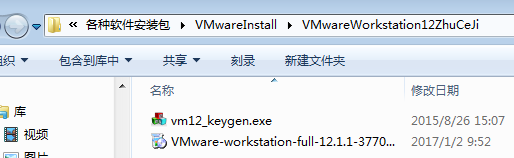
# 操作环境准备

## 1.安装Vmware

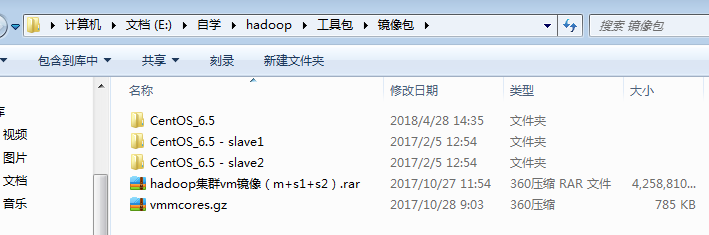


2.获得许可证

1.安装文件

.

## 2获得配置好的linux



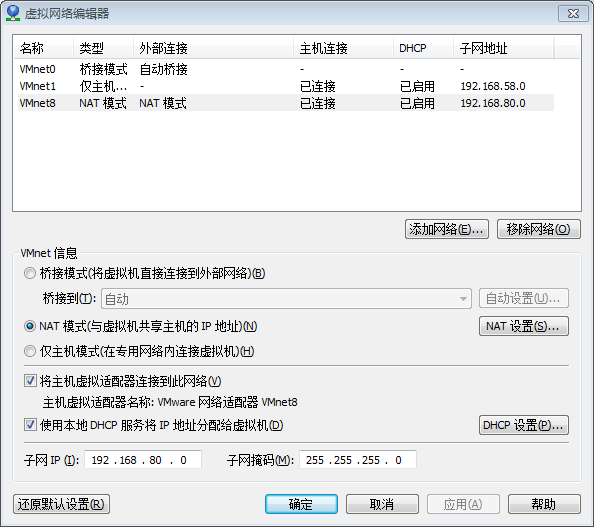
2.解压即可

1.先删除3个文件夹

## 3在VMware中打开Linux

## 4.配置Linux

在VMware左上角选择‘编辑’—‘虚拟网络配置器’



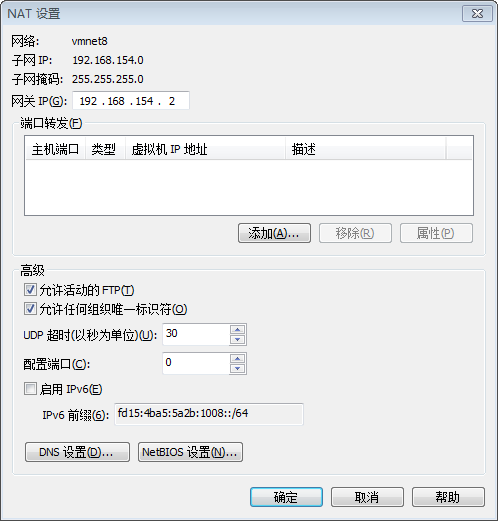
5.点击

4.选择net模式

3.添加Vmnet8

1.点中VMnet8

2.点击

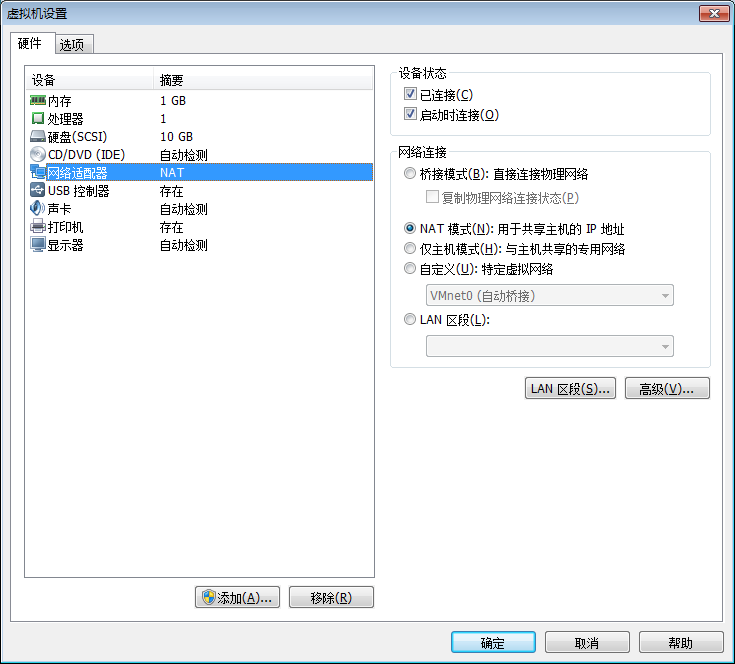


记住网关

应用—确定

看VMware右下角，选择小圈那个，选择设置





2.回来后，再选择net模式，初始化才完成

1.选择桥接模式，确定后再选择上图小圈，回到这个界面

进入Linux

打开Terminal，依次输入以下命令

1. **su**  //进入root，而后输入密码111111
2. **cd /etc/sysconfig/network-scripts**
3. **ls**
4. **vim ifcfg-eth0** //输入到ifcfg-e时按下Tab键自动补全
5. 按下’i’或’insert‘，进入编辑模式
6. 将**BOOTPROTO**=”dhbc”改成BOOTPROTO=“static”

**IPADDR**=192.168.183.10 //添加IPADDR或将其改成上面要求记住的ip，下同

**NETMASK**=255.255.255.0 // 添加此句

**GATEWAY**=192.168.183.2 //改成要求记住的网关

**DNS1**=202.106.0.20 //添加此句

1. 按Esc，输入**”:wq**”，保存退出。（退不出时，用**”:q!”**强制不保存退出，然后用su获得root模式再进入编辑）
2. 新建一个Terminal 输入以下命令，重启网关

