Nginx的代码是由一个核心和一系列的模块组成, 核心主要用于提供Web Server的基本功能，以及Web和Mail反向代理的功能；还用于启用网络协议，创建必要的运行时环境以及确保不同的模块之间平滑地进行交互。不过，大多跟协议相关的功能和某应用特有的功能都是由nginx的模块实现的。

在nginx内部，进程间的通信是通过模块的pipeline或chain实现的；换句话说，每一个功能或操作都由一个模块来实现。例如，压缩、通过FastCGI或uwsgi协议与upstream服务器通信，以及与memcached建立会话等。

一、安装Nginx：

1、解决依赖关系

编译安装nginx需要事先需要安装开发包组"Development Tools"和 "Development Libraries"。同时，还需要专门安装pcre-devel包：

# yum -y install pcre-devel

2、安装

首先添加用户nginx，实现以之运行nginx服务进程：

# groupadd -r nginx

# useradd -r -g nginx -s /bin/false -M nginx

接着开始编译和安装：

# ./configure \

--prefix=/usr \

--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/nginx.pid \

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

--user=nginx \

--group=nginx \

--with-http\_ssl\_module \

--with-http\_flv\_module \

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

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

--http-client-body-temp-path=/var/tmp/nginx/client/ \

--http-proxy-temp-path=/var/tmp/nginx/proxy/ \

--http-fastcgi-temp-path=/var/tmp/nginx/fcgi/ \

--http-uwsgi-temp-path=/var/tmp/nginx/uwsgi \

--http-scgi-temp-path=/var/tmp/nginx/scgi \

--with-pcre=/usr

# make && make install

说明：如果想使用nginx的perl模块，可以通过为configure脚本添加--with-http\_perl\_module选项来实现，但目前此模块仍处于实验性使用阶段，可能会在运行中出现意外，因此，其实现方式这里不再介绍。如果想使用基于nginx的cgi功能，也可以基于FCGI来实现，具体实现方法请参照网上的文档。

3、为nginx提供SysV init脚本:

新建文件/etc/rc.d/init.d/nginx，内容如下：

#!/bin/sh

#

# nginx - this script starts and stops the nginx daemon

#

# chkconfig: - 85 15

# description: Nginx is an HTTP(S) server, HTTP(S) reverse \

# proxy and IMAP/POP3 proxy server

# processname: nginx

# config: /etc/nginx/nginx.conf

# config: /etc/sysconfig/nginx

# pidfile: /var/run/nginx.pid

# Source function library.

. /etc/rc.d/init.d/functions

# Source networking configuration.

. /etc/sysconfig/network

# Check that networking is up.

[ "$NETWORKING" = "no" ] && exit 0

nginx="/usr/sbin/nginx"

prog=$(basename $nginx)

NGINX\_CONF\_FILE="/etc/nginx/nginx.conf"

[ -f /etc/sysconfig/nginx ] && . /etc/sysconfig/nginx

lockfile=/var/lock/subsys/nginx

make\_dirs() {

# make required directories

user=`nginx -V 2>&1 | grep "configure arguments:" | sed 's/[^\*]\*--user=\([^ ]\*\).\*/\1/g' -`

options=`$nginx -V 2>&1 | grep 'configure arguments:'`

for opt in $options; do

if [ `echo $opt | grep '.\*-temp-path'` ]; then

value=`echo $opt | cut -d "=" -f 2`

if [ ! -d "$value" ]; then

# echo "creating" $value

mkdir -p $value && chown -R $user $value

fi

fi

done

}

start() {

[ -x $nginx ] || exit 5

[ -f $NGINX\_CONF\_FILE ] || exit 6

make\_dirs

echo -n $"Starting $prog: "

daemon $nginx -c $NGINX\_CONF\_FILE

retval=$?

echo

[ $retval -eq 0 ] && touch $lockfile

return $retval

}

stop() {

echo -n $"Stopping $prog: "

killproc $prog -QUIT

retval=$?

echo

[ $retval -eq 0 ] && rm -f $lockfile

return $retval

}

restart() {

configtest || return $?

stop

sleep 1

start

}

reload() {

configtest || return $?

