**基于Hadoop2.6.0的Spark1.3.1大数据处理平台的搭建**

目 录

[一、虚拟化软件、实验虚拟机准备 4](#_Toc418549839)

[（一）VMware Workstation 11 4](#_Toc418549840)

[（二）模版机安装 4](#_Toc418549841)

[（三）安装VMware tools 4](#_Toc418549842)

[（四）安装FTP服务 4](#_Toc418549843)

[二、Hadoop与Spark的安装配置 4](#_Toc418549844)

[（一）登陆和使用系统 4](#_Toc418549845)

[（二）下载和安装jdk-7u79 5](#_Toc418549846)

[（三）配置单机模式hadoop 7](#_Toc418549847)

[1. 安装SSH和rsync 7](#_Toc418549848)

[2. 安装hadoop2.6.0 8](#_Toc418549849)

[3. 编辑Hadoop环境配置文件 8](#_Toc418549850)

[4. 运行单机例子 9](#_Toc418549851)

[（四）配置伪分布式hadoop 10](#_Toc418549852)

[1. 创建分布式文件系统所需目录 10](#_Toc418549853)

[2. 配置分布式部署描述符文件 10](#_Toc418549854)

[3. 编辑Hadoop环境配置文件（参见单机模式） 11](#_Toc418549855)

[4. 编辑masters和slaves文件 11](#_Toc418549856)

[5. 格式化namenode 11](#_Toc418549857)

[6. 启动hadoop 12](#_Toc418549858)

[7. 停止Hadoop 12](#_Toc418549859)

[（五）配置hadoop分布式集群 12](#_Toc418549860)

[1.配置IP地址 12](#_Toc418549861)

[2.修改主机名 15](#_Toc418549862)

[3.安装hadoop2.6.0 15](#_Toc418549863)

[4.编辑Hadoop环境配置文件 16](#_Toc418549864)

[5.创建分布式文件系统的目录 17](#_Toc418549865)

[6.配置分布式部署描述符文件 17](#_Toc418549866)

[7.编辑masters和slaves文件 19](#_Toc418549867)

[（六）安装Scala、Spark和IDEA 19](#_Toc418549868)

[1.分别解压到相关目录 19](#_Toc418549869)

[2.编辑当前用户环境变量配置文件 20](#_Toc418549870)

[3.编辑spark运行环境配置文件 21](#_Toc418549871)

[4.编辑Spark的slaves文件 21](#_Toc418549872)

[5.Idea IDEA安装 21](#_Toc418549873)

[（七）克隆其他slave节点 22](#_Toc418549874)

[1.克隆slave节点 22](#_Toc418549875)

[2.配置集群SSH无密码验证 22](#_Toc418549876)

[3.保持配置文件同步 23](#_Toc418549877)

[三、Hadoop和Spark集群测试 24](#_Toc418549878)

[（一）启动hadoop分布式集群 24](#_Toc418549879)

[（二）启动Spark分布式集群 28](#_Toc418549880)

[（三）服务启动后启动webUI 30](#_Toc418549881)

[（四）在Hadoop分布式集群中运行wordcount示例 33](#_Toc418549882)

[（五）在Spark分布式集群中运行wordcount示例 38](#_Toc418549883)

[附录 40](#_Toc418549884)

[附录一：64位Ubuntu Linux虚拟机中手动安装或升级 VMware Tools 40](#_Toc418549885)

[附录二：FTP工具Win-SCP 44](#_Toc418549886)

[附录三：SecureCRT SSH登陆管理 45](#_Toc418549887)

[附录四：Ubuntu下火狐浏览器安装Flash及书签使用相关事项 45](#_Toc418549888)

[附录五：Hadoop2.6.0在Ubuntu14.04.2 64位系统中使用的编译方法 46](#_Toc418549889)

**前言**

本文为零基础的同学准备，主要是引导大家搭建平台，深入学习可参考Spark亚太研究院的《**Spark实战高手之路 从零开始**》系列教程。参考链接：<http://book.51cto.com/art/201408/448416.htm>

# 虚拟化软件、实验虚拟机准备

## VMware Workstation 11

注册码/key ：1F04Z-6D111-7Z029-AV0Q4-3AEH8

* 开发中可使用桌面版VMware Workstation 11，便于向vShpere管理的ESXi Server服务器“上载”PC机中配置好虚拟机，便于把调试好的开发环境迁移到生产环境的服务器上。

## 模版机安装

OS:ubuntu-14.04.2-desktop-amd64.iso

\*\*\*在Ubuntu 中安装VMwareTools以便于在宿主机和虚拟机之间共享内存，可以互相拷贝文本和文件，这个功能很方便，具体参见附录1：《Linux虚拟机中手动安装或升级 VMware Tools》。

* 自定义用户lolo及密码ljl，这个安装时候设置，该用户在后面的FTP和SSH服务中用到该用户。

## 安装VMware tools

详见附录一。

## 安装FTP服务

详见附录二

# Hadoop与Spark的安装配置

## 登陆和使用系统

* 以下用vim和gedit修改相应脚本文件均可，如果是命令行就用vim，如果是图形界面就用gedit。
* 进入root用户权限

lolo@lolo-virtual-machine:~$ sudo -s

* 安装vim编辑

注意：关于校园网linux无法上网的问题，如果你用的是WIFI上网，建议接入360wifi的访问点中。

虚拟机使用

root@lolo-virtual-machine:~# apt-get install vim

* 修改lightdm.conf环境变量

root@lolo-virtual-machine:~# vim /etc/lightdm/lightdm.conf

#允许用户登陆并关闭guest用户

[SeatDefaults]

user-session=ubuntu

greeter-session=unity-greeter

greeter-show-manual-login=true

allow-guest=false

* 设置root用户密码

root@lolo-virtual-machine:~# sudo passwd root

设置密码：ljl

* 修改/root/.profile：

备注：为避免root登录开机出现以下提示：

Error found when loading /root/.profile

stdin:is not a tty

…………

root@lolo-virtual-machine:~# gedit /root/.profile

打开文件后找到“mesg n”，

将其更改为“tty -s && mesg n”

