LNMP(linux+nginx+mysql+php)
服务器环境配置

一、简介

Nginx 是俄罗斯人编写的十分轻量级的 HTTP 服务器,Nginx，它的发音为 “engine X”， 是一个高性能的 HTTP 和反向代理服务器，同时也是一个 IMAP/POP3/SMTP 代理服务器. Nginx 是由俄罗斯人 Igor Sysoev 为俄罗斯访问量第二的 Rambler.ru 站点开发的，它已经在该站点运行超过三年了。Igor Sysoev 在建立的项目时,使用基于 BSD 许可。

在高并发连接的情况下，Nginx 是 Apache 服务器不错的替代品。Nginx 同时也可以作为 7 层负载均衡服务器来使用。Nginx 0.8.46 + PHP 5.2.14 (FastCGI) 可以承受 3 万以上的并发连接数，相当于同等环境下 Apache 的 10 倍。

Nginx 超越 Apache 的高性能和稳定性，使得国内使用 Nginx 作为 Web 服务器的网站也越来越多，其中包括新浪博客、新浪播客、网易新闻、腾讯网、搜狐博客等门户网站频道，六间房、56.com 等视频分享网站，Discuz!官方论坛、水木社区等知名论坛，盛大在线、金山逍遥网等网络游戏网站，豆瓣、人人网、YUPOO 相册、金山爱词霸、迅雷在线等新兴 Web 2.0 网站。

为什么 Nginx 的性能要比 Apache 高得多？这得益于 Nginx 使用了最新的 epoll（Linux 2.6 内核）和 kqueue（freebsd）网络 I/O 模型，而 Apache 则使用的是传统的 select 模型。目前 Linux 下能够承受高并发访问的 Squid、Memcached 都采用的是 epoll 网络 I/O 模型。

处理大量的连接的读写，Apache 所采用的 select 网络 I/O 模型非常低效。下面用一个比喻来解析 Apache 采用的 select 模型和 Nginx 采用的 epoll 模型进行之间的区别：

假设你在大学读书，住的宿舍楼有很多间房间，你的朋友要来找你。select 版宿管大妈就会带着你的朋友挨个房间去找，直到找到你为止。而 epoll 版 宿管大妈会先记下每位同学的房间号，你的朋友来时，只需告诉你的朋友你住在哪个房间即可，不用亲自带着你的朋友满大楼找人。如果来了 10000 个人，都要 找自己住这栋楼的同学时，select 版和 epoll 版宿管大妈，谁的效率更高，不言自明。同理，在高并发服务器中，轮询 I/O 是最耗时间的操作之 一，select 和 epoll 的性能谁的性能更高，同样十分明了。

Nginx 的官方中文维基：http://wiki.nginx.org/NginxChs

二、系统环境

系统平台：RHEL 5.4（系统要求：Linux 2.6+ 内核）

Nginx 版本：nginx/1.0.15

Mysql 版本：5.1.35-log Source distribution

Php 版本：php-5.2.10

三、安装准备

1、获取相关开源程序并安装

RedHat 等其他 Linux 发行版可从安装光盘中找到这些程序库的 RPM 包 RedHat 可以直接利用 CentOS 的 RPM 包安装。

可以用 rpm 安装以下包，如有关联包，安装时一起安装。

```
gcc gcc-c++ autoconf libjpeg libjpeg-devel libpng libpng-devel
freetype freetype-devel libxml2 libxml2-devel zlib zlib-devel glibc
glibc-devel glib2 glib2-devel bzip2 bzip2-devel ncurses ncurses-devel
curl curl-devel e2fsprogs e2fsprogs-devel krb5 krb5-devel libidn
libidn-devel openssl openssl-devel openldap openldap-devel nss_ldap
openldap-clients openldap-servers
```

以上包如果安装了的话，不需要再安装了。

2、RPM 包搜索网站

http://rpm.pbone.net/
http://www.rpmfind.net/
http://code.google.com/p/zed-lnmp/

3、nginx 软件包准备

Nginx 所需要的软件包可以从下面位置获取，也可以记住名字在 google 中搜索。

nginx-0.7.61.tar.gz

php-5.2.10.tar.gz

php-5.2.10-fpm-0.5.11.diff.gz

mysql-5.1.35.tar.gz

libiconv-1.13.tar.gz

libmcrypt-2.5.8.tar.gz

mcrypt-2.6.8.tar.gz

memcache-2.2.5.tgz

mhash-0.9.9.9.tar.gz

pcre-7.9.tar.gz

eaccelerator-0.9.5.3.tar.bz2

PDO_MYSQL-1.0.2.tgz

ImageMagick.tar.gz

imagick-2.2.2.tgz

四、安装 PHP 5.2.10（FastCGI 模式）

编译安装 PHP 5.2.10 所需的支持库：

1.安装 libiconv

对文本进行编码间的转换，用它来处理中文各种编码之间的转换。

```
#tar zxvf libiconv-1.13.tar.gz
#cd libiconv-1.13/
#./configure --prefix=/usr/local
#make
#make install
cd ../
```

