g;
```

缓存路径，levels目录层次2级，定义了一个存储区域名字，缓存大小，不活动的数据在缓存中多长时间，目录总大小

在server location标签添加如下：

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

{

fastcgi_pass 127.0.0.1:9000;

fastcgi_index index.php;

include fastcgi.conf;

fastcgi_cache ngx_fcgi_cache;

fastcgi_cache_valid 200 302 1h;

fastcgi_cache_valid 301 1d;

fastcgi_cache_valid any 1m;

fastcgi_cache_min_uses 1;

fastcgi_cache_use_stale error timeout invalid_header http_500;

fastcgi_cache_key http:// (http://)$host$request_uri;

}
```

3、fastcgi cache资料：

官方文档：

http://nginx.org/en/docs/http/ngx_http_fastcgi_module.html#fastcgi_cache

相关文档

http://kb.cnblogs.com/page/96115/

http://www.linuxyan.com/web-server/78.html

1.13 配置nginx gzip压缩功能（重要）

1、优点：

a.节约贷款，省钱

b.传输速度快，用户体验好

2、使用模块：

nginx依赖ngx_http_gzip_module模块。

apache使用的是mod_deflate压缩功能

3、需要压缩的内容：纯文本（js，css，html），对于图片，视频，FLASH什么的不压缩，gzip_types参数控制，因为压缩占用cpu啊。

4、对应参数含义如下：

```
01 gzip on;
02
03 ###开启压缩功能
04
05 gzip_min_length  1k;
06
07 ###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，不管页面多大都进行压缩，建议设置成大于1K，如果小与1K可能会越压越大。
08
09 gzip_buffers     4 32k;
10
11 ###压缩缓冲区大小，表示申请4个单位为32K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。
12
13 gzip_http_version 1.1;
14
15 ###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
16
17 gzip_comp_level 9;
18
19 ###压缩比例，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，但是处理慢，也比较消耗CPU资源。
20
21 gzip_types  text/css text/xml application/javascript;
22
23 ###用来指定压缩的类型，‘text/html’类型总是会被压缩。
24
25 gzip_vary on;
26 ###vary header支持，改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。
```

5、具体配置如下

在http标签中配置

```
1 gzip on;
2 gzip_min_length  1k;
3 gzip_buffers     4 32k;
4 gzip_http_version 1.1;
5 gzip_comp_level 9;
6 gzip_types  text/css text/xml application/javascript;
7 gzip_vary on;
```

1.14 配置nginx expires缓存功能（重要）

对于图片，CSS，JS等元素更改的机会较少，特别是图片，这时可以将图片设置在浏览器本地缓存365天或更长，CSS，JS，html等代码缓存10天，这样用户第一次打开页面后，会在本地缓存上述内容，提高了以后打开的页面加载速度，节省服务端大量贷款，此功能同apache的expires。这里通过location，将需要缓存的扩展名列出来，然后指定缓存时间。

1、根据文件扩展名进行判断，添加expires功能

在server字段添加

```
范例1:

location ~ .*\.(gif|jpg|jpeg|png|bmp|swf)$
{
expires      3650d;
}
范例2:
location ~ .*\.(js|css)?$
{
expires      30d;
}

百度的logo就是10年
```

2、根据目录及其他进行判断，添加expires功能范例

```
## Add expires header according to dir.
location ~ ^/(images|javascript|js|css|flash|media|static)/ {
expires 360d;
}

location ~(robots.txt) {
expires 7d;
break;
}
```