**su**

**/etc/init.d/network restart**

1. 使用**ping** [**www.baidu.com**](http://www.baidu.com)

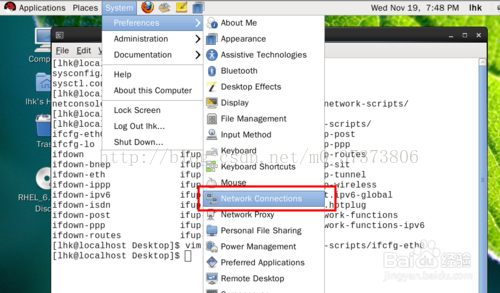
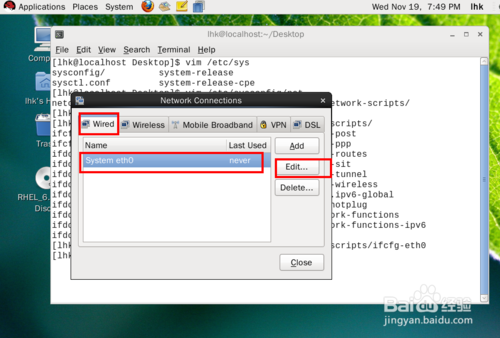
和 curl [**www.baidu.com**](http://www.baidu.com)

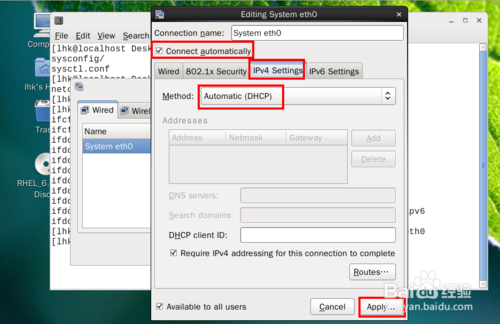
1. 检验是否连上网

11.从root回到用户，用su badou（su 用户名） 连用两次

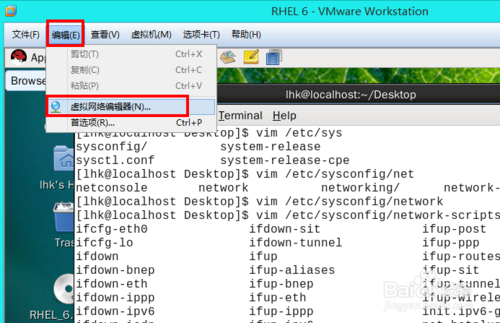
## 5虚拟机VMware安装linux无法上网解决办法

Linux网络设置：

1. 依次单击【System】-->【Preferences】-->【Network Connections】，如下图所示： 
2. 依次单击【Wired】-->【System eth0】-->【Edit】，如下图所示： 
3. 勾选【Connect automaticlly】，然后点击【IPv4 Settings】，在下面的Method中选择【Automatic(DHCP)】,最后点击【Apply】，如下图所示：

END

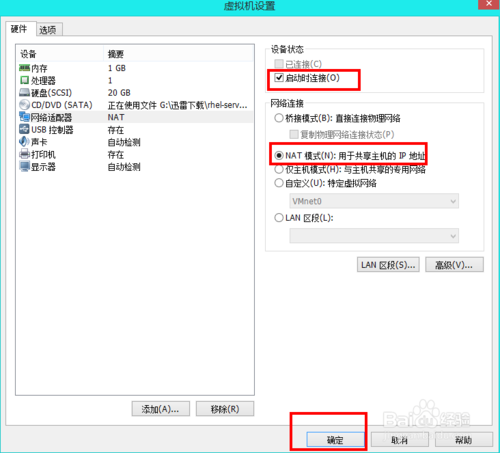
2、恢复虚拟网络默认设置(在断网状态下)：

1. 依次单击【编辑】-->【虚拟网络编辑器】，如下图所示：
2. 点击【恢复默认设置】，如下图所示：



3、设置虚拟机上网方式为NAT：

1. 在虚拟机关机状态下，双击【网络适配器】，如下图所示：
2. 勾选【启动时连接】，选中【NAT模式】，最后点击【确定】如下图所示：

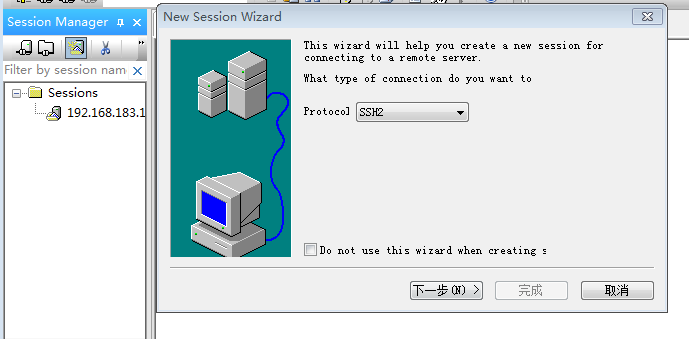


4、见证成果：

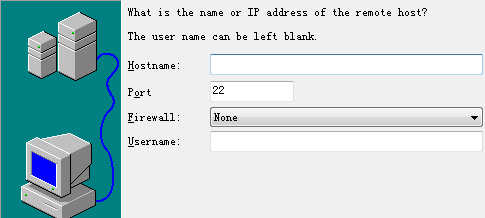
1. 1

在经过以上的设置之后，先在宿主机上联入互联网，然后启动虚拟机Linux，什么也不用做，直接打开Firefox就可以网上冲浪了。

## 6用Secure CRT连接vmware

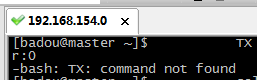


1.点这个

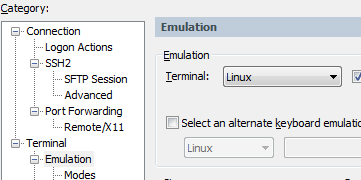


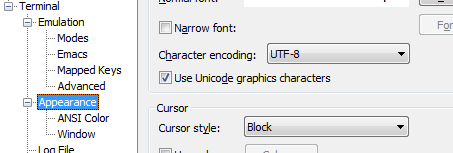
3.刚刚配置的linux的ip号

必须在VMware开启下才有用



右击这个标签，选择最后一个Session Option





# 开发环境准备

从root回到用户，用su badou（su 用户名） 连用两次

创建一个文件，用来安装开发需要的东西 mkdir python

## 安装anaconda

**查看版本 python -V (大写的)**

1.确定服务器是32位的还是64位的

$ file /bin/ls

* 1

输出为:

/bin/ls: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked(uses shared libs), for GNU/Linux 2.6.18, stripped

* 1

根据服务器的位数选择Anaconda的版本，我选择了Linux 64位的。   
2.下载Anaconda   
从[https://www.continuum.io/downloads/](https://www.continuum.io/downloads/" \t "_blank)下载自己需要的Anaconda的版本。   
3.安装   
进入Anaconda所在的目录，执行下面的命令：

$ bash Anaconda2-4.4.0-Linux-x86\_64.sh（根据具体安装包修改）

* 1

此后依照提示操作即可（最简便的方式就是，让ENTER就ENTER，问yes或no，输入yes，三个yes分别代表同意license、使用默认的安装路径、自动向.bashrc写入路径)   
4. 使.bashrc生效   
此时Anaconda并未安装完成，若在终端输入python将会发现依然是Centos自带的python版本，这是因为.bashrc的更新还没有生效，执行下述命令使其生效即可。

**$ source ~/.bashrc**

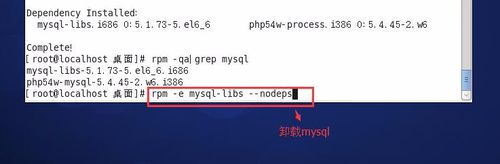
## 2 安装mysql

1. 第一步就是看linu是否安装了mysql，经过rpm -qa|grep mysql查看到centos下安装了mysql5.1，那就开始卸载咯

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=1)

1. 2

接下来就是卸载mysql5.1了，命令：rpm -e mysql-libs --nodeps

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=2)

1. 3

yum中之后mysql5.1，安装还是5.1，现在就要去增加一个新的repo

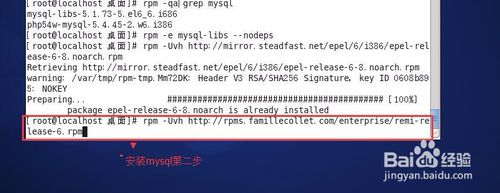
rpm -Uvh http://mirror.steadfast.net/epel/6/i386/epel-release-6-8.noarch.rpm

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=3)

1. 4

一共需要增加两个repo，现在增加另一个

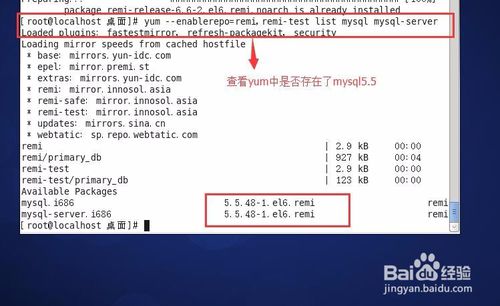
rpm -Uvh http://rpms.famillecollet.com/enterprise/remi-release-6.rpm

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=4)

1. 5

现在查看一些是否增加成功了，yum --enablerepo=remi,remi-test list mysql mysql-server

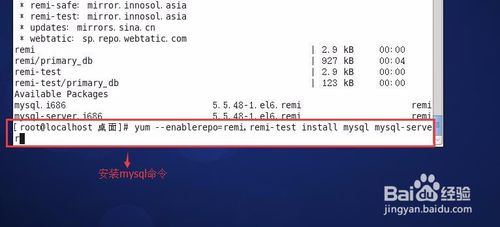
看到下面这些就是安装成功了。

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=5)

1. 6

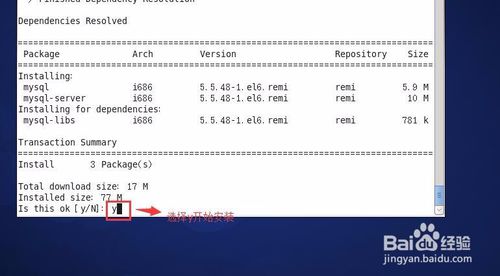
接下来就是安装mysql5.5了

yum --enablerepo=remi,remi-test install mysql mysql-server

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=6)

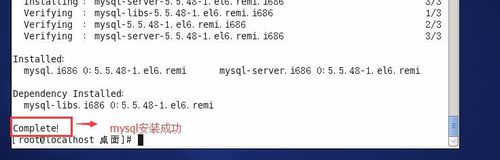
1. 7

出现这个东西的时候一定要选择y。不然神都救不了你，就要重新安装了。

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=7)

1. 8

看到下面这个图片证明你的mysql安装成功了。

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=8)

1. 9

接下来需要启动一下mysql

/etc/init.d/mysqld start

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=9)

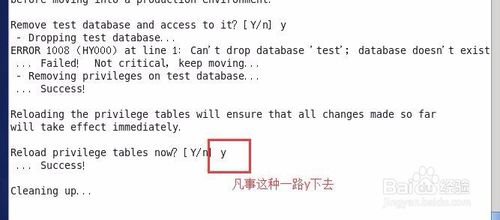
1. 10

下面这条命令设置开机自启动，省的我们每次开机就去启动一下mysql了。

chkconfig --levels 345 mysqld on

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=10)

1. 11

[](http://jingyan.baidu.com/album/922554465e471a851648f4ed.html?picindex=12)

