

---

## Nginx 网站服务

# 目录

目录.....	2
文档信息 .....	3
文档约定 .....	3
NGINX 简介 .....	3
部署 NGINX 软件.....	5
NGINX.CONF 文件结构 .....	8
状态统计模块.....	9
虚拟主机应用.....	10
LNMP 架构及应用部署 .....	12
1、安装 MySQL 数据库 .....	12
2、安装 PHP 解析环境 .....	16
3、配置 NGINX 支持 PHP 环境 .....	17
4、LNMP 平台中部署 WEB 应用.....	20

## 文档信息

文档作者	房佳亮
文档版本	Version1.0
文档版权	内部资料禁止传播
文档归类	Linux 运维架构师系列
系统环境	CentOS-7.X-x86_64
作者邮箱	crushlinux@163.com
修订信息	2021-01-26
技术交流	

## 文档约定

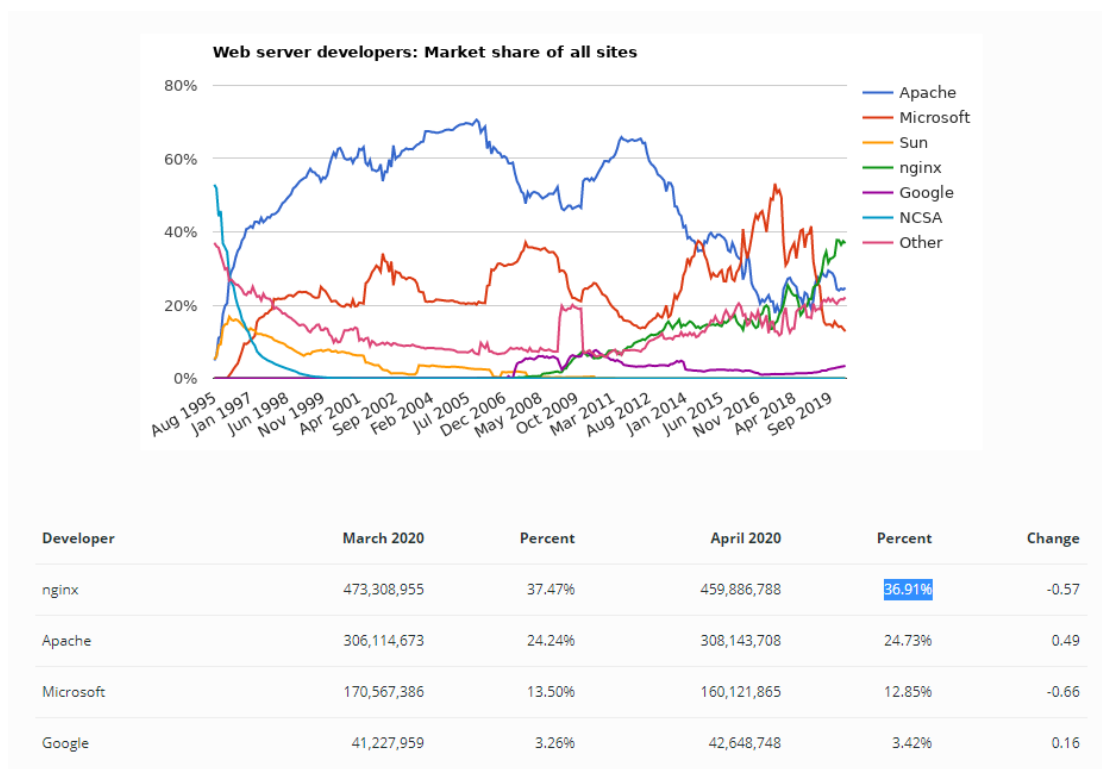
[绿色背景]	知识重点
[红色背景]	错误警告
[黄色背景]	注意事项

### 执行命令

## Nginx 简介

Nginx（发音 engine x）专为性能优化而开发的开源软件，是 HTTP 及较好的反向代理软件，由俄罗斯的作者 [Igor Sysoev](#) 开发，其最知名的优点是它的稳定性和低系统资源消耗（硬件资源占用较低），以及对 HTTP 并发连接的高处理能力（单台物理服务器可支持 30000~50000 个并发请求），是一个轻量级 WEB 服务器软件。正因为如此，大量提供社交网站、新闻资讯、电子商务以及虚拟主机等服务的企业纷纷选择 Nginx 来提供 WEB 服务。如新浪，淘宝（Tengine），京东，金山，网易，腾讯，百度文库，51cto，人人网等。

网站应用统计站点：<http://news.netcraft.com/>



Ngix 官方网站: <http://www.nginx.org> <http://www.nginx.com>

**nginx news**

2018-12-25 [nginx-1.15.8](#) mainline version has been released.

2018-12-25 [njs-0.2.7](#) version has been released, featuring rest parameters syntax and [more](#).

2018-12-20 [unit-1.7](#) version has been [released](#), featuring improved Node.js support with a number of bug fixes.

2018-12-04 [nginx-1.14.2](#) stable version has been released.

2018-11-27 [nginx-1.15.7](#) mainline version has been released.

2018-11-27 [njs-0.2.6](#) version has been released, featuring initial support for mutable prototypes and [more](#).

2018-11-15 [unit-1.6](#) version has been [released](#), featuring improved Node.js support.

2018-11-06 [nginx-1.14.1](#) stable and [nginx-1.15.6](#) mainline versions have been released, with fixes for [vulnerabilities in HTTP/2](#) (CVE-2018-16843, CVE-2018-16844) and [the MP4 module](#) (CVE-2018-16845).

2018-10-30 [njs-0.2.5](#) version has been released, featuring arguments object and [more](#).

2018-10-25 [unit-1.5](#) version has been [released](#), featuring preliminary Node.js support.