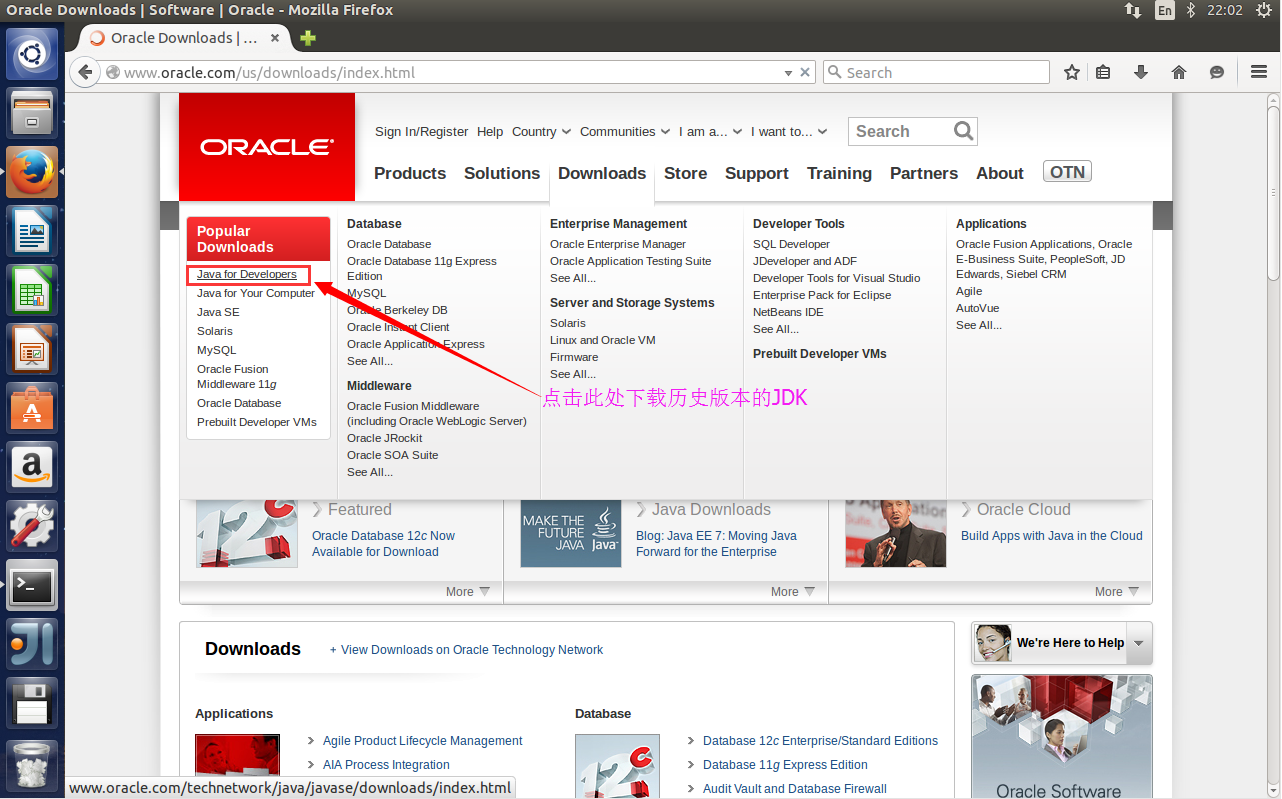
* 重启

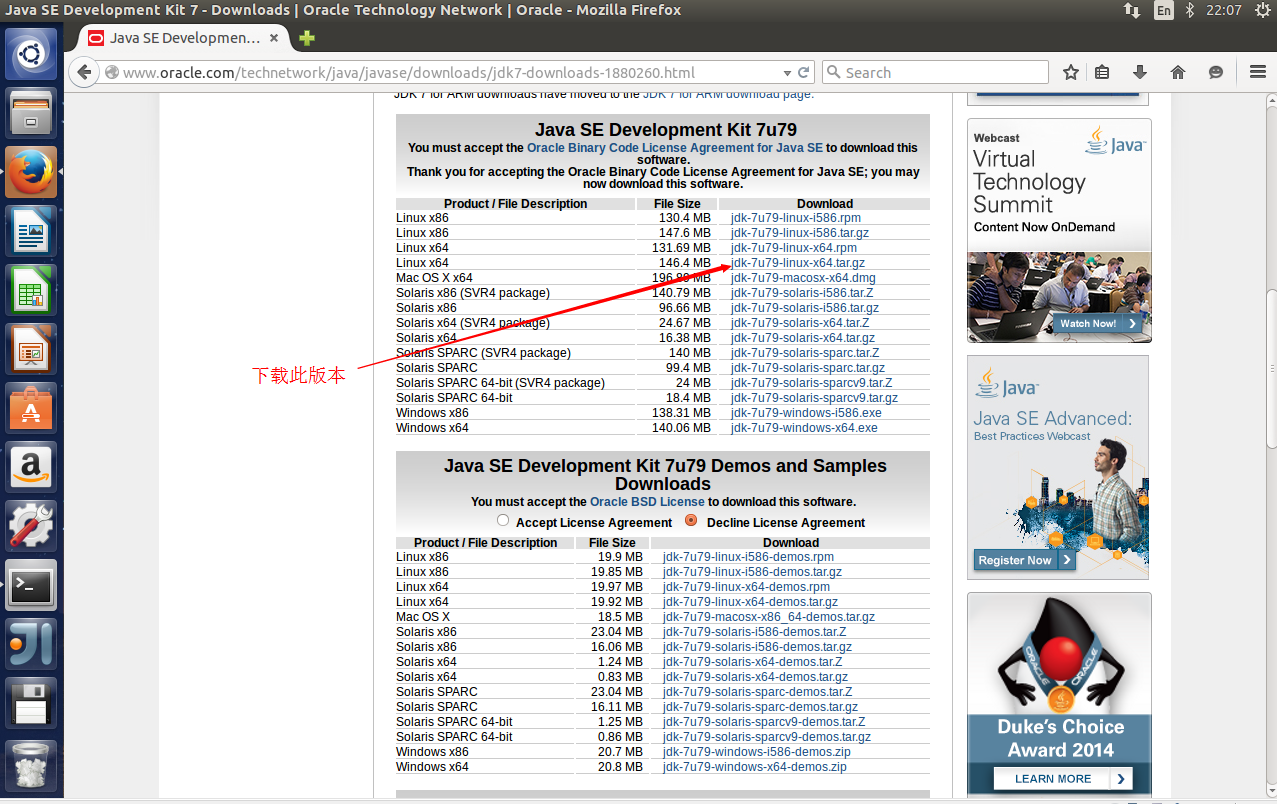
root@lolo-virtual-machine:~#reboot –h now

## 下载和安装jdk-7u79

* 注意：目前JDK1.7是hadoop2.6.0和Spark1.3.1能够稳定运行的最新版本，目前测试jdk-7u79-linux-x64.tar.gz可稳定运行，推荐。jdk-7u80-linux-x64.tar.gz和JDK1.8有些不稳定，不推荐使用。

下载链接http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html





JDK会被下载到当前用户的Downloads目录下。

* 创建java安装目录

root@lolo-virtual-machine:~# mkdir /usr/lib/java

* 将压缩包copy到安装目录

root@lolo-virtual-machine:~# mv /root/Downloads/jdk-7u79-linux-x64.tar.gz /usr/lib/java

* 进入安装目录

root@lolo-virtual-machine:~# cd /usr/lib/java

* 解压缩JDK压缩包

root@lolo-virtual-machine:/usr/lib/java# tar -xvf jdk-7u79-linux-x64.tar.gz

（也可以用图形化界面来解压缩）

* 编辑配置文件，添加环境变量。

root@lolo-virtual-machine:~# vim ~/.bashrc

“i”

加入：

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

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

export CLASS\_PATH=.:${JAVA\_HOME}/lib:${JRE\_HOME}/lib

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

敲“esc” 键 输入“：wq”保存退出。

* 使脚本配置生效

root@lolo-virtual-machine:~# source ~/.bashrc

## 配置单机模式hadoop

下载链接： <http://mirrors.cnnic.cn/apache/hadoop/common/hadoop-2.6.0/>

此处下载的hadoop2.6.0已经是64位编译的，可以在64位linux系统下使用。

### 1. 安装SSH和rsync

root@lolo-virtual-machine:~# apt-get install ssh

或者：sudo apt-get install ssh openssh-server

（必要时reboot一下，校园网有时更新源有问题）

* 启动服务

root@lolo-virtual-machine:~# /etc/init.d/ssh start

* 测试服务

root@lolo-virtual-machine:~# ps -e |grep ssh

* 设置免密码登陆

root@lolo-virtual-machine:~# ssh-keygen -t rsa -P ""

root@lolo-virtual-machine:~# cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys

* 测试本地ssh服务：

root@lolo-virtual-machine:~# ssh localhost

root@lolo-virtual-machine:~#exit

* 安装rsync

 root@lolo-virtual-machine:~# apt-get install rsync

### 2. 安装hadoop2.6.0

注意：目前最新版本为2.7.0，属于测试版本，不稳定，建议使用2.6.0.

root@lolo-virtual-machine:~# mkdir /usr/local/hadoop

root@lolo-virtual-machine:~# cd /root/Downloads/

root@lolo-virtual-machine:~# mv /root/Downloads/ hadoop-2.6.0.tar.gz /usr/local/hadoop/

root@lolo-virtual-machine:~/Downloads# cd /usr/local/hadoop/

root@lolo-virtual-machine: /usr/local/hadoop # tar -xzvf hadoop-2.6.0.tar.gz

root@lolo-virtual-machine: /usr/local/hadoop # cd /usr/local/hadoop/hadoop-2.6.0/etc/Hadoop

查JDK路径

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#${JAVA\_HOME}

bash: /usr/lib/java/jdk1.7.0\_79: Is a directory

### 3. 编辑Hadoop环境配置文件

#### 1）hadoop-env.sh

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#vim hadoop-env.sh

备注：此处用gedit命令替代vim也可，看习惯。

键入“i”

将export JAVA\_HOME=${JAVA\_HOME}

改为export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

（其他两个文件加入本句代码）：

敲“esc”键，输入“:wq”保存退出。

应用该配置：

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#source hadoop-env.sh

#### 2）yarn-env.sh

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#gedit yarn-env.sh

在# export JAVA\_HOME=/home/y/libexec/jdk1.6.0/下面加入：

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#source yarn-env.sh

#### 3）mapred-env.sh

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#gedit mapred-env.sh

在# export JAVA\_HOME=/home/y/libexec/jdk1.6.0/下面加入：

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#source mapred-env.sh

#### 4）修改~/.bashrc文件中的环境变量

root@lolo-virtual-machine:/# vim ~/.bashrc

* 插入

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

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

export CLASS\_PATH=.:${JAVA\_HOME}/lib:${JRE\_HOME}/lib

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

#HADOOP VARIABLES START

export HADOOP\_INSTALL=/usr/local/hadoop/hadoop-2.6.0

export PATH=$PATH:$HADOOP\_INSTALL/bin

export PATH=$PATH:$HADOOP\_INSTALL/sbin

export PATH=$PATH:$HADOOP\_INSTALL/etc/hadoop

export HADOOP\_MAPRED\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_HOME=$HADOOP\_INSTALL

export HADOOP\_HDFS\_HOME=$HADOOP\_INSTALL

export YARN\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_INSTALL/lib/native

export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_INSTALL/lib"

#HADOOP VARIABLES END

* 应用配置

root@lolo-virtual-machine:~# source ~/.bashrc

* 查看Hadoop版本

root@lolo-virtual-machine:~# hadoop version

### 4. 运行单机例子

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0#mkdir input

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0#cp README.txt input

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0# bin/hadoop jar share/hadoop/mapreduce/sources/hadoop-mapreduce-examples-2.6.0-sources.jar org.apache.hadoop.examples.WordCount input output