2.安装 libmcrypt 实现加密功能的库。

```
# tar zxvf libmcrypt-2.5.8.tar.gz

# cd libmcrypt-2.5.8/

# ./configure

# make

# make install

# /sbin/ldconfig

# 注：这里不要退出去了。

# cd libltdl/

# ./configure --enable-ltdl-install

# make

# make install

# cd ../../
```

3. 安装 mhash(哈稀函数库)

```
# tar zxvf mhash-0.9.9.9.tar.gz

# cd mhash-0.9.9.9/

# ./configure

# make

# make install

# cd ../

ln -s /usr/local/lib/libmcrypt.la /usr/lib/libmcrypt.la

ln -s /usr/local/lib/libmcrypt.so /usr/lib/libmcrypt.so

ln -s /usr/local/lib/libmcrypt.so.4 /usr/lib/libmcrypt.so.4

ln -s /usr/local/lib/libmcrypt.so.4.4.8 /usr/lib/libmcrypt.so.4.4.8

ln -s /usr/local/lib/libmhash.a /usr/lib/libmhash.a
```

```
ln -s /usr/local/lib/libmhash.la /usr/lib/libmhash.la

ln -s /usr/local/lib/libmhash.so /usr/lib/libmhash.so

ln -s /usr/local/lib/libmhash.so.2 /usr/lib/libmhash.so.2

ln -s /usr/local/lib/libmhash.so.2.0.1 /usr/lib/libmhash.so.2.0.1
```

4. 安装 mcrypt

```
# tar zxvf mcrypt-2.6.8.tar.gz

# cd mcrypt-2.6.8/

# /sbin/ldconfig

#./configure

# make

# make install

# cd ../
```

注：DG 库所需要安装包，可以采用 rpm 包来安装，减少时间，由于 php 已经集成 GD 库，但前提支持包应事先安装，如 zlib,png,jpeg,freetype 等。如果完全想采用 tar 包安装的话，请参考 lamp 相关内容。

五、编译安装 MySQL

建立 mysql 组，建立 mysql 用户并且加入到 mysql 组中

```
# groupadd mysql

# useradd mysql -g mysql

[root@linux lnmp]# tar zxvf mysql-5.1.35.tar.gz

[root@linux lnmp]# cd mysql-5.1.35

# ./configure --prefix=/usr/local/mysql

--without-debug

--with-extra-charsets=gbk
```

```
--with-extra-charsets=all

--enable-assembler

--with-pthread

--enable-thread-safe-client

--with-mysqld-ldflags=-all-static  /*不带共享库的形式编译 mysqld*/

--with-client-ldflags=-all-static

--with-big-tables

--with-readline    /*要采用 rpm 方式安装 ncurses 或 tar 包安装*/

--with-ssl            /*要采用 rpm 方式安装 openssl*/

--with-embedded-server

--enable-local-infile

--with-plugins=innobase

# make && make install

# /usr/local/mysql/bin/mysql_install_db --user=mysql

#以 mysql 身份初始化数据库

# cp ./support-files/mysql.server /etc/init.d/mysql

#复制 Mysql 启动服务至系统

# cp ./support-files/my-medium.cnf /etc/my.cnf

# chmod 755 /etc/init.d/mysql

# cd /usr/local/mysql/   #切换到 cd /usr/local/mysql/目录下

# chown -R mysql .       #改变当前目录下的所有者为 mysql 用户

# chown -R mysql var     #修改数据库目录的权限

# chgrp -R mysql .       #改变当前目录下的 mysql 用户的文件为 mysql 组

# /usr/local/mysql/bin/mysqld_safe --user=mysql&
```

```
# /usr/local/mysql/bin/mysqladmin -u root password 'admin' #设置管理员密码

[root@linux html]# /usr/local/mysql/bin/mysql -u root -p   #测试密码输入

Enter password:

Welcome to the MySQL monitor.  Commands end with ; or \g.

Your MySQL connection id is 5

Server version: 5.1.35-log Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> \q       /*退出 mysql*/

# chkconfig --add mysql  #添加 mysqld 服务到系统

# chkconfig mysql on     #打开 myslqd 服务

# service mysql start    #启动 Mysql

#/usr/local/mysql/bin/mysqladmin shutdown  #关闭数据库

#查看 mysql 端口的打开情况

# netstat -tunlp

Active Internet connections (only servers)

Proto Recv-Q Send-Q Local Address   Foreign Address  State   PID/P
name

tcp        0      0
0.0.0.0:3306    0.0.0.0:*        LISTEN  2936/

#查看是否启动:

#ps -ef | grep mysql
```

