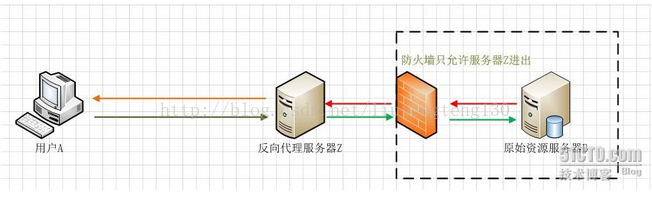
**负载均衡服务器Nginx**

Nginx (engine x) 是一个高性能的HTTP和反向代理服务器，也是一个IMAP/POP3/SMTP服务器。Nginx 可以在大多数 UnixLinux OS 上编译运行，并有 Windows 移植版。

代理服务器:一般是指局域网内部的机器通过代理服务器发送请求到互联网上的服务器,代理服务器一般作用在客户端。

反向代理（Reverse Proxy）方式是指以代理服务器来接受internet上的连接请求，然后将请求转发给内部网络上的服务器，并将从服务器上得到的结果返回给internet上请求连接的客户端，此时代理服务器对外就表现为一个服务器。

这里讲得很直白。反向代理方式实际上就是一台负责转发的代理服务器，貌似充当了真正服务器的功能，但实际上并不是，代理服务器只是充当了转发的作用，并且从真正的服务器那里取得返回的数据。



负载均衡的作用：

例如一台服务器本身的性能能够支持1w个业务并发处理

如果业务并发少于1w个，机器也能负重前行

但是如果有5w个怎么办呢

简单的办法是使用nginx做前面的类似于堡垒机

10w个并发都打到这个nginx

但是nginx本身不处理业务，所以他能接纳10w个并发，但是他本身没有处理能力

而是把这个10w个并发，按照一定的策略分配给后面的其他机器

这样的好处是

1.通过1个nginx+n个后面的机器组成一个小集群，能处理超过单台机器接纳的上线

2.对外提供服务一般会固定一个ip，而往往你的业务一个ip（也就是单台服务器）不能完全处理，那么找一个性能不错的专门负责转发外面的业务请求（一般这种对服务器的压力不会太大），然后发给真正需要处理业务的后面的例如tomcat去，这样能够对外提供一致的提供服务的点

如果并发有50w个怎么办？

这种即使仅仅是转发，单个nginx都没法处理，性能要求太高，而且即使性能能够跟上，操作系统的端口数也是有限的

这个时候就需要高级的转发服务器了

这方面有硬件的例如f5或者软件的ha

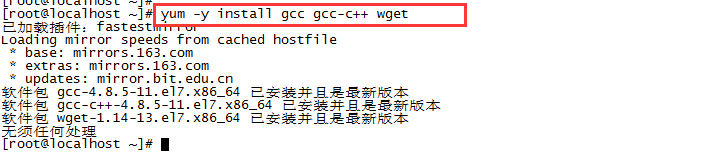
然后将请求分发给nginx，nginx再分发给具体处理业务的例如tomcat

但是功能都是一样的：提供统一的对外访问入口，转发请求给真正的执行者

**安装步骤：**

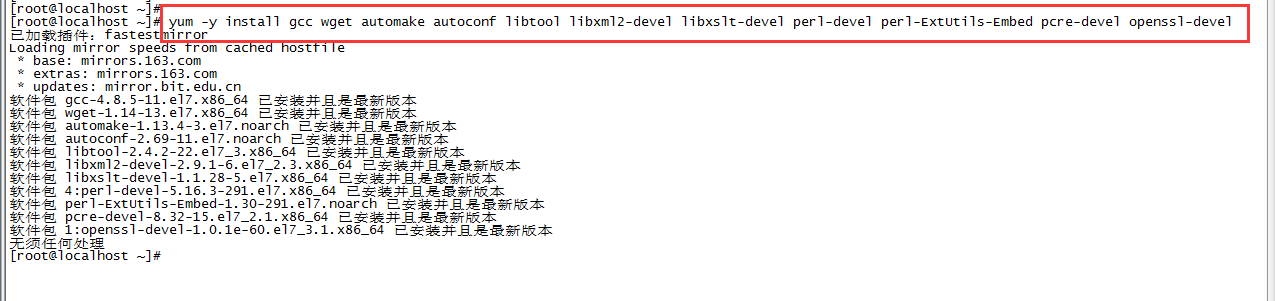
1，先安装gcc 等

yum -y install gcc gcc-c++ wget



2，安装一些库

yum -y install gcc wget automake autoconf libtool libxml2-devel libxslt-devel perl-devel perl-ExtUtils-Embed pcre-devel openssl-devel