2018-10-02 [nginx-1.15.5](#) mainline version has been released.

**NGINX**

english  
[русский](#)

news  
[2017](#)  
[2016](#)  
[2015](#)  
[2014](#)  
[2013](#)  
[2012](#)  
[2011](#)  
[2010](#)  
[2009](#)

[about](#)  
[download](#)  
[security](#)  
[documentation](#)  
[faq](#)  
[books](#)  
[support](#)  
[trac](#)

淘宝 Tengine: <http://tengine.taobao.org/>



分析网站后台软件产品：

[http://toolbar.netcraft.com/site\\_report?url=undefined#last\\_reboot](http://toolbar.netcraft.com/site_report?url=undefined#last_reboot)

最新的稳定版：1.8.0

最新的开发板：1.19.2

## 部署 Nginx 软件

### 1) 安装支持软件：

Nginx 的配置及运行需要 pcre、zlib 等软件包的支持，因此应预先安装这些软件的开发包（devel），以便提供相应的库和头文件，确保 Nginx 的安装顺利完成。

```
[root@nginx ~]# systemctl stop firewalld
[root@nginx ~]# iptables -F
[root@nginx ~]# setenforce 0

[root@nginx ~]# yum -y install pcre-devel zlib-devel openssl-devel
```

### 2) 创建运行用户、组：

Nginx 服务程序默认以 nobody 身份运行，建议为其创建专门的用户账号，以便更准确地控制其访问权限，增加灵活性、降低安全风险。如：创建一个名为 nginx 的用户，不建立宿主目录，也禁止登录到 shell 环境。

```
[root@nginx ~]# useradd -M -s /sbin/nologin nginx
```

### 3) 编译安装 nginx：

释放 nginx 源码包

```
[root@nginx ~]# tar xf nginx-1.18.0.tar.gz -C /usr/src/
```

#### 4) 配置编译:

```
[root@nginx ~]# cd /usr/src/nginx-1.18.0/
[root@nginx nginx-1.18.0]# ./configure --prefix=/usr/local/nginx --user=nginx --group=nginx --
with-http_stub_status_module --with-http_ssl_module --with-http_flv_module --with-
http_gzip_static_module && make && make install
```

注: 配置前可以参考: ./configure --help 给出说明

- --prefix 设定 Nginx 的安装目录
- --user 和 --group 指定 Nginx 运行用户和组
- --with-http\_stub\_status\_module 启用 http\_stub\_status\_module 模块以支持状态统计
- --with-http\_ssl\_module 启用 SSL 模块
- --with-http\_flv\_module 启用 FLV 模块, 提供寻求内存使用基于时间的偏移量文件

为了使 Nginx 服务器的运行更加方便, 可以为主程序 nginx 创建链接文件, 以便管理员直接执行 nginx 命令就可以调用 Nginx 的主程序。

```
[root@nginx nginx-1.18.0]# ln -s /usr/local/nginx/sbin/nginx /usr/local/bin/
[root@nginx nginx-1.18.0]# ll /usr/local/bin/nginx
lrwxrwxrwx 1 root root 27 12-29 07:24 /usr/local/bin/nginx -> /usr/local/nginx/sbin/nginx
```

#### 5) Nginx 的运行控制:

与 Apache 的主程序 httpd 类似, Nginx 的主程序也提供了 "-t" 选项用来对配置文件进行检查, 以便找出不当或错误的配置。配置文件 nginx.conf 默认位于安装目录 /usr/local/nginx/conf/ 目录中。若要检查位于其他位置的配置文件, 可使用 "-c" 选项来指定路径。

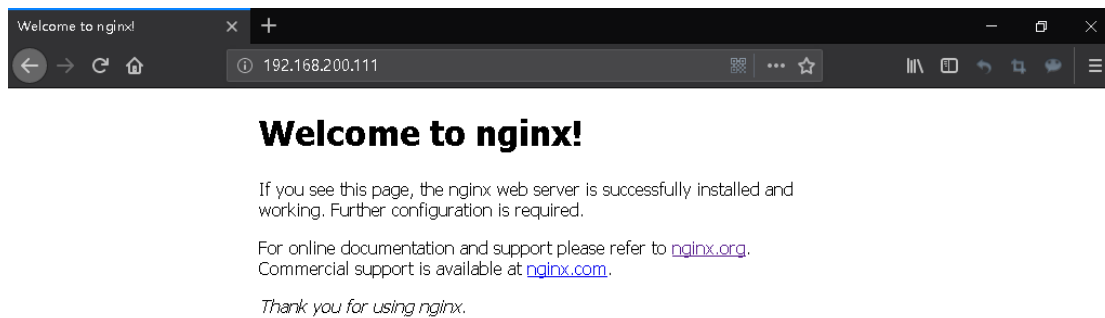
```
[root@nginx conf]# nginx -t
nginx: the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok
nginx: configuration file /usr/local/nginx/conf/nginx.conf test is successful
```

#### 6) 启动、停止 Nginx:

直接运行 nginx 即可启动 Nginx 服务器, 这种方式将使用默认的配置文​​件, 若要改用其他配置文件, 需添加 "-c 配置文件路径" 选项来指定路径。需要注意的是, 若服务器中已安装有 httpd 等其他 WEB 服务软件, 应采取措施 (修改端口, 停用或卸载其他软件) 避免冲突。

```
[root@nginx conf]# netstat -anpt | grep :80
[root@nginx conf]# nginx
[root@nginx conf]# netstat -anpt | grep :80
tcp        0      0 0.0.0.0:80          0.0.0.0:*          LISTEN
6810/nginx: master
```

通过检查 Nginx 程序的监听状态, 或者在浏览器中访问此 WEB 服务 (默认页面将显示 "Welcome to nginx!"), 可以确认 Nginx 服务是否正常运行。