* 查看结果

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0# cat output/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*至此Hadoop单机模式配置成功\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## 配置伪分布式hadoop

### 1. 创建分布式文件系统所需目录

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0#mkdir tmp

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0#mkdir dfs

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0#mkdir dfs/data

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0#mkdir dfs/name

或

cd /usr/local/hadoop/hadoop-2.6.0

mkdir tmp dfs dfs/name dfs/data

### 2. 配置分布式部署描述符文件

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit core-site.xml

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit hdfs-site.xml

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit mapred-site.xml

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit yarn-site.xml

#### 1）core-site.xml

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0# cd /etc/hadoop

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit core-site.xml

* 伪分布式（Pseudo-Distributed Operation）

<configuration>

<property>

<name>fs.defaultFS</name>

<value>hdfs://localhost:9000</value>

</property>

</configuration>

#### 2）hdfs-site.xml

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# vim hdfs-site.xml

* 伪分布式（Pseudo-Distributed Operation）

<configuration>

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

</configuration>

#### 3）mapred-site.xml

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# vim mapred-site.xml

* 伪分布式（Pseudo-Distributed Operation）

<configuration>

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

</configuration>

#### 4）yarn-site.xml

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#gedit yarn-site.xml

* 伪分布式（Pseudo-Distributed Operation）

<configuration>

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

</configuration>

### 3. 编辑Hadoop环境配置文件（参见单机模式）

### 4. 编辑masters和slaves文件

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit masters

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit slaves

或：

sudo gedit /usr/local/hadoop/etc/hadoop/masters 添加：localhost

sudo gedit /usr/local/hadoop/etc/hadoop/slaves 添加：localhost

### 5. 格式化namenode

root@lolo-virtual-machine:~# hdfs namenode -format

2015-02-11 14:47:20,657 INFO [main] namenode.NameNode (StringUtils.java:startupShutdownMessage(633)) - STARTUP\_MSG:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

STARTUP\_MSG: Starting NameNode

STARTUP\_MSG: host = lolo-virtual-machine/127.0.1.1

STARTUP\_MSG: args = [-format]

STARTUP\_MSG: version = 2.6.0

### 6. 启动hadoop

root@lolo-virtual-machine:/# start-dfs.sh

root@lolo-virtual-machine:/# start-yarn.sh

root@lolo-virtual-machine:/# jps

几个hadoop集群运行状态监控Web页面：

<http://localhost:50030/jobtracker.jsp>

<http://localhost:50060/tasktracker.jsp>

<http://localhost:50070/dfshealth.jsp>

### 7. 停止Hadoop

root@lolo-virtual-machine:/# stop-dfs.sh

root@lolo-virtual-machine:/# stop-yarn.sh

## 配置hadoop分布式集群

### 配置IP地址

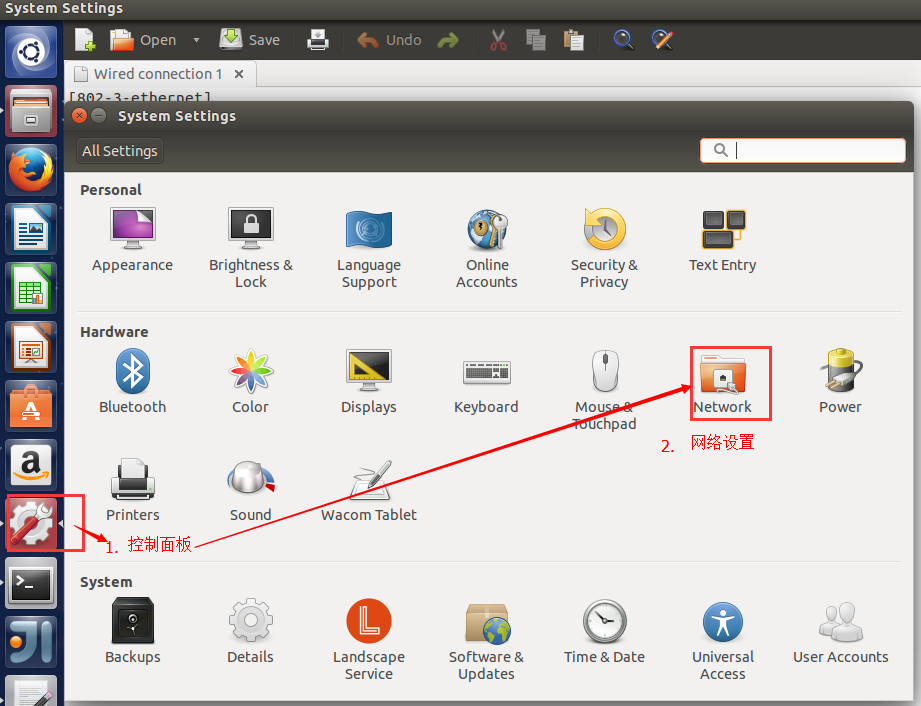
查看网卡IP配置命令

root@lolo-virtual-machine:/# ifconfig

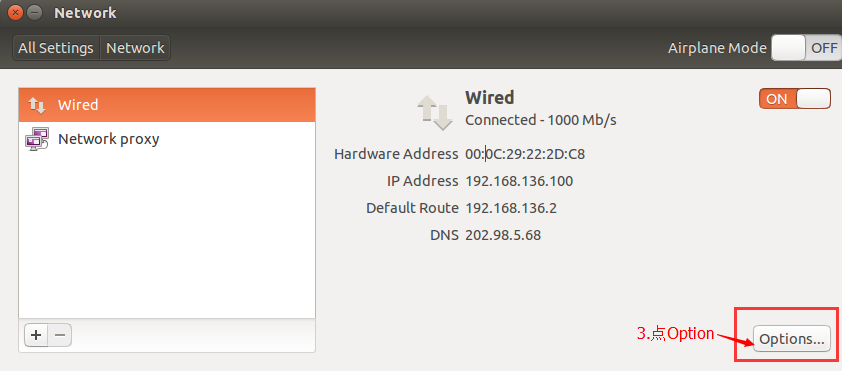
eth0 Link encap:Ethernet  HWaddr 00:0c:29:02:4f:ac

inet addr:192.168.207.136  Bcast:192.168.207.255  Mask:255.255.255.0

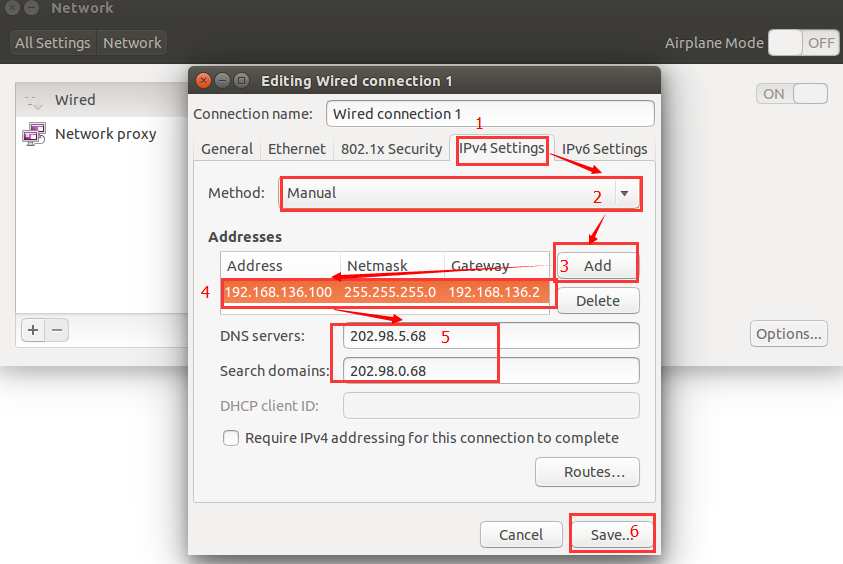
* **第一种方法：使用管理面板设置IP**



* 打开控制面板，点击“Network”



* 点击Option，添加IP、网关和DNS



* **第二种方法：手动**[**设置静态IP**](http://www.cnblogs.com/vincedotnet/p/4013099.html)**（）**

1） 找到配置文件并作如下修改：

root@SparkMaster:/etc/NetworkManager/system-connections# vim Wired\ connection\ 1

修改如下部分：

[802-3-ethernet]

duplex=full

mac-address=00:0C:29:22:2D:C8

[connection]

id=Wired connection 1

uuid=de16d53e-bb1a-47c1-a2e8-70b9107b20ec

type=802-3-ethernet

timestamp=1430738836

[ipv6]

method=auto

[ipv4]

method=manual

dns=202.98.5.68;

dns-search=202.98.0.68;

address1=192.168.136.100/24,192.168.136.2

本例中使用图形界面修改的，地址配置信息被保在了: /etc/NetworkManager/system-connections/目录下的Wired connection 1文件中。

