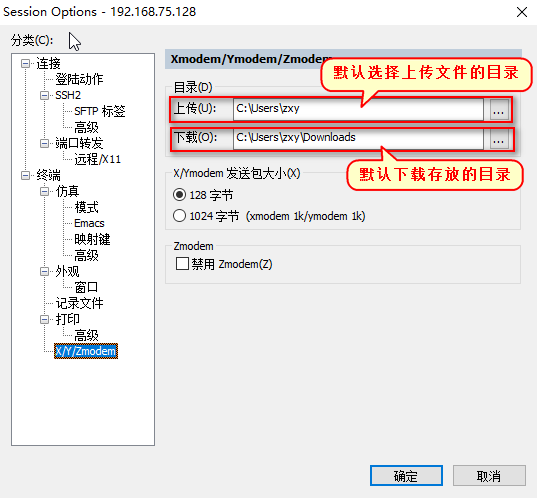
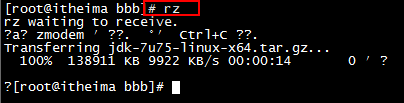
**在linux上安装lrzsz工具，安装方式如下：安装： yum install lrzsz 卸载：yum remove lrzsz**

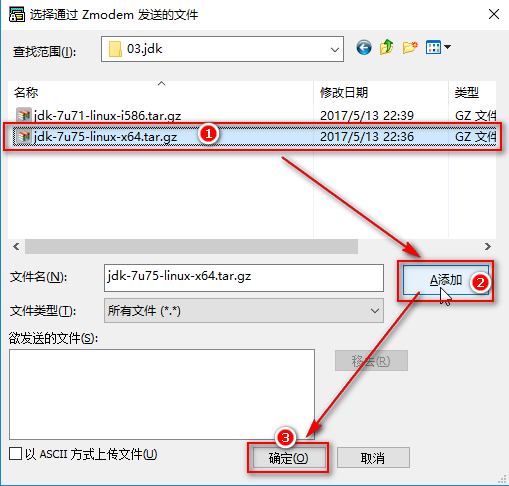


**安装完成之后需要设置windows上传和下载的目录：CTR--->选项--->会话选项**

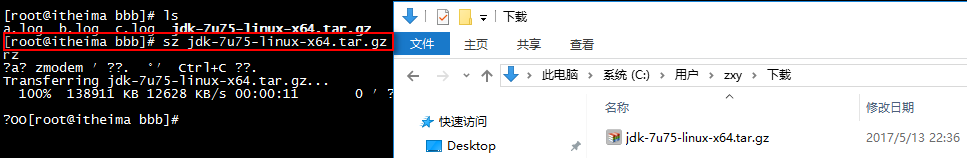


**上传文件：**



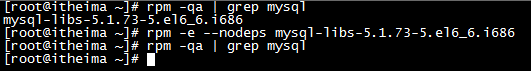


**下载文件：**



# dlinux下软件安装-mysql上传与解包

1. rpm -qa | grep mysql 查找jdk
2. rpm -e --nodeps mysql-libs-5.1.73-8.el6\_8.x86\_64 卸载mysql
3. rpm -e --nodeps java-1.7.0-openjdk-1.7.0.79-2.5.5.4.el6.i686 卸载jdk
   1. 卸载已经自带的mysql；



[root@pc ~]# cd /usr/local/src/

[root@pc src]# yum -y install bison expat expat-devel ncurses ncurses-devel libaio libaio-devel libc.so.6

[root@pc src]# wget https://cmake.org/files/v3.5/cmake-3.5.2.tar.gz

[root@pc src]# tar -xzf cmake-3.5.2.tar.gz

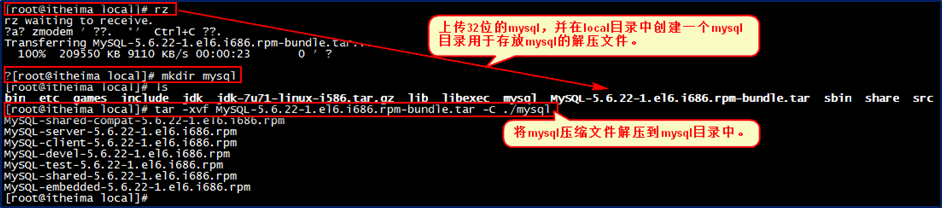
[root@pc src]# cd cmake-3.5.2

[root@pc cmake-3.5.2]# ./configure

[root@pc cmake-3.5.2]# make

[root@pc cmake-3.5.2]# make install

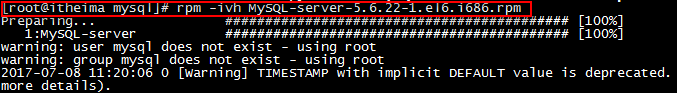
* 1. 上传并解压我们的mysql

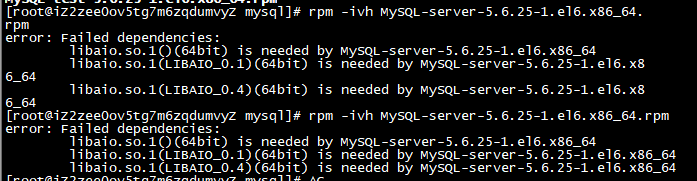


# linux下软件安装-mysql服务器与客户端安装

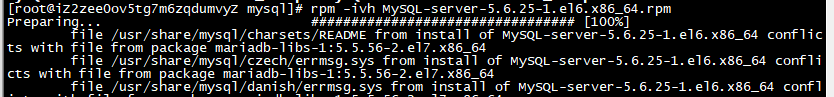
参考：**[资料\实战文档\实战二：Linux下安装mysql.docx](C:/xuchao/javaweb/资料/实战文档/实战二：Linux下安装mysql.docx)**

