# Lnmp安装笔记

准备工作

Linux系统(这里使用的VM虚拟机)

Nginx源码包

MySQL源码包

PHP源码包

这里使用源码包安装，而不是yum安装

开始操作(准备)

登录Linux，ping命令检查网络状态，用Xshll远程连接到Linux，方便操作

将下载好的源码压缩包使用Xftp传输到目录/usr/local/src下

cd /usr/local/src 第二条命令：yum -y install gcc\*

安装gcc编译器 (个人理解：nginx属于C语言开发的软件，编译过程需要C编译器，GUN开发的GCC属于编译器套件)

开始安装

nginx和PHP篇

提前安装部分依赖 命令如下

yum -y install gcc gcc-c++ make libtool zlib zlib-devel openssl openssl-devel pcre pcre-devel libcurl-devel libjpeg-devel libpng-devel freetype-devel php-mcrypt  libmcrypt  libmcrypt-devel libxml2-devel  php-mcrypt  libmcrypt  libmcrypt-devel  mysql-devel

在目录/usr/local/src下，把nginx-1.17.0.tar.gz的压缩包解压缩到当前目录

tar zxvf nginx-1.17.0.tar.gz

cd命令切换到nginx目录下，添加一个用户

useradd www

不需要设置密码

开始配置nginx，需要确定已经在nginx目录下

./configure --prefix=/usr/local/nginx --user=www --group=www --prefix=/usr/local/nginx --with-http\_stub\_status\_module --with-http\_ssl\_module --with-http\_realip\_module --http-log-path=/usr/local/nginx/logs/access.log --error-log-path=/usr/local/nginx/logserror.log

因为已经提前安装了依赖，所以不会报错，可以使用make和make install

make 编译需要时间执行 make install

当make install结束后，nginx的安装就到此结束

现在做一些别的设置

systemctl status firewalld

查看一下防火墙设置，当然啦，如果之前没有关闭过，现在的结果肯定是运行中

你会看到action (running)

systemctl stop firewalld    该命令可以临时关闭一下防火墙systemctl disable firewalld    该命令可以永久关闭防火墙

为了日后使用方便，防火墙应该永久关闭，但是记得应该先执行stop停止防火墙的工作状态，在永久关闭

这个时候可能依旧无法访问，修改selinux

vim /etc/selinux/config将里面改成SELINUX=disabled保存退出

访问方式，网址输入Linux的ip地址

这时候，nginx就可以通过浏览器访问了

当你看到nginx的欢迎页面，就是成功访问了，也就是nginx已经搭建成功

现在开始安装PHP

cd /usr/local/srctar zxvf php-7.3.6.tar.gz

切换回到src目录

其实是如果还在src目录下的nginx目录

cd ../

这样的方式更快，前提是对目录结构足够了解

解压完成后，切换到php目录，开始配置php

./configure --prefix=/usr/local/php --with-config-file-path=/usr/local/php/etc --enable-fpm --with-fpm-user=php-fpm --with-fpm-group=php-fpm --with-mysql=mysqlnd --with-mysql-sock=/tmp/mysql.sock --with-libxml-dir --with-gd --with-jpeg-dir --with-png-dir --with-freetype-dir --with-iconv-dir --with-zlib-dir --with-mcrypt --enable-soap --enable-gd-native-ttf --enable-ftp --enable-mbstring --enable-exif --disable-ipv6 --with-pear --with-curl --with-openssl --enable-bcmath

由于前面已经安装了依赖，所以不会报错，如果有报警告，警告可能是某些变量或者配置不适用于当前版本，也许是当前版本废弃的功能，可以不理会

开始make和make install就可以了

cp php.ini-production /usr/local/php/etc/php.ini复制配置文件cp /usr/local/php/etc/php-fpm.conf.default /usr/local/php/etc/php-fpm.conf复制启动文件配cp /usr/local/php/etc/php-fpm.d/www.conf.default /usr/local/php/etc/php-fpm.d/www.conf复制用户配置文件

当三个文件复制好了后，开始修改配置文件

vim /usr/local/php/etc/php-fpm.d/www.confuser=php-fpm修改为user=wwwgroup=php-fpm修改为group=www

修改后保存退出

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

检查PHP启动，看到success就是成功了

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

启动PHP

但是浏览器还不能访问到php

访问方式，网址栏输入ip地址/index.php

开始修改nginx的配置文件，让nginx和PHP连接起来

cp /usr/local/nginx/conf/nginx.conf /usr/local/nginx/conf/nginx.conf.bak

规范操作，备份一个nginx配置文件，以免当前修改发生不可逆转的错误

vim /usr/local/nginx/conf/nginx.conf

打开配置文件

#location ~ \.php$ { #root html; #fastcgi\_pass 127.0.0.1:9000; #fastcgi\_index index.php; #fastcgi\_param SCRIPT\_FILENAME /www/$fastcgi\_script\_name; #include fastcgi\_params;        #}

