





# LNMP环境搭建

Auchor: bottle  
Email: bottle.friday@gmail.com

LINUX: CentOS 6.4 final  
NGINX: nginx-1.4.4  
MYSQL: mysql-5.5.16  
PHP: php-5.4.23

<http://frame.fridayws.com>

注:  红色字体表示标注;  
 绿色表示bash指令;  
 蓝色表示配置文件;  
 黑色为文档的描述文字;

安装所需包: `yum -y install gcc gcc-c++ glibc glibc-devel glib2 glib2-devel bzip2 bzip2-devel ncurses ncurses-devel patch zip gettext gettext-devel`

需要的源码包:

- 源码包下载地址: <http://download.csdn.net/detail/otypedef/6814845>
- bison-3.0.tar.gz
- expat-2.1.0.tar
- freetype-2.4.0.tar.gz
- jpegsrc.v9.tar.gz
- libxml2-2.9.0.tar.gz
- pcre-8.34.tar.bz2
- cmake-2.8.12.1.tar.gz
- fontconfig-2.10.93.tar.bz2
- gd-2.0.33.tar.gz
- libmcrypt-2.5.7.tar.gz
- mysql-5.5.16.tar.gz
- php-5.4.23.tar.gz
- curl-7.34.0.tar.gz
- fontconfig-2.4.2.tar.gz
- gzip-1.2.4.tar
- libpng-1.6.8.tar
- nginx-1.4.4.tar.gz
- zlib-1.2.8.tar.gz

安装过程:

(最好全程用root用户安装, 如果不放心可以用普通用户的sudo操作, 本示例都是以root权限执行指令);

注: 本示例软件(除去安装到默认位置的)大多安装在/workspace/local/目录下, 笔者的/workspace是另外做的一个分区。

指令中#开头的表示本文档的注释。

安装前准备:

创建一个通用的用户, 作为nginx, php, mysql等的默认用户。  
我们这边使用www用户, 下面创建www用户和组。

`groupadd www`  
`useradd -g www www`

我们的软件安装包放在 /workspace/lnmp/ 目录中。

安装pcre:

`tar -jxvf pcre-8.34.tar.bz2`  
`cd pcre-8.34`

```
./configure --prefix=/workspace/local/pcre --enable-utf8 --enable-unicode-properties
```

注: 这里的`--enable-utf8 --enable-unicode-properties` 是开启正则utf8支持  
`make && make install`

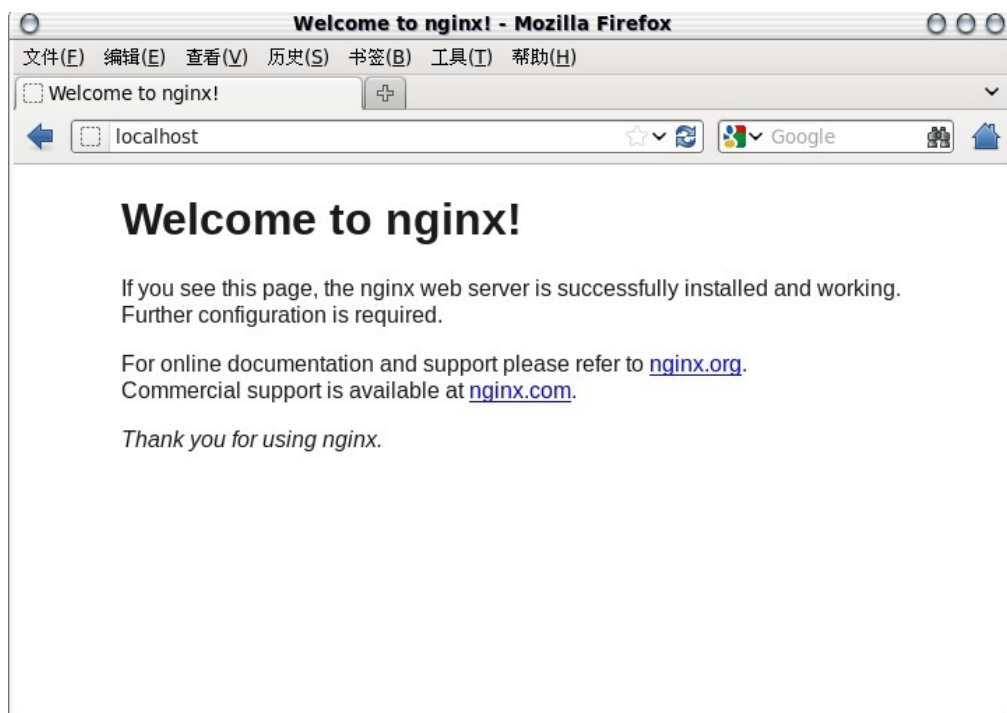
安装zlib:

```
tar -zxvf zlib-1.2.8.tar.gz
cd zlib-1.2.8
./configure --prefix=/workspace/local/zlib
make && make install
```