echo -n $"Reloading $prog: "

killproc $nginx -HUP

RETVAL=$?

echo

}

force\_reload() {

restart

}

configtest() {

$nginx -t -c $NGINX\_CONF\_FILE

}

rh\_status() {

status $prog

}

rh\_status\_q() {

rh\_status >/dev/null 2>&1

}

case "$1" in

start)

rh\_status\_q && exit 0

$1

;;

stop)

rh\_status\_q || exit 0

$1

;;

restart|configtest)

$1

;;

reload)

rh\_status\_q || exit 7

$1

;;

force-reload)

force\_reload

;;

status)

rh\_status

;;

condrestart|try-restart)

rh\_status\_q || exit 0

;;

\*)

echo $"Usage: $0 {start|stop|status|restart|condrestart|try-restart|reload|force-reload|configtest}"

exit 2

esac

而后为此脚本赋予执行权限：

# chmod +x /etc/rc.d/init.d/nginx

添加至服务管理列表，并让其开机自动启动：

# chkconfig --add nginx

# chkconfig nginx on

而后就可以启动服务并测试了：

# service nginx start

二、安装mysql-5.5.24

1、准备数据存放的文件系统

新建一个逻辑卷，并将其挂载至特定目录即可。这里不再给出过程。

这里假设其逻辑卷的挂载目录为/mydata，而后需要创建/mydata/data目录做为mysql数据的存放目录。

2、新建用户以安全方式运行进程：

# groupadd -r mysql

# useradd -g mysql -r -s /sbin/nologin -M -d /mydata/data mysql

# chown -R mysql:mysql /mydata/data

3、安装并初始化mysql-5.5.24

首先下载平台对应的mysql版本至本地，这里是32位平台，因此，选择的为mysql-5.5.24-linux2.6-i686.tar.gz，其下载位置为ftp://172.16.0.1/pub/Sources/mysql-5.5。

# tar xf mysql-5.5.24-linux2.6-i686.tar.gz -C /usr/local

# cd /usr/local/

# ln -sv mysql-5.5.24-linux2.6-i686 mysql

# cd mysql

# chown -R mysql:mysql .

# scripts/mysql\_install\_db --user=mysql --datadir=/mydata/data

# chown -R root .

4、为mysql提供主配置文件：

# cd /usr/local/mysql

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

并修改此文件中thread\_concurrency的值为你的CPU个数乘以2，比如这里使用如下行：

thread\_concurrency = 2

另外还需要添加如下行指定mysql数据文件的存放位置：

datadir = /mydata/data

5、为mysql提供sysv服务脚本：

# cd /usr/local/mysql

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

添加至服务列表：

# chkconfig --add mysqld

# chkconfig mysqld on

而后就可以启动服务测试使用了。

为了使用mysql的安装符合系统使用规范，并将其开发组件导出给系统使用，这里还需要进行如下步骤：

6、输出mysql的man手册至man命令的查找路径：

编辑/etc/man.config，添加如下行即可：

MANPATH /usr/local/mysql/man

7、输出mysql的头文件至系统头文件路径/usr/include：

这可以通过简单的创建链接实现：

# ln -sv /usr/local/mysql/include /usr/include/mysql

8、输出mysql的库文件给系统库查找路径：

# echo '/usr/local/mysql/lib' > /etc/ld.so.conf.d/mysql.conf

而后让系统重新载入系统库：

# ldconfig

9、修改PATH环境变量，让系统可以直接使用mysql的相关命令。具体实现过程这里不再给出。

三、编译安装php-5.4.4

1、解决依赖关系：

请配置好yum源（可以是本地系统光盘）后执行如下命令：

# yum -y groupinstall "X Software Development"

如果想让编译的php支持mcrypt、mhash扩展和libevent，此处还需要下载ftp://172.16.0.1/pub/Sources/ngnix目录中的如下几个rpm包并安装之：