找到上面的注释内容，修改为

location ~ \.php$ { #root html; fastcgi\_pass 127.0.0.1:9000; fastcgi\_index index.php; #fastcgi\_param SCRIPT\_FILENAME /www/$fastcgi\_script\_name;            include        fastcgi.conf; }

然后还需要修改一个地方

location / {           root html;           index index.html  index.xml; }

这个地方原内容如上，不指定网站根目录内容，所以修改成

location / { root html;           index index.html index.xml index.php; }

保存退出后，重启nginx

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

在html目录下编辑index.php文件

cd /usr/local/nginx/htmlvim index.php

php文件内容如下

<?phpphpinfo();?>

浏览器访问该文件

出现PHP的配置信息即成功了

接下来安装mysqli扩展

切换到mysqli扩展目录

Cd /usr/local/src/php-7.3.6/ext/mysqli/

启动动态扩展安装

/usr/local/php/bin/phpize

./configure --prefix=/usr/local/mysqli --with-php-config=/usr/local/php/bin/php-config --with-mysqli=/usr/lib64/mysql/mysql\_config

由于我们已经提前安装了依赖，所以报错的信息不会有依赖缺失

ext/mysqlnd/mysql\_float\_to\_double.h: No such file or directorymake: \*\*\* [mysqli\_api.lo] Error 1

如上报错只需要修改一个路径即可

首先检查mysql\_float\_to\_double.h的路径

find / -name mysql\_float\_to\_double.h

将得到的路径复制下来，打开配置文件同时定位到34行

vim /usr/local/src/php-7.3.6/ext/mysqli/mysqli\_api.c +34修改#include "复制下来的路径粘贴到这儿"

重新执行配置命令，然后make和make install

接着修改php.ini

vim /usr/local/php/etc/php.ini将924行的;extension=mysqli.so去掉;注释

重启PHP后，浏览器访问index.php  
kill -9 php的id

会看到已经有mysqli模块了

MySQL篇

解压mysql压缩包

tar -zxvf mysql-boost-5.7.24.tar.gz

cd到mysql目录下

安装cmake工具

yum -y install cmake

开始安装依赖

yum -y install make apr\* autoconf automake curl-devel gcc gcc-c++ zlib-devel openssl openssl-devel pcre-devel gdyum -y install libXpm\* freetype libjpeg\* libpng\* php-common php-gd ncurses\* libtool\* libxml2 libxml2-develyum -y install patch wget ncurses ncurses-devel cmake make perl bison gcc\* libxml2 libxml2-devel curl-devel libjpeg\* freetype\*yum -y install kernel keyutils patch perl kernel-headers compat\* mpfr cpp glibc libgomp libstdc++-devel ppl --skip-broken

这里说明一下第四个条yum，我的系统执行第四条yum时发生了一个冲突，所以根据系统的提示，在后面带上参数--skip-broken

添加mysql用户，创建数据库存放目录和日志存放目录，修改所有者和组，修改权限

useradd mysqldmkdir -p /usr/local/mysql/datamkdir -p /usr/local/mysql/logchown -R mysql:mysql /usr/local/mysql/datachown -R mysql:mysql /usr/local/mysql/log/chmod 750 /usr/local/mysql/data      chmod 750 /usr/local/mysql/log

这些工作准备好了后，开始cmake

cmake -DCMAKE\_INSTALL\_PREFIX=/usr/local/mysql -DMYSQL\_DATADIR=/usr/local/mysql/data -DSYSCONFDIR=/etc -DWITH\_MYISAM\_STORAGE\_ENGINE=1 -DDEFAULT\_CHARSET=utf8 -DWITH\_INNOBASE\_STORAGE\_ENGINE=1 -DWITH\_MEMORY\_STORAGE\_ENGINE=1 -DMYSQL\_UNIX\_ADDR=/tmp/mysqld.sock -DMYSQL\_TCP\_PORT=3306 -DWITH\_READLINE=1 -DENABLED\_LOCAL\_INFILE=1 -DWITH\_BOOST=/usr/local/src/mysql-5.7.24/boost/boost\_1\_59\_0 -DWITH\_PARTITION\_STORAGE\_ENGINE=1

然后开始make

make命令的执行，受电脑性能影响，八代低压i5编译时间是30来分钟

make结束后执行make install命令

当所有结束后，开始初始化数据库

/usr/local/mysql/bin/mysqld --initialize --user=mysql

我的电脑这样执行的时候报了一个错误

错误的处理，在命令后面加参数，命令写为下面形式

/usr/local/mysql/bin/mysqld --initialize --user=mysql --collation-server=utf8\_general\_ci