主程序 Nginx 支持标准的进程信号，通过 kill 或者 killall 命令传送

- HUP 重载配置 等同于-1
- QUIT 退出进程 等同于-3
- KILL 杀死进程 等同于-9

```
[root@nginx ~]# killall -s HUP nginx
[root@nginx ~]# killall -s QUIT nginx
[root@nginx ~]# netstat -anpt |grep :80
```

当 Nginx 进程运行时，PID 号默认存放在 /usr/local/nginx/logs/ 目录下的 nginx.pid 文件中，因此若改用 kill 命令，也可以根据 nginx.pid 文件中的 PID 号来进行控制。

为了使 Nginx 服务的启动、停止、重载等操作更加方便，可以编写 Nginx 服务脚本，并使用 chkconfig 和 systemctl 工具来进行管理，也更加符合 RHEL 系统的管理习惯。

```
[root@nginx ~]# vim /etc/init.d/nginx
#!/bin/bash
# chkconfig: 2345 99 20
# description: Nginx Server Control Script
PROG="/usr/local/nginx/sbin/nginx"
PIDF="/usr/local/nginx/logs/nginx.pid"

case "$1" in
start)
    $PROG
    ;;
stop)
    kill -s QUIT $(cat $PIDF)
    ;;
restart)
    $0 stop
    $0 start
    ;;
reload)
    kill -s HUP $(cat $PIDF)
    ;;
*)
```

```
echo "Usage: $0 {start|stop|restart|reload}"
exit 1
esac
exit 0

[root@nginx ~]# chmod +x /etc/init.d/nginx
[root@nginx ~]# chkconfig --add nginx
[root@nginx ~]# chkconfig nginx on
[root@nginx ~]# chkconfig --list nginx
nginx          0:关闭  1:关闭  2:启用  3:启用  4:启用  5:启用  6:关闭
```

这样就可以通过 nginx 脚本来启动、停止、重启、重载 Nginx 服务器了。

## nginx.conf 文件结构

在 Nginx 服务器的主配置文件 nginx.conf 中，包括全局配置、I/O 事件配置、HTTP 配置这三大块内容，配置语句的格式为"**关键字 值**;"(末尾以分号表示结束)，以"#"开始的部分表示注释。

### 1) 全局配置

由各种配置语句组成，不使用特定的界定标记。全局配置部分包括运行用户、工作进程数、错误日志、PID 存放位置等基本设置。

常用配置项：

- user nginx [nginx]; //运行用户，Nginx 的运行用户实际是编译时指定的 nginx，若编译时未指定则默认为 nobody。
- worker\_processes 2; //指定 nginx 启动的工作进程数量，建议按照 cpu 数目来指定，一般和 CPU 核心数相等。
- worker\_cpu\_affinity 00000001 00000010; //为每个进程分配 cpu 核心，上例中将 2 个进程分配到两个 cpu，当然可以写多个，或者将一个进程分配到多个 cpu
- worker\_rlimit\_nofile 102400; //这个指令是指当一个 nginx 进程打开的最多文件数目，理论值应该是最多打开文件数（ulimit -n）与 nginx 进程数相除，但是 nginx 分配请求并不是那么均匀，所以最好与 ulimit -n 的值保持一致。(通过"ulimit -n 数值"可以修改打开的最多文件数目)
- error\_log logs/error.log; //全局错误日志文件的位置
- pid logs/nginx.pid; //PID 文件的位置

### 2) I/O 事件配置：

使用"events {"界定标记，用来指定 Nginx 进程的 I/O 响应模型，每个进程的连接数等设置 events {

use epoll; //使用 epoll 模型，对于 2.6 以上的内核，建议使用 epoll 模型以提高性能

worker\_connections 4096; //每个进程允许的最多连接数(默认为 1024)，每个进程的连接数应根据实际需要来定，一般在 10000 以下，理论上每台 nginx 服务器的最大连接数为 worker\_processes\*worker\_connections，具体还要看服务器的硬件、带宽等。



}

### 3) HTTP 配置

使用"http{}"界定标记，包括访问日志、HTTP 端口、网页目录、默认字符集、连接保持、以及虚拟主机、PHP 解析等一系列设置。其中大部分配置语句包含在子界定标记"server {}"内。

```
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;           //访问日志位
    sendfile     on;                //支持文件发送（下载）
    keepalive_timeout 65;           //连接保持超时

    server {                                //web 服务的监听配置
        listen    80;                  //监听地址及端口（IP: PORT）
        server_name www.crushlinux.com; //网站名称（FQDN）
        charset utf-8;                 //网页的默认字符集

        location / {                    //跟目录配置
            root    html;               //网站根目录的位置安装位置的 html 中
            index   index.html index.htm; //默认首页（索引页）
        }

        error_page 500 502 503 504 /50x.html; //内部错误的反馈页面
        location = /50x.html {              //错误页面配置
            root    html;
        }
    }
}
```

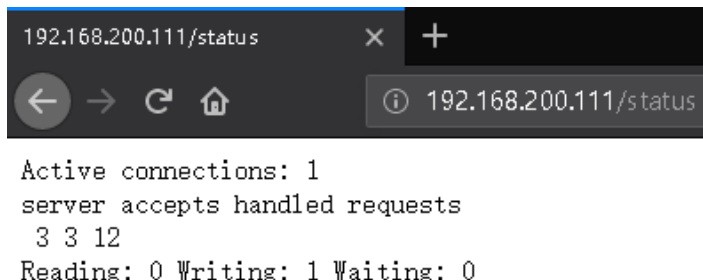
## 状态统计模块

Nginx 内置了 HTTP\_STUB\_STATUS 状态统计模块，用来反馈当前的 WEB 访问情况。配置编译参数时可添加 `--with-http_stub_status_module` 来启用此模块。要使用 Nginx 的状态统计功能，除了启用内建模块以外，还需要修改 nginx.conf 文件，指定访问位置并打开 stub\_status 配置。在 http{} 配置的 server{} 子配置内添加如下配置项

```
[root@nginx ~]# vim /usr/local/nginx/conf/nginx.conf
location /status {
    stub_status on;           //打开状态统计功能
```

```
access_log off;           //关闭此位置的日志记录
}
[root@nginx conf]# systemctl restart nginx
```