3、expire总结

expire功能优点

（1）expires可以降低网站购买的贷款，节约成本

（2）同时提升用户访问体验

（3）减轻服务的压力，节约服务器成本，甚至可以节约人力成本，是web服务非常重要的功能。

expire功能缺点：

被缓存的页面或数据更新了，用户看到的可能还是旧的内容，反而影响用户体验。

解决办法：

第一个 缩短缓存时间，例如：1天，不彻底，除非更新频率大于1天

第二个 对缓存的对象改名

a.图片，附件一般不会被用户修改，如果用户修改了，实际上也是更改文件名重新传了而已

b.网站升级对于js，css元素，一般可以改名，把css，js，推送到CDN。

企业网站缓存日期案例

1、51cto 1周

2、sina 15天

3、京东 25年

4、淘宝 10年

网站不希望被缓存的内容

1）广告图片

2）网站流量统计工具

3）更新频繁的文件（google的logo）

1.15 nginx日志相关优化与安全

1、配置日志切割脚本并写入计划任务

```
[root@web-1 conf]# cd /server/scripts/

[root@web-1 scripts]# cat cut_nginx_log.sh

#!/bin/sh

cd /app/logs

mv www_access.log www_access_$(date +%F -d -1day).log

mv bbs_access.log bbs_access_$(date +%F -d -1day).log

mv blog_access.log blog_access_$(date +%F -d -1day).log

/application/nginx/sbin/nginx -s reload

cat >>/var/spool/cron/root<<eof

#cut nginx log by lxy at 20150327#

00 00 * * * /bin/sh /server/scripts/cut_nginx_log.sh >/dev/null 2>&1

eof
```

2、不记录不需要的访问日志

对于健康检查或某些（图片，js，css）日志，一般不记录日志，因为在统计PV时是按照页面计算，而且日志写入频繁会消耗磁盘IO，降低服务器性能。

```
location ~ .*\.(js|jpg|JPG|jpeg|JPEG|css|bmp|gif|GIF)$ {
access_log off;
}
```

3、访问日志的权限设置

假设日志目录为/app/logs，则授权

```
chown -R root.root /app/logs

chmod -R 700 /app/logs
```

不需要再日志目录给nginx用户读或者写许可。因为nginx的master进程是root，不要担心权限不够写不进去日志

1.16 nginx站点目录及文件URL访问控制（重要:防止恶意解析）

1、根据扩展名限制程序和文件访问

作用：禁止目录下指定文件北风文，或者禁止指定目录下所有内容被访问

最佳应用场景:集群的共享存储，本来就应该只是资源文件，禁止指定扩展名程序被执行，例如：.php，.sh，.pl

nginx下禁止访问资源目录下的php程序文件，配置方法如下：

范例1：nginx配置限制指定目录下的php程序被解析

```
location ~ ^/images/.*\.(php|php5|.sh|.pl|.py)$

{

deny all;

}

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

## Only allow these request methods ##
if ($request_method !~ ^(GET|HEAD|POST)$ ) {
return 444;
}
禁止请求方法

---------------------------------------
```

```
location ~ ^/static/.*\.(php|php5|.sh|.pl|.py)$

{

deny all;

}

location ~* ^/data/(attachment|avatar)/.*\.(php|php5)$

{

deny all;

}

需要放在php解析location的下面
```

范例2：Nginx下配置禁止访问*.txt文件

```
location ~* \.(txt|doc)$ {
if (-f $request_filename) {
root /data/www/www;
#rewrite …..可以重定向到某个URL
break;
}
}
location ~* \.(txt|doc)${
root /data/www/www;
deny all;
}

location ~ ^/(static)/ {
deny all;
}

location ~ ^/static {
deny all;
}
location ~ ^/(static|js) {
deny all;
}
location /admin/ { return 404; }
location /templates/ { return 403; }
```

2、限制来源ip访问

使用ngx_http_access_module限制ip访问

例如：phpmyadmin数据库web客户端，内部开发人员用

范例1：禁止某目录让外界访问，但允许某ip访问改目录，且支持php解析。

```
location ~ ^/oldboy/ {
allow 202.111.12.211;
deny all;
}
```

范例2限制及指定ip或ip段访问

```
location / {
deny 192.168.1.1;
allow 192.168.1.0/24;
allow 10.1.1.0/16;
deny all;
}
```

官网资料

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

其他写法

```
if ( $remote_addr = 10.0.0.7 ) {
return 403;
}
```

企业问题案例：nginx做反向代理的时候可以限制客户端IP吗？
解答：
法1：使用if来控制

```
if ( $remote_addr = 10.0.0.7 ) {
return 403;
}
if ( $remote_addr = 218.247.17.130 ) {
set $allow_access_root 'true';
}
```

3、限制使用网站ip访问网站

防止别的网站指定到你的ip，盗用你的流量，占你贷款，假冒你。

方法1：ngnix第一个虚拟主机设置403，不够友好

```
server {
listen 80 default_server;
server_name _;
return 403;
}
```

