**Mapnik安装部署**

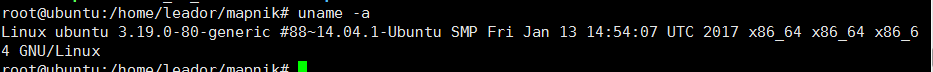
**二维研发部**

**夏神保**

**2018年7月19日**

**前言：**

**本示例系统环境为Ubuntu 14.04.1；**



# 升级GCC：

[https://blog.csdn.net/qingrenufo/article/details/78661513](https://blog.csdn.net/qingrenufo/article/details/78661513" \t "_blank)

sudo add-apt-repository ppa:ubuntu-toolchain-r/test

如果此步报ppa的错误，请参考下面步骤：

1.需要创建一个文件/etc/resolvconf/resolv.conf.d/tail

sudo vi /etc/resolvconf/resolv.conf.d/tail

2.在该文件中写入自己需要的dns服务器，格式与/etc/resolv.conf相同

nameserver 8.8.8.8

3.重启下resolvconf程序

sudo /etc/init.d/resolvconf restart

再去看看/etc/resolv.conf文件,可以看到自己添加的dns服务器已经加到该文件中

$cat /etc/resolv.conf

如果没有nameserver 8.8.8.8，则手动添加：

$vim /etc/resolv.conf

加入nameserver 8.8.8.8

sudo apt-get update

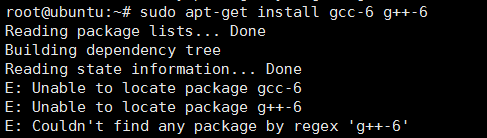
sudo apt-get install software-properties-common

sudo apt-get upgrade

可能需要个把小时

sudo apt-get install gcc-6 g++-6

如果报以下错误：



则安装build-essential：

sudo apt-get install build-essential

sudo apt-get update

sudo apt-get upgrade

然后再执行sudo apt-get install gcc-6 g++-6

gcc -v

发现此时的版本未更改，那么需要更新一下连接。   
gcc4.8更新链接



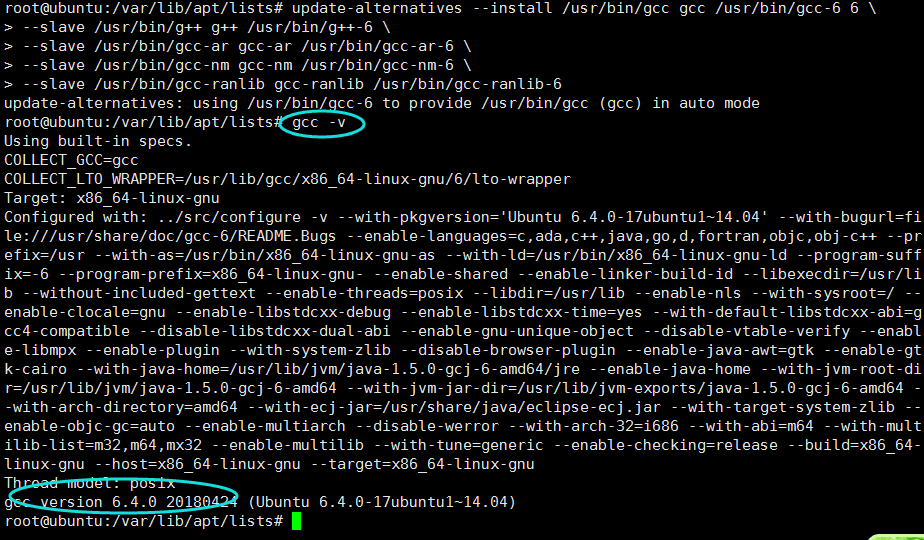
update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-6 6 \

--slave /usr/bin/g++ g++ /usr/bin/g++-6 \

--slave /usr/bin/gcc-ar gcc-ar /usr/bin/gcc-ar-6 \

--slave /usr/bin/gcc-nm gcc-nm /usr/bin/gcc-nm-6 \

--slave /usr/bin/gcc-ranlib gcc-ranlib /usr/bin/gcc-ranlib-6



# 安装mapnik

sudo apt-get install python-mapnik  
这种安装方式是基于python支持的，相当于python安装了一模块，系统无法识别mapnik，输入mapnik-config –v没有版本信息。这种情况恐怕会影响到后来的render调用mapnik。  
按照提示运行以下命令  
sudo apt-get install libmapnik-dev  
再次输入mapnik-config -v  
有版本信息就OK啦

# 安装boost

<https://blog.csdn.net/big_bit/article/details/51258895>

$cd /home/leador

1.首先下载安装包，然后解压， 切换目录

wget –O boost\_1\_55\_0.tar.gz http://sourceforge.net/projects/boost/files/boost/1.55.0/boost\_1\_55\_0.tar.gz/download

tar xzvf boost\_1\_55\_0.tar.gz

cd boost\_1\_55\_0/

安装包也可以在http://www.boost.org/users/history/version\_1\_55\_0.html下载

2.安装附加依赖库

sudo apt-get update

sudo apt-get install build-essential g++ python-dev autotools-dev libicu-dev build-essential libbz2-dev libboost-all-dev

3.运行脚本文件，--prefix参数，可以指定安装路径，如果不带--prefix参数的话（推荐），默认路径是 /usr/local/include 和 /usr/local/lib，分别存放头文件和各种库。执行完成后，会生成bjam，已经存在的脚本将会被自动备份。

cd /home/leador/boost\_1\_55\_0

./bootstrap.sh --prefix=/usr/local

4.安装boost，也就是将头文件和生成的库，放到指定的路径（--prefix）下

./b2 install --prefix=PREFIX  where PREFIX is a directory where you want Boost.Build to be installed.