这时候还会报错

修改配置文件

cd /etc/my.cn在[mysqld]下面添加explicit\_defaults\_for\_timestamp=true

重新初始化

这时候能够得到临时密码

修改配置文件

cd /etc/my.cn在最上面添加[client]default-character-set=utf8在[mysqld]下面添加character-set-server=utf8将datadir后面的路径修改为datadir=/usr/local/mysql/data需要注释的地方#socket=/var/lib/mysql/mysql.sock[mysqld\_safe]#log-error=/var/log/mariadb/mariadb.log#pid-file=/var/run/mariadb/mariadb.pid

保存退出后，再次初始化数据库

记录下生成临时密码

接着复制启动文件到init.d目录下

cp /usr/local/mysql/support-files/mysql.server /etc/init.d/

启动mysql

service mysql.server start

给mysql设置环境变量

vim /etc/profile在最底下添加PATH=/usr/local/mysql/bin:$PATH

保存退出后要刷新生成

source /etc/profile

现在可以登录mysql

mysql -uroot -p输入密码

密码是刚刚初始化的时候生成的临时密码

临时密码只能让你登录，登录后不能做任何操作，必须修改密码

mysql > set password=password('新密码');

做完这些后，LNMP基本做完

开始做web授权

web授权很简单

**1.查看用户**

1. use mysql；
2. select \* from user;

**2.创建用户**

**CREATE USER  'user\_name'@'host'  IDENTIFIED BY  'password'；**

user\_name：要创建用户的名字。

host：表示要这个新创建的用户允许从哪台机登陆，如果只允许从本机登陆，则 填　‘localhost’ ，如果允许从远程登陆，则填 ‘%’

password：新创建用户的登陆数据库密码，如果没密码可以不写。

例：

**CREATE USER  ‘aaa’@‘localhost’ IDENTIFIED BY ‘123456’；**         //表示创建的新用户，名为aaa，这个新用户密码为123456，只允许本机登陆

**CREATE USER  'bbb'@'%' IDENTIFIED BY '123456'；**//表示新创建的用户，名为bbb，这个用户密码为123456，可以从其他电脑远程登陆mysql所在服务器

**CREATE USER  ‘ccc’@‘%’ ；**//表示新创建的用户ccc，没有密码，可以从其他电脑远程登陆mysql服务器

我用 CREATE USER  'aaa'@‘%’；创建新用户，再用 select \* from user；查看用户列表：

**3.授权用户**

命令：**GRANT privileges ON  databasename.tablename  TO  ‘username’@‘host’**

privileges：表示要授予什么权力，例如可以有 select ， insert ，delete，update等，如果要授予全部权力，则填 ALL

databasename.tablename：表示用户的权限能用在哪个库的哪个表中，如果想要用户的权限很作用于所有的数据库所有的表，则填 \*.\*，\*是一个通配符，表示全部。

’username‘@‘host’：表示授权给哪个用户。

例：

**GRANT  select，insert  ON  zje.zje  TO ‘aaa’@‘%’；**         //表示给用户aaa授权，让aaa能给zje库中的zje表 实行 insert 和 select。

**GRANT  ALL  ON  \*.\*  TO  ‘aaa’@‘%’；**//表示给用户aaa授权，让aaa能给所有库所有表实行所有的权力。

用GRANT  ALL  ON  \*.\*  TO  ‘aaa’@‘%’ ；再看用户列表，可以发现权限都变成 Y了。

注意：

用以上命令授权的用户不能给其他用户授权，如果想这个用户能够给其他用户授权，就要在后面加上   WITH GRANT OPTION

如：**GRANT  ALL  ON   \*.\*   TO  ’aaa‘@'%'  WITH GRANT OPTION；**

**4.删除用户**

命令：**DROP  USER ‘user\_name’@‘host’**

例：

**DROP USER 'aaa'@‘%’；//表示删除用户aaa；**

**5.设置与更改用户密码**

**SET  PASSWORD  FOR  ‘username’@‘host’ = PASSWORD(‘newpassword’)；**

如果是设置当前用户的密码：

**SET  PASSWORD = PASSWORD('newpassword')；**

如：**SET  PASSWORD = PASSWORD(‘123456’)；**

**6.撤销用户权限：**

命令：**REVOKE   privileges   ON  database.tablename  FROM  ‘username’@‘host’；**

例如：**REVOKE  SELECT ON  \*.\*  FROM  ‘zje’@‘%’；**

使用navicat远程连接mysql，如果成功连接，那么可以创建属于自己的数据库

# centos:Mysql主从复制

创建PHP文件

<?php

$servername = "127.0.0.1";

$username = "root";

$password = "123456";

// 创建连接

$conn = mysqli\_connect($servername, $username, $password);

// 检测连接

if (!$conn) {

die("连接失败: " . mysqli\_connect\_error());