方法2：添加301跳转，也是第一个虚拟主机（一般不这么搞，一般用上面的）

```
server {
listen 80 default_server;
server_name _;
rewrite ^(.*) http://blog.etiantian.org/ (http://blog.etiantian.org/)$1 permanent;
}
```

nginx的所有变量 http://nginx.org/en/docs/varindex.html

1.17 配置nginx图片及目录防盗链

1、什么是防盗链

简单的说，就是某些不法的网站，通过在其自身网站程序里未经许可非法调用其他网站的资源，然后在自己的网站上显示这些调用的资源，达到了填充自身网站显示的效果，但是浪费了调用资源网站的网站流量，造成其他网站的带宽及服务压力吃紧，甚至宕机。

2、解决办法

（1）图片，视频上打水印，品牌。

（2）防火墙控制，根据ip控制。

（3）防盗链（根据referer控制）

运维的价值：轻松应得IDC机房流量暴涨问题。

http://oldboy.blog.51cto.com/2561410/909696

如何及时发现问题

第一、对IDC及CDN带宽做监控报警。

第二、作为高级运维或者运维经理，每天上班的一个重要任务，就是经常查看网站流量图，关注流量变化，关注异常流量。

第三、对访问日志做分析，对于异常流量能迅速定位，并且和公司市场推广等有比较好的默契沟通交流，以便调度贷款和服务器资源。确保网站正常的访问体验得到保证。

3、利用http referer设置防盗链

HTTP Referer是header的一部分，当浏览器向web服务器发送请求的时候，一般会带上Referer，告诉服务器我是从哪个页面链接过来的，服务器籍此可以获得一些信息用于处理。

利用referer并且针对扩展名rewrite重定向

```
location ~* \.(jpg|gif|png|swf|flv|wma|wmv|asf|mp3|mmf|zip|rar)$ {
valid_referers none blocked *.etiantian.org etiantian.org;
if ($invalid_referer) {
rewrite ^/ http://www.etiantian.org/img/nolink.jpg; (http://www.etiantian.org/img/nolink.jpg;)
}
}
```

说明：设定一个location，如果访问为指定的扩展名文件，则进行判断，如果不是来自我指定的网站，则转交给我提前设置的其他页面及图片

特别说明：防盗链跳转的地址，不能再是设置防盗链的虚拟主机地址，要用第三个虚拟主机，要不就成死循环了！

```
# Stop deep linking or hot linking
location /images/ {
valid_referers none blocked www.example.com example.com;
if ($invalid_referer) {
return   403;
}
}
```

1.18 nginx错误页面优雅显示

范例：403跳转

```
###www
server {
listen       80;
server_name  www.etiantian.org;
location / {
root   html/www;
index  index.html index.htm;
}
error_page  403  /403.html; 此路径相对于root   html/www;的

#error_page  404  http;//www.baudu.com 写法2

#error_page   500 502 503 504  /50x.html;

}
```

生产场景状态码列表

http://oldboy.blog.51cto.com/2561410/716294

```
error_page   404  http://oldboy.blog.51cto.com; (http://oldboy.blog.51cto.com;)
```

阿里门户天猫网站的nginx优雅显示配置案例：

```
error_page   500 501 502 503 504  http://err.tmall.com/error2.html;
(http://err.tmall.com/error2.html;)
error_page 400 403 404 405 408 410 411 412 413 414 415 http://err.tmall.com/error1.html;
(http://err.tmall.com/error1.html;)
```

1.19 nginx防爬虫优化

Robots协议（也称为爬虫协议、机器人协议等）的全称是“网络爬虫排除标准”（Robots Exclusion Protocol），网站通过Robots协议告诉搜索引擎哪些页面可以抓取，哪些页面不能抓取。

问题：可能会暴漏网站目录结构

nginx防爬虫优化

nginx根据$http_user_agent获取客户端agent，然后判断是否允许或者返回指定页面

网上资料：http://blog.csdn.net/xifeijian/article/details/38615695

##添加如下内容可防止爬虫（防搜索引擎的，但是网站展示宣传就不行了，所以看情况搞）

```
if ($http_user_agent ~* "qihoobot|Baiduspider|Googlebot|Googlebot-Mobile|Googlebot-
Image|Mediapartners-Google|Adsbot-
Google|Yahoo! Slurp China|YoudaoBot|Sosospider|Sogou spider|Sogou web spider|MSNBot")
{
return 403;
}
```