或者

./b2 install --build-dir=PREFIX  where PREFIX is a directory where you want Boost.Build to be installed.

</div>

5.测试是否安装成功 ，建立一个boost\_test.cpp 文件

vim boost\_test.cpp

6.编辑boost\_test.cpp 文件

#include <iostream>

#include <boost/array.hpp>

using namespace std;

int main(){

boost::array<int, 4> arr = {{1,2,3,4}};

cout << "hi" << arr[0]<<endl;

return 0;

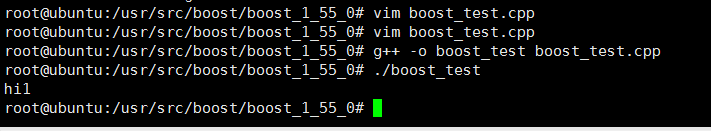
}

7.编译boost\_test.cpp

g++ -o boost\_test boost\_test.cpp

8.运行boost\_test，查看结果

./boost\_test



sudo /sbin/ldconfig

sudo apt-get install libprotobuf-dev protobuf-c-compiler

# 安装依赖库

sudo apt-get install libboost-all-dev subversion git-core tar unzip wget bzip2 build-essential autoconf libtool libxml2-dev libgeos-dev libgeos++-dev libpq-dev libbz2-dev libproj-dev munin-node munin libprotobuf-c0-dev protobuf-c-compiler libfreetype6-dev libpng12-dev libtiff4-dev libicu-dev libgdal-dev libcairo-dev libcairomm-1.0-dev apache2 apache2-dev libagg-dev liblua5.2-dev ttf-unifont lua5.1 liblua5.1-dev libgeotiff-epsg node-carto

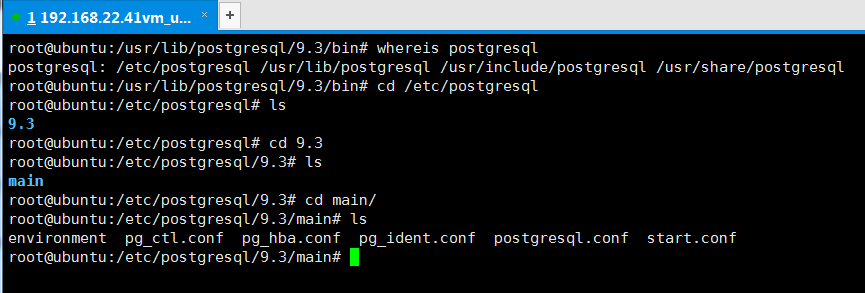
# 安装postgresql / postgis

sudo apt-get install postgresql postgresql-contrib postgis postgresql-9.3-postgis-2.1