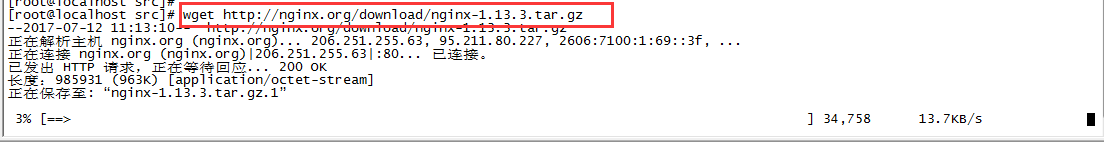
3，进入默认的软件目录

cd /usr/local/src/



4，下载nginx软件（进入下载页面查看最新版本http://nginx.org/download）

wget http://nginx.org/download/nginx-1.13.3.tar.gz



5，解压

tar zxvf nginx-1.13.3.tar.gz



6，进入 nginx1.13.3的源码 如果想改版本号 可以进入源码目录src/core/nginx.h更改

cd nginx-1.13.3/



7，创建一个nginx目录用来存放运行的临时文件夹

mkdir -p /var/cache/nginx



8，开始configure

./configure \

--prefix=/usr/local/nginx \

--sbin-path=/usr/sbin/nginx \

--conf-path=/etc/nginx/nginx.conf \

--error-log-path=/var/log/nginx/error.log \

--http-log-path=/var/log/nginx/access.log \

--pid-path=/var/run/nginx.pid \

--lock-path=/var/run/nginx.lock \

--http-client-body-temp-path=/var/cache/nginx/client\_temp \

--http-proxy-temp-path=/var/cache/nginx/proxy\_temp \

--http-fastcgi-temp-path=/var/cache/nginx/fastcgi\_temp \

--http-uwsgi-temp-path=/var/cache/nginx/uwsgi\_temp \

--http-scgi-temp-path=/var/cache/nginx/scgi\_temp \

--user=nobody \

--group=nobody \

--with-pcre \

--with-http\_v2\_module \

--with-http\_ssl\_module \

--with-http\_realip\_module \

--with-http\_addition\_module \

--with-http\_sub\_module \

--with-http\_dav\_module \

--with-http\_flv\_module \