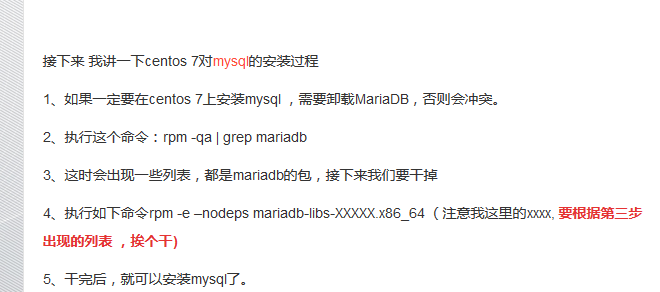
【安装mysql服务端】：进入到mysql目录中,使用rpm -ivh MySQL-server-5.6.22-1.el6.i686.rpm安装mysql服务端

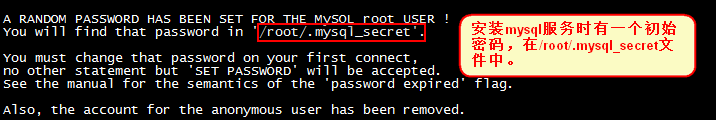




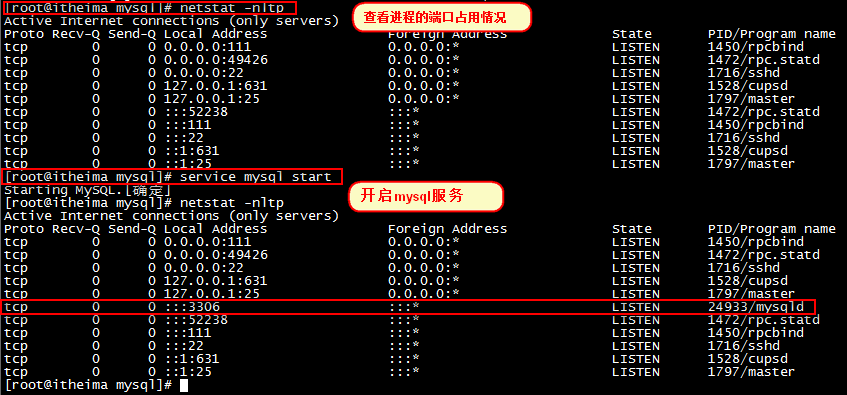








**安装完mysql服务端之后,开启mysql服务:**



【安装mysql客户端】:在mysql目录中,使用rpm -ivh MySQL-client-5.6.22-1.el6.i686.rpm安装mysql客户端：

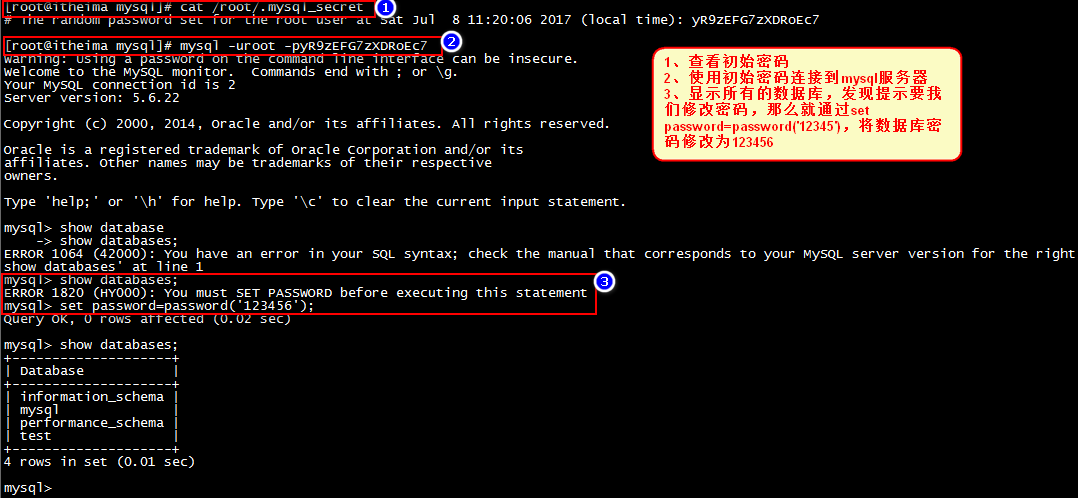


查看初始密码，访问mysql数据库：

cat /root/.mysql\_secret

mysql -uroot -pR30NJZwqCvMvGN35

set password=password('123456');



# linux下软件安装-mysql远程访问

**将mysql服务添加到系统服务并开机启动：**

加入到系统服务：

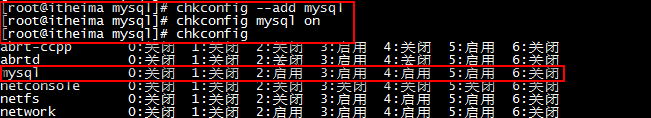
chkconfig --add mysql

自动启动：

chkconfig mysql on

查询列表：

chkconfig



**mysql远程访问：**

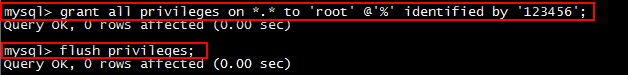
**登录：mysql -uroot –p123456**

**mysql -u root –p**

**设置远程访问（使用root密码）：**

grant all privileges on \*.\* to 'root' @'%' identified by '123456'; //后面的root是你数据库的密码

flush privileges;