浏览器访问 <http://192.168.200.111/status>



```
Active connections: 1
server accepts handled requests
3 3 12
Reading: 0 Writing: 1 Waiting: 0
```

Active connections 表示当前活跃的连接数，

第三行的三个数字表示 Nginx 当前总共处理了 3 个连接，成功创建 3 次握手，总共处理了 12 个请求。

Reading 表示 Nginx 读取到客户端 Header 信息数，

Writing 表示 Nginx 返回给客户端的 Header 信息数

Waiting 表示 Nginx 已经处理完，正在等候下一次请求指令时的驻留连接数。

## 虚拟主机应用

虚拟主机分类：

- 基于域名：多个域名解析为一个 IP 地址，不同域名访问到不同网站内容。  
<http://www.a.com> <http://www.b.com>
- 基于 IP 地址：服务器拥有多个 IP 地址，不同 IP 访问到不同网站内容。  
<http://192.168.1.1> <http://192.168.1.2>
- 基于端口：相同 IP 地址的不同端口访问到不同网站内容。  
<http://192.168.1.1:80> <http://192.168.1.1:81>

使用 Nginx 搭建虚拟主机服务器时，每个虚拟 WEB 站点拥有独立的"server {}"配置段，各自监听的 IP 地址、端口号可以单独指定，当然网站名称也是不同的。

例如：要创建两个站点 [www.crushlinux.com](http://www.crushlinux.com) 和 [www.cloud.com](http://www.cloud.com) 为两个虚拟 WEB 主机分别建立根目录，并准备测试首页

```
[root@nginx ~]# mkdir /usr/local/nginx/html/crushlinux
[root@nginx ~]# mkdir /usr/local/nginx/html/cloud
[root@nginx ~]# echo "<h1>www.crushlinux.com</h1>" >
/usr/local/nginx/html/crushlinux/index.html
[root@nginx ~]# echo "<h1>www.cloud.com</h1>" > /usr/local/nginx/html/cloud/index.html
[root@nginx ~]# vim /usr/local/nginx/conf/nginx.conf
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;
keepalive_timeout 65;
server {
    listen 80;
    server_name www.crushlinux.com;
    charset utf-8;
    access_log logs/crushlinux.access.log main;

    location / {
        root html/crushlinux;
        index index.html index.htm;
    }
}

server {
    listen 80;
    server_name www.cloud.com;
    charset utf-8;
    access_log logs/cloud.access.log main;

    location / {
        root html/cloud;
        index index.html index.htm;
    }
}
}

[root@nginx ~]# systemctl restart nginx

[root@nginx ~]# vim /etc/hosts
192.168.200.111 www.crushlinux.com
192.168.200.111 www.cloud.com
```

虚拟主机访问测试

```
[root@nginx ~]# elinks --dump http://www.crushlinux.com
www.crushlinux.com

[root@nginx ~]# elinks --dump http://www.cloud.com
```

# LNMP 架构及应用部署

LNMP 也称为 LEMP，E 来自于 Nginx 的发音[engine x]

LNMP 就像构建 LAMP 平台一样，构建 LNMP 平台也需要用到 Linux 服务器，MySQL 数据库，PHP 解释环境等应用。P（PHP、Perl、Python）

wamp 是什么？xampp

## 1、安装 Mysql 数据库

1) 基于源码包安装 MySQL

检查是否有 rpm 格式的安装包

```
[root@nginx ~]# rpm -q mysql mysql-server mariadb mariadb-server
```

ncurses-devel 是 cmake 的依赖包

```
[root@nginx ~]# yum -y install ncurses-devel
[root@nginx ~]# rpm -q ncurses-devel
ncurses-devel-5.9-14.20130511.el7_4.x86_64
```

安装配置工具 cmake

```
[root@nginx ~]# tar xf cmake-3.13.1.tar.gz -C /usr/src/
[root@nginx ~]# cd /usr/src/cmake-3.13.1/
[root@nginx cmake-3.13.1]# ./configure && make && make install
```

建议采用 yum 安装方式

```
[root@nginx ~]# yum -y install cmake
[root@nginx ~]# rpm -q cmake
cmake-2.8.12.2-2.el7.x86_64
```

创建运行用户

```
[root@nginx ~]# useradd -M -s /sbin/nologin mysql
```

解包，配置，编译，安装

```
[root@nginx ~]# tar xf mysql-5.7.24.tar.gz -C /usr/src/
[root@nginx ~]# cd /usr/src/mysql-5.7.24/
[root@nginx mysql-5.7.24]# cmake -DCMAKE_INSTALL_PREFIX=/usr/local/mysql -
DDEFAULT_CHARSET=utf8 -DDEFAULT_COLLATION=utf8_general_ci -
DWITH_EXTRA_CHARSETS=all -DSYSCONFDIR=/etc && make && make install
```

- -DCMAKE\_INSTALL\_PREFIX=/usr/local/mysql //数据库程序安装目录

- -DDEFAULT\_CHARSET=utf8 //指定字符集编码
- -DDEFAULT\_COLLATION=utf8\_general\_ci //默认的字符集校对规则，utf8\_general\_ci 适用于 utf-8 字符集的通用规则
- -DWITH\_EXTRA\_CHARSETS=all //指定额外支持的字符集编码
- -DSYSCONFDIR=/etc //指定配置文件存放目录

### 报错处理：

CMake Error at cmake/boost.cmake:81 (MESSAGE):

You can download it with -DDOWNLOAD\_BOOST=1 -DWITH\_BOOST=<directory>

This CMake script will look for boost in <directory>. If it is not there, it will download and unpack it (in that directory) for you.