**第2步、设置开机启动**  
[root@stonex ~]#  chkconfig mysqld on  
**第3步、启动MySql服务**  
[root@stonex ~]#  service mysqld start  
**第4步、设置MySQL的root用户设置密码**  
[root@stonex ~]#  mysql -u root  
Welcome to the MySQL monitor.  Commands end with ; or \g.  
...... 省略了一些行  
mysql> select user,host,password from mysql.user;  
查询用户的密码，都为空，用下面的命令设置root的密码为root  
mysql> set password for root@localhost=password('root')  
mysql> exit;  
**第5步、用新密码登陆**  
[root@stonex ~]#  mysql -u root -p

**本机是 mysql -uroot -p111111**  
**第6步、基本命令**  
show databases; //查看系统已存在的数据库  
use databasesname;   //选择需要使用的数据库  
drop database databasename; //删除选定的数据库  
exit    //退出数据库的连接  
create database test01;    //建立名为test的数据库  
show tables;        // 列出当前数据库下的表  
其他基本的增删改查使用标准SQL即可  
**第7步、开放远程登录权限**  
GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'ip' IDENTIFIED BY'123456' WITH GRANT OPTION; 这个\*可以指定某一数据库名。如test.\* 也可以直接用\*，表示全部数据库都开放 （注意空格问题！！！@前后不能有空格，密码那里也不能有空格！！！）

FLUSH PRIVILEGES;

exit;

通过 find /\* -name my.cnf 找到文件

通过vim编辑该文件，找到bind-address = 127.0.0.1这一句，然后在前面加个#号注释掉，保存退出

**第8步、开放3306端口**

输入命令：vim /etc/sysconfig/iptables

添加：-A INPUT -m state --state NEW -m tcp -p tcp --dport 3306 -j ACCEPT

增加的开放3306端口的语句一定要在icmp-host-prohibited之前，否则将仍不能进行访问

然后重启 防火墙：service iptables restart

**第9步 修改mysql服务端的编码**

输入：vi /etc/my.cnf



　　在黄色区域添加如下代码：

|  |  |
| --- | --- |
| 1  2  3 | default-character-set=utf8  [client]  default-character-set=utf8 |

四、重启mysql服务

　　输入：service mysqld restart

## 卸载MySQL5.5

1. [root@localhost usr]# yum remove mysql mysql-server mysql-libs compat-mysql51
2. [root@localhost usr]# rm -rf /var/lib/mysql
3. [root@localhost usr]# rm /etc/my.cnf

如果装了mysql-devel(其他一样add command)，卸载为：

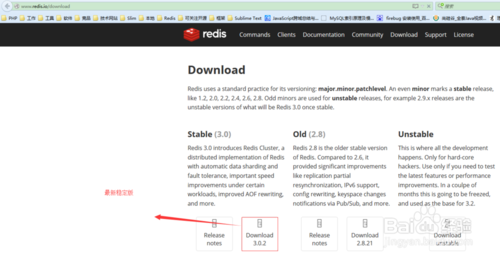
1. [root@Tony\_ts\_tian init.d]# yum remove mysql mysql-devel mysql-server mysql-libs compat-mysql51

## 3 安装Redis

**[Redis源码获取](http://jingyan.baidu.com/album/6dad507510ea07a123e36e95.html?picindex=6" \t "_self)**

[1、进入Redis官网获取Redis最新稳定版下载地址](http://jingyan.baidu.com/album/6dad507510ea07a123e36e95.html?picindex=6" \t "_self)

[2、通过wget命令下载 Redis 源代码。](http://jingyan.baidu.com/album/6dad507510ea07a123e36e95.html?picindex=6" \t "_self)

[](http://jingyan.baidu.com/album/6dad507510ea07a123e36e95.html?picindex=1)

[](http://jingyan.baidu.com/album/6dad507510ea07a123e36e95.html?picindex=2)

wget http://download.redis.io/releases/redis-3.2.8.tar.gz

1. **Redis编译**

1、通过tar -xvf redis-3.2.8.tar.gz命令解压下载Redis源码压缩包redis-3.0.2.tar.gz；

2.

yum -y install gcc

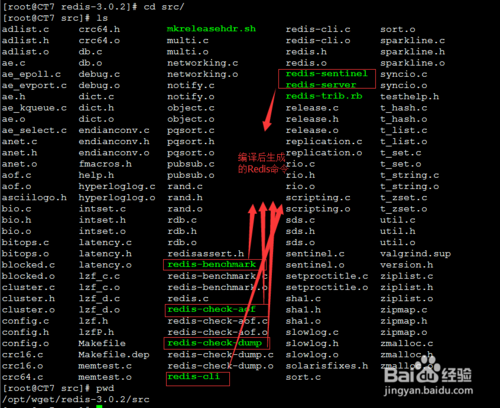
yum -y install gcc-c++

yum install make

yum install gcc gcc-c++ kernel-devel

2、编译Redis。通过cd redis-3.2.8/进入Redis源码目录内，执行make编译Redis；

注意：make命令执行完成编译后，会在src目录下生成6个可执行文件，分别是redis-server、redis-cli、redis-benchmark、redis-check-aof、redis-check-dump、redis-sentinel。

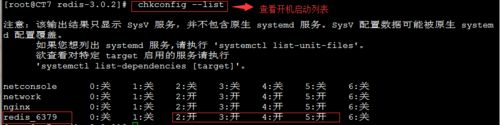
[](http://jingyan.baidu.com/album/6dad507510ea07a123e36e95.html?picindex=3)

1. **Redis安装配置**

1、回到 redis-3.2.8/ cd .. 安装Redis，执行make install。会将make编译生成的可执行文件拷贝到/usr/local/bin目录下；

2、执行./utils/install\_server.sh配置Redis配置之后Redis能随系统启动。一路回车

[](http://jingyan.baidu.com/album/6dad507510ea07a123e36e95.html?picindex=4)

[](http://jingyan.baidu.com/album/6dad507510ea07a123e36e95.html?picindex=5)