安装nginx:

```
tar -zxvf nginx-1.4.4.tar.gz
cd nginx-1.4.4
./configure --user=www --group=www --prefix=/workspace/local/nginx \
--with-http_stub_status_module --with-http_ssl_module \
--with-pcre=/workspace/lnmp/pcr-8.34 \
--with-zlib=/workspace/lnmp/zlib
```

注: 这里我们的pcre和zlib需要的是它们的源码包(并已经configure过)的位置. 并且因pcre需要开启utf8支持, 而默认安装是没有开启的. 我们在执行完configure之后, 需要编辑目录下的 Makefile文件, 找到: `./configure --disable-shared` 并在后面添加:  
`--enable-utf8 --enable-unicode-properties` 以支持utf8 .  
`make && make install`



到这里nginx已经安装完成. 我们可以做一个测试.

首先启动nginx服务, 执行以下指令:

```
/workspace/local/nginx/sbin/nginx
```

打开浏览器, 输入: <http://localhost> 回车, 然后如果我们看到如下图的页面, 则安装成功.

安装cmake(安装cmake是为了给安装mysql作准备):

```
tar -zxvf cmake-2.8.12.1.tar.gz
cd cmake-2.8.12.1
./bootstrap #这里的安装和之前的有点不同
gmake && gmake install
```

安装bison(它也是给安装mysql做准备):

```
tar -zxvf bison-3.0.tar.gz
cd bison-3.0
./configure
make && make install
```

安装Mysql:

我们的数据和日志目录放在/workspace/mysql下面,  
我们需要创建这些目录:

```
mkdir -p /workspace/mysql/data
mkdir -p /workspace/mysql/log
chown -R www:www /workspace/mysql/
然后
cd /workspace/lnmp
tar -zxvf mysql-5.5.16.tar.gz
cd mysql-5.5.16
```

```
cmake -DCMAKE_INSTALL_PREFIX=/workspace/local/mysql \
-DMYSQL_UNIX_ADDR=/tmp/mysql.sock \
-DDEFAULT_CHARSET=utf8 \
-DDEFAULT_COLLATION=utf8_general_ci \
-DWITH_EXTRA_CHARSETS:STRING=utf8,gbk \
-DWITH_MYISAM_STORAGE_ENGINE=1 \
-DWITH_INNOBASE_STORAGE_ENGINE=1 \
-DWITH_MEMORY_STORAGE_ENGINE=1 \
-DWITH_READLINE=1 \
-DENABLED_LOCAL_INFILE=1 \
-DMYSQL_DATADIR=/workspace/mysql/data \
-DMYSQL_USER=www
```

```
make && make install
```

处理配置文件和启动脚本:

```
cp /workspace/lnmp/mysql-5.5.16/support-files/my-large.cnf /etc/my.cnf
cp /workspace/lnmp/mysql-5.5.16/support-files/mysql.server /etc/init.d/mysqld
chmod +x /etc/init.d/mysqld
vim /etc/init.d/mysqld #编辑此文件
basedir=/workspace/local/mysql
datadir=/workspace/mysql/data
```