If you are inside a firewall, you may need to use an http proxy:

```
export http_proxy=http://example.com:80
```

Call Stack (most recent call first):

cmake/boost.cmake:238 (COULD\_NOT\_FIND\_BOOST)

CMakeLists.txt:507 (INCLUDE)

-- Configuring incomplete, errors occurred!

See also "/usr/src/mysql-5.7.24/CMakeFiles/CMakeOutput.log".

See also "/usr/src/mysql-5.7.24/CMakeFiles/CMakeError.log".

### 解决办法：

a. 在/usr/local 下创建一个名为 boost 的文件夹

```
[root@nginx ~]# mkdir /usr/local/boost
```

b. 进入目录并下载 boost

```
[root@nginx ~]# cd /usr/local/boost
```

```
[root@nginx boost]# wget
```

```
https://sourceforge.net/projects/boost/files/boost/1.59.0/boost_1_59_0.tar.gz
```

c. 解压 boost

```
[root@nginx boost]# tar xf boost_1_59_0.tar.gz
```

d. 继续 cmake，添加上红色部分

```
[root@nginx ~]# cd /usr/src/mysql-5.7.24/
```

```
[root@nginx mysql-5.7.24]# cmake -DCMAKE_INSTALL_PREFIX=/usr/local/mysql -  
DDEFAULT_CHARSET=utf8 -DDEFAULT_COLLATION=utf8_general_ci -
```

```
DWITH_EXTRA_CHARSETS=all -DSYSCONFDIR=/etc -DWITH_BOOST=/usr/local/boost && make
&& make install
```

## 2) 安装后的调整

对数据库目录进行权限设置

```
[root@nginx ~]# cd /usr/local/mysql/
[root@nginx mysql]# chown -R mysql:mysql ./
```

建立配置文件(CentOS7 系统默认支持 MariaDB 数据库，系统默认的/etc/my.cnf 配置文件是 MariaDB 的配置文件 )

```
[root@nginx mysql]# vim /etc/my.cnf
[mysqld]
datadir=/usr/local/mysql/data
socket=/tmp/mysql.sock

[mysqld_safe]
log-error=/usr/local/mysql/data/mysql.log
pid-file=/usr/local/mysql/data/mysql.pid
```

## 3) 初始化数据库

```
[root@nginx mysql]# ./bin/mysql --user=mysql --basedir=/usr/local/mysql --
datadir=/usr/local/mysql/data --initialize
2018-12-08T01:51:39.798903Z 1 [Note] A temporary password is generated for root@nginx:
Tvc:Rm1ZlxtG
```

- --basedir=/usr/local/mysql/ //指定安装目录（产品目录）
- --datadir=/usr/local/mysql/data //指定数据目录
- --user=mysql //指定用户身份

## 4) 设置环境变量

```
[root@nginx mysql-5.7.24]# echo "PATH=$PATH:/usr/local/mysql/bin" >> /etc/profile
[root@nginx mysql-5.7.24]# . /etc/profile = source /etc/profile
```

## 5) 添加系统服务

添加 MySQL 为系统服务，以便通过 systemctl 命令进行管理

方法一：不推荐使用

```
[root@nginx mysql-5.7.24]# cp support-files/mysql.server /usr/local/mysql/bin/mysqld.sh
[root@nginx mysql-5.7.24]# chmod +x /usr/local/mysql/bin/mysqld.sh
[root@nginx ~]# vim /usr/lib/systemd/system/mysqld.service
[Unit]
Description=MySQL Server
After=network.target

[Service]
```

```
User=mysql #指定程序运行的用户账户
Group=mysql #指定程序运行的组账户

Type=forking
PIDFile=/usr/local/mysql/data/localhost.pid #指定 PID 文件的位置，默认为“主机名.pid”
ExecStart=/usr/local/mysql/bin/mysqld.sh start
ExecStop=/usr/local/mysql/bin/mysqld.sh stop

[Install]
WantedBy=mutli-user.target

[root@nginx ~]# systemctl start mysqld
[root@nginx ~]# systemctl enable mysqld
Created symlink from /etc/systemd/system/multi-user.target.wants/mysqld.service to
/usr/lib/systemd/system/mysqld.service.

[root@nginx ~]# systemctl status mysqld
● mysqld.service - MySQL Server
Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
Active: active (running) since 六 2018-12-08 09:54:04 CST; 42s ago
Main PID: 2520 (mysqld)
CGroup: /system.slice/mysqld.service
└─2364 /bin/sh /usr/local/mysql/bin/mysqld_safe --datadir=/usr/local/mysql/data --pid...
└─2520 /usr/local/mysql/bin/mysqld --basedir=/usr/local/mysql --datadir=/usr/local/my...

12 月 08 09:53:57 localhost systemd[1]: Starting MySQL Server...
12 月 08 09:54:04 localhost systemd[1]: mysqld.service: Supervising process 2520 which is
not...ts.
12 月 08 09:54:04 localhost systemd[1]: Started MySQL Server.
12 月 08 09:54:16 localhost systemd[1]: mysqld.service: Supervising process 2520 which is
not...ts.
Hint: Some lines were ellipsized, use -l to show in full.

[root@nginx ~]# netstat -lnpt | grep mysql
tcp6      0      0 :::3306          :::*              LISTEN
23892/mysqld
```