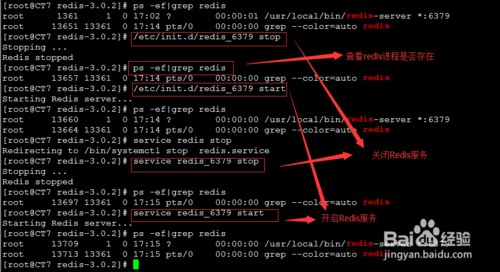
1. **Redis服务查看、开启、关闭**

1、通过ps -ef|grep redis命令查看Redis进程；

2、开启Redis服务操作通过/etc/init.d/redis\_6379 start命令，也可通过（service redis\_6379 start）；

redis 强制关闭方法  
ps auxf|grep redis |grep -v grep|xargs kill -9

3、关闭Redis服务操作通过/etc/init.d/redis\_6379 stop命令，也可通过（service redis\_6379 stop）；

[](http://jingyan.baidu.com/album/6dad507510ea07a123e36e95.html?picindex=6)

## 安装github

* **安装需求**

**# yum install curl-devel expat-devel gettext-devel openssl-devel zlib-devel  
# yum install  gcc perl-ExtUtils-MakeMaker**

* **卸载Centos自带的git1.7.1**

**通过git --version查看系统带的版本，Cento6.5应该自带的是git版本是1.7.1**

**# yum remove git**

* **下载git2.1.2并将git添加到环境变量中**

**下载git最新版本**

**# cd /usr/src  
# wget htt**[**ps**](http://www.111cn.net/fw/photo.html)**://www.kernel.org/pub/software/scm/git/git-2.1.2.tar.gz  
# tar xzf git-2.1.2.tar.gz**

* **安装git并添加到环境变量中**

**# cd git-2.1.2  
# make prefix=/usr/local/git all  
# make prefix=/usr/local/git install  
# echo "export PATH=$PATH:/usr/local/git/bin" >> /etc/bashrc  
# source /etc/bashrc**

* **查看版本号**

**# git --version**

* **git version 2.1.2（即可成功)**
* **下载git的文件：git clone** <https://github.com/WiseDoge/ProxyPool.git>

**报错则改为 git clone** [git://github.com/WiseDoge/ProxyPool.git](https://github.com/WiseDoge/ProxyPool.git)

**区别在于将https改为git**

## 5 安装 FTP

1 检测是否已经安装 FTP

rpm -qa | grep vsftpd

2 若没有，则进行安装

yum install vsftpd

二 设置 vsftpd 开机启动

chkconfig --level 35 vsftpd on

三 配置 FTP 服务器（开启基于用户的访问控制）

1 配置文件的修改

vim /etc/vsftpd/vsftpd.conf

anonymous\_enable=NO # 是否开启匿名登录

local\_enable=YES # 是否允许本地用户登录

write\_enable=YES # 是否允许上传

local\_umask=022 # 默认的umask码

diremssage\_enable=YES # 是否显示目录说明文件

xferlog\_enable=YES # 是否记录ftp传输过程

connect\_from\_prot\_20=YES # 是否确定端口传输来自20

xferlog\_ftd\_format=YES # 是否使用标准的ftp xferlog模式

chroot\_list\_enable=YES 去掉注释 # 是否将系统用户限制在自己的home目录下

chroot\_list\_file=/etc/vsftpd/chroot\_list 去掉注释# 列表不受限制的用户

listen=YES # 是否开启监听

pam\_service\_name=vsftpd # 服务名称

userlist\_enable=YES

tcp\_wrappers=YES

2 修改 selinux

getsebool -a | grep ftp

若报错，getsebool: SELinux is disabled。则

vim /etc/selinux/config

修改 SELINUX=1。

然后重启 LINUX，"shutdown -r now" 立刻重启（root 用户使用）。

重新执行 "getsebool -a | grep ftp"。

修改 selinux，允许用户在家目录写入数据

setsebool -P allow\_ftpd\_anon\_write off

setsebool -P ftp\_home\_dir on

## 6.jupyter notebook

**1. 登陆远程服务器**

**2. 生成配置文件**

$jupyter notebook --generate-config

**3. 生成密码**

打开ipython，创建一个密文的密码：

In [1]: from notebook.auth import passwd

In [2]: passwd()

Enter password:

Verify password:

Out[2]: 'sha1:ce23d945972f:34769685a7ccd3d08c84a18c63968a41f1140274'

In [3]: exit() 退出

**4. 修改默认配置文件**

$vim ~/.jupyter/jupyter\_notebook\_config.py   
进行如下修改：

c.NotebookApp.ip='\*'

c.NotebookApp.password = u'sha:ce...刚才复制的那个密文'

c.NotebookApp.open\_browser = False

c.NotebookApp.port =8888 #随便指定一个端口

**5. 启动jupyter notebook：**

$jupyter notebook

**6. 远程访问**

此时应该可以直接从本地浏览器直接访问http://address\_of\_remote:8888就可以看到jupyter的登陆界面。

**7. 建立ssh通道**

如果登陆失败，则有可能是服务器防火墙设置的问题，此时最简单的方法是在本地建立一个ssh通道：   
在本地终端中输入ssh username@address\_of\_remote -L127.0.0.1:1234:127.0.0.1:8888   
便可以在localhost:1234直接访问远程的jupyter了。

**windows客户端配置**

（1）需要安装一个windows远程ssh登录Linux的工具，可以是Xshell或SecureCRT，这两个工具的安装简单，如果遇到问题请自行百度；

（2）Xshell或SecureCRT第一次启动需要新建连接，一般输入服务器IP地址、用户名、密码即可，这里不细述；

（3）如果安装了Xshell，在Xshell的菜单栏，点击文件-属性-SSH-隧道-添加，侦听端口输入8888，目标主机设为服务器的IP地址，目标端口设为8888，“说明”中填入jupyter，确定。如果安装了SecureCRT，点击Options-Session Options-Port Forwarding-Add，name中填入jupyter，第一个port填入8888，勾选“Destination host is different from the SSH server”，填入服务器的IP地址，port填入8888，点两次OK；

（4）在第（2）步成功远程登录命令行窗口中输入 $ jupyter notebook；

（5）打开本地windows的浏览器，地址栏输入localhost:8888，回车，输入Linux服务器端配置时第（2）步设置的密码，回车，jupyter界面就出来了。