}

// 创建数据库

$sql = "CREATE DATABASE myDB";

if (mysqli\_query($conn, $sql)) {

echo "数据库创建成功";

} else {

echo "Error creating database: " . mysqli\_error($conn);

}

mysqli\_close($conn);

?>

大体思路：

配置数据库配置

创建用户

开启复制：change

测试和验证

规划：

192.168.0.103 主库

192.168.0.101 从库

主服务器：192.168.0.103;

从服务器：192.168.0.101;

配置过程

1.进入主服务器,停止主服务mysql服务: systemctl stop mysqld;

 主节点编辑配置文件

vim /etc/my.cnf

添加如下配置：

server-id=1

log-bin=master-bin

log-bin-index=master-bin.index

注： 多台节点的server-id必须不同，否则会出现异常。

重启数据库，使得配置生效：

然后启动mysql服务:

systemctl start mysqld;

2. 进入主节点mysql，创建用户：

create user 'slave'@'192.168.0.103' identified by '123456';

注：ip即允许连接的ip地址，安全起见可以写slave从节点的ip(10.1.1.163)，或者直接写%代表所有ip均可以连接。

3. 对用户进行权限设置：

格式：grant replication slave on \*.\* to 'username'@'ip';

操作：grant replication slave on \*.\* to 'slave'@'192.168.0.101';

注： 表示从ip使用username连接过来的请求，允许复制。所以，ip为从节点的ip，username为上面创建的username

使配置生效：

flush privileges;

4. 进入从节点服务器,停止mysql服务: systemctl stop mysqld;

从节点slave编辑配置文件

vim /etc/my.cnf 添加如下配置：

server-id=2

relay-log=slave-relay-bin

relay-log-index=slave-relay-bin

重启mysql服务，使配置生效：

service mysqld restart

5. 从节点mysql进行配置：

change master to  
     master\_host='192.168.0.103',  
     master\_user='slave',  
     master\_password='123456',  
     master\_log\_file='mysql-bin.000004',  
     master\_log\_pos=154;

注： master\_host是主节点的ip地址，master\_user 和master\_password是前面步骤2主服务器创建的用于连接主服务器的帐号和密码,master\_log\_file和master\_log\_pos需要从master节点，进入Mysql数据库后进行查询，命令如下：

进入主服务器登录 mysql: mysql -uroot -p ，输入密码后进入mysql,输入如下指令：

show master STATUS;

6. 在从节点上启动主从复制

start slave;

如需要关闭，使用如下命令：

stop slave;

查看从节点状态：

show slave status \G;

如果没有异常信息，则说明启动成功。

7.如果步骤6报错，：

ERROR 1872 (HY000): Slave failed to initialize relay log info structure from the repository  
意思是：salve复制线程停止，尝试start slave 时报ERROR 1872错误

解决办法：先执行reset slave all;然后再重新执行步骤5，6即可;

查看从节点状态：

show slave status \G; 完美！

报错信息为：The slave I/O thread stops because master and slave have equal MySQL server UUIDs; these UUIDs must be different for replication to work.

/usr/local/mysql/data/auto.cnf

主从两台数据库的UUID相同了，即data目录下的 auto.cnf 中的 server-uuid值相同了找个字母随便改一下就行 比如把 9改成8 ,重启数据库

重新开启slave

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

这两个都为也是表示成功

## 测试：

在主库创建数据库

create database rsync2;

在从库查看是否创建了rsync2

show databases;

1. 首先远程连接进入服务器，在cmd中输入mysql -u root -p，然后回车，输入密码后回车进入mysql命令行。

2.输入use mysql;

3.输入select user,password,host from user;可以看到host中只有localhost主机。我们需要将xxx.xxx.xxx.xxx也添加到这里才对。

4.添加方法如下：

输入grant all privileges on \*.\* to root@”xxx.xxx.xxx.xxx” identified by “密码”;

或者 ​GRANT ALL PRIVILEGES ON \*.\* TO ‘root’@’xxx.xxx.xxx.xxx’ IDENTIFIED BY ‘123456’ WITH GRANT OPTION;

这相当于是给IP-xxx.xxx.xxx.xxx赋予了所有的权限，包括远程访问权限。

然后再输入  
flush privileges;  
这相当于是重新加载一下mysql权限，这一步必须有。

5.再次输入select user,password,host from user;  
可以看到host中已经有了新加的IP。

6.现在再次用Navicat for MySQl访问远程mysql数据库，已经能正常打开了。  
问题解决。

7.如果还是连接不到，那是不是 MySQL Server 绑定了本地地址，打开 /etc/mysql/my.cnf，

找到：bind-address = 127.0.0.1

去除 IP 地址绑定，把它改为：bind-address = 0.0.0.0

然后重启 MySQL Server：/etc/init.d/mysql restart