--with-http\_mp4\_module \

--with-http\_gunzip\_module \

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

--with-http\_random\_index\_module \

--with-http\_secure\_link\_module \

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

--with-http\_auth\_request\_module \

--with-mail \

--with-mail\_ssl\_module \

--with-file-aio \

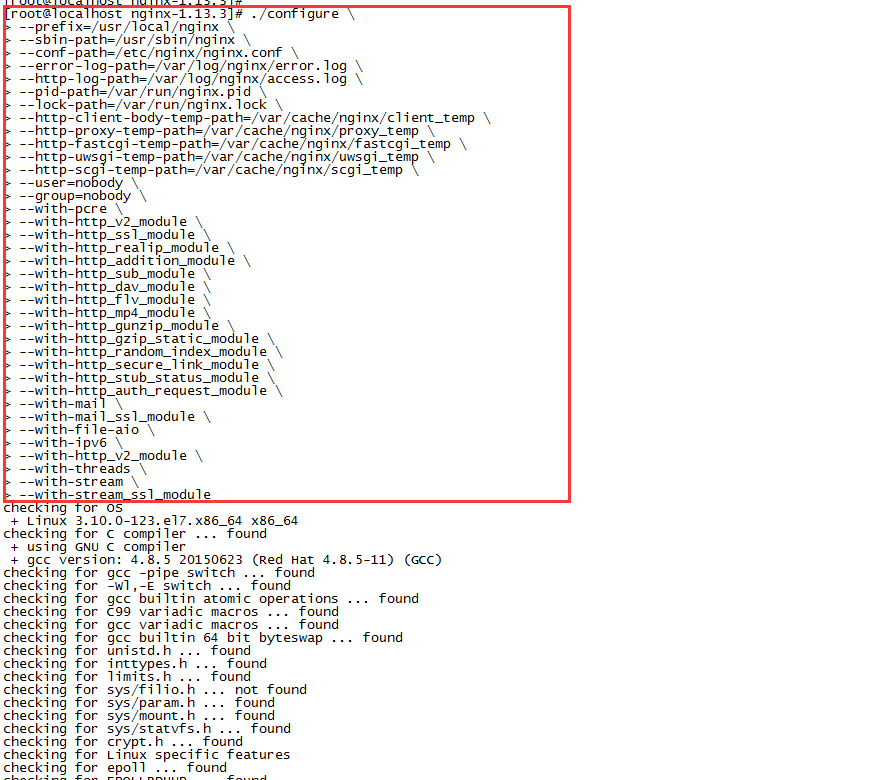
--with-ipv6 \

--with-http\_v2\_module \

--with-threads \

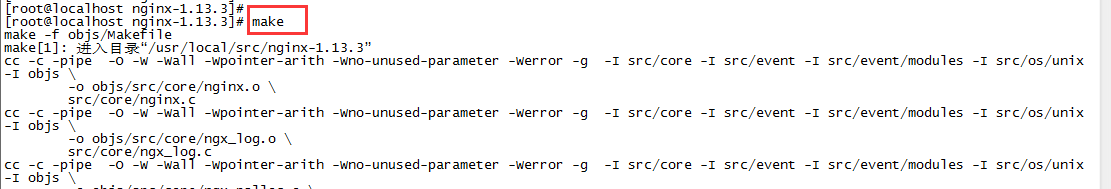
--with-stream \

--with-stream\_ssl\_module



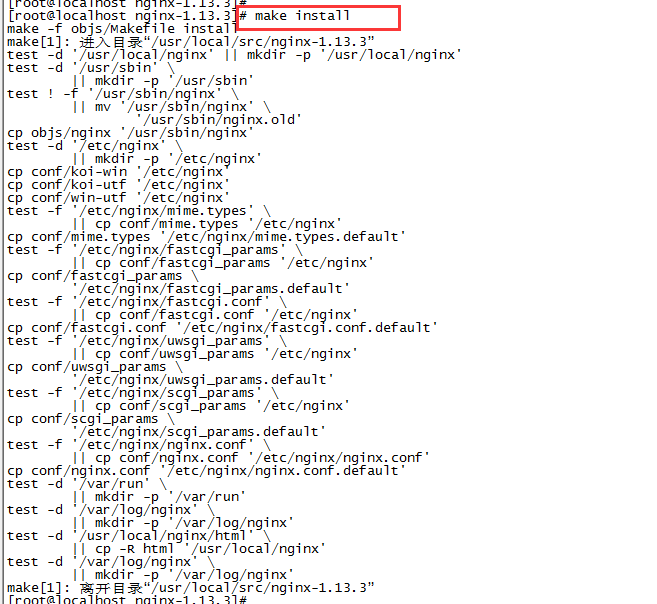
9，编译

make



10，安装

make install



11，启动nginx

/usr/sbin/nginx



12，配置服务

vi /usr/lib/systemd/system/nginx.service

输入以下内容后保存

[Unit]

Description=nginx - high performance web server

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

After=network.target remote-fs.target nss-lookup.target

[Service]

Type=forking

PIDFile=/var/run/nginx.pid

ExecStartPre=/usr/sbin/nginx -t -c /etc/nginx/nginx.conf

ExecStart=/usr/sbin/nginx -c /etc/nginx/nginx.conf

ExecReload=/bin/kill -s HUP $MAINPID

ExecStop=/bin/kill -s QUIT $MAINPID

PrivateTmp=true

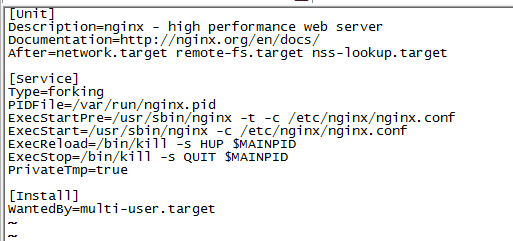
[Install]