## 配置访问权限：

$whereis postgresql

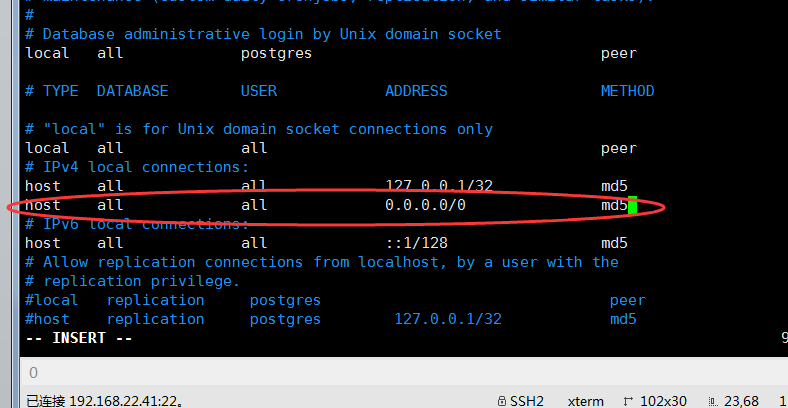
$cd /etc/postgresql/9.3/main



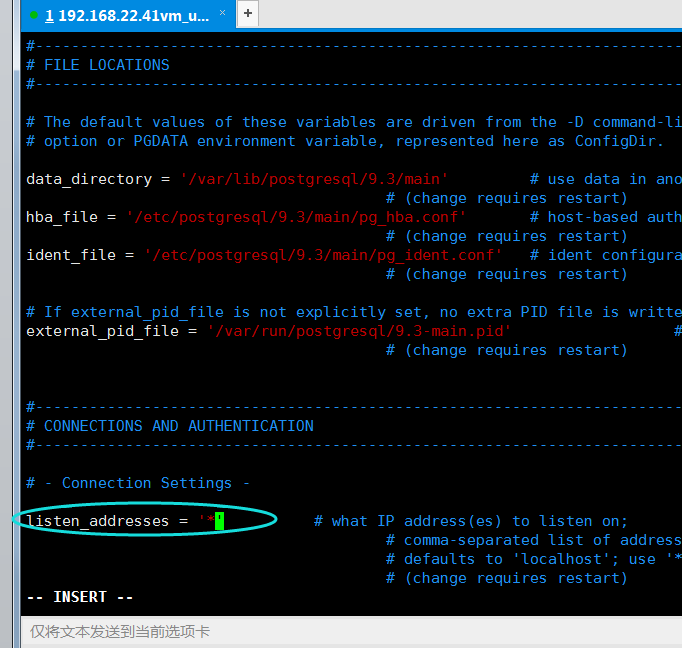
vim pg\_hba.conf

在host all all 127.0.0.1/32 md5行下添加以下配置：

host all all 0.0.0.0/0 md5



vim postgresql.conf



## 配置用户

初次安装后，默认生成一个名为postgres的数据库和一个名为postgres的数据库用户，同时还生成了一个名为postgres的操作系统用户。

1、切换到postgresql用户：

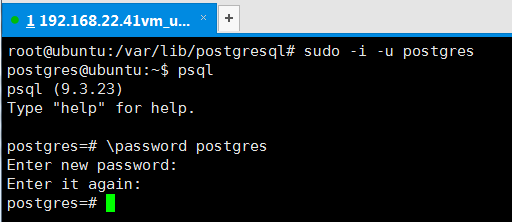
sudo -i -u postgres

2、进入数据库：

psql

3、修改postgresql用户密码：

\password postgres



## 重启postgresql：

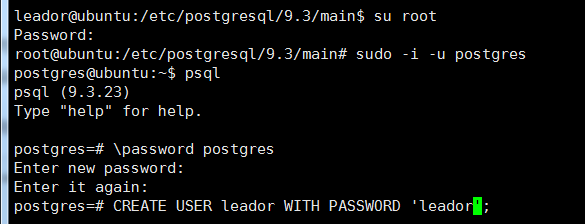
$\q

$exit

service postgresql restart

## 创建用户leador：

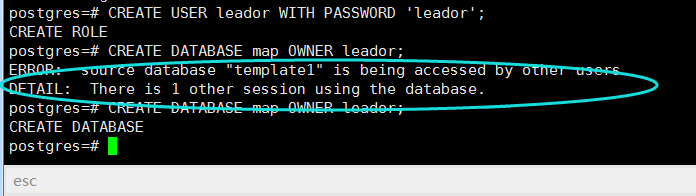
<https://blog.csdn.net/zhangzeyuaaa/article/details/77941039>



## 创建用户数据库，如map：

postgres=# CREATE DATABASE map OWNER leador;

如果报错如下图，则应该关闭其他客户端连接：



再执行创建数据库即可；

## 把map数据库赋权给leador用户：

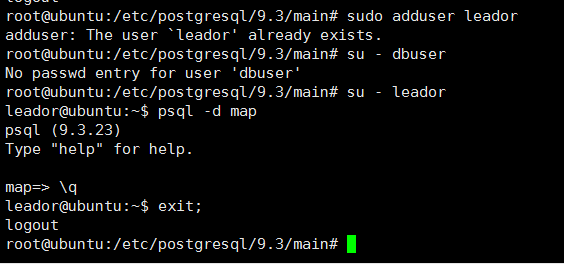
GRANT ALL PRIVILEGES ON DATABASE map TO leador;



退出psql

**\q**

## 以leador的身份连接数据库map

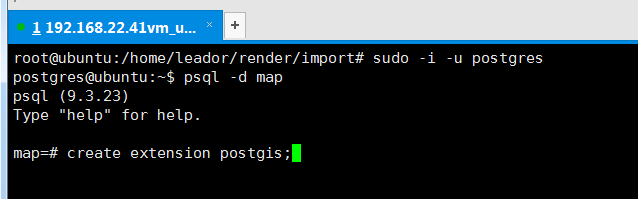


## 添加postgis扩展

sudo -i -u postgres

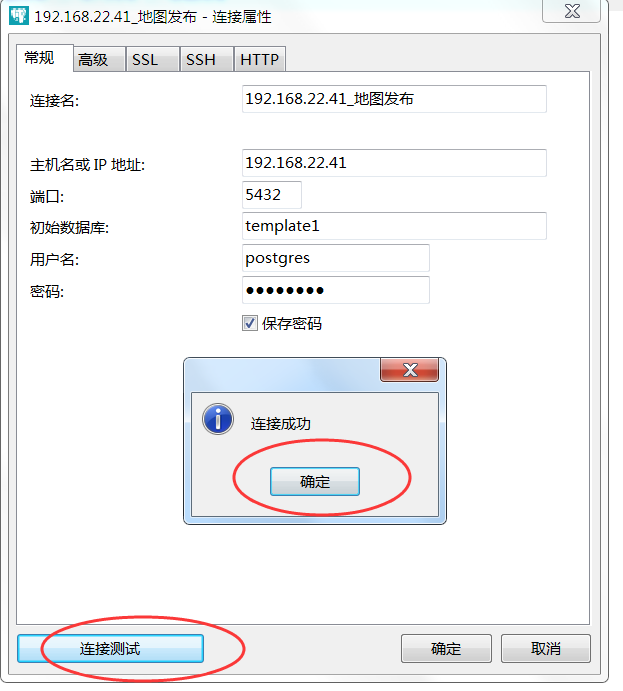
psql –d map

create extension postgis;