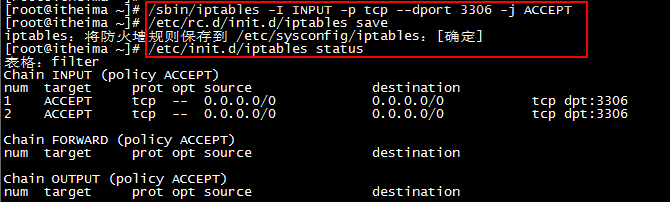


**防火墙打开3306端口：**

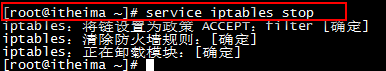
/sbin/iptables -I INPUT -p tcp --dport 3306 -j ACCEPT

/etc/rc.d/init.d/iptables save

/etc/init.d/iptables status

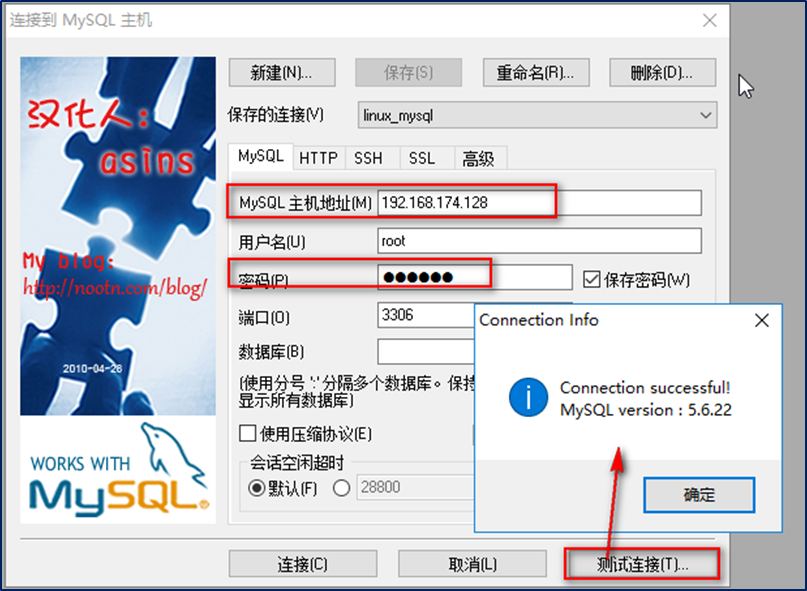


**或者关闭防火墙：service iptables stop**



**打开SQLyog进行远程连接测试：**

chkconfig iptables off 禁止防火墙自启



在安装nginx前首先要确认系统中安装了**gcc、pcre-devel、zlib-devel、openssl-devel**。

Linux下检查是否安装过某软件包：[http://www.cnblogs.com/xxoome/p/5866553.html](http://www.cnblogs.com/xxoome/p/5866553.html" \t "http://www.cnblogs.com/xxoome/p/_blank)

安装命令：

yum -y install gcc pcre-devel zlib-devel openssl openssl-devel

nginx下载地址：[https://nginx.org/download/](https://nginx.org/download/" \t "http://www.cnblogs.com/xxoome/p/_blank)

1. **linxu下安装nginx**

**3.1、环境要求**

nginx是C语言开发，建议在linux上运行，本教程使用Centos6.4作为安装环境。

* gcc

安装nginx需要先将官网下载的源码进行编译，编译依赖gcc环境，如果没有gcc环境，需要安装gcc：

**yum install gcc-c++**

* PCRE

PCRE(Perl Compatible Regular Expressions)是一个Perl库，包括 perl 兼容的正则表达式库。nginx的http模块使用pcre来解析正则表达式，所以需要在linux上安装pcre库。

**yum install -y pcre pcre-devel**

注：pcre-devel是使用pcre开发的一个二次开发库。nginx也需要此库。

* zlib

zlib库提供了很多种压缩和解压缩的方式，nginx使用zlib对http包的内容进行gzip，所以需要在linux上安装zlib库。

**yum install -y zlib zlib-devel**

* openssl

OpenSSL 是一个强大的安全套接字层密码库，囊括主要的密码算法、常用的密钥和证书封装管理功能及SSL协议，并提供丰富的应用程序供测试或其它目的使用。

nginx不仅支持http协议，还支持https（即在ssl协议上传输http），所以需要在linux安装openssl库。

**yum install -y openssl openssl-devel**

[root@pc ~]# cd /usr/local/src/

root@pc src]# wget http://nginx.org/download/nginx-1.9.9.tar.gz

[root@pc src]# tar -xzf nginx-1.9.9.tar

[IMG_256](http://www.cnblogs.com/xxoome/p/javascript:void(0);)

## 解压

tar -zxvf nginx-1.9.9.tar.gz

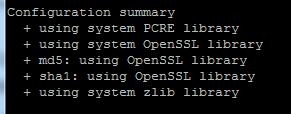
##进入nginx目录

cd nginx-1.9.9

## 配置  
./configure --prefix=/usr/local/nginx  
  
# make  
make  
make install

[IMG_257](http://www.cnblogs.com/xxoome/p/javascript:void(0);)