libmcrypt-2.5.8-4.el5.centos.i386.rpm

libmcrypt-devel-2.5.8-4.el5.centos.i386.rpm

mhash-0.9.9-1.el5.centos.i386.rpm

mhash-devel-0.9.9-1.el5.centos.i386.rpm

libevent-2.0.17-2.i386.rpm

libevent-devel-2.0.17-2.i386.rpm

mcrypt-2.6.8-1.el5.i386.rpm

最好使用升级的方式安装上面的rpm包，命令格式如下：

# rpm -Uvh

2、编译安装php-5.4.4

首先下载源码包至本地目录，下载位置ftp://172.16.0.1/pub/Sources/new\_lamp。

# tar xf php-5.4.4.tar.bz2

# cd php-5.4.4

# ./configure --prefix=/usr/local/php4nginx --with-mysql=/usr/local/mysql --with-openssl --enable-fpm --enable-sockets --enable-sysvshm --with-mysqli=/usr/local/mysql/bin/mysql\_config --enable-mbstring --with-freetype-dir --with-jpeg-dir --with-png-dir --with-zlib-dir --with-libxml-dir=/usr --enable-xml --with-mhash --with-mcrypt --with-config-file-path=/etc --with-config-file-scan-dir=/etc/php.d --with-bz2 --with-curl

说明：如果前面第1步解决依赖关系时安装mcrypt相关的两个rpm包，此./configure命令还可以带上--with-mcrypt选项以让php支持mycrpt扩展。--with-snmp选项则用于实现php的SNMP扩展，但此功能要求提前安装net-snmp相关软件包。

# make

# make test

# make intall

为php提供配置文件：

# cp php.ini-production /etc/php.ini

为php-fpm提供Sysv init脚本，并将其添加至服务列表：

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

# chmod +x /etc/rc.d/init.d/php-fpm

# chkconfig --add php-fpm

# chkconfig php-fpm on

为php-fpm提供配置文件：

# cp /usr/local/php/etc/php-fpm.conf.default /usr/local/php/etc/php-fpm.conf

编辑php-fpm的配置文件：

# vim /usr/local/php/etc/php-fpm.conf

配置pm.的相关选项为你所需要的值，并启用pid文件（如下最后一行）：

pm.max\_children = 50

pm.start\_servers = 5

pm.min\_spare\_servers = 2

pm.max\_spare\_servers = 8

pid = /usr/local/php/var/run/php-fpm.pid

接下来就可以启动php-fpm了：

# service php-fpm start

使用如下命令来验正（如果此命令输出有中几个php-fpm进程就说明启动成功了）：

# ps aux | grep php-fpm

四、整合nginx和php5

1、编辑/etc/nginx/nginx.conf，启用如下选项：

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;

}

2、编辑/etc/nginx/fastcgi\_params，将其内容更改为如下内容：

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\_name;

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;

并在所支持的主页面格式中添加php格式的主页，类似如下：//注意改主机名

location / {

root html;

index index.php index.html index.htm;

}

而后重新载入nginx的配置文件：

# service nginx reload

3、在/usr/html新建index.php的测试页面，测试php是否能正常工作：

# cat > /usr/html/index.php << EOF

<?php

phpinfo();

?>

测试mysql连接

接着就可以通过浏览器访问此测试页面了。

五、安装xcache，为php加速：

1、安装

# tar xf xcache-2.0.0.tar.gz

# cd xcache-2.0.0

# /usr/local/php/bin/phpize

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

# make && make install

安装结束时，会出现类似如下行：

Installing shared extensions: /usr/local/php/lib/php/extensions/no-debug-zts-20100525/

2、编辑php.ini，整合php和xcache：

首先将xcache提供的样例配置导入php.ini

# mkdir /etc/php.d

# cp xcache.ini /etc/php.d

说明：xcache.ini文件在xcache的源码目录中。

接下来编辑/etc/php.d/xcache.ini，找到zend\_extension开头的行，修改为如下行：

zend\_extension = /usr/local/php/lib/php/extensions/no-debug-zts-20100525/xcache.so

注意：如果php.ini文件中有多条zend\_extension指令行，要确保此新增的行排在第一位。

3、重新启动php-fpm

# service php-fpm restart

六、补充说明

如果要在SSL中使用php，需要在php的location中添加此选项：

fastcgi\_param HTTPS on;