2）重启网卡：

sudo /etc/init.d/networking restart

### 修改主机名

root@lolo-virtual-machine:/# vim /etc/hostname

lolo-virtual-machine改为：SparkMaster

重启后测试：

root@lolo-virtual-machine:/#sudo reboot –h now

root@SparkMaster:/# hostname

SparkMaster

SparkWorker1，SparkWorker2同上

 SparkWorker1的IP规划为192.168.136.101

 SparkWorker2的IP规划为192.168.136.102

root@SparkMaster:/# vim /etc/hosts

将：

127.0.0.1       localhost

127.0.1.1       lolo-virtual-machine

改为：

127.0.0.1 localhost

192.168.136.100 SparkMaster

192.168.136.101 SparkWorker1

192.168.136.102 SparkWorker2

### 安装hadoop2.6.0

注意：目前最新版本为2.7.0，属于测试版本，不稳定，建议使用2.6.0.

root@SparkMaster:~# mkdir /usr/local/hadoop

root@SparkMaster:~# cd /root/Downloads/

root@SparkMaster:~# mv /root/Downloads/ hadoop-2.6.0.tar.gz /usr/local/hadoop/

root@SparkMaster:~ # cd /usr/local/hadoop/

root@SparkMaster:/usr/local/hadoop # tar -xzvf hadoop-2.6.0.tar.gz

root@SparkMaster:/usr/local/hadoop # cd /usr/local/hadoop/hadoop-2.6.0/etc/Hadoop

查JDK路径

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#${JAVA\_HOME}

bash: /usr/lib/java/jdk1.7.0\_79: Is a directory

### 编辑Hadoop环境配置文件

#### 1）hadoop-env.sh

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#vim hadoop-env.sh

备注：此处用gedit命令替代vim也可，看习惯。

键入“i”

将export JAVA\_HOME=${JAVA\_HOME}

改为export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

（其他两个文件加入本句代码）：

敲“esc”键，输入“:wq”保存退出。

应用该配置：

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#source hadoop-env.sh

#### 2）yarn-env.sh

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#gedit yarn-env.sh

在# export JAVA\_HOME=/home/y/libexec/jdk1.6.0/下面加入：

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#source yarn-env.sh

#### 3）mapred-env.sh

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#gedit mapred-env.sh

在# export JAVA\_HOME=/home/y/libexec/jdk1.6.0/下面加入：

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#source mapred-env.sh

#### 4）修改~/.bashrc文件中的环境变量

root@SparkMaster:/# vim ~/.bashrc

* 插入

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

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

export CLASS\_PATH=.:${JAVA\_HOME}/lib:${JRE\_HOME}/lib

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

#HADOOP VARIABLES START

export HADOOP\_INSTALL=/usr/local/hadoop/hadoop-2.6.0

export PATH=$PATH:$HADOOP\_INSTALL/bin

export PATH=$PATH:$HADOOP\_INSTALL/sbin

export PATH=$PATH:$HADOOP\_INSTALL/etc/hadoop

export HADOOP\_MAPRED\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_HOME=$HADOOP\_INSTALL

export HADOOP\_HDFS\_HOME=$HADOOP\_INSTALL

export YARN\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_INSTALL/lib/native

export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_INSTALL/lib"

export JAVA\_LIBRARY\_PATH=$HADOOP\_INSTALL/lib/native

#HADOOP VARIABLES END

* 应用配置

root@SparkMaster:~# source ~/.bashrc

* 查看Hadoop版本

root@SparkMaster:~# hadoop version

### 创建分布式文件系统的目录

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0#mkdir tmp

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0#mkdir dfs

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0#mkdir dfs/data

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0#mkdir dfs/name

或

cd /usr/local/hadoop/hadoop-2.6.0

mkdir tmp dfs dfs/name dfs/data

### 配置分布式部署描述符文件

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit core-site.xml

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit hdfs-site.xml

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit mapred-site.xml

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit yarn-site.xml

#### 1）core-site.xml

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0# cd /etc/hadoop

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# gedit core-site.xml

* 分布式

<configuration>

<property>

<name> fs.defaultFS </name>

<value>hdfs://SparkMaster:9000</value>

</property>

<property>

<name>hadoop.tmp.dir</name>

<value>file:/usr/local/hadoop/hadoop-2.6.0/tmp</value>

</property>

<property>

<name>hadoop.native.lib</name>

<value>true</value>

<description>Should native hadoop libraries, if present, be used.</description>

</property>

</configuration>

#### 2）hdfs-site.xml

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# vim hdfs-site.xml

* 分布式

<configuration>

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<property>

<name>dfs.namenode.name.dir</name>

<value>file:/usr/local/hadoop/hadoop-2.6.0/dfs/name</value>

</property>

<property>

<name>dfs.datanode.data.dir</name>

<value>file:/usr/local/hadoop/hadoop-2.6.0/dfs/data</value>

</property>

</configuration>

**注意：**

<name>dfs.replication</name>

<value>3</value>把1改为3这样数据就有了3份副本，本例中SparkMaster也充当slave参与工作。

#### 3）mapred-site.xml

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop# vim mapred-site.xml

* 分布式

<configuration>

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

<description>Execution framework set to Hadoop YARN.</description>

</property>

<property>

<name>mapred.job.tracker</name>

<value>SparkMaster:9001</value>

<description>Host or IP and port of JobTracker.</description>

</property>

</configuration>

#### 4）yarn-site.xml

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/etc/hadoop#gedit yarn-site.xml

* 分布式

<configuration>

<property>

<name>yarn.resourcemanager.hostname</name>

<value>SparkMaster</value>

</property>

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

</configuration>

### 编辑masters和slaves文件

sudo gedit /usr/local/hadoop/etc/hadoop/masters

分布式：

SparkMaster

sudo gedit /usr/local/hadoop/etc/hadoop/slaves

分布式：

SparkMaster

SparkWorker1

SparkWorker2

备注：本例把master也当作slave来用，所以把SparkMaster也加到了slaves文件里了。

## 安装Scala、Spark和IDEA

### 分别解压到相关目录

注意：如果想用Scala-2.11.6，需要下载spark-1.3.1源码进行重新编译。

解压scala-2.10.5到

usr/lib/scala/

生成

usr/lib/scala/scala-2.10.5/

解压spark-1.3.1-bin-hadoop2.6到

/usr/local/spark/

生成

/usr/local/spark/spark-1.3.1-bin-hadoop2.6/

### 编辑当前用户环境变量配置文件

root@SparkMaster:~# gedit ~/.bashrc

# for examples

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

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

export SCALA\_HOME=/usr/lib/scala/scala-2.10.5

export SPARK\_HOME=/usr/lib/spark/spark-1.3.1-bin-hadoop2.6

export CLASS\_PATH=.:${JAVA\_HOME}/lib:${JRE\_HOME}/lib

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

#HADOOP VARIABLES START

export HADOOP\_INSTALL=/usr/local/hadoop/hadoop-2.6.0

export PATH=$PATH:$HADOOP\_INSTALL/bin

export PATH=$PATH:$HADOOP\_INSTALL/sbin

export PATH=$PATH:$HADOOP\_INSTALL/etc/hadoop

export HADOOP\_MAPRED\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_HOME=$HADOOP\_INSTALL

export HADOOP\_HDFS\_HOME=$HADOOP\_INSTALL

export YARN\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_INSTALL/lib/native

export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_INSTALL/lib"

#HADOOP VARIABLES END

使环境变量生效

root@SparkMaster:~# source ~/.bashrc

### 编辑spark运行环境配置文件

root@SparkMaster:~# gedit /usr/local/spark/spark-1.3.1-bin-hadoop2.6/conf/spark-env.sh

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

export SCALA\_HOME=/usr/lib/scala/scala-2.10.5

export HADOOP\_HOME=/usr/local/hadoop/hadoop-2.6.0

export HADOOP\_CONF\_DIR=/usr/local/hadoop/hadoop-2.6.0/etc/hadoop

export SPARK\_MASTER\_IP=SparkMaster

export SPARK\_MEMORY=2g

“export SPARK\_MEMORY=2g”可与虚拟机内存大小一致

### 编辑Spark的slaves文件

gedit /usr/local/spark/spark-1.3.1-bin-hadoop2.6/conf/slaves

SparkMaster

SparkWorker1

SparkWorker2

----------–----SparkMaster作为两种角色--------------

scp /usr/local/spark/spark-1.3.1-bin-hadoop2.6/conf/slaves root@SparkWorker1:/usr/local/spark/spark-1.3.1-bin-hadoop2.6/conf/

scp /usr/local/spark/spark-1.3.1-bin-hadoop2.6/conf/slaves root@SparkWorker2:/usr/local/spark/spark-1.3.1-bin-hadoop2.6/conf/

### Idea IDEA安装

下载路径：http://www.jetbrains.com/idea/download/

安装路径：/usr/local/idea/idea-IC-141.731.2/

scala插件下载路径：http://plugins.jetbrains.com/files/1347/19130/scala-intellij-bin-1.4.15.zip