OK，现在可以执行make 了。



执行make、make install命令

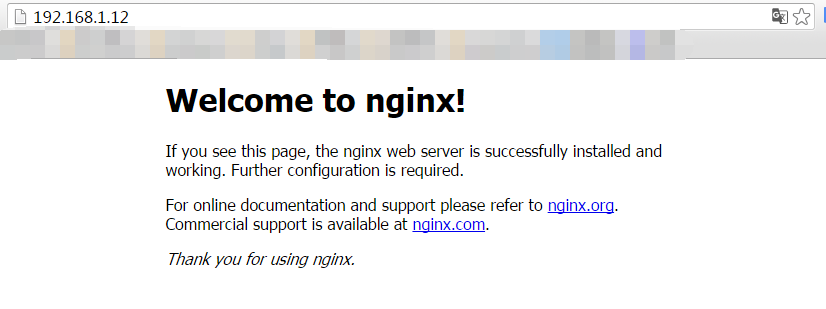
启动nginx

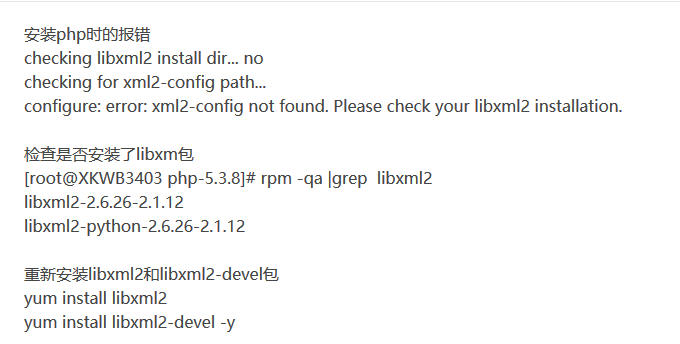
cd /usr/local/nginx/sbin

./nginx //启动nginx

在浏览器中输入服务器的ip地址，如：192.168.1.12

刷新浏览器





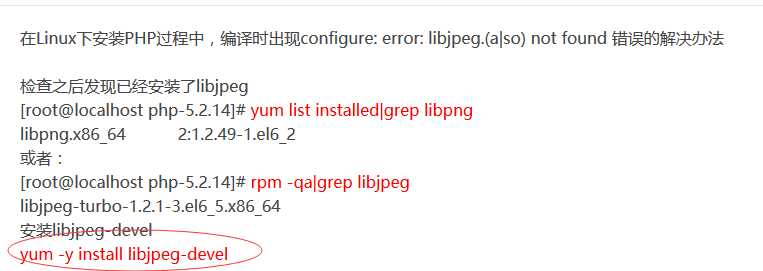
yum -y install libxml2-devel -y

yum -y install libxml2



yum -y install curl-devel

yum -y install libjpeg-devel







yum -y install libpng-devel

yum -y install libpng



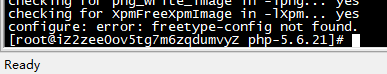


yum -y install libXpm-devel

yum install bzip2 bzip2-devel

yum -y install freetype-devel

yum -y install libxslt-devel



**安装必要的依赖软件**

如果已经安装了可能会进行升级，版本完全一致则不会进行任何操作。

yum -y install libjpeg libjpeg-devel libpng libpng-devel libvpx libvpx-devel libXpm libXpm-devel libxml2 libxml2-devel fontconfig fontconfig-devel freetype freetype-devel zlib zlib-devel bzip2 bzip2-devel curl curl-devel openssl openssl-devel

yum -y install bzip2-devel curl-devel freetype-devel gcc libjpeg-devel libpng-devel libxslt-devel libxml2-devel openssl-devel pcre-devel pcre-devel zlib-devel

1. **安装php**

[root@pc ~]# cd /usr/local/src/

[root@pc src]# wget ftp://mcrypt.hellug.gr/pub/crypto/mcrypt/libmcrypt/libmcrypt-2.5.7.tar.gz

[root@pc src]# tar -xzf libmcrypt-2.5.7.tar.gz

[root@pc src]# cd libmcrypt-2.5.7

[root@pc libmcrypt-2.5.7]# ./configure

[root@pc libmcrypt-2.5.7]# make

[root@pc libmcrypt-2.5.7]# make install

[root@pc libmcrypt-2.5.7]# ldconfig

[root@pc libmcrypt-2.5.7]# cd libltdl/

[root@pc libltdl]# ./configure --enable-ltdl-install

[root@pc libltdl]# make

[root@pc libltdl]# make install

[root@pc libltdl]# cd ../../

[root@pc src]# wget http://jaist.dl.sourceforge.net/project/mhash/mhash/0.9.9.9/mhash-0.9.9.9.tar.gz

[root@pc src]# tar -zxf mhash-0.9.9.9.tar.gz

[root@pc src]# cd mhash-0.9.9.9

[root@pc mhash-0.9.9.9]# ./configure

[root@pc mhash-0.9.9.9]# make

[root@pc mhash-0.9.9.9]# make install

[root@pc mhash-0.9.9.9]# cd ../

[root@pc src]# wget http://jaist.dl.sourceforge.net/project/lnmpaio/web/mcrypt/mcrypt-2.6.8.tar.gz

[root@pc src]# tar -xzf mcrypt-2.6.8.tar.gz

[root@pc src]# cd mcrypt-2.6.8

[root@pc mcrypt-2.6.8]# ./configure LD\_LIBRARY\_PATH=/usr/local/lib

[root@pc mcrypt-2.6.8]# make

[root@pc mcrypt-2.6.8]# make install

[root@pc mcrypt-2.6.8]# cd ../

[root@pc src]# wget http://mirror.centos.org/centos/6/os/x86\_64/Packages/libzip-0.9-3.1.el6.x86\_64.rpm