# 爬虫环境准备

## 安装所需依赖库

conda update -n base conda

conda install requests

conda install selenium

conda install pyquery

conda install pymysql

conda install pymongo

conda install redis

conda install django

以下是已安装的

conda install jupyter

conda install flask

## Windows远程连接linux的Redis

1. 进入rediis.3.2.8 里的src,开启redis服务器

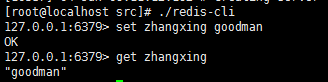
cd redis-3.2.8

**cd src**

**./redis-server**

1. 开启redis客户端

./redis-cli



ok,这样redis就在linux下安装成功了。

exit 退出

.redis的相关配置

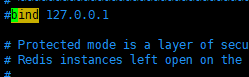
1. 进入redis, 编辑redis.conf

cd ..

vim redis.conf



②将bind 127.0.0.1加上注释，变成 蓝色或绿色



1. 按住下箭头，直到看到要改的的地方。

定位到###################GENERAL#############

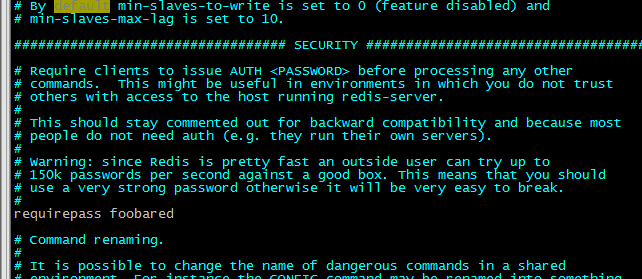
将daemonize(守护进行)的no改成yes

https://img-blog.csdn.net/20170605133812578?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvemhhbmd4aW5nNTIwNzc=/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

接着下拉

继续下拉，定位到###################SECURITY#############

找到requirepass foobared ，将它前面的注释去掉，变成白色



foobared是密码，可以改

4.桌面redis客户端的连接

重启redis

redis-cli -h 127.0.0.1 -p 6379 shutdown

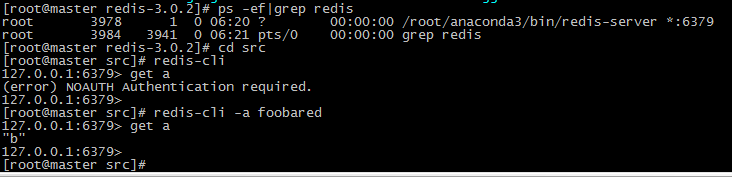
执行./utils/install\_server.sh配置Redis配置之后Redis能随系统启动

通过ps -ef|grep redis命令查看Redis进程,有进程才成功重启

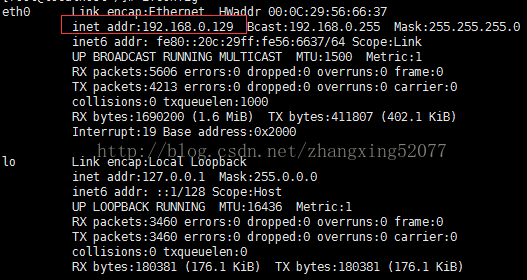
逐步执行下图命令，只有需要输入密码的情况下，才能取值，说明配置成功！

输入密码格式：redis-cli -a 密码

还连不上，就用iptables -F 。再重复上面操作



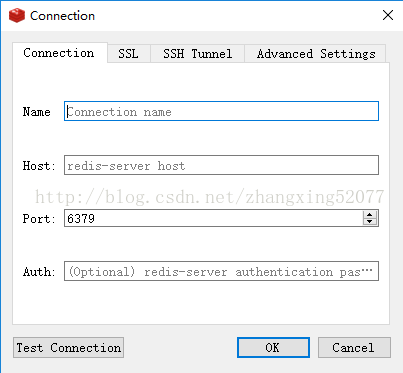
①查看linux主机的ip



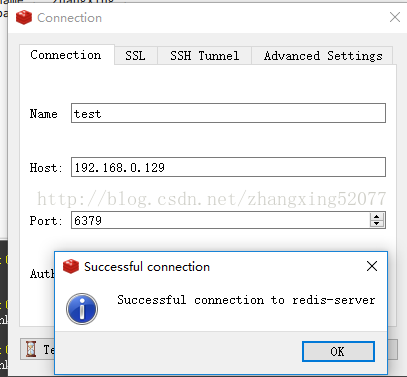
②在win系统下载Redis DeskTop Manager



③点击底部的Connect to Redis Server，弹出如下界面



1. 根据服务器配置编辑



完成之后,重启服务器

# 分布式准备

## 获取和使用代理池

1. 在<https://github.com/WiseDoge/ProxyPool>下载安装

在terminal使用python run.py运行起来

1. 先定义使用接口的变量：从网页获取IP，固定写法

proxy\_pool\_url = ‘http://127.0.0.1:5000/get’，获取代理了

1. 接下来爬虫文件中使用代理：

def get\_proxy():

try:

response = requests.get(proxy\_pool\_url)

if response.status\_code == 200: # 因为是从网页上获取IP的

return response.text # 返回IP

return None

except：

return None

声明一个全局变量 proxy =None

调用该方法即可 proxy =get\_proxy(),

然后声明一个字典 Proxies = {‘http’:’http://’+proxy}构成一个协议

最后执行requests.get(url，prooxies=Proxies)进行爬取

如果获取了代理，就用上面的方法，如果没有获取，就用原来的方法。

当需要判断返回状态时，用response.status\_code，同时在  
requset.set中加入参数：allow\_redirect=false，该参数关闭自动处理状态，让我可以自己处理状态码非200的情况。

requests.get(url，allow\_redirect=false,prooxies=Proxies)