环境变量配置：

gedit ~/.bashrc

# for examples

export JAVA\_HOME=/usr/lib/java/jdk1.7.0\_79

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

export SCALA\_HOME=/usr/lib/scala/scala-2.10.5

export SPARK\_HOME=/usr/lib/spark/spark-1.3.1-bin-hadoop2.6

export CLASS\_PATH=.:${JAVA\_HOME}/lib:${JRE\_HOME}/lib

export PATH=/usr/local/idea/idea-IC-141.731.2/bin:${SPARK\_HOME}/bin:${SCALA\_HOME}/bin:${JAVA\_HOME}/bin:$PATH

#HADOOP VARIABLES START

export HADOOP\_INSTALL=/usr/local/hadoop/hadoop-2.6.0

export PATH=$PATH:$HADOOP\_INSTALL/bin

export PATH=$PATH:$HADOOP\_INSTALL/sbin

export PATH=$PATH:$HADOOP\_INSTALL/etc/hadoop

export HADOOP\_MAPRED\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_HOME=$HADOOP\_INSTALL

export HADOOP\_HDFS\_HOME=$HADOOP\_INSTALL

export YARN\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_INSTALL/lib/native

export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_INSTALL/lib"

#HADOOP VARIABLES END

**备注：这个版本的 .bashrc文件是最完整的！**

## 克隆其他slave节点

### 克隆slave节点

如果你使用的VMware虚拟机，可以使用clone的功能，克隆SparkWorker1、SparkWorker2，建议完全克隆，不是用链接模式，避免依赖。

**克隆完修改IP和主机名。**

用Ping命令测试

root@SparkMaster:/#ping SparkWorker1

ping 主机名Master、SparkWorker1、SparkWorker2

ctl+c结束

### 配置集群SSH无密码验证

**1）验证**

**备注：参考单机版SSH的配置**

root@SparkMaster:~# ssh SparkWorker1

root@SparkWorker1:~# exit

root@SparkMaster:~# cd /root/.ssh

root@SparkMaster:~/.ssh# ls

authorized authorized\_keys  id\_rsa  id\_rsa.pub known\_hosts

**2）从slave向Master上传公钥文件id\_rsa.pub**

 SparkWorker1上传公钥给SparkMaster：

root@SparkWorker1: ~#cd /root/.ssh

root@SparkWorker1:~/.ssh#ls

authorized authorized\_keys  id\_rsa  id\_rsa.pub known\_hosts

root@SparkWorker1:~/.ssh#scp id\_rsa.pub root@SparkMaster:/root/.ssh/id\_rsa.pub.SparkWorker1

id\_rsa.pub                                    100%  407    0.4KB/s   00:00

 SparkWorker2上传公钥给Master：

root@SparkWorker2:~/.ssh# scpid\_rsa.pub root@SparkMaster:/root/.ssh/id\_rsa.pub.SparkWorker2

id\_rsa.pub                                    100% 407     0.4KB/s   00:00

**3）Master组合公钥并分发**

Master上看到公钥已经传过来：

root@SparkMaster:~/.ssh# ls

authorized       id\_rsa      id\_rsa.pub.SparkWorker1  known\_hosts

authorized\_keys  id\_rsa.pub id\_rsa.pub.SparkWorker2

在Master上综合所有公钥：

root@SparkMaster:~/.ssh# cat id\_rsa.pub>>authorized\_keys

root@SparkMaster:~/.ssh# cat id\_rsa.pub.SparkWorker1>>authorized\_keys

root@SparkMaster:~/.ssh# cat id\_rsa.pub.SparkWorker2>>authorized\_keys

Master分发公钥给SparkWorker1和SparkWorker2

root@SparkMaster:~/.ssh# scp authorized\_keys root@SparkWorker1:/root/.ssh/authorized\_keys

root@SparkMaster:~/.ssh# scp authorized\_keys root@SparkWorker2:/root/.ssh/authorized\_keys

### 保持配置文件同步

如果调试过程中修改了配置文件，需要进行主从同步，需要同步的文件包括：

* Hadoop需要的：

~/.bashrc、hadoop-env.sh、yarn-env.sh、mapred-env.sh、core-site.xml、hdfs-site.xml、mapred-site.xml、yarn-site.xml、masters、slaves、hosts。

* Spark需要的：

~/.bashrc、spark-env.sh，spark目录下的slaves。

更简便的方法是使用root用户拷贝：java、hadoop（scala、spark、idea顺便带上，后面具体介绍）目录到另两台机器上。

root@SparkMaster:~# scp ~/.bashrc root@sparkworker1:/root/.bashrc

root@SparkMaster:~# scp -r /usr/lib/java root@sparkworker1:/usr/lib/

root@SparkMaster:~# scp -r /usr/local/hadoop root@sparkworker1:/usr/local/

root@SparkMaster:~# scp -r /usr/lib/scala root@sparkworker1:/usr/lib/

root@SparkMaster:~# scp -r /usr/local/spark root@sparkworker1:/usr/local/

root@SparkMaster:~# scp -r /usr/local/idea root@sparkworker1:/usr/local/

**sparkworker2同上**

# Hadoop和Spark集群测试

注意：spark1.3.1（spark-1.3.1-bin-hadoop2.6）需要使用scala2.10.x版本。

如果想使用最新的scala2.11.6需要下载spark-1.3.1.tgz，并重新编译，再使用。

## 启动hadoop分布式集群

* **格式化集群文件系统**

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin#./hdfs namenode -format

或root@SparkMaster:/# hdfs namenode -format

15/05/01 18:37:29 INFO namenode.NameNode: STARTUP\_MSG:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

STARTUP\_MSG: Starting NameNode

STARTUP\_MSG: host = SparkMaster/192.168.136.100

STARTUP\_MSG: args = [-format]

。。。。。。

STARTUP\_MSG: version = 2.6.0Re-format filesystem in Storage Directory /usr/local/hadoop/hadoop-2.6.0/dfs/name ? (Y or N) Y

15/05/01 18:37:33 INFO namenode.FSImage: Allocated new BlockPoolId: BP-77366057-192.168.136.100-1430476653791

15/05/01 18:37:33 INFO common.Storage: Storage directory /usr/local/hadoop/hadoop-2.6.0/dfs/name has been successfully formatted.

15/05/01 18:37:33 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0

15/05/01 18:37:33 INFO util.ExitUtil: Exiting with status 0

15/05/01 18:37:33 INFO namenode.NameNode: SHUTDOWN\_MSG:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SHUTDOWN\_MSG: Shutting down NameNode at SparkMaster/192.168.136.100

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

* **启动hadoop服务**

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/sbin# ./start-dfs.sh

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/sbin# jps

3218 DataNode

4758 Jps

3512 SecondaryNameNode

4265 NodeManager

3102 NameNode

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/sbin# ./start-yarn.sh

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/sbin# jps

3218 DataNode

4758 Jps

3512 SecondaryNameNode

4265 NodeManager

3102 NameNode

4143 ResourceManager

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/sbin# ./mr-jobhistory-daemon.sh start historyserver

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/sbin# jps

4658 JobHistoryServer

3218 DataNode

4758 Jps

3512 SecondaryNameNode

4265 NodeManager

3102 NameNode

4143 ResourceManager

* **典型故障排除**

root@SparkMaster:~# stop-all.sh

报错：

SparkMaster: stopping tasktracker

SparkWorker2: stopping tasktracker

SparkWorker1: stopping tasktracker

stopping namenode

Master: stopping datanode

SparkWorker2: no datanode tostop

SparkWorker1: no datanode tostop

Master: stopping secondarynamenode

**解决：**

* + **清空以下目录中的所有内容**

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0#rm -rf tmp/\*

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0#rm -rf dfs/data/\*

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0#rm -rf dfs/name/\*

* + **格式化和启动集群**

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin# hadoop namenode –format

…………

Re-format filesystem in /usr/local/hadoop/hadoop-2.6.0/hdfs/name? (Y or N) Y（\*\*\*此处一定要用大写的Y，否则无法格式化）

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

* + 重新启动hadoop服务

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/sbin#./start-dfs.sh

[root@SparkMaster](mailto:root@SparkMaster):/usr/local/hadoop/hadoop-2.6.0/sbin#./start-yarn.sh

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/sbin#./mr-jobhistory-daemon.sh start historyserver

（停止historyserver：sudo mr-jobhistory-daemon.sh stop historyserver）

root@SparkMaster:~#start-all.sh(可不用启动)

* **看一下各节点的运行状况：**

root@SparkMaster:~# hdfs dfsadmin -report

Configured Capacity: 53495648256 (49.82 GB)

Present Capacity: 29142274048 (27.14 GB)

DFS Remaining: 29141831680 (27.14 GB)

DFS Used: 442368 (432 KB)

DFS Used%: 0.00%

Under replicated blocks: 0

Blocks with corrupt replicas: 0

Missing blocks: 0

-------------------------------------------------

Live datanodes (3):

Name: 192.168.136.102:50010 (SparkWorker2)

Hostname: SparkWorker2

Decommission Status : Normal

Configured Capacity: 17831882752 (16.61 GB)

DFS Used: 147456 (144 KB)