WantedBy=multi-user.target

文件保存后，刚刚配置的服务需要让systemctl能识别，就必须刷新配置

systemctl daemon-reload





13，设置开机启动

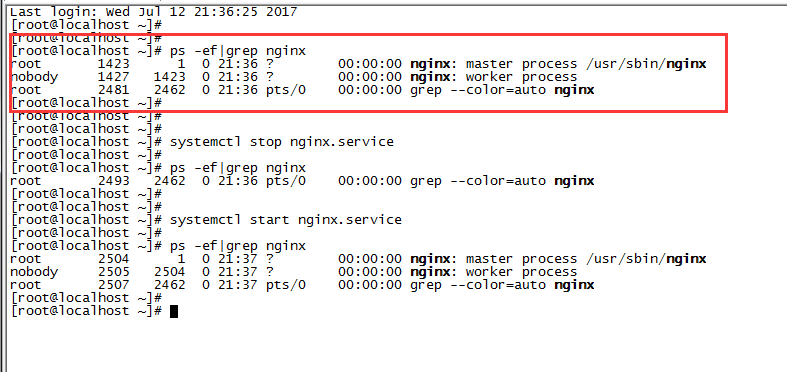
systemctl enable nginx.service



重启电脑，看到nginx已自启。

reboot

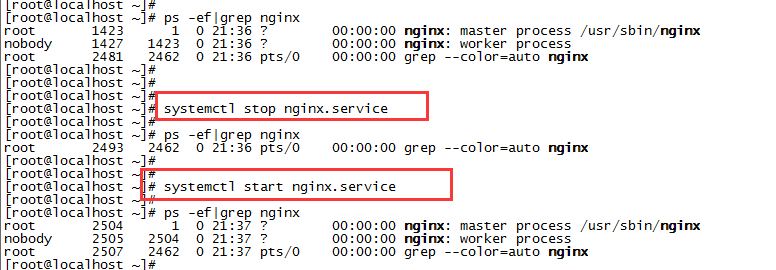
ps -ef|grep nginx



14，后面可以用systemctl来操作nginx.service

systemctl start nginx.service

systemctl stop nginx.service



如果使用命令出现如下报错

Job for nginx.service failed because the control process exited with error code. See "systemctl status nginx.service" and "journalctl -xe" for details.

修改文件/usr/lib/systemd/system/nginx.service测试是否能启用服务。

vi /usr/lib/systemd/system/nginx.service

输入以下内容后保存

[Unit]

Description=nginx - high performance web server

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

After=network.target remote-fs.target nss-lookup.target

[Service]

Type=forking

PIDFile=/var/run/nginx.pid

ExecStartPre=/usr/sbin/nginx

ExecStart=/usr/sbin/nginx

ExecStop=/bin/kill nginx

PrivateTmp=true

[Install]