注:在配置过程中,整体的配置思路从上往下,其中的一些功能可以用在实际的配置过程中,主要用于测试环节中。

Mysql 的优化部分这里没有给出，需要参考其它资料。

六、编译安装 PHP

```
# tar zxvf php-5.2.10.tar.gz

# gzip -cd php-5.2.10-fpm-0.5.11.diff.gz | patch -d php-5.2.10 -p1

# 将 php-5.2.10-fpm-0.5.11.diff.gz 以补丁形式加到 php-5.2.10 里面

# cd php-5.2.10/

# ./configure --prefix=/usr/local/php

--with-config-file-path=/usr/local/php/etc

--with-mysql=/usr/local/mysql

--with-mysqli=/usr/local/mysql/bin/mysql_config

--with-iconv-dir=/usr/local

--with-freetype-dir

--with-jpeg-dir

--with-png-dir

--with-zlib

--with-gd

--enable-gd-native-ttf

--with-libxml-dir=/usr

--enable-xml

--disable-rpath

--enable-discard-path

--enable-safe-mode

--enable-bcmath
```

```
--enable-shmop

--enable-sysvsem

--enable-inline-optimization

--with-curl

--with-curlwrappers

--enable-mbregex

--enable-fastcgi

--enable-fpm

--enable-force-cgi-redirect

--enable-mbstring

--with-mcrypt

--with-openssl

--with-mhash

--enable-pcntl

--enable-sockets

--with-ldap

--with-ldap-sasl

--with-xmlrpc

--enable-zip

--enable-soap

--without-pear

#注：make 的时候一定要加上后面的参数，才能成功。

# make ZEND_EXTRA_LIBS='-liconv'

# make install
```

```
# cp php.ini-dist /usr/local/php/etc/php.ini

# cd ../
```

注：在安装过程中采用了 tar 包与 rpm 混合安装的情况，对于库的指定确实出现了很大的麻烦。如果采用 rpm 安装的话，不需要指定支持包的位置就可以了，tar 安装的话，需要指定安装位置。

七、编译安装 PHP5 扩展模块

1.安装 memcache

```
# tar zxvf memcache-2.2.5.tgz

# cd memcache-2.2.5/

# /usr/local/php/bin/phpize

# ./configure --with-php-config=/usr/local/php/bin/php-config

# make

# make install

# 说明：memcache 库的位置

Installing shared extensions:

/usr/local/php/lib/php/extensions/no-debug-non-zts-20060613/

# cd ..
```

2.安装 eaccelerator php 加速

```
# tar jxvf eaccelerator-0.9.5.3.tar.bz2

# cd eaccelerator-0.9.5.3/

# /usr/local/php/bin/phpize

# ./configure
--enable-eaccelerator=shared  --with-php-config=/usr/local/php/bin
/php-config

# make
```

```
# make install

Installing shared extensions:

/usr/local/php/lib/php/extensions/no-debug-non-zts-20060613/

# cd ../
```

3.安装 PDO_MYSQL(数据库连接的支持)

```
# tar zxvf PDO_MYSQL-1.0.2.tgz

# cd PDO_MYSQL-1.0.2/

# /usr/local/php/bin/phpize

#./configure --with-php-config=/usr/local/php/bin/php-config

--with-pdo-mysql=/usr/local/mysql

# make

# make install

Installing shared extensions:

/usr/local/php/lib/php/extensions/no-debug-non-zts-20060613/

# cd ../
```

4.安装 ImageMagick 是 Linux 下非常强大的图象处理函数与 GD 类似.

```
# tar zxvf ImageMagick.tar.gz

# cd ImageMagick-6.5.1-2/

#./configure

# make

# make install

# cd ../
```