方法二：

```
[root@localhost mysql-5.7.24]# cp support-files/mysql.server /etc/init.d/mysqld
[root@localhost mysql-5.7.24]# chmod +x /etc/init.d/mysqld
[root@localhost mysql-5.7.24]# /etc/init.d/mysqld start
Starting MySQL.Logging to '/usr/local/mysql/data/mysql.log'.
SUCCESS!
```

后期修改数据库用户的密码：

```
[root@nginx ~]# mysqladmin -u root -p'TvC:Rm1ZlxtG' password '123456'
[root@nginx ~]# mysql -uroot -p123456
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.7.24 Source distribution

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql>
```

## 2、安装 PHP 解析环境

较新版本（如 5.6）的 PHP 已经自带 FPM(fastCGI process manager,FastCGI 进程管理器)模块，用来对 PHP 解析实例进行管理，优化解析效率，因此在配置 PHP 编译选项时应添加"--enable-fpm"以启用此模块。

### 1) 编译安装 php

```
[root@nginx ~]# yum -y install gd libxml2-devel libjpeg-devel libpng-devel
[root@nginx ~]# tar xf php-5.6.39.tar.gz -C /usr/src/
[root@nginx ~]# cd /usr/src/php-5.6.39/
[root@nginx php-5.6.39]# ./configure --prefix=/usr/local/php5 --with-gd --with-zlib --with-mysql=/usr/local/mysql --with-mysqli=/usr/local/mysql/bin/mysql_config --with-config-file-path=/usr/local/php5 --enable-mbstring --enable-fpm --with-jpeg-dir=/usr/lib && make && make install
```

### 2) 安装后的调整

```
[root@nginx php-5.6.39]# cp php.ini-production /usr/local/php5/php.ini
[root@nginx php-5.6.39]# ln -s /usr/local/php5/bin/* /usr/local/bin/
[root@nginx php-5.6.39]# ln -s /usr/local/php5/sbin/* /usr/local/sbin/
```

3) 为了提高 PHP 解析效率，建议将相应版本的 ZendGuardLander 也装上。  
安装 ZendGuardLander(注意：若是 64 位系统，该软件得到其官网下载 64 位的相应软件包，若用 32 位的就会报错。下载地址：<http://www.zend.com/en/products/guard/downloads>

```
[root@nginx ~]# tar xf zend-loader-php5.6-linux-x86_64_update1.tar.gz -C /usr/src/
[root@nginx ~]# cd /usr/src/zend-loader-php5.6-linux-x86_64/
```



```
[root@nginx zend-loader-php5.6-linux-x86_64]# cp ZendGuardLoader.so
/usr/local/php5/lib/php/

[root@nginx zend-loader-php5.6-linux-x86_64]# vim /usr/local/php5/php.ini
zend_extension=/usr/local/php5/lib/php/ZendGuardLoader.so
zend_loader.enable=1
```

### 3、配置 Nginx 支持 PHP 环境

若要让 Nginx 能够解析 PHP 网页，有两种方法可以选择；其一：Nginx 充当中介，将访问 PHP 页面的 WEB 请求转交给其他服务器（LAMP）去处理；其二：通过使用 PHP 的 FPM 模块来调用本机的 PHP 环境。

如果选用 FPM 方式，则需要先启动 php-fpm 进程，以便监听 PHP 解析请求。参考范例建立 php-fpm.conf 配置文件，并修改其中的 PID 文件、运行用户、服务数（进程数量）等相关设置，然后启动 php-fpm 程序即可（默认监听本机的 9000 端口）

```
[root@nginx ~]# cd /usr/local/php5/etc/
[root@nginx etc]# cp php-fpm.conf.default php-fpm.conf
[root@nginx etc]# useradd -M -s /sbin/nologin php
[root@nginx etc]# vim php-fpm.conf
25 pid = run/php-fpm.pid          //确认 pid 文件位置
149 user = php                    //运行用户
150 group = php                   //运行组
246 pm.start_servers = 20         //启动时开启的进程数
251 pm.min_spare_servers = 5     //最少空闲进程数
256 pm.max_spare_servers = 35    //最大空闲进程数
241 pm.max_children = 50         //最多空闲进程数

[root@nginx etc]# /usr/local/sbin/php-fpm
[root@nginx etc]# netstat -anpt |grep php-fpm
tcp        0      0 127.0.0.1:9000          0.0.0.0:*               LISTEN
23027/php-fpm.conf)
```

在 php-fpm.conf 文件中，pid 配置行指出了 PID 信息的存放位置，对应的实际路径为 /usr/local/php5/var/run/php-fpm.pid，根据上述信息，可以修改 Nginx 服务脚本，以便在启动/停止 Nginx 服务器时将 php-fpm 进程也自动启动/停止。

```
[root@nginx etc]# vim /etc/init.d/nginx
#!/bin/bash
# chkconfig: 2345 99 20
# description: Nginx Server Control Script
PROG="/usr/local/nginx/sbin/nginx"
PIDF="/usr/local/nginx/logs/nginx.pid"
PROG_FPM="/usr/local/sbin/php-fpm"
```

```
PIDF_FPM="/usr/local/php5/var/run/php-fpm.pid"
case "$1" in
start)
    $PROG
    $PROG_FPM
;;
stop)
    kill -s QUIT $(cat $PIDF)
    kill -s QUIT $(cat $PIDF_FPM)
;;
restart)
    $0 stop
    $0 start
;;
reload)
    kill -s HUP $(cat $PIDF)
;;
*)
    echo "Usage: $0 (start|stop|restart|reload)"
    exit 1
esac
exit 0
```