## 测试远程访问数据库

用navicate连接postgresql进行测试：

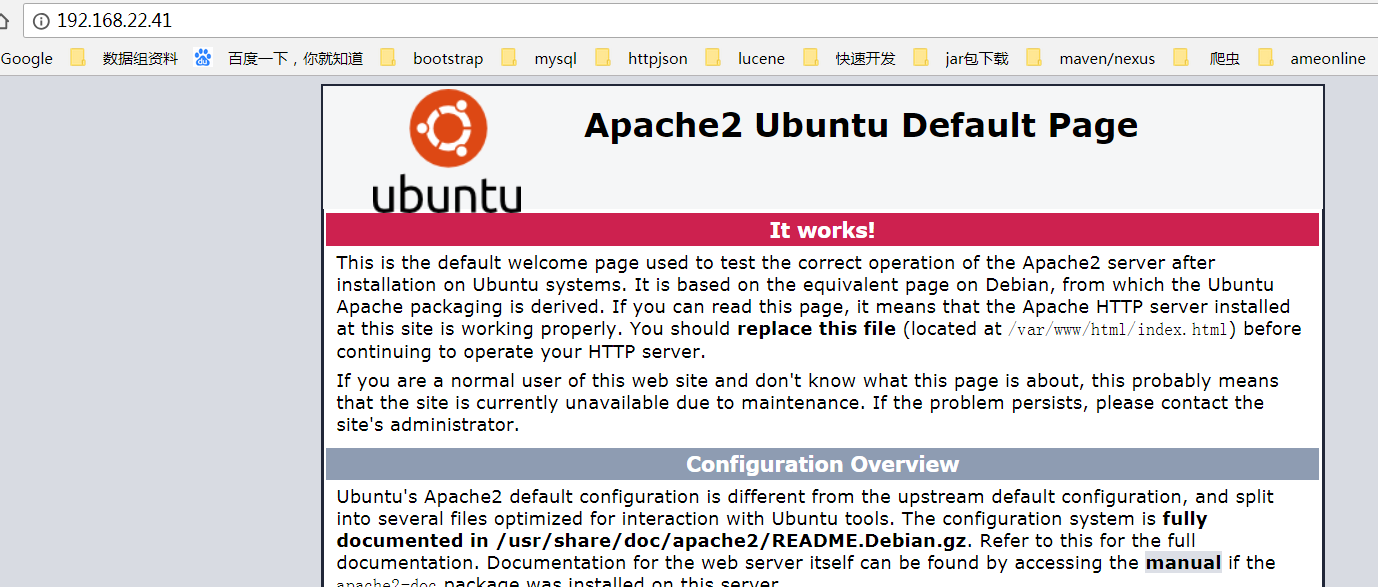


# 安装apache2（mod\_tile依赖）

sudo apt-get install apache2

测试是否安装成功：

http://服务器IP地址



        Apache的安装路径为/var

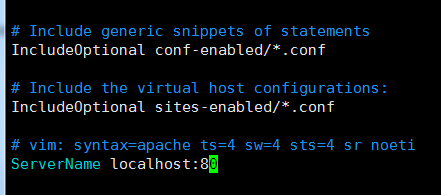
        默认的网站根目录的路径为/var/www/html

## 配置Apache自启动

vi /etc/apache2/apache2.conf

在最下面添加：

ServerName localhost:80



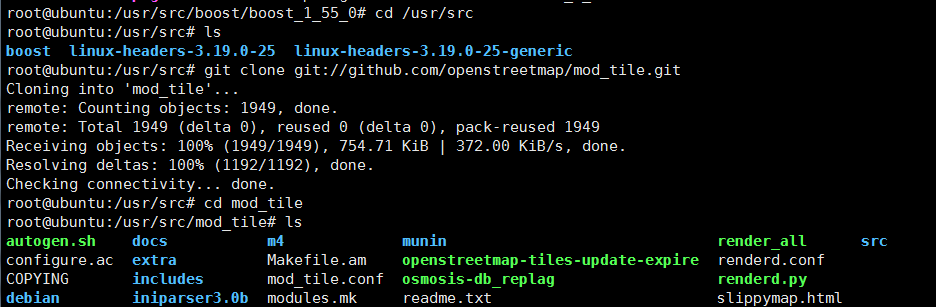
重启apache

sudo /etc/init.d/apache2 restart

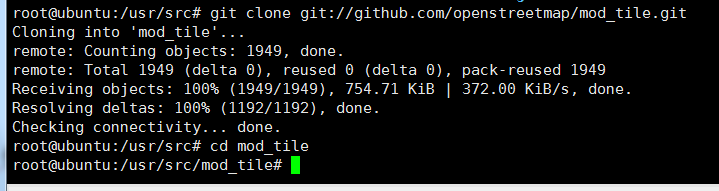
# 安装mod\_tile（依赖mapnik）和renderd

cd /usr/src/

git clone git://github.com/openstreetmap/mod\_tile.git



cd mod\_tile



./autogen.sh

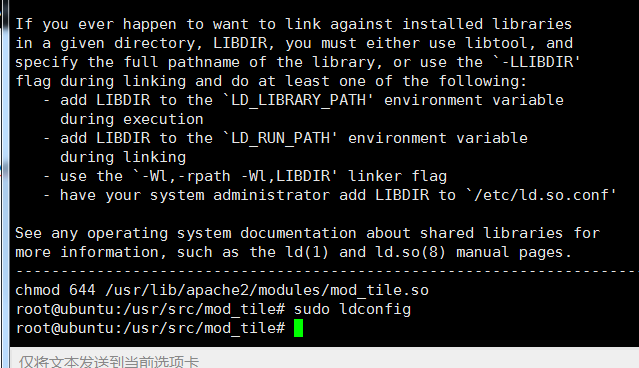
./configure

make

sudo make install

sudo make install-mod\_tile