5.安装 imagick(连接 PHP 和 ImageMagick 的通道)

```
# tar zxvf imagick-2.2.2.tgz
```

```
# cd imagick-2.2.2/

# /usr/local/php/bin/phpize

# ./configure --with-php-config=/usr/local/php/bin/php-config

# make

# make install

Installing shared extensions:

/usr/local/php/lib/php/extensions/no-debug-non-zts-20060613/

# cd ../
```

6.修改 php.ini 文件,已使 php 支持扩展的功能

```
vi /usr/local/php/etc/php.ini
```

查找

```
extension_dir = "./"
```

修改为

```
extension_dir="/usr/local/php/lib/php/extensions/no-debug-non-zts
-20060613/"
```

并在此行后增加以下几行，然后保存：

```
extension = "memcache.so"

extension = "pdo_mysql.so"

extension = "imagick.so"
```

再查找 output_buffering = Off

修改为 output_buffering = On

7.配置 eAccelerator 加速 PHP：

```
mkdir -p /usr/local/eaccelerator_cache

vi /usr/local/php/etc/php.ini
```

到配置文件的最末尾，粘上以下内容：

```
[eaccelerator]

zend_extension="/usr/local/php/lib/php/extensions/no-debug-non-zts-20060613/eaccelerator.so"

eaccelerator.shm_size="64"

eaccelerator.cache_dir="/usr/local/eaccelerator_cache"

eaccelerator.enable="1"

eaccelerator.optimizer="1"

eaccelerator.check_mtime="1"

eaccelerator.debug="0"

eaccelerator.filter=""

eaccelerator.shm_max="0"

eaccelerator.shm_ttl="3600"

eaccelerator.shm_prune_period="3600"

eaccelerator.shm_only="0"

eaccelerator.compress="1"

eaccelerator.compress_level="9"
```

八、PHP-fpm 配置

1.创建 php-fpm 配置文件

php-fpm 是为 PHP 打的一个 FastCGI 管理补丁，可以平滑变更 php.ini 配置而无需重启 php-cgi：

在/usr/local/php/etc/目录中创建 php-fpm.conf 文件，也可以在原有的基础上进行修改。

如果您安装 Nginx + PHP 用于程序调试

请将以下的

<value name="display_errors">0</value>改为

<value name="display_errors">1</value>，以便显示 PHP 错误信息，否则，Nginx 会报状态为 500 的空白错误页。

说明：创建 www 用户与组，这里创建了下面就不用创建了。

```
#/usr/sbin/groupadd www -g 48

#/usr/sbin/useradd -u 48 -g www www

rm -f /usr/local/php/etc/php-fpm.conf

vi /usr/local/php/etc/php-fpm.conf

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

#输入或者是修改为以下内容：

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

<?xml version="1.0" ?>

<configuration>

<section name="global_options">

<value name="pid_file">/usr/local/php/logs/php-fpm.pid</value>

<value name="error_log">/usr/local/php/logs/php-fpm.log</value>

<value name="log_level">notice</value>

<value name="emergency_restart_threshold">10</value>

<value name="emergency_restart_interval">1m</value>

<value name="process_control_timeout">5s</value>

<value name="daemonize">yes</value>

</section>

<workers>
```

```
<section name="pool">

<value name="name">default</value>

<value name="listen_address">127.0.0.1:9000</value>

<value name="listen_options">

<value name="backlog">-1</value>

<value name="owner"></value>

<value name="group"></value>

<value name="mode">0666</value>

</value>

<value name="php_defines">

<value name="sendmail_path">/usr/sbin/sendmail -t -i</value>

<value name="display_errors">1</value>

</value>

<value name="user">www</value>

<value name="group">www</value>

<value name="pm">

<value name="style">static</value>

<value name="max_children">128</value>

<value name="apache_like">

<value name="StartServers">20</value>

<value name="MinSpareServers">5</value>

<value name="MaxSpareServers">35</value>

</value>

</value>
```

```
<value name="request_terminate_timeout">0s</value>

<value name="request_slowlog_timeout">0s</value>

<value name="slowlog">logs/slow.log</value>

<value name="rlimit_files">51200</value>

<value name="rlimit_core">0</value>

<value name="chroot"></value>

<value name="chdir"></value>

<value name="catch_workers_output">yes</value>

<value name="max_requests">500</value>

<value name="allowed_clients">127.0.0.1</value>

<value name="environment">

<value name="HOSTNAME">$HOSTNAME</value>

<value name="PATH">/usr/local/bin:/usr/bin:/bin</value>

<value name="TMP">/tmp</value>

<value name="TMPDIR">/tmp</value>

<value name="TEMP">/tmp</value>

<value name="OSTYPE">$OSTYPE</value>

<value name="MACHTYPE">$MACHTYPE</value>

<value name="MALLOC_CHECK_">2</value>

</value>

</section>

</workers>

</configuration>
```