如果response.status\_code==302，则获取新的ip

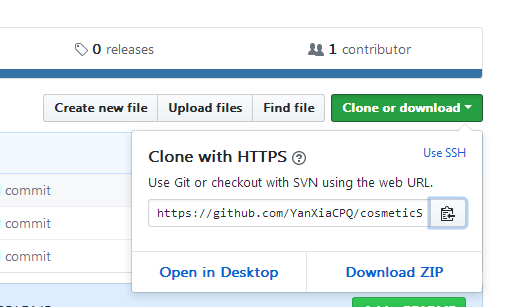
4.因为terminal在后台运行了，所以可以直接在控制台运行爬虫文件

## 获取和安装scrapy-reids

1. scrapy-redis改写了Scrapy的调度器，队列等组件，可以方便地实现scrapy分布式架构
2. 在 <https://github.com/rmax/scrapy-redis>，下载，根据该页面提示要求，安装
3. 修改下载文件的setting.py。Host、PORT、PASSWORD
4. 看分布式第二个11分钟处

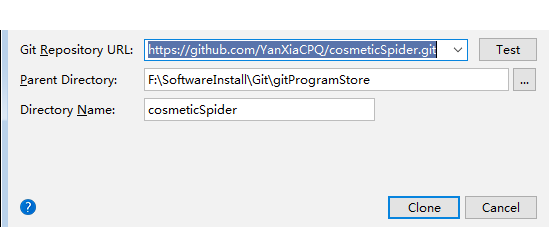
## 分布式搭建

### 1.上传爬虫文件到GitHub

1. 在ip代理池的代码中，setting里面修改必要的参数，比如用户，端口，密码（虚拟机一律为111111）
2. 将爬虫代码放到github上，然后：：

### 2.从Git下载文件

1. windows从GitHub下载代码，在Pychram通过VCS—checkout from version control –GitHub



1. Linux下载

git clone <https://github.com/WiseDoge/ProxyPool.git>

或者 git clone [git://github.com/WiseDoge/ProxyPool.git](https://github.com/WiseDoge/ProxyPool.git)

从该网页中寻找使用方法

查看文件 cat xxx

修改dbop.py文件,vim dbop.py

将redis\_pool = redis.ConnectionPool(host=HOST, port=PORT, max\_connections=20)

改为

redis\_pool = redis.ConnectionPool(host=’IP’, port=6379,password=’111111’, max\_connections=20)

### 3.部署分支

1. 在terminal新建且切换一个分支，不影响源代码。

git checkout -b distributed //新建分支

git branch //切换分支

1. 在setting里加入一些配置（查看<https://github.com/rmax/scrapy-redis>）具体如**Usage**

简略版：添加或修改代码

1.SCHEDULER = "scrapy\_redis.scheduler.Scheduler" # 修改调度器

2.DUPEFILTER\_CLASS = "scrapy\_redis.dupefilter.RFPDupeFilter" # 修改去重的#class

3.在ITEM\_PIPELINES 添加

'scrapy\_redis.pipelines.RedisPipeline': 301（数字改大一些，越大优先级越小。让爬虫代码的pipeline优先）

4.REDIS\_URL = 'redis://user:pass@hostname:9001' #需要每个scrapy共享爬取队列，所以要指定redis连接信息。改user和pass(密码), hostname(IP),9001（端口）

REDIS\_URL = 'redis://root:111111@192.168.174.128:6379' # 写的是服务器（虚拟机）的。设置好后，就可以远程连接服务器的数据库了

**Usage**

Use the following settings in your project:

# Enables scheduling storing requests queue in redis.

SCHEDULER = "scrapy\_redis.scheduler.Scheduler"

# Ensure all spiders share same duplicates filter through redis.

DUPEFILTER\_CLASS = "scrapy\_redis.dupefilter.RFPDupeFilter"

# Default requests serializer is pickle, but it can be changed to any module

# with loads and dumps functions. Note that pickle is not compatible between

# python versions.

# Caveat: In python 3.x, the serializer must return strings keys and support

# bytes as values. Because of this reason the json or msgpack module will not

# work by default. In python 2.x there is no such issue and you can use

# 'json' or 'msgpack' as serializers.

#SCHEDULER\_SERIALIZER = "scrapy\_redis.picklecompat"

# Don't cleanup redis queues, allows to pause/resume crawls.

#SCHEDULER\_PERSIST = True

# Schedule requests using a priority queue. (default)

#SCHEDULER\_QUEUE\_CLASS = 'scrapy\_redis.queue.PriorityQueue'

# Alternative queues.

#SCHEDULER\_QUEUE\_CLASS = 'scrapy\_redis.queue.FifoQueue'

#SCHEDULER\_QUEUE\_CLASS = 'scrapy\_redis.queue.LifoQueue'

# Max idle time to prevent the spider from being closed when distributed crawling.

# This only works if queue class is SpiderQueue or SpiderStack,

# and may also block the same time when your spider start at the first time (because the queue is empty).

#SCHEDULER\_IDLE\_BEFORE\_CLOSE = 10

# Store scraped item in redis for post-processing.

ITEM\_PIPELINES = {

'scrapy\_redis.pipelines.RedisPipeline': 300

}

# The item pipeline serializes and stores the items in this redis key.

#REDIS\_ITEMS\_KEY = '%(spider)s:items'

# The items serializer is by default ScrapyJSONEncoder. You can use any

# importable path to a callable object.

#REDIS\_ITEMS\_SERIALIZER = 'json.dumps'

# Specify the host and port to use when connecting to Redis (optional).

#REDIS\_HOST = 'localhost'

#REDIS\_PORT = 6379

# Specify the full Redis URL for connecting (optional).

# If set, this takes precedence over the REDIS\_HOST and REDIS\_PORT settings.

#REDIS\_URL = 'redis://user:pass@hostname:9001'

# Custom redis client parameters (i.e.: socket timeout, etc.)

#REDIS\_PARAMS = {}

# Use custom redis client class.

#REDIS\_PARAMS['redis\_cls'] = 'myproject.RedisClient'