Non DFS Used: 8084967424 (7.53 GB)

DFS Remaining: 9746767872 (9.08 GB)

DFS Used%: 0.00%

DFS Remaining%: 54.66%

Configured Cache Capacity: 0 (0 B)

Cache Used: 0 (0 B)

Cache Remaining: 0 (0 B)

Cache Used%: 100.00%

Cache Remaining%: 0.00%

Xceivers: 1

Last contact: Fri May 01 22:13:37 CST 2015

Name: 192.168.136.101:50010 (SparkWorker1)

Hostname: SparkWorker1

Decommission Status : Normal

Configured Capacity: 17831882752 (16.61 GB)

DFS Used: 147456 (144 KB)

Non DFS Used: 7672729600 (7.15 GB)

DFS Remaining: 10159005696 (9.46 GB)

DFS Used%: 0.00%

DFS Remaining%: 56.97%

Configured Cache Capacity: 0 (0 B)

Cache Used: 0 (0 B)

Cache Remaining: 0 (0 B)

Cache Used%: 100.00%

Cache Remaining%: 0.00%

Xceivers: 1

Last contact: Fri May 01 22:13:37 CST 2015

Name: 192.168.136.100:50010 (SparkMaster)

Hostname: SparkMaster

Decommission Status : Normal

Configured Capacity: 17831882752 (16.61 GB)

DFS Used: 147456 (144 KB)

Non DFS Used: 8595677184 (8.01 GB)

DFS Remaining: 9236058112 (8.60 GB)

DFS Used%: 0.00%

DFS Remaining%: 51.80%

Configured Cache Capacity: 0 (0 B)

Cache Used: 0 (0 B)

Cache Remaining: 0 (0 B)

Cache Used%: 100.00%

Cache Remaining%: 0.00%

Xceivers: 1

Last contact: Fri May 01 22:13:37 CST 2015

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*至此，分布式Hadoop集群构建完毕\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## 启动Spark分布式集群

root@SparkMaster:/usr/local/spark/spark-1.3.1-bin-hadoop2.6/sbin# ./start-all.sh

starting org.apache.spark.deploy.master.Master, logging to /usr/local/spark/spark-1.3.1-bin-hadoop2.6/sbin/../logs/spark-root-org.apache.spark.deploy.master.Master-1-SparkMaster.out

SparkMaster: starting org.apache.spark.deploy.worker.Worker, logging to /usr/local/spark/spark-1.3.1-bin-hadoop2.6/sbin/../logs/spark-root-org.apache.spark.deploy.worker.Worker-1-SparkMaster.out

SparkWorker1: starting org.apache.spark.deploy.worker.Worker, logging to /usr/local/spark/spark-1.3.1-bin-hadoop2.6/sbin/../logs/spark-root-org.apache.spark.deploy.worker.Worker-1-SparkWorker1.out

SparkWorker2: starting org.apache.spark.deploy.worker.Worker, logging to /usr/local/spark/spark-1.3.1-bin-hadoop2.6/sbin/../logs/spark-root-org.apache.spark.deploy.worker.Worker-1-SparkWorker2.out

root@SparkMaster:/usr/local/spark/spark-1.3.1-bin-hadoop2.6/sbin# jps

13018 Master

11938 NameNode

12464 ResourceManager

13238 Worker

13362 Jps

12601 NodeManager

12296 SecondaryNameNode

12101 DataNode

10423 JobHistoryServer

root@SparkWorker1:~# jps

5344 NodeManager

5535 Worker

5634 Jps

5216 DataNode

root@SparkWorker2:~# jps

4946 NodeManager

5246 Jps

5137 Worker

4818 DataNode

root@SparkMaster:/usr/local/spark/spark-1.3.1-bin-hadoop2.6/bin# ./spark-shell

15/05/01 19:12:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

15/05/01 19:12:24 INFO spark.SecurityManager: Changing view acls to: root

15/05/01 19:12:24 INFO spark.SecurityManager: Changing modify acls to: root

15/05/01 19:12:24 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(root); users with modify permissions: Set(root)

15/05/01 19:12:24 INFO spark.HttpServer: Starting HTTP Server

15/05/01 19:12:24 INFO server.Server: jetty-8.y.z-SNAPSHOT

15/05/01 19:12:24 INFO server.AbstractConnector: Started SocketConnector@0.0.0.0:42761

15/05/01 19:12:24 INFO util.Utils: Successfully started service 'HTTP class server' on port 42761.

Welcome to

\_\_\_\_ \_\_

/ \_\_/\_\_ \_\_\_ \_\_\_\_\_/ /\_\_

\_\ \/ \_ \/ \_ `/ \_\_/ '\_/

/\_\_\_/ .\_\_/\\_,\_/\_/ /\_/\\_\ version 1.3.1 /\_/

Using Scala version 2.10.4 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0\_79)

Type in expressions to have them evaluated.

。。。。。。。。。。。。。。。。

scala>

root@SparkMaster:~# jps

13391 SparkSubmit

13018 Master

11938 NameNode

12464 ResourceManager

13238 Worker

13570 Jps

12601 NodeManager

12296 SecondaryNameNode

12101 DataNode

10423 JobHistoryServer

root@SparkMaster:~#

## 服务启动后启动webUI

[http://sparkmaster:50070](http://sparkmaster:50070/)

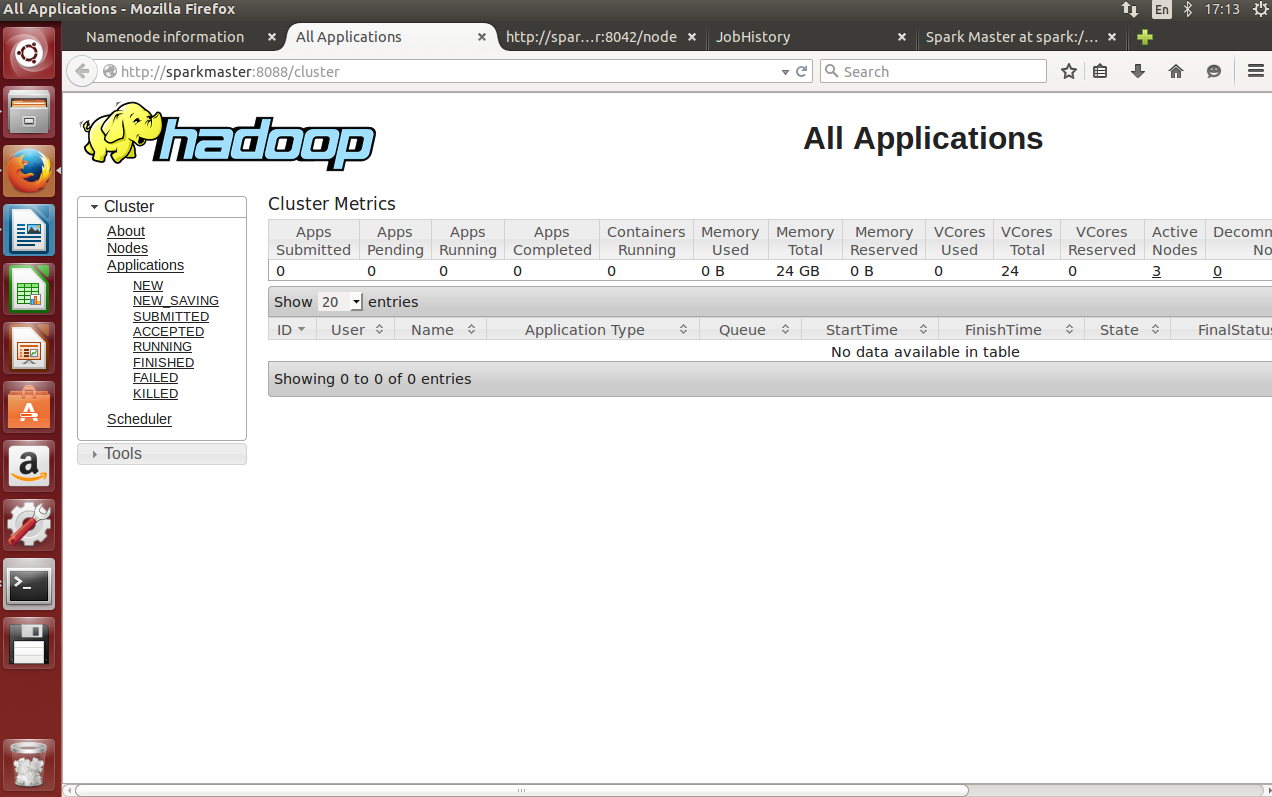
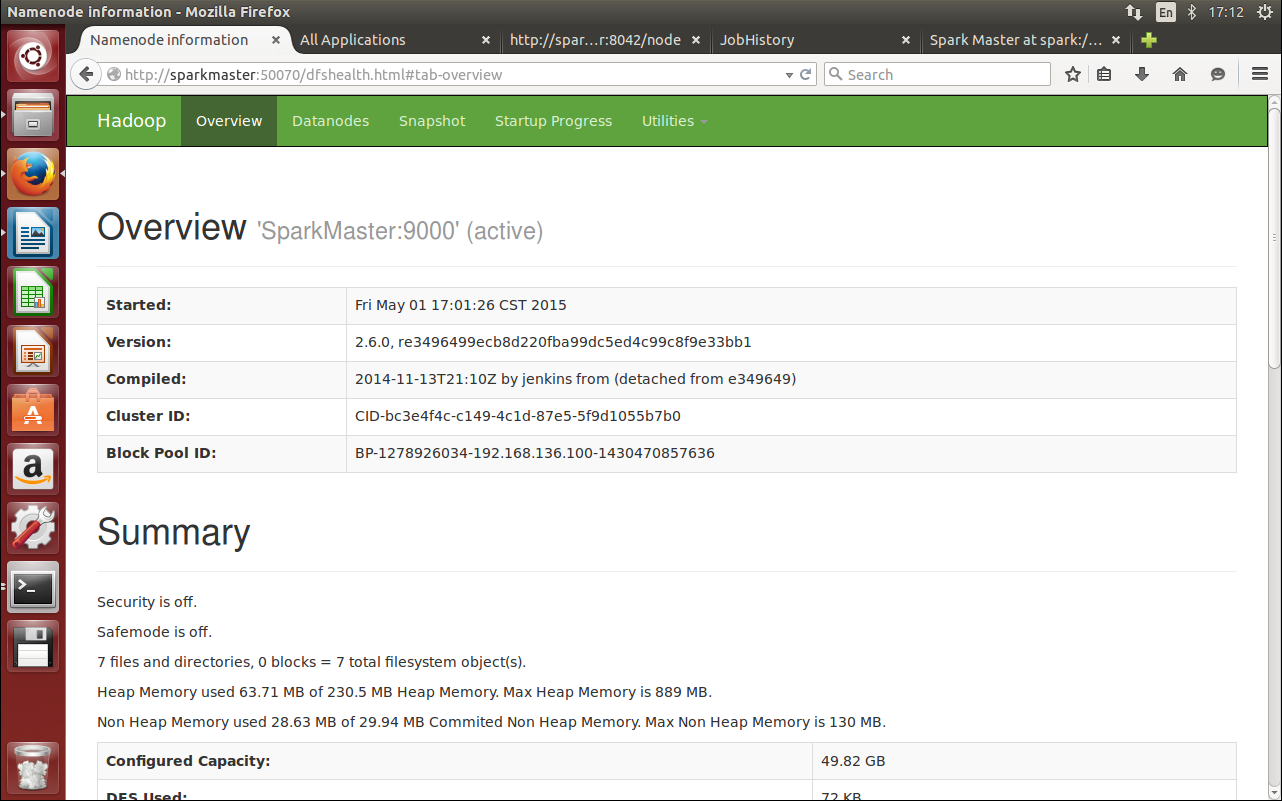
[http://sparkmaster:8088](http://sparkmaster:8088/)