2.php-fpm 启动与管理

/usr/local/php/sbin/php-fpm start

注：/usr/local/php/sbin/php-fpm 还有其他参数，包括：

start|stop|quit|restart|reload|logrotate,修改 php.ini 后不重启 php-cgi，重新加载配置文件使用 reload，就保持了在 php 的 fastcgi 进程持续运行的状态下，又重新加载了 php.ini。

九、Nginx 安装

Nginx 只是 web 服务器，配合 php 技术实现的 fastcgi 来提高性能。

1、安装 rewrite 模块支持包 pcre 库：

```
# tar zxvf pcre-7.8.tar.gz

# cd pcre-7.8/

# ./configure

# make && make install

cd ../
```

2.安装 Nginx

说明:创建 www 用户组及 www 用户，如果之前 php-fpm 没有创建，这里要创建。

```
# /usr/sbin/groupadd www

# /usr/sbin/useradd -g www www

# tar zxvf nginx-1.0.15.tar.gz

# cd nginx-1.0.15/

#./configure --user=www --group=www --prefix=/usr/local/nginx
--with-http_stub_status_module --with-http_ssl_module

  nginx path prefix: "/usr/local/nginx"

  nginx binary file: "/usr/local/nginx/sbin/nginx"

  nginx configuration prefix: "/usr/local/nginx/conf"
```

```
  nginx configuration file: "/usr/local/nginx/conf/nginx.conf"

  nginx pid file: "/usr/local/nginx/logs/nginx.pid"

  nginx error log file: "/usr/local/nginx/logs/error.log"

  nginx http access log file: "/usr/local/nginx/logs/access.log"

  nginx http client request body temporary files: "client_body_temp"

  nginx http proxy temporary files: "proxy_temp"

  nginx http fastcgi temporary files: "fastcgi_temp"


# make && make install

# cd ../
```

3. Nginx 安装后只有一个程序文件，本身并不提供各种管理程序，它是使用参数和系统信号机制对 Nginx 进程本身进行控制的。

Nginx 的参数包括有如下几个：

-c <path_to_config>：使用指定的配置文件而不是 conf 目录下的 nginx.conf 。

-t：测试配置文件是否正确，在运行时需要重新加载配置的时候，此命令非常重要，用来检测所修改的配置文件是否有语法错误。

-v：显示 nginx 版本号。

-V：显示 nginx 的版本号以及编译环境信息以及编译时的参数。

例如我们要测试某个配置文件是否书写正确，我们可以使用以下命令

```
sbin/nginx -t -c conf/nginx.conf
```

十、nginx 配置

1.在/usr/local/nginx/conf/目录中创建 nginx.conf 文件：

```
rm -f /usr/local/nginx/conf/nginx.conf

vi /usr/local/nginx/conf/nginx.conf
```

```
=======================================

nginx.conf 才是 nginx web 服务器的配置文件

=======================================

user  www www;         /*启动 nginx 服务的用户与组*/

worker_processes 1;    /*启动 nginx 服务的工作进程*/

error_log  logs/nginx_error.log  crit; /*错误日志，以及等级*/

pid        /usr/local/nginx/nginx.pid;  /*nginx 服务进程 PID*/

worker_rlimit_nofile 51200;

events

{

 use epoll;                 /*工作模式*/

 worker_connections 51200; /*每进程允许最大的同时连接数*/

}

http

{

 include       mime.types;

 default_type  application/octet-stream;

 #charset  gb2312;

 server_names_hash_bucket_size 128;

 client_header_buffer_size 32k;

 large_client_header_buffers 4 32k;


 sendfile on;

 tcp_nopush     on;
```

```
 keepalive_timeout 60;

 tcp_nodelay on;

 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 on;

 gzip_min_length  1k;

 gzip_buffers     4 16k;

 gzip_http_version 1.0;

 gzip_comp_level 2;

 gzip_types text/plain application/x-javascript text/css
application/xml;

 gzip_vary on;

 #limit_zone  crawler  $binary_remote_addr  10m;

 server

 {

   listen       80;               /*监听端口*/

   server_name  localhost;        /*服务器名称*/

   index index.html index.htm index.php;  /*缺省主页名称*/

   root  /usr/local/nginx/html;   /*网站根目录，也可以采用下面内容*/
```

```
#也可以采用相对路径,下面注释部分*/
    #location / {
    #          root   html;
    #          index  index.html index.htm;
    # }
    #limit_conn   crawler  20;
```