# If True, it uses redis' ``SPOP`` operation. You have to use the ``SADD``

# command to add URLs to the redis queue. This could be useful if you

# want to avoid duplicates in your start urls list and the order of

# processing does not matter.

#REDIS\_START\_URLS\_AS\_SET = False

# Default start urls key for RedisSpider and RedisCrawlSpider.

#REDIS\_START\_URLS\_KEY = '%(name)s:start\_urls'

# Use other encoding than utf-8 for redis.

#REDIS\_ENCODING = 'latin1'

1. .

shutdown -r now 重启

shutdown -h now 关机

# Spark

### ****安装Java SE****

1、下载JAVA SE linux版本。下载地址为：   
<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

注意选择Linux版本，根据机器类型选择32位或者64位。

2、下载jdk-8u131-linux-x64.tar.gz后，解压到自己指定目录。

cd ~

cd /usr/local

sudo mkdir java

sudo mkdir scala

sudo mkdir spark

$ tar -zxvf jdk-8u131-linux-x64.tar.gz -C /usr/local/java

3、配置JAVA环境变量。

打开.bashrc文件。

$ nano ~/.bashrc

在.bashrc文件末尾增加如下内容：

export JAVA\_HOME=/usr/local/java/jdk1.8.0\_131 (根据下载版本决定，下同)

export JRE\_HOME=${JAVA\_HOME}/jre

export CLASSPATH=.:${JAVA\_HOME}/local:${JRE\_HOME}/local

export PATH=${JAVA\_HOME}/bin:${JRE\_HOME}/bin:$PATH

保存后退出，运行如下命令，使修改环境变量即可生效：

$ source ~/.bashrc

4、检测java是否成功安装。

$ java -version

java version "1.8.0\_131"

如果显示了java安装的版本，则表示已正确安装，可以进行下一步了。

### ****安装Scala****

1、下载Scala的压缩文件。下载地址为：   
<http://www.scala-lang.org/download/>

2、下载scala-2.12.2.tgz后，解压到指定的目录：

$ tar -zxvf scala-2.12.2.tgz -C /usr/local/scala/ (上面已创建该文件夹，下同)

3、配置scala环境变量。

打开.bashrc文件。

$ nano ~/.bashrc

在.bashrc文件末尾增加如下内容：

export SCALA\_HOME=/usr/local/scala/scala-2.12.2

export PATH=${SCALA\_HOME}/bin:$PATH

保存后退出，运行如下命令，使修改环境变量即可生效：

$ source ~/.bashrc

4、检测scala是否正确安装。

$ scala -version

Scala code runner version 2.12.2 -- Copyright 2002-2017, LAMP/EPFL and Lightbend, Inc

若显示如上信息，则表示scala已安装正确。

### ****安装Spark****

1、下载Spark的压缩文件。下载地址为：   
<http://spark.apache.org/downloads.html>

2、下载spark-2.1.1-bin-hadoop2.7.tgz后，解压到指定的目录：

$ tar -zxvf spark-2.1.1-bin-hadoop2.7.tgz -C /opt/spark/

3、配置scala环境变量。

打开.bashrc文件。

$ nano ~/.bashrc

在.bashrc文件末尾增加如下内容：

export SPARK\_HOME=/usr/local/spark/spark-2.1.1-bin-hadoop2.7

export PATH=${SPARK\_HOME}/bin:$PATH

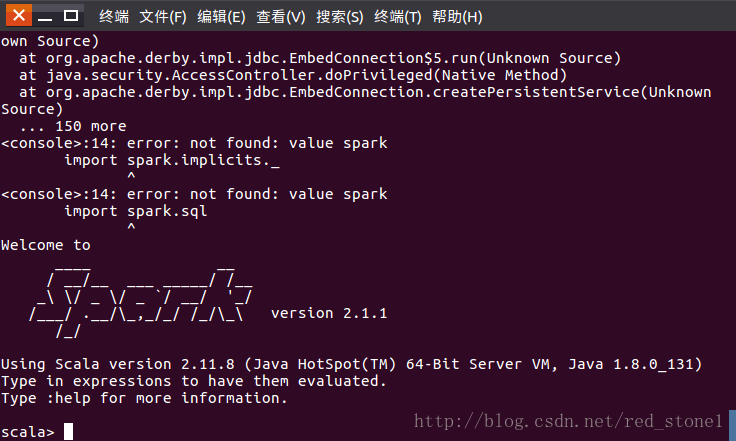
保存后退出，运行如下命令，使修改环境变量即可生效：

$ source ~/.bashrc

4、检测spark是否正确安装。

$ spark-shell

* 1

如果出现如下界面，则表示spark已正确安装：   


### ****安装sbt****

1、下载Spark的压缩文件。下载地址为：   
<http://www.scala-sbt.org/download.html>

2、下载sbt-0.13.15.tgz后，解压到指定的目录：

$ tar -zxvf sbt-0.13.15.tgz -C /opt/scala/sbt/

3、配置sbt环境变量。

打开.bashrc文件。

$ nano ~/.bashrc

在.bashrc文件末尾增加如下内容：

export SBT\_HOME=/opt/scala/sbt

export PATH=${SBT\_HOME}/bin:$PATH

保存后退出，运行如下命令，使修改环境变量即可生效：

$ source ~/.bashrc

4、建立启动sbt的脚本。

$ mkdir /opt/scala/sbt/

$ cd /opt/scala/sbt/

$ touch sbt

脚本内容如下，注意sbt-launch.jar的路径

SBT\_OPTS="-Xms512M -Xmx1536M -Xss1M -XX:+CMSClassUnloadingEnabled -XX:MaxPermSize=256M"

java $SBT\_OPTS -jar /opt/scala/sbt/bin/sbt-launch.jar "$@"

修改sbt文件的权限

$ chmod u+x sbt

1

5、检测sbt是否正确安装。

$ sbt sbt-version

[info] 0.13.15

第一次执行的时候会下载很多东西，不用管它，后来就好了。   
如果出现sbt的版本信息就表示安装成功。

至此，Linux下Spark单机环境已经成功搭建完毕！