这样，一旦启动或关闭 nginx 服务，php-fpm 程序也会随之启动或关闭，不需要额外再启动或关闭 php-fpm。

配置 Nginx 支持 PHP 解析：

无论是将 PHP 页面交给 LAMP 服务器去解析，还是调用本机的 php-fpm 进程进行解析，都需要在"server{"配置段中添加 location 设置，以便指定当访问.php 页面时采取何种操作。对于第一种方法使用 Nginx 的反向代理功能（转交给其他 WEB 服务器处理），使用的配置语句如下所示，例如，交给 IP 地址为 192.168.200.112 的 LAMP 服务器处理，从而实现由 Nginx 负责静态页面，LAMP 负责动态页面的分离效果。

```
[root@nginx etc]# vim /usr/local/nginx/conf/nginx.conf
server {
.....
    location ~ \.php$ {                //访问.php 页面的配置段
        proxy_pass http://192.168.200.112:80;    //LAMP 服务器的监听地址
    }
}
```

对于第二种方法（调用本机的 php-fpm 进程），使用的配置语句如下所示。在 conf/目录下的 fastcgi.conf 文件中已经包含必需的宏设置，可通过 include 语句添加进来。

```
[root@nginx etc]# vim /usr/local/nginx/conf/nginx.conf
server {
..... //省略部分信息
```

```

        location / {
            root    html;
            index   index.php index.html index.htm;
        }
        location ~ \.php$ {
            root html;                //访问 php 页面的配置段
            fastcgi_pass 127.0.0.1:9000; //PHP 网页文档根目录
            fastcgi_index index.php;    //php-fpm 的监听地址
            include fastcgi.conf;      //PHP 首页文件
            //包括 fastcgi.conf 样本配置
        }
    }
}

```

本文中我选择的是第二种方法，完成修改后重新加载 nginx 服务即可生效。

```
[root@nginx etc]# killall -HUP nginx
```

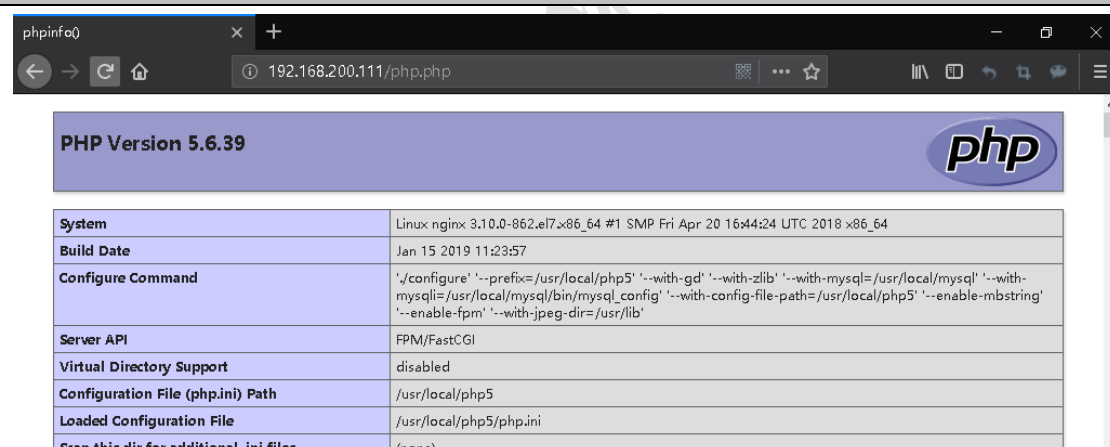
在 PHP 文档根目录下创建一个测试网页，以便测试 PHP 语言能否正常解析，以及能否访问 MySQL 数据库。测试页内容如下：

```
[root@nginx ~]# cat /usr/local/nginx/html/php.php
```

```

<?php
phpinfo();
?>

```



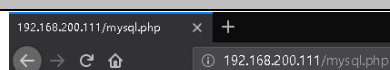
System	Linux nginx 3.10.0-862.el7.x86_64 #1 SMP Fri Apr 20 16:44:24 UTC 2018 x86_64
Build Date	Jan 15 2019 11:29:57
Configure Command	./configure '--prefix=/usr/local/php5' '--with-gd' '--with-zlib' '--with-mysql=/usr/local/mysql' '--with-mysqli=/usr/local/mysql/bin/mysqli_config' '--with-config-file-path=/usr/local/php5' '--enable-mbstring' '--enable-fpm' '--with-jpeg-dir=/usr/lib'
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/php5
Loaded Configuration File	/usr/local/php5/php.ini
Scan this dir for additional .ini files	(none)

```
[root@nginx etc]# vim /usr/local/nginx/html/mysql.php
```

```

<?php
$link=mysqli_connect('localhost','root','123456'); //连接 mysql 数据库
if($link) echo "<h1>恭喜你，大功告成!! </h1>"; //连接成功则返回信息
mysqli_close(); //关闭数据库连接
?>

```



恭喜你，大功告成！！

## 4、LNMP 平台中部署 WEB 应用

LNMP 平台与 LAMP 平台是非常相似的，区别主要在于所用 WEB 服务软件的不同，而这与使用 PHP 开发的 WEB 应用程序并无太大关系，因此 PHP 应用的部署方法也是类似的。

部署 ComsenzDiscuz BBS 论坛系统

Discuz! 社区论坛是一个采用 PHP 与 MySQL 等多种数据库构建的性能优异，功能全面的且安全稳定的社区论坛（BBS）软件，其官方网站为 <http://www.discuz.net>

```
[root@nginx ~]# unzip ComsenzDiscuz-DiscuzX-master.zip
[root@nginx ~]# cd DiscuzX/
[root@nginx DiscuzX]# ls upload/
admin.php api.php config crossdomain.xml favicon.ico group.php index.php m misc.php
portal.php search.php static uc_clientapi archiver connect.php data forum.php home.php
install member.php plugin.php robots.txt source template uc_server
```

上传 bbs 代码：

```
[root@nginx DiscuzX]# mv upload/ /usr/local/nginx/html/bbs
```

浏览器访问 <http://192.168.200.111/bbs/install/index.php>



设置权限及模板文件

```
[root@nginx ~]# cd /usr/local/nginx/html/bbs/config/
[root@nginx config]# cp config_global_default.php config_global.php
[root@nginx config]# cp config_ucenter_default.php config_ucenter.php

[root@nginx ~]# cd /usr/local/nginx/html/bbs
[root@nginx bbs]# chmod -R 777 config/ data/ uc_client/ uc_server/
```

Discuz!