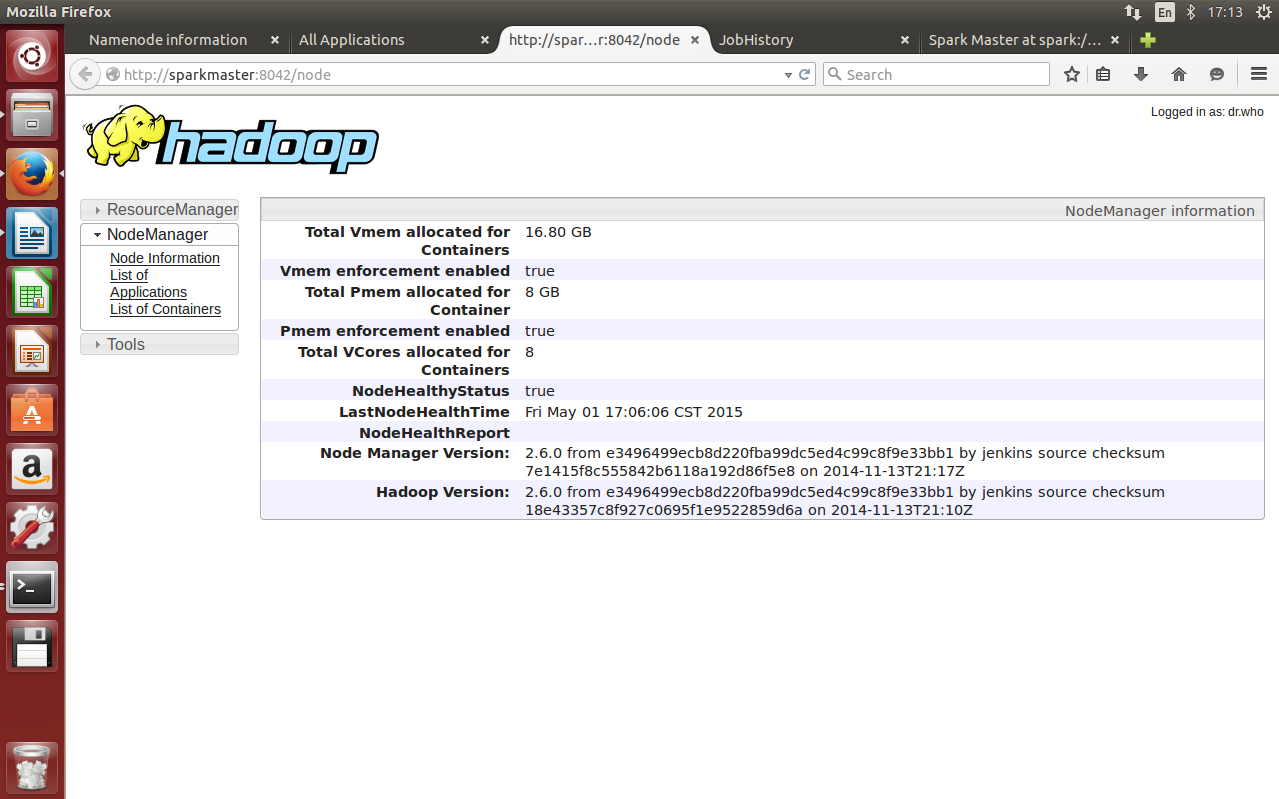
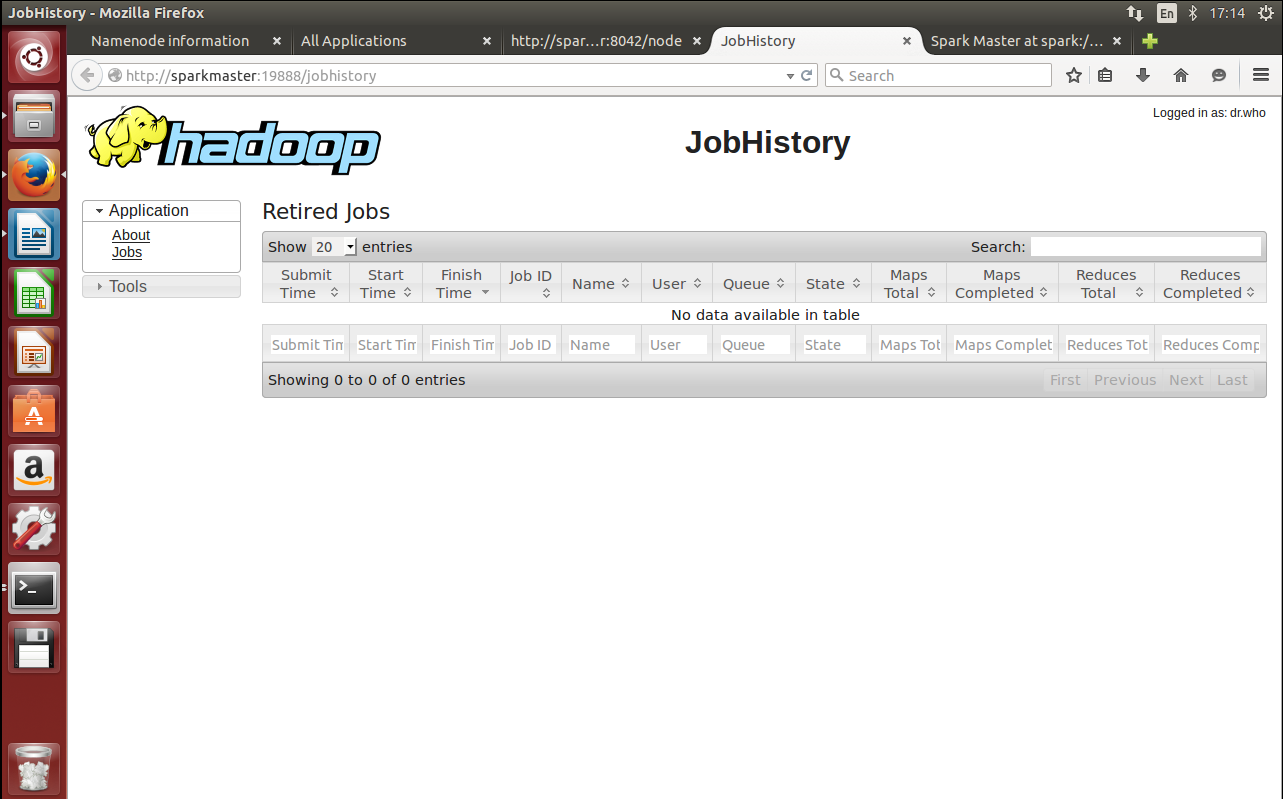
[http://sparkmaster:8042](http://sparkmaster:8042/)