实战演示：禁止不同浏览器软件访问：

```
if ($http_user_agent ~* "Firefox|MSIE")
{
return 403;
rewrite ^(.*) http://blog.etiantian.org/ (http://blog.etiantian.org/)$1 permanent;
}
```

1.20 使用tmpfs文件系统给/tmp

提高效率，部分程序切图片操作临时放到/tmp下，可以把tmp设置成内存文件系统，占用内存空间的，就是从内存里拿出一块来当磁盘用

mount -t tmpfs -o size=16m tmpfs /tmp

1.21 防DOS攻击

1、控制单个ip的并发请求防止DOS攻击

使用limit_conn_zone进行控制，控制单个ip或域名的访问次数，限制连续访问

在http标签添加控制，可添加多个，在server或location中使用，

```
limit_conn_zone $binary_remote_addr zone=perip:10m;
limit_conn_zone $server_name zone=perserver:10m;
server {
...
limit_conn perip 10;
limit_conn perserver 100;
}

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

limit_conn_zone $binary_remote_addr zone=addr:10m;
server {
location /download/ {
limit_conn addr 1;
}
```

2、限制单个ip的请求速率防止DOS攻击

使用limit_req_zone进行控制，控制单个ip的访问速率

1.22 防DDOS策略

http://oldboy.blog.51cto.com/2561410/845349

1.23 限制客户端请求的HTTP方法

```
#Only allow these request methods
if ($request_method !~ ^(GET|HEAD|POST)$ ) {
return 501;
}
#Do not accept DELETE, SEARCH and other methods
```

1.24 为web服务增加用户身份验证（适合内部机密网址）

控制力度：可以是一个虚拟主机，或一个目录

场景：内部使用的网址，例如phpmyadmin客户端

配置方法如下：

nginx配置：

```
location /phpmyadmin/ {
auth_basic      "oldboy training";
auth_basic_user_file /application/nginx/conf/htpasswd;
}
```

创建密码文件：

```
[root@nginx conf]# htpasswd -cb /application/nginx/conf/htpasswd oldboy 123456
Adding password for user oldboy
[root@nginx conf]# cat /application/nginx/conf/htpasswd
oldboy:Pl80EhiC2Z/5I

chmod 400 /application/nginx/conf/htpasswd

chown nginx /application/nginx/conf/htpasswd
```

1.25 让Nginx运行于（A Chroot Jail（Containers））监牢模式

即降权思想，将nginx以普通用户运行，所有相关配置都放到普通用户的家目录中，类似于Docker的思想

下面知识举例：

1、给nginx服务降权用inca用户跑服务，站点inca，给开发设置普通账户和inca同组。

2、开发重启nginx，管理站点程序，看日志项目负责制：出问题你来负责

参考资料：到底要不要给开发root权限：

http://down.51cto.com/data/844517

具体做法如下：

创建普通用户，将站点目录以及nginx.conf放到普通用户家目录，之后用

/application/nginx/sbin/nginx -c 配置文件来启服务

注意相关日志所在路径，指定pid ，端口不能用80（前端用反向代理解决外部80问题）

```
useradd inca

cd /home/inca

mkdir conf www log

echo inca > www/index.html
```

修改配置文件

```
error_log /home/inca/log/error.log

pid /home/inca/log/nginx.pid
```

指定站点目录

日志

端口不能用80

用普通用户启动

解决端口问题

用反向代理解决端口问题haproxy，nginx，f5.

解决方案的优点：

a、服务降权网站更安全了

b、按用户设置站点权限。站点更独立了（无需虚拟化隔离）

c、开发不需要root可以管理服务，不要吵来吵去

d、责任划分：网络问题：运维责任，网站打不开开发责任。（共同承担）

1.26 其他优化

1、安全优化：web磁盘挂载优化

LABEL=/nginx /nginx ext3 defaults,nosuid,noexec,nodev 1 2

2、内核优化

3、移除不想要的nginx模块（最小化原则）

编译时加–without-http-**不安装模块

4、开启iptables防护

正文部分到此结束

返回 目录 (http://www.z-dig.com/team-lixinyu/)