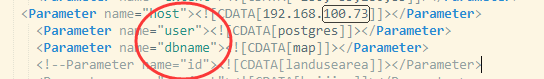
sudo ldconfig



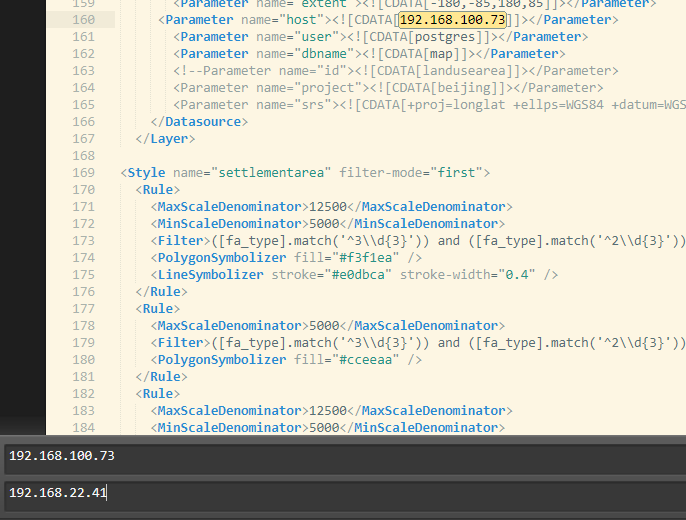
# 配图

## 修改XML文件

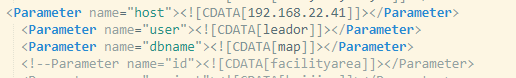
将/home/leador/render/service\_render\_data/ame\_platform/beijing.xml里面数据库IP地址，用户名和密码都修改一下：



全部替换



最后为：



## 配置render

$vim /usr/local/etc/renderd.conf

设置如下属性

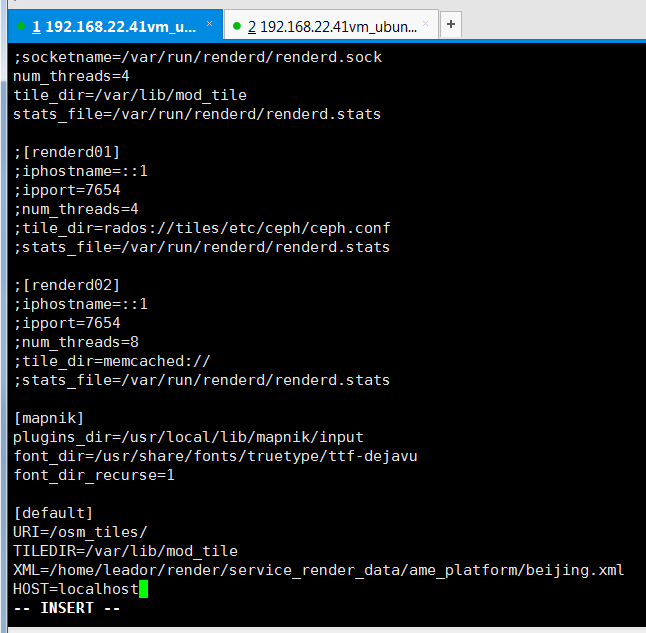
socketname=/var/run/renderd/renderd.sock

plugins\_dir=/usr/lib/mapnik/2.2/input

font\_dir=/usr/share/fonts/truetype/ttf-dejavu

XML=/home/leador/render/service\_render\_data/ame\_platform/beijing.xml

HOST=localhost



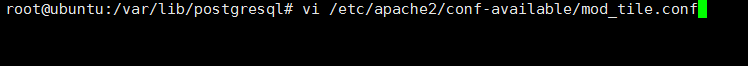
不管有没有对应的文件夹，直接配置接口，但是要拷贝beijing.xml到对应的文件夹位置；（我们直接把100.73下的/home/leador/render文件夹拷贝到此服务器/home/leador/render中即可）；