#通过 FastCGI 方式支持 PHP，php 页面由 fastcgi 代理处理，这也是反向代理的一个应用，这里可以是 jsp/asp 等脚本。

```
    location ~ .*\.(php|php5)?$
    {
      #fastcgi_pass  unix:/tmp/php-cgi.sock;
      fastcgi_pass  127.0.0.1:9000;   /*fastcgi 监听端口*/
      fastcgi_index index.php;
      include fcgi.conf;        /*fastcgi 配置文件，修改为以下内容*/
    }
```

#对于某一类型的文件，设置过期时间，静态的页面通常设置长一点。

```
    #静态文件，nginx 自己处理
    location ~ .*\.(gif|jpg|jpeg|png|bmp|swf|js|css)$
    {
      expires      30d;
    }
#日志的格式
```

```
   log_format  access  '$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  access;

     }

}
```

说明:以上配置文件只是基本配置文件，要实现其它功能的话，需要在此基础上进行修改。

2.在/usr/local/nginx/conf/目录中创建 fcgi.conf 文件：

说明：可以直接粘贴以下内容。

```
vi /usr/local/nginx/conf/fcgi.conf

fastcgi_param  GATEWAY_INTERFACE  CGI/1.1;

fastcgi_param  SERVER_SOFTWARE    nginx;

fastcgi_param  QUERY_STRING       $query_string;

fastcgi_param  REQUEST_METHOD     $request_method;

fastcgi_param  CONTENT_TYPE       $content_type;

fastcgi_param  CONTENT_LENGTH     $content_length;

fastcgi_param  SCRIPT_FILENAME    $document_root$fastcgi_script_na
me;

fastcgi_param  SCRIPT_NAME        $fastcgi_script_name;

fastcgi_param  REQUEST_URI        $request_uri;

fastcgi_param  DOCUMENT_URI       $document_uri;

fastcgi_param  DOCUMENT_ROOT      $document_root;

fastcgi_param  SERVER_PROTOCOL    $server_protocol;

fastcgi_param  REMOTE_ADDR        $remote_addr;
```

```
fastcgi_param  REMOTE_PORT        $remote_port;

fastcgi_param  SERVER_ADDR        $server_addr;

fastcgi_param  SERVER_PORT        $server_port;

fastcgi_param  SERVER_NAME        $server_name;

# PHP only, required if PHP was built with --enable-force-cgi-redirect

fastcgi_param  REDIRECT_STATUS    200;
```

十一、nginx 启动与管理

1．启动 nginx

```
/usr/local/nginx/sbin/nginx
```

2．测试 nginx 配置文件

修改/usr/local/nginx/conf/nginx.conf 配置文件后，请执行以下命令检查配置文件是否正确：

```
# /usr/local/nginx/sbin/nginx -t
```

如果屏幕显示以下两行信息，说明配置文件正确：

```
the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok

the configuration file /usr/local/nginx/conf/nginx.conf was tested
successfully
```

3．查看 Nginx 主进程号

```
ps -ef | grep "nginx: master process" | grep -v "grep" | awk -F ' '
'{print $2}'
```

屏幕显示的即为 Nginx 主进程号，例如：

```
6302
```

这时，执行以下命令即可使修改过的 Nginx 配置文件生效：

```
kill -HUP 6302
```

或者无需这么麻烦，找到 Nginx 的 Pid 文件：

```
kill -HUP `cat /usr/local/nginx/logs/nginx.pid`
```

4. 配置开机自动启动 Nginx + PHP

```
vi /etc/rc.local
```

加入以下内容：

```
ulimit -SHn 51200

/usr/local/php/sbin/php-fpm start

/usr/local/nginx/sbin/nginx
```

5.测试 nginx

```
vi /usr/local/nginx/html/test.php

<?

phpinfo();

?>
```

6.测试结果