[root@pc src]# rpm -ivh libzip-0.9-3.1.el6.x86\_64.rpm

[root@pc src]# wget http://cn2.php.net/distributions/php-5.6.21.tar.gz

[root@pc src]# tar -xzf php-5.6.21.tar.gz

[root@pc src]# cd php-5.6.21

[root@pc php-5.6.21]# echo '/usr/local/lib' >> /etc/ld.so.conf.d/local.conf

[root@pc php-5.6.21]# echo '/usr/local/lib64' >> /etc/ld.so.conf.d/local.conf

[root@pc php-5.6.21]# ldconfig

**3、配置安装变量**

./configure --prefix=/usr/local/php-5.6.21 --with-curl --with-freetype-dir --with-gd --with-gettext --with-iconv-dir --with-kerberos --with-libdir=lib64 --with-libxml-dir --with-mysqli --with-openssl --with-pcre-regex --with-pdo-mysql --with-pdo-sqlite --with-pear --with-png-dir --with-jpeg-dir --with-xmlrpc --with-xsl --with-zlib --with-bz2 --with-mhash --enable-fpm --enable-bcmath --enable-libxml --enable-inline-optimization --enable-gd-native-ttf --enable-mbregex --enable-mbstring --enable-opcache --enable-pcntl --enable-shmop --enable-soap --enable-sockets --enable-sysvsem --enable-sysvshm --enable-xml --enable-zip

./configure \--prefix=/usr/local/php-5.6.21 \--with-config-file-path=/usr/local/php-5.6.21/etc \--enable-fpm \--enable-mysqlnd \--with-mysqli\--with-pdo-mysql\--with-libxml-dir \--enable-zip \--with-zlib \--with-zlib-dir \--with-curl \--with-mcrypt \--with-gd \--enable-gd-native-ttf \--with-openssl \--with-mhash \--with-xmlrpc \--with-jpeg-dir \--with-png-dir \--with-xpm-dir \--with-freetype-dir \--enable-shared \--enable-xml \--disable-rpath \--enable-bcmath \--enable-shmop \--enable-sysvsem \--enable-inline-optimization \--enable-mbregex \--enable-mbstring \--enable-pcntl \--enable-sockets \--enable-soap

“/usr/local/php-5.6.21”是安装路径，可以改成自己喜欢的安装路径。

**4、编译源码**

# 在解压目录执行编译命令：make  
cd php-5.6.21

make

编译会花费一些时间，耐心等待即可。编译完成的最后会提示你执行make test命令。就是对上一步 make 的检查，

要确保 make 是没有错误的，也就是这一步的 test、check要全部是 OK 的，error 为0。

**5、安装php**

make install

**6、配置PHP**

[root@pc php-5.6.21]# cd /usr/local/php-5.6.21/etc

[root@pc etc]# cp /usr/local/src/php-5.6.21/php.ini-development php.ini

[root@pc etc]# cp php-fpm.conf.default php-fpm.conf

root@pc etc]# vi php-fpm.conf

pm.max\_children = 40

pm.start\_servers = 20

pm.min\_spare\_servers = 5

pm.max\_spare\_servers = 35

root@pc etc]# vi php.ini

date.timezone = 'PRC'

error\_log = /data/logs/php\_errors.log

[root@pc etc]# /usr/local/php-5.6.21/sbin/php-fpm

[root@pc etc]# ps -ef | grep fpm

root 9516 1 0 15:52 ? 00:00:00 php-fpm: master process (/usr/local/php-5.6.21/etc/php-fpm.conf)

www 9517 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9518 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9519 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9520 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9521 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9522 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9523 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9524 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9525 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9526 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9527 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9528 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9529 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9530 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9531 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9532 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9533 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9534 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9535 9516 0 15:52 ? 00:00:00 php-fpm: pool www

www 9536 9516 0 15:52 ? 00:00:00 php-fpm: pool www

root 9545 1666 0 15:54 pts/0 00:00:00 grep fpm

[root@pc etc]# /usr/local/php-5.6.21/bin/php -v

PHP 5.6.21 (cli) (built: May 7 2016 15:29:00)

Copyright (c) 1997-2016 The PHP Group

Zend Engine v2.6.0, Copyright (c) 1998-2016 Zend Technologies

配置php.ini “cgi.fix\_pathinfo=0”

启动php-fpm服务：

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

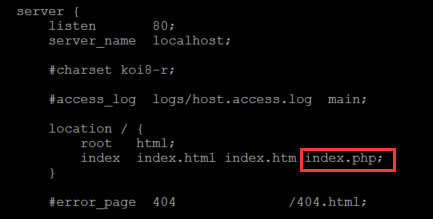
启动完毕之后，php-fpm服务默认使用9000端口，使用 netstat -tln | grep 9000 可以查看端口使用情况：

IMG_256

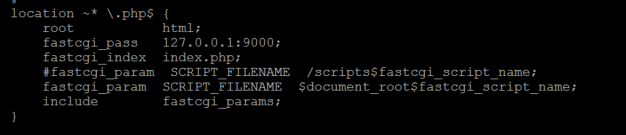
**配置nginx**

nginx安装教程：[http://www.cnblogs.com/xxoome/p/5866475.html](http://www.cnblogs.com/xxoome/p/5866475.html" \t "https://www.cnblogs.com/xxoome/p/_blank)

编辑nginx配置文件/usr/local/nginx/conf/nginx.conf，主要修改nginx的server {}配置块中的内容，修改location块，追加index.php让nginx服务器默认支持index.php为首页：