## 配置mod\_tile

创建文件：

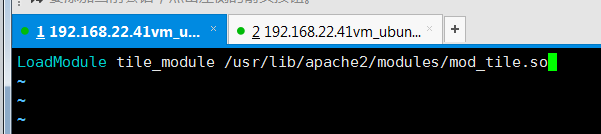
/etc/apache2/conf-available/mod\_tile.conf

whereis apache2



内容为：

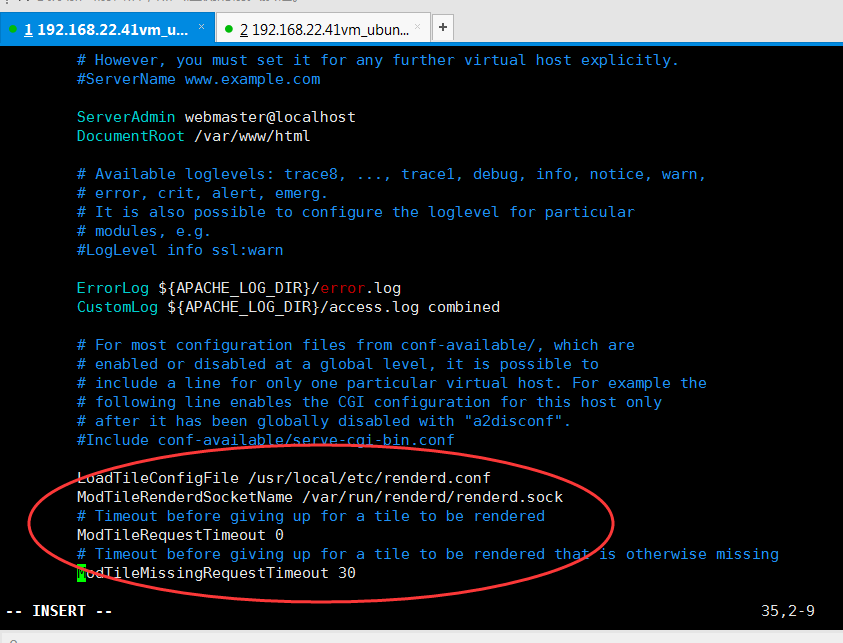
LoadModule tile\_module /usr/lib/apache2/modules/mod\_tile.so