如果有需要可以修改/etc/my.cnf:

一般我们会把/etc/my.cnf中的user mysql 改成 user www 如果没有这一行则在:

```
[mysqld] #这个下面添加
user    = www
```

mysql初始化系统表:

```
/workspace/local/mysql/scripts/mysql_install_db \
--defaults-file=/etc/my.cnf \
--basedir=/workspace/local/mysql \
--datadir=/workspace/mysql/data \
--user=www
```

将mysql加入开机启动:

```
chkconfig --add mysqld
chkconfig --level 345 mysqld on
启动mysql:
service mysqld start # 正常启动.
```

我们可以在本机连接mysql:

```
mysql -h localhost -u root #直接进入. 修改密码等操作参见mysql相关内容. 这里不多
做介绍.
```

至此, **MYSQL**安装完毕, 后面需要安装**php**, 在安装**PHP**前需要很多的准备工作.

安装curl:

```
tar -zxvf curl-7.34.0.tar.gz
cd curl-7.34.0
./configure --prefix=/workspace/local/curl
make && make install
```

安装expat:

```
tar -xvf expat-2.1.0.tar
cd expat-2.1.0
./configure --prefix=/workspace/local/expat
make && make install
```

安装libxml2

```
tar -zxvf libxml2-2.9.0.tar.gz
cd libxml2-2.9.0
./configure --prefix=/workspace/local/libxml2
make && make install
```

安装jpeg9

```
tar -zxvf jpegsrc.v9.tar.gz
cd jpeg-9
./configure --prefix=/workspace/local/jpeg-9 --enable-shared --enable-static
make && make install
```

安装libpng

```
tar -xvf libpng-1.6.8.tar
```

```
cd libpng-1.6.8
./configure
make && make install
```

安装freetype:

```
tar -zxvf freetype-2.4.0.tar.gz
cd freetype-2.4.0
./configure --prefix=/workspace/local/freetype
make && make install
```

安装libmcrypt:

```
tar -zxvf libmcrypt-2.5.7.tar.gz
cd libmcrypt-2.5.7
./configure
make && make install
```

安装fontconfig:

```
tar -zxvf fontconfig-2.4.2.tar.gz
cd fontconfig-2.4.2
./configure --prefix=/workspace/local/fontconfig \
--with-freetype-config=/workspace/local/freetype/bin/freetype-config
make && make install
```

安装gd:

```
tar -zxvf gd-2.0.33.tar.gz
cd gd-2.0.33
./configure --prefix=/workspace/local/gd --with-jpeg=/workspace/local/jpeg-9 \
--with-png --with-zlib=/workspace/local/zlib \
--with-freetype=/workspace/local/freetype \
--with-fontconfig=/workspace/local/fontconfig
make && make install
```

安装PHP:

```
tar -zxvf php-5.4.23.tar.gz
cd php-5.4.23
./configure --prefix=/workspace/local/php \
--with-config-file-path=/workspace/local/php/etc \
--with-gd=/workspace/local/gd \
--with-mysql=/workspace/local/mysql/ \
--with-mysqli=/workspace/local/mysql/bin/mysqli_config \
--with-jpeg-dir=/workspace/local/jpeg-9/ \
--with-zlib-dir=/workspace/local/zlib/ \
--enable-mbstring=all \
--with-pdo-mysql \
--with-freetype-dir=/workspace/local/freetype \
--with-mcrypt \
--enable-sockets \
--enable-mbstring \
--enable-fpm
```

注: 因php-5.4 有一个bug, 需要编辑 /workspace/local/gd/include/gd\_io.h文件, 给结构体gdIOCtx 添加一个成员 void (\*data);

几个调试选项, 一般在开发扩展的时候需要用到:

—enable-debug \ //开启DEBUG

—enable-maintainer-zts \ //开启线程安全

—enable-embed //这个一般是用作php开发嵌入式的开发场景中的, 这个选项使得我们编译后得到一个与我们设定的SAPI相对应的结果.

make && make install

php基本配置:

cp php.ini-production /workspace/local/php/etc/php.ini

可以给php做一些基本配置:

vim /workspace/local/php/etc/php.ini

添加date.timezone = Asia/Shanghai

更改short\_open\_tag = Off 为short\_open\_tag = On (开启短格式支持)

更改expose\_php = on 为expose\_php = off (在curl中隐藏php版本号)

配置php-fpm

cp /workspace/local/php/etc/php-fpm.conf.default /workspace/local/php/etc/php-fpm.conf

vim /workspace/local/php/etc/php-fpm.conf

下面对php-fpm做一些修改:

pid = run/php-fpm.pid

pm.max\_child = 35

pm.start\_servers = 5

pm.min\_spare\_servers = 5

pm.max\_spare\_servers = 35

user = www

group = www

添加php-fpm服务

cp sapi/fpm/init.d.php-fpm /etc/init.d/php-fpm

chmod 0755 /etc/init.d/php-fpm

加入开机启动

chkconfig —add php-fpm

chkconfig —level 345 php-fpm on

启动php-fpm

service php-fpm start

至此, php相关所有服务已经配置完毕.

编写nginx配置文件

---

整合nginx和php:

我们的http根目录要放在/workspace/wwwroot下面, 那么我们要先创建这个目录:

mkdir -p /workspace/wwwroot

chown -R www:www /workspace/wwwroot

然后我们再去修改nginx的配置文件:

vim /workspace/local/nginx/conf/nginx.conf

```

user bottle bottle;
worker_processes 8;

pid logs/nginx.pid;

events {
    use epoll;
    worker_connections 1024;
}

http {
    include mime.types;
    default_type application/octet-stream;
    server_names_hash_bucket_size 128;
    client_header_buffer_size 32k;
    large_client_header_buffers 4 32k;
    client_max_body_size 8m;

    sendfile on;
    tcp_nopush on;
    server_tokens off;
    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;

    keepalive_timeout 65;
    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;

```

见下页

接上页

添加fastcgi.conf

`vim /workspace/local/nginx/conf/fastcgi.conf`

```
server {
    listen      80;
    server_name localhost;
    root        /workspace/wwwroot;
    index        index.html index.htm index.php;

    location    ~.*\.(php|php5)?$ {
        fastcgi_pass    127.0.0.1:9000;
        fastcgi_index    index.php;
        include          fastcgi.conf;
    }

    location    ~.*\.(gif|jpg|jpeg|png|bmp|swf)$
    {
        expires        30d;
    }

    location    ~.*\.(js|css)?$
    {
        expires        12h;
    }

    error_page  500 502 503 504 /50x.html;
    location = /50x.html {
        root    html;
    }
}
```

里面可做一些修改, 我们这里保持默认就可以.

测试配置文件:

`/workspace/local/nginx/sbin/nginx -t`

测试完成后启动nginx:

`/workspace/local/nginx/sbin/nginx`

nginx指令:

重启: `/workspace/local/nginx/sbin/nginx -s restart`

重载入配置: `/workspace/local/nginx/sbin/nginx -s reload`

关闭: `/workspace/local/nginx/sbin/nginx -s stop`

后面可以做一些php的测试, 比如在/workspace/wwwroot中新建一个php文件.

`touch phpinfo.php`



```
echo "<?php phpinfo(); ?>" > phpinfo.php
```

然后打开浏览器, 输入<http://localhost/phpinfo.php> 回车;  
看到如下页面, 表示安装成功.



好了, 到现在为止, 我们的LNMP环境就搭建好了. 如果大家有什么疑问, 上面有我的E-mail 可以直接联系我.