安装向导

Discuz!X3.4 简体中文 UTF8 版 20180101

1.

开始安装

环境以及文件目录权限检查

检查安装环境

设置运行环境

创建数据库

安装

环境检查

项目	Discuz! 所需配置	Discuz! 最佳	当前服务器
操作系统	不限制	类Unix	✔ Linux
PHP 版本	5.3	7.1	✔ 7.3.0
附件上传	不限制	2M	✔ 2M
GD 库	1.0	2.0	✔ bundled (2.1.0 compatible)
cURL 库	不限制	开启	✔ 开启 7.29.0
OPcache	不限制	开启	✔ 关闭
磁盘空间	30MB	不限制	✔ 36GB

目录、文件权限检查

目录文件	所需状态	当前状态
./config/config_global.php	✔ 可写	✔ 可写
./config/config_ucenter.php	✔ 可写	✔ 可写
./config	✔ 可写	✔ 可写
./data	✔ 可写	✔ 可写
./data/cache	✔ 可写	✔ 可写
./data/avatar	✔ 可写	✔ 可写
./data/plugindata	✔ 可写	✔ 可写
./data/download	✔ 可写	✔ 可写
./data/addonmd5	✔ 可写	✔ 可写
./data/template	✔ 可写	✔ 可写
./data/threadcache	✔ 可写	✔ 可写
./data/attachment	✔ 可写	✔ 可写
./data/attachment/album	✔ 可写	✔ 可写
./data/attachment/forum	✔ 可写	✔ 可写
./data/attachment/group	✔ 可写	✔ 可写



准备数据库并配置相关授权

```
[root@nginx ~]# mysql -uroot -p123456
mysql> create database bbs;
Query OK, 1 row affected (0.00 sec)

mysql> grant all on bbs.* to 'bbs'@'localhost' identified by '123456';
Query OK, 0 rows affected, 1 warning (0.07 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

Discuz!

安装向导

Discuz!X3.4 简体中文 UTF8 版 20180101

3.

安装数据库

正在执行数据库安装

检查安装环境

设置运行环境

创建数据库

安装

填写数据库信息

数据库服务器:

localhost

数据库服务器地址, 一般为 localhost

数据库名:

bbs

数据库用户名:

bbs

数据库密码:

bbs123456

数据表前缀:

pre\_

同一数据库运行多个论坛时, 请修改前缀

系统信箱 Email:

admin@admin.com

用于发送程序错误报告

填写管理员信息

管理员账号:

admin

管理员密码:

.....

管理员密码不能为空

重复密码:

.....

管理员 Email:

admin@admin.com

下一步

©2001 - 2017 Comsenz Inc.



# 安装向导

Discuz!X3.4 简体中文 UTF8 版 20180101

## Discuz! 应用中心

应用中心特意为您准备了一批优秀应用，插件、模板应有尽有，无限制扩充站点功能，建站必备。  
快来应用中心装个应用吧！





专注微信应用

拼团 砍价 交友

助力 抽奖 投票

维清双12感恩回馈

全场应用 **68折**

文章自动采集发布

西瓜手机应用

50多款 任你挑



**SEO百度熊掌号推送**

安装: 228 ★★★★★



**顶象无感验证码**

安装: 5,867 ★★★★★



**手机注册V手机登录**

安装: 3,617 ★★★★★



**Discuz! 短信通**

安装: 2.3万 ★★★★★



**【同盾】论坛防灌水**

安装: 1.7万 ★★★★★

您的论坛已完成安装，[点此访问](#)

©2001 - 2017 Comsenz Inc.



社区动力

论坛

输入搜索内容

帖子 热搜: 活动 交友 discuz

论坛

今日: 0 | 昨日: 0 | 帖子: 0 | 会员: 1 | 欢迎新会员: admin

Discuz!

默认皮肤

0 / 0 从串

在线会员: 0 人在线 · 0 会员(0 隐身), 0 位游客 · 最高记录是 0 于 2018-12-0

管理员 超等版主 版主 会员

当前只有管理员和版主会员在线

官方论坛

提供最新 Discuz! 产品新闻、软件下载与技术交流

Comsenz 常用主机

Powered by Discuz! X3.4

© 2001-2017 Comsenz Inc.

Archiver (手机版) 小黑屋 Comsenz Inc.

GMT+8, 2018-12-0 16:49, Processed in 0.090001 seconds(s), 12 queries.

```
[root@nginx ~]# vim /usr/local/nginx/conf/nginx.conf
user  nginx nginx;
worker_processes  2;
error_log  logs/error.log;
pid        logs/nginx.pid;
```

版权所有© CRUSHLINUX

24 / 26



```
worker_rlimit_nofile 102400;

events {
    use epoll;
    worker_connections 4096;
}

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;
    keepalive_timeout 65;

    server {
        listen 80;
        server_name localhost;
        charset utf-8;

        location / {
            root html/bbs;
            index index.php index.html index.htm;
        }

        location ~ \.php$ {
            root html/bbs;
            fastcgi_pass 127.0.0.1:9000;
            fastcgi_index index.php;
            include fastcgi.conf;
        }

        location /status {
            stub_status on;
            access_log off;
        }

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

```
}  
}  
}  
[root@nginx etc]# killall -HUP nginx
```

浏览器访问 <http://192.168.200.111>