WantedBy=multi-user.target

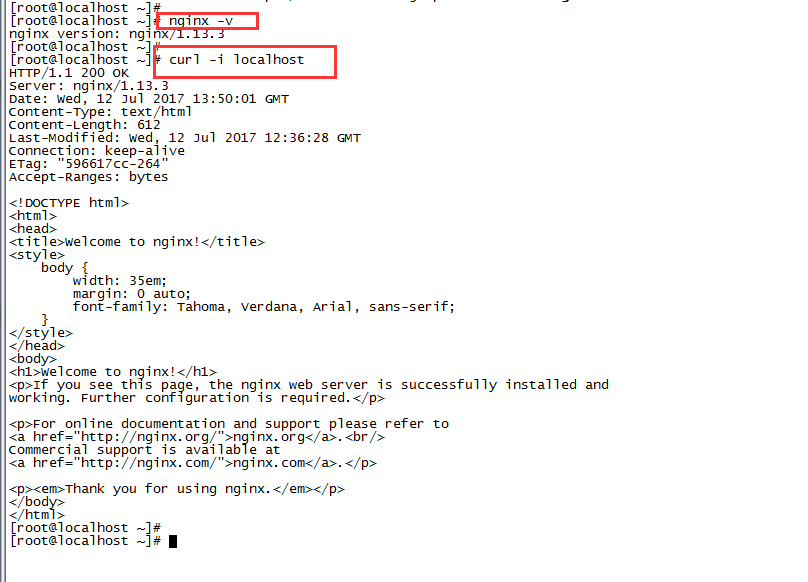
15，

查看nginx版本

nginx -v

访问nginx，现在你可以通过公网ip (本地可以通过 localhost /或 127.0.0.1 ) 查看nginx 服务返回的信息。

curl -i localhost



16，17，18是修改nginx配置文件的例子，配置文件位于/usr/local/src/nginx-1.13.3/conf/

使用命令

vi /usr/local/src/nginx-1.13.3/conf/nginx.conf

16，常见例子->访问静态文件

#定义Nginx运行的用户和用户组

user nginx nginx;

#nginx进程数，建议设置为等于CPU总核心数。用lscpu命令查看cou核数

worker\_processes 1;

#nginx默认是没有开启利用多核cpu的配置的。需要通过增加worker\_cpu\_affinity配置参数来充分利用多核cpu。

#worker\_cpu\_affinity 00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000;

#一个nginx进程打开的最多文件描述符数目，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。

worker\_rlimit\_nofile 65536;

#全局错误日志定义类型，[ debug | info | notice | warn | error | crit ]

#error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.log info;

#进程文件

#pid /var/run/nginx.pid;

events {

#单个进程最大连接数（最大连接数=连接数\*进程数）

worker\_connections 65536;

}

#设定http服务器

http {

#文件扩展名与文件类型映射表

include mime.types;

#默认文件类型

default\_type application/octet-stream;

log\_format main '$remote\_addr [$time\_local] $upstream\_addr $upstream\_status $upstream\_response\_time "$request" $status $body\_bytes\_sent "$http\_referer" "$http\_user\_agent"';

log\_format cachelog '$time\_local - $upstream\_cache\_status - Cache-Control:$upstream\_http\_cache\_control - $request($status) - ';

access\_log logs/access.log main;

sendfile on;

tcp\_nopush on;

proxy\_ignore\_client\_abort on;

#长连接超时时间，单位是秒

keepalive\_timeout 65;

#默认编码

charset utf-8;

#gzip模块设置

gzip on; #开启gzip压缩输出

gzip\_min\_length 10k; #最小压缩文件大小

gzip\_buffers 4 16k; #压缩缓冲区

gzip\_comp\_level 2; #压缩等级

#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\_types text/plain text/javascript application/javascript application/x-javascript text/css application/xml application/octet-stream;

gzip\_vary on;

#虚拟主机的配置

server {

#被监听的端口号和网址

listen 80;

#域名可以有多个，用空格隔开

server\_name www.test.com;

#charset koi8-r;

#定义本虚拟主机的访问日志

access\_log logs/test\_access.log main;

#对 "/" 启用反向代理

location / {

#这个地方指定被访问的文件夹位置

root /data/test;

index index.html index.htm;

}

#error\_page 404 /404.html;

# redirect server error pages to the static page /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

#图片缓存时间设置

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

root /data/test;

expires 30d;

}

}

#location ~ /purge(/.\*){

# allow 192.168.0.0/16;

# deny all;

# proxy\_cache\_purge resource $host$1$is\_args$args;

#}

}

17，常见例子->一个负载均衡例子

user nginx nginx;

worker\_processes 8;

worker\_cpu\_affinity 00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000;

worker\_rlimit\_nofile 65536;

events {

worker\_connections 65536;

}

http {

include mime.types;

default\_type application/octet-stream;

# log\_format 日志格式

log\_format main '$remote\_addr [$time\_local] $upstream\_addr $upstream\_status $upstream\_response\_time "$request" $status $body\_bytes\_sent "$http\_referer" "$http\_user\_agent"';

log\_format cachelog '$time\_local - $upstream\_cache\_status - Cache-Control:$upstream\_http\_cache\_control - $request($status) - ';

access\_log logs/access.log main;

sendfile on;

tcp\_nopush on;

proxy\_ignore\_client\_abort on;

keepalive\_timeout 65;

#keepalive\_timeout 1000;

charset utf-8;

gzip on;

gzip\_min\_length 10k;

gzip\_buffers 4 16k;

gzip\_comp\_level 2;

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

gzip\_vary on;

upstream balance{

server 192.168.21.77:8080;