## 配置/etc/apache2/sites-available/000-default.conf

$vim /etc/apache2/sites-available/000-default.conf

加入以下配置：



LoadTileConfigFile /usr/local/etc/renderd.conf

ModTileRenderdSocketName /var/run/renderd/renderd.sock

# Timeout before giving up for a tile to be rendered

ModTileRequestTimeout 0

# Timeout before giving up for a tile to be rendered that is otherwise missing

ModTileMissingRequestTimeout 30

## 重启Apache加载mod\_tile模块

sudo a2enconf mod\_tile

sudo service apache2 reload

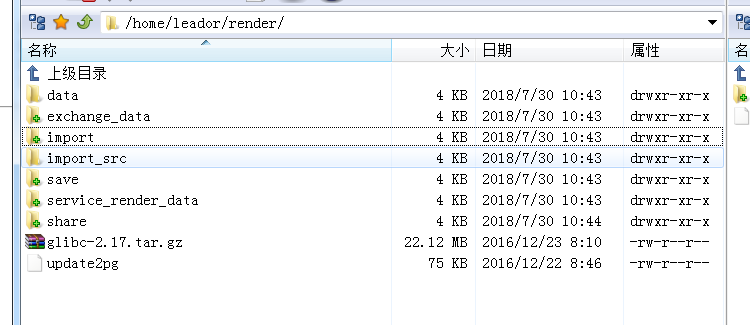
~~Centos httpd~~

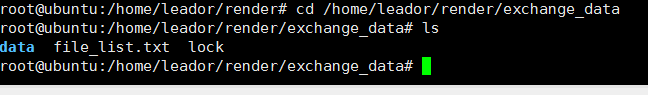
~~/etc/httpd/conf/httpd.conf~~

~~LoadModule tile\_module modules/mod\_tile.so~~

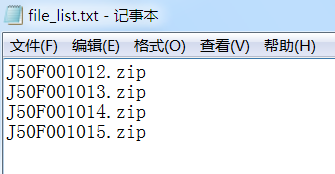
# 导入数据

## 配置需要加载的图幅号

$cd /home/leador/render/exchange\_data

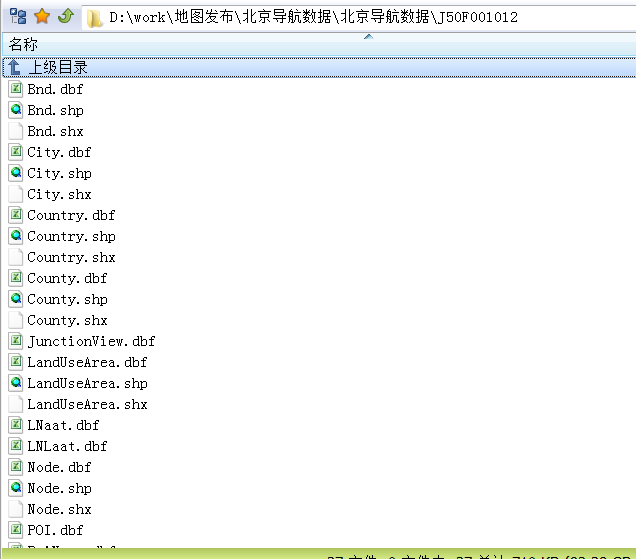


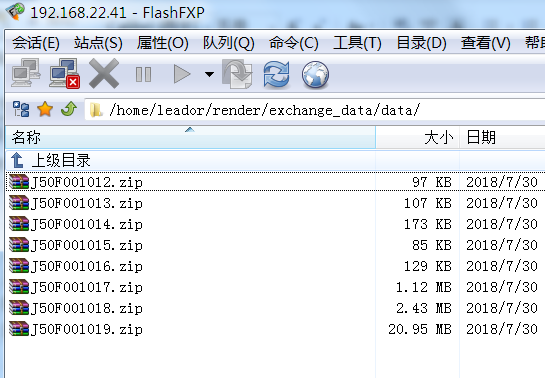
$vim file\_list.txt



## 上传图幅数据

将数据上传到/home/leador/render/exchange\_data/data文件夹下：



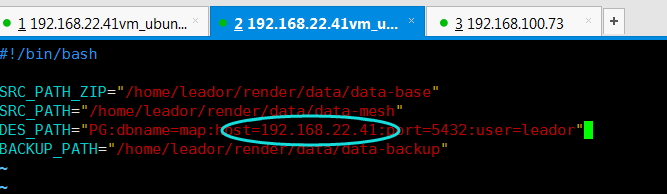


## 导入数据

修改数据库连接文件：

$cd /home/leador/render/import/fflush/

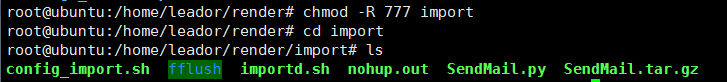
$vim configure.sh



注意：在import文件夹下，importd.sh需要做如上修改，子文件夹fflush下的unzip.sh、importall\_road.sh、importall\_all.sh、fflush.sh需要做如上修改。注意修改ip、用户名和密码。

$cd /home/leador/render

赋权：



切换到leador用户，以防止root用户执行下面脚本导致系统崩溃：

$su leador

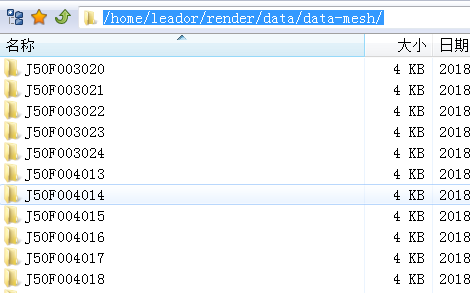
$cd /home/leador/render/import/

nohup bash ./importd.sh > ./importd.log 2>&1 &

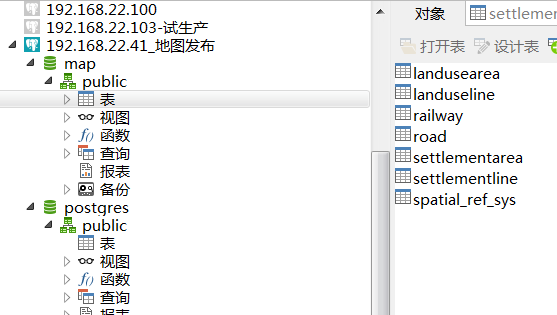
~~nohup sh importd.sh 2>&1 &~~

## 验证是否所有图幅上传完毕

1. 判断/home/leador/render/exchange\_data/data/文件夹下所有zip文件是否清空；
2. 判断/home/leador/render/data/data-mesh/文件夹下是否有对应的图幅号文件夹：



1. 判断数据库是否生成对应的表：



~~启动mapnik渲染~~

~~renderd -f -c /usr/local/etc/renderd.conf~~

~~链接查看：<http://192.168.100.73/preview.html>~~

# 配置邮件

$cd /home/leador/render/import/

$vim SendMail.py

# 切片服务

## 启动切片

创建mod\_tile文件夹：

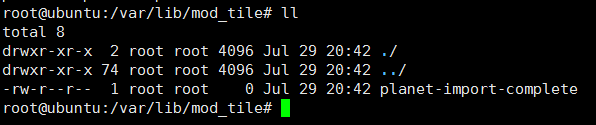
$cd /var/lib

$mkdir mod\_tile



创建planet-import-complete文件：

$vim planet-import-complete



sudo mkdir /var/run/renderd

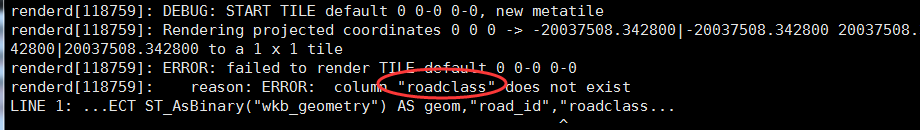
sudo chown username /var/run/renderd

sudo chown username /var/lib/mod\_tile

sudo -u leador renderd -f -c /usr/local/etc/renderd.conf

## 报错集锦

1. 如果报错：



则手动添加roadclass字段

/d road查看表结构

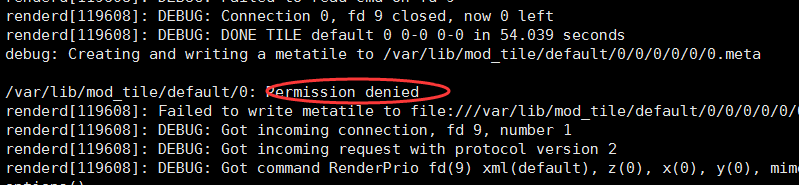
alter table road add roadclass roadclass numeric(8,0);

update road set roadclass=road\_class;