然后配置.php请求被传送到后端的php-fpm模块，默认情况下php配置块是被注释的，此时去掉注释并修改为以下内容：



这里面很多都是默认的，root是配置php程序放置的根目录，主要修改的就是fastcgi\_param中的/scripts为$document\_root

修改完这些保存并退出，然后重启nginx：/usr/local/nginx/sbin/nginx -s reload

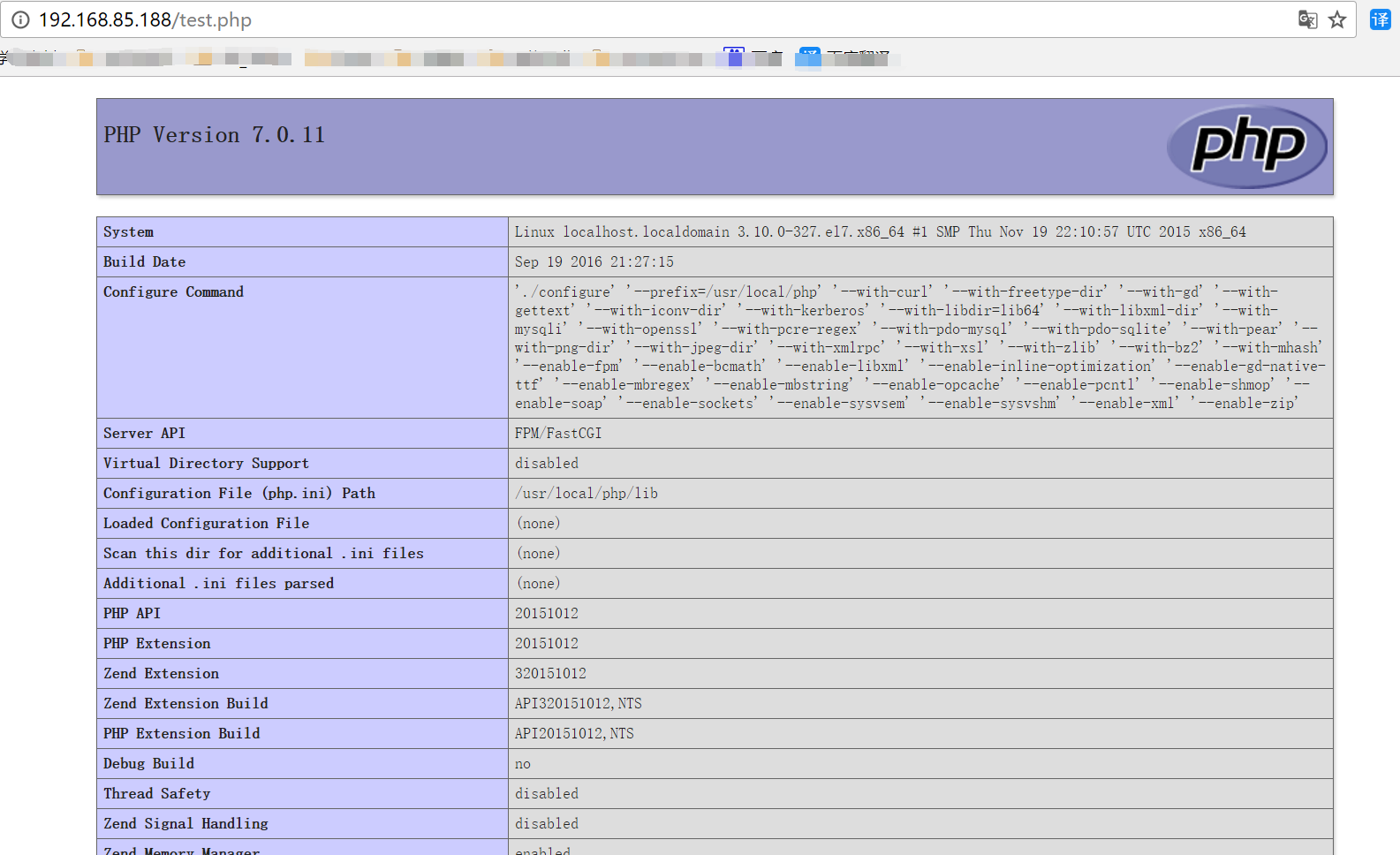
接下来编辑一个测试的php程序，在nginx下的html目录下创建test.php文件，打印一下php配置：

<?php

phpinfo();

?>

　　然后打开浏览器输入对应的地址进行访问，看到输出页面，说明nginx和php都配置成功了：



在/usr/local/nginx/conf/添加一个

**pathinfo.conf配置文件内容如下**

set $real\_script\_name $fastcgi\_script\_name;