}

server {

listen 80;

server\_name www.website.com;

#charset koi8-r;

access\_log logs/website.log main;

#request proxy server

location / {

proxy\_set\_header Host $host;

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

proxy\_set\_header REMOTE-HOST $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;

#这里设置代理的位置

proxy\_pass http://balance;

proxy\_redirect default;

}

#error\_page 404 /404.html;

# redirect server error pages to the static page /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

#设定查看Nginx状态的地址

location /nginx\_status{

stub\_status on;

access\_log off;#指定全局的 log 是否打开

allow all;

# deny all;

}

}

}

18，常见例子->多个负载均衡，多个服务，多个端口

user nginx nginx;

worker\_processes 8;

worker\_cpu\_affinity 00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000;

worker\_rlimit\_nofile 65536;

#error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.log info;

#pid logs/nginx.pid;

events {

worker\_connections 65536;

}

http {

include mime.types;

default\_type application/octet-stream;

log\_format main '$remote\_addr [$time\_local] $upstream\_addr $upstream\_status $upstream\_response\_time "$request" $status $body\_bytes\_sent "$http\_referer" "$http\_user\_agent"';

log\_format cachelog '$time\_local - $upstream\_cache\_status - Cache-Control:$upstream\_http\_cache\_control - $request($status) - ';

#全局日志记录记录的位置及日志格式

access\_log logs/access.log main;

sendfile on;

tcp\_nopush on;

proxy\_ignore\_client\_abort on;

keepalive\_timeout 65;

#keepalive\_timeout 1000;

charset utf-8;

gzip on;

gzip\_min\_length 10k;

gzip\_buffers 4 16k;

gzip\_comp\_level 2;

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

gzip\_vary on;

#负载均衡示例 1

upstream balance1{

server 192.168.21.76:8093 max\_fails=3 fail\_timeout=30s;

}

#负载均衡示例 2

upstream balance2{

server 192.168.21.76:8070;

server 192.168.21.76:8071 down;

}

#负载均衡示例 3

upstream balance3{

server 192.168.21.76:8080 max\_fails=3 fail\_timeout=30s;

}

#web 服务 1

server {

listen 80;

server\_name www.website1.com;

#charset koi8-r;

#当前 web 服务的日志 位置、格式

access\_log logs/404\_access.log main;

#request proxy server

location / {

proxy\_set\_header Host $host;

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

proxy\_set\_header REMOTE-HOST $remote\_addr;

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

proxy\_pass http://balance1;

proxy\_redirect default;

}

#error\_page 404 /404.html;

# redirect server error pages to the static page /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

location /nginx\_status{

stub\_status on;

access\_log off;#指定全局的 access\_log 是否打开

allow all;

# deny all;

}

}

server {

listen 80;

server\_name website2.com;

#charset koi8-r;

#

access\_log logs/website2.log main;

#request proxy server

location / {

proxy\_set\_header Host $host;

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

proxy\_set\_header REMOTE-HOST $remote\_addr;

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

proxy\_pass http://balance3;

proxy\_redirect default;

}

#error\_page 404 /404.html;

# redirect server error pages to the static page /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

location /nginx\_status{

stub\_status on;

access\_log off;

allow all;

# deny all;

}

}

server {

listen 80;

server\_name www.website3.com;

#charset koi8-r;

access\_log logs/website3.log main;

#request proxy server

location / {

proxy\_set\_header Host $host;

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

proxy\_set\_header REMOTE-HOST $remote\_addr;

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

proxy\_pass http://balance1;

proxy\_redirect default;

}

#error\_page 404 /404.html;

# redirect server error pages to the static page /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

location /nginx\_status{

stub\_status on;

access\_log off;

allow all;

# deny all;

}

}

server {

listen 8060;

server\_name 192.168.1.111;

#charset koi8-r;

access\_log logs/website4.log main;

#request proxy server

location / {

proxy\_set\_header Host $host;

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

proxy\_set\_header REMOTE-HOST $remote\_addr;

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

proxy\_pass http://balance2;

proxy\_redirect default;

}

#error\_page 404 /404.html;

# redirect server error pages to the static page /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

location /nginx\_status{

stub\_status on;

access\_log off;

allow all;

# deny all;

}

}

}