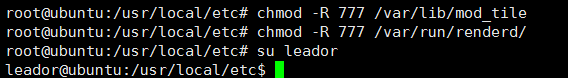
1. 如果报对应的表找不到

则修改beijing.xml，找到对应的Layer表，注释一下，最后再跑切片；

1. 如果报权限问题



则赋权



d)roadname 字段不存在

新建一个与name\_chn数据类型相同的字段

并执行

update road set roadname=name\_chn;

## 重启Apache加载mod\_tile模块

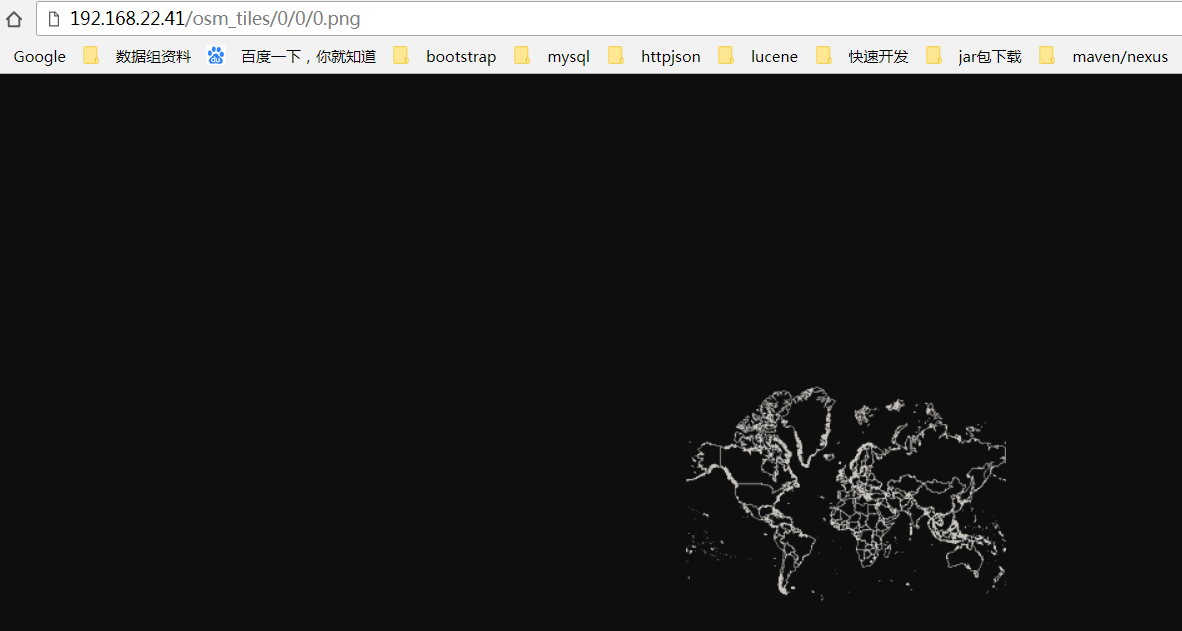
sudo a2enconf mod\_tile

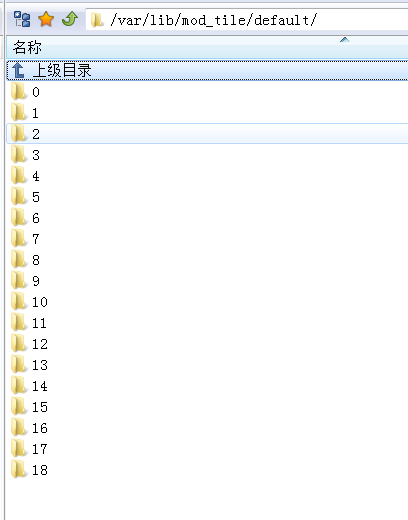
sudo service apache2 reload

## 验证切片是否成功

查看/var/lib/mod\_tile目录，是否生成default文件夹，并且里面有0到18级的目录，或是直接访问IP地址：

<http://192.168.22.41/osm_tiles/0/0/0.png>





# 设置开机启动

为了让开机运行，需要发到 /etc/init.d里

sudo cp /mod\_tile/debian/renderd.init /etc/init.d/renderd

sudo chmod u+x /etc/init.d/renderd

注：以上目录请根据mod\_tile的实际路径做相应修改

而后编辑/etc/init.d/renderd，

DAEMON=/usr/local/bin/$NAME

DAEMON\_ARGS="-c /usr/local/etc/renderd.conf"

保存

使用以下命令可以开启或关闭renderd

sudo /etc/init.d/renderd start

sudo /etc/init.d/renderd stop

最后，设置随机启动

sudo ln -s /etc/init.d/renderd /etc/rc2.d/S20renderd

重启，访问

http://localhost/osm\_tiles/1/1/0.png

就知道wms行不行