if ($fastcgi\_script\_name ~ "(.+?\.php)(/.\*)") {

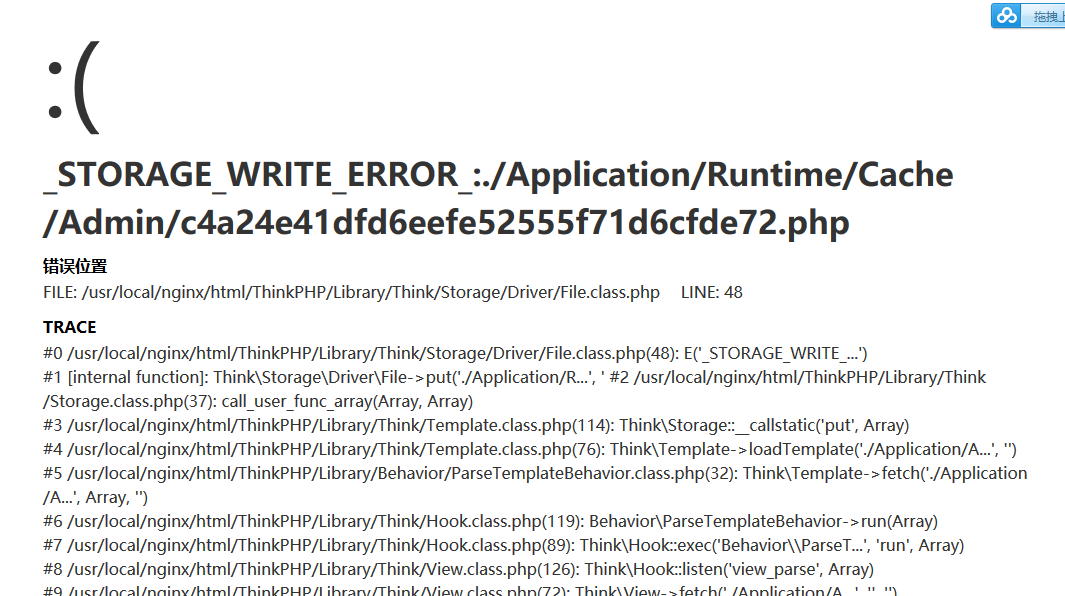
set $real\_script\_name $1;

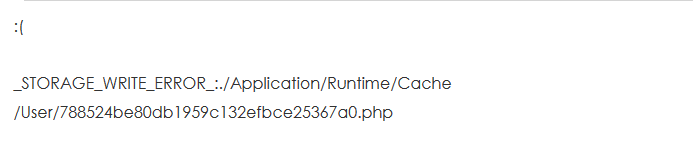
set $path\_info $2;

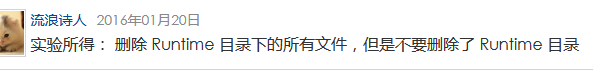
}

fastcgi\_param SCRIPT\_NAME $fastcgi\_script\_name;

fastcgi\_param PATH\_INFO $path\_info;









**Nginx.Conf配置**

#user nobody;

worker\_processes 1;

#error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.log info;

#pid logs/nginx.pid;

events {

worker\_connections 1024;

}

http {

include mime.types;

default\_type application/octet-stream;

#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;

#keepalive\_timeout 0;

keepalive\_timeout 65;

#gzip on;

server {

listen 80;

server\_name localhost;

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

root html;

index index.html index.htm index.php;

if (!-e $request\_filename) {

rewrite ^/(.\*)$ /index.php?s=$1 last;

}

}

#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;

}

# proxy the PHP scripts to Apache listening on 127.0.0.1:80

#

#location ~ \.php$ {

# proxy\_pass http://127.0.0.1;

#}

# pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000

#

location ~ \.php$ {

root html;

fastcgi\_pass 127.0.0.1:9000;

fastcgi\_index index.php;

include fastcgi\_params;

include fastcgi.conf;

include pathinfo.conf;

#fastcgi\_param SCRIPT\_FILENAME /scripts$fastcgi\_script\_name;

fastcgi\_split\_path\_info ^(.+\.php)(/.+)$;

fastcgi\_param PATH\_INFO $fastcgi\_path\_info;

fastcgi\_param PATH\_TRANSLATED $document\_root$fastcgi\_path\_info;

}

#location ~ \.php$ {

# root html;

# fastcgi\_pass 127.0.0.1:9000;

# fastcgi\_index index.php;

# fastcgi\_param SCRIPT\_FILENAME /scripts$fastcgi\_script\_name;

# include fastcgi\_params;

#}

# deny access to .htaccess files, if Apache's document root

# concurs with nginx's one

#

#location ~ /\.ht {

# deny all;

#}

}

# another virtual host using mix of IP-, name-, and port-based configuration

#

#server {

# listen 8000;

# listen somename:8080;

# server\_name somename alias another.alias;

# location / {

# root html;

# index index.html index.htm;

# }

#}

# HTTPS server

#

#server {

# listen 443 ssl;

# server\_name localhost;

# ssl\_certificate cert.pem;

# ssl\_certificate\_key cert.key;

# ssl\_session\_cache shared:SSL:1m;

# ssl\_session\_timeout 5m;

# ssl\_ciphers HIGH:!aNULL:!MD5;

# ssl\_prefer\_server\_ciphers on;

# location / {

# root html;

# index index.html index.htm;

# }

#}

}

[root@pc ~]# cd /usr/local/src/

root@pc src]# wget https://launchpad.net/libmemcached/1.0/1.0.18/+download/libmemcached-1.0.18.tar.gz

[root@pc src]# tar -xzf libmemcached-1.0.18.tar.gz

[root@pc src]# cd libmemcached-1.0.18

[root@pc libmemcached-1.0.18]# ./configure --prefix=/usr/local/libmemcached-1.0.18

[root@pc libmemcached-1.0.18]# make

[root@pc libmemcached-1.0.18]# make install

[root@pc libmemcached-1.0.18]# cd ../

[root@pc src]# wget http://pecl.php.net/get/memcached-2.2.0.tgz

[root@pc src]# tar -xzf memcached-2.2.0.tgz

[root@pc src]# cd memcached-2.2.0

[root@pc memcached-2.2.0]# /usr/local/php-5.6.21/bin/phpize

[root@pc memcached-2.2.0]# ./configure --enable-memcached \

--with-php-config=/usr/local/php-5.6.21/bin/php-config \

--with-libmemcached-dir=/usr/local/libmemcached-1.0.18

checking sasl/sasl.h usability... no

checking sasl/sasl.h presence... no

checking for sasl/sasl.h... no

configure: error: no, sasl.h is not available. Run configure with --disable-memcached-sasl to disable this check

[root@pc memcached-2.2.0]# yum -y install cyrus-sasl-devel

[root@pc memcached-2.2.0]# /usr/local/php-5.6.21/bin/phpize

[root@pc memcached-2.2.0]# ./configure --enable-memcached --disable-memcached-sasl \

--with-php-config=/usr/local/php-5.6.21/bin/php-config \

--with-libmemcached-dir=/usr/local/libmemcached-1.0.18

[root@pc memcached-2.2.0]# make

[root@pc memcached-2.2.0]# make install

[root@pc memcached-2.2.0]# vi /usr/local/php-5.6.21/etc/php.ini

extension=memcached.so

[root@pc memcached-2.2.0]# service php-fpm restart

[root@pc memcached-2.2.0]# /usr/local/php-5.6.21/bin/php -i |grep memcached

[root@pc memcached-2.2.0]# cd ../

[root@pc src]# wget http://pecl.php.net/get/memcache-2.2.7.tgz

[root@pc src]# tar -xzf memcache-2.2.7.tgz

[root@pc src]# cd memcache-2.2.7

[root@pc memcache-2.2.7]# /usr/local/php-5.6.21/bin/phpize

[root@pc memcache-2.2.7]# ./configure --with-php-config=/usr/local/php-5.6.21/bin/php-config

[root@pc memcache-2.2.7]# make

[root@pc memcache-2.2.7]# make install

[root@pc memcache-2.2.7]# vi /usr/local/php-5.6.21/etc/php.ini

extension=memcache.so

[root@pc ~]# cd /usr/local/src/

[root@pc src]# wget https://github.com/downloads/libevent/libevent/libevent-2.0.21-stable.tar.gz

[root@pc src]# tar -xzf libevent-2.0.21-stable.tar.gz

[root@pc src]# cd libevent-2.0.21-stable

[root@pc libevent-2.0.21-stable]# ./configure -prefix=/usr

[root@pc libevent-2.0.21-stable]# make

[root@pc libevent-2.0.21-stable]# make install

[root@pc libevent-2.0.21-stable]# ls -al /usr/lib | grep libevent

[root@pc libevent-2.0.21-stable]# cd ../

[root@pc src]# wget http://memcached.org/files/memcached-1.4.25.tar.gz

[root@pc src]# tar -xzf memcached-1.4.25.tar.gz

[root@pc src]# cd memcached-1.4.25

[root@pc memcached-1.4.25]# ./configure -with-libevent=/usr

[root@pc memcached-1.4.25]# make

[root@pc memcached-1.4.25]# autoreconf -ivf

[root@pc memcached-1.4.25]# make

[root@pc memcached-1.4.25]# make install

[root@pc memcached-1.4.25]# ls -al /usr/local/bin/mem\*

[root@pc memcached-1.4.25]# /usr/local/bin/memcached -d -m 512 -u root -l 127.0.0.1 -p 11211 -c 256 -P /data/logs/memcached.pid

[root@pc memcached-1.4.25]# service memcached restart

|  |  |
| --- | --- |
|  |  |

netstat -lp | grep memcached

ps -ef | grep memcached

/usr/local/php/bin/php -i |grep memcached

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

ps -ef | grep nginx

ps -ef | grep fpm

/usr/local/php-5.6.21/sbin/php-fpm

/usr/local/php-5.6.21/lib/php/extensions/no-debug-non-zts-20131226/

/usr/local/php-5.6.21/bin/phpize

/usr/local/bin/memcached -d -m 1024 -u root -l 127.0.0.1 -p 11211 -c 256 -P /data/logs/memcached.pid

/usr/local/nginx/sbin/nginx -s reload

**ps 正在运行的某个进程的状态**

ps –ef 查看所有进程

ps –ef | grep ssh 查找某一进程

**kill 杀掉某一进程**

kill 2868 杀掉2868编号的进程

kill -9 2868 强制杀死进程

enforcing=0

/usr/local/php-5.6.21/lib/php/extensions/no-debug-non-zts-20131226/

1. <!-- <value name="user">nobody</value> -->
2. Unix group of processes
3. <!-- <value name="group">nobody</value> -->

/usr/local/php-5.6.21/lib/php/extensions/no-debug-non-zts-20131226/

find / -name mysql

service mysql restart

sql\_mode=NO\_ENGINE\_SUBSTITUTION,STRICT\_TRANS\_TABLES

[root@iZ2zee0ov5tg7m6zqdumvyZ ~]# df -h

Filesystem Size Used Avail Use% Mounted on

/dev/vda1 40G 6.6G 31G 18% /

tmpfs 3.9G 0 3.9G 0% /dev/shm

[root@iZ2zee0ov5tg7m6zqdumvyZ ~]# free -m

total used free shared buffers cached

Mem: 8001 7877 124 0 142 6524

-/+ buffers/cache: 1210 6791

Swap: 0 0 0

[root@iZ2zee0ov5tg7m6zqdumvyZ ~]#

/usr/local/php-5.6.21/sbin/php-fpm -c /usr/local/php-5.6.21/etc/php.ini -y /usr/local/php-5.6.21/etc/php-fpm.conf