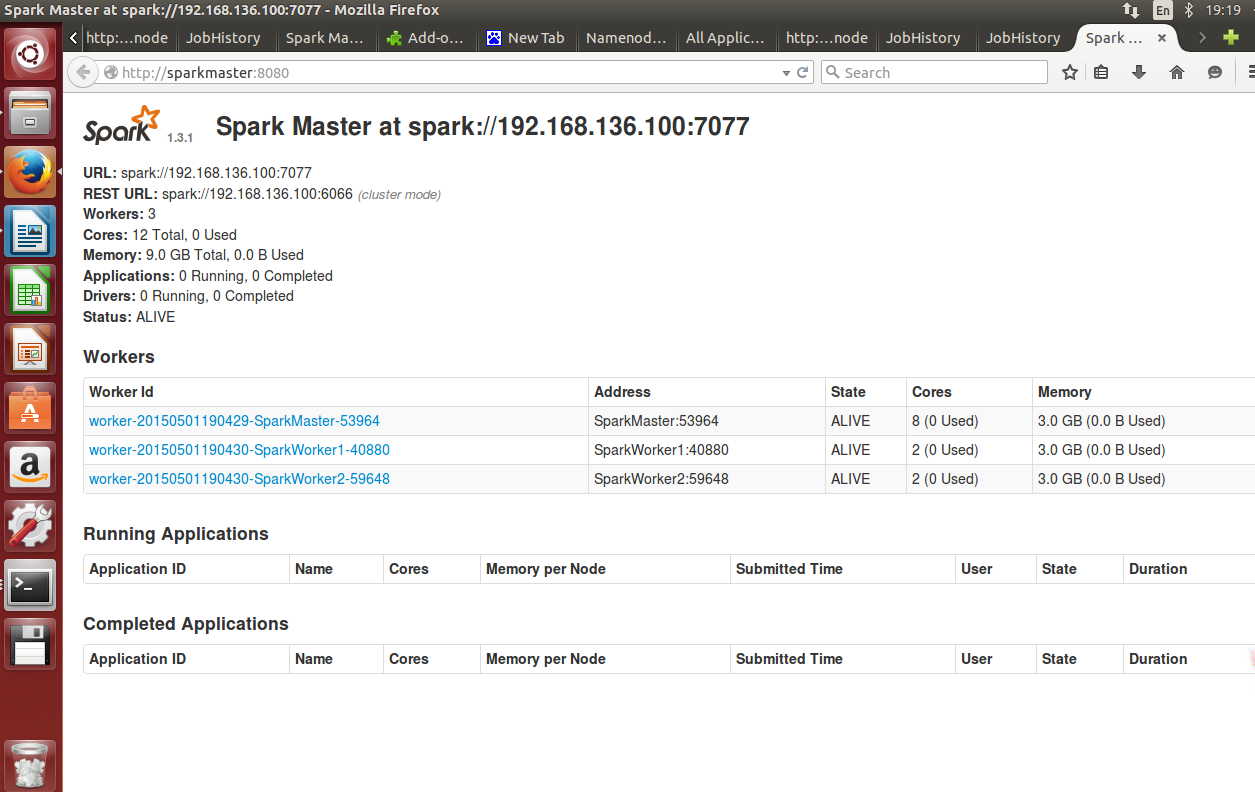
<http://sparkmaster:19888/>

<http://sparkmaster:8080/>

[http://sparkmaster:4040](http://sparkmaster:4040/)





## 在Hadoop分布式集群中运行wordcount示例

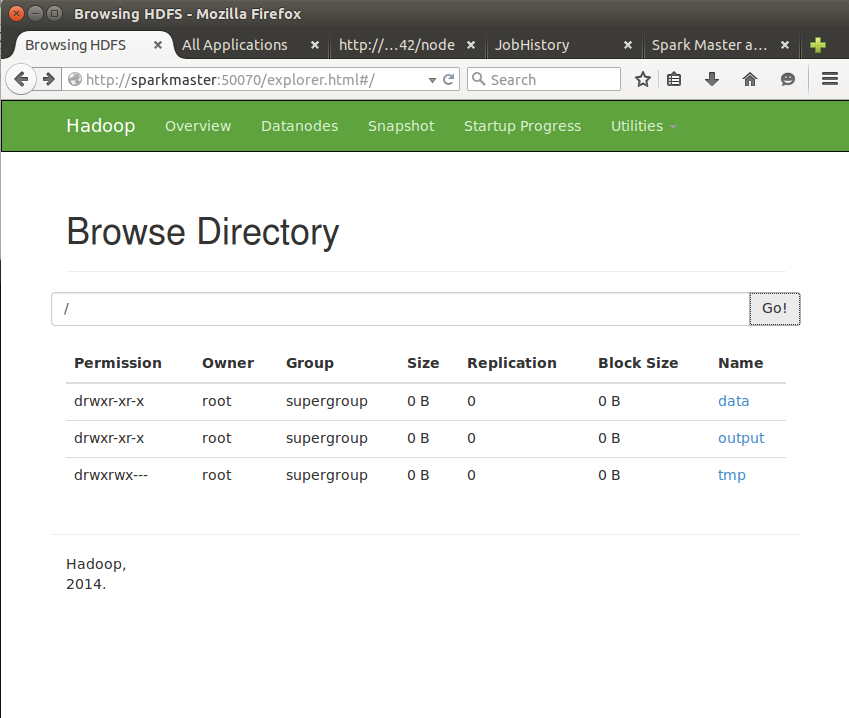
* **准备Hadoop分布式文件目录**

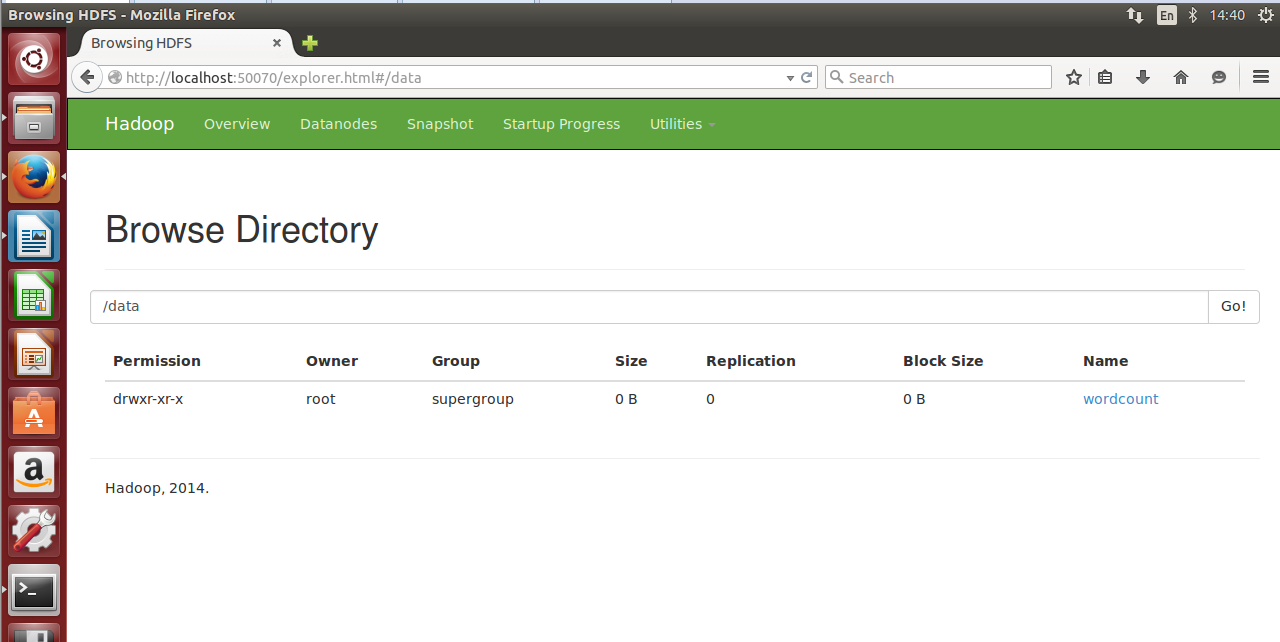
首先在hdfs文件系统上创建两个目录，wordcount用于存放准备单词级数的文件，output用于存放结果。

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin# hadoop fs -mkdir -p /data/wordcount

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin# hadoop fs -mkdir -p /output/

备注：新版本建议用hdfs dfs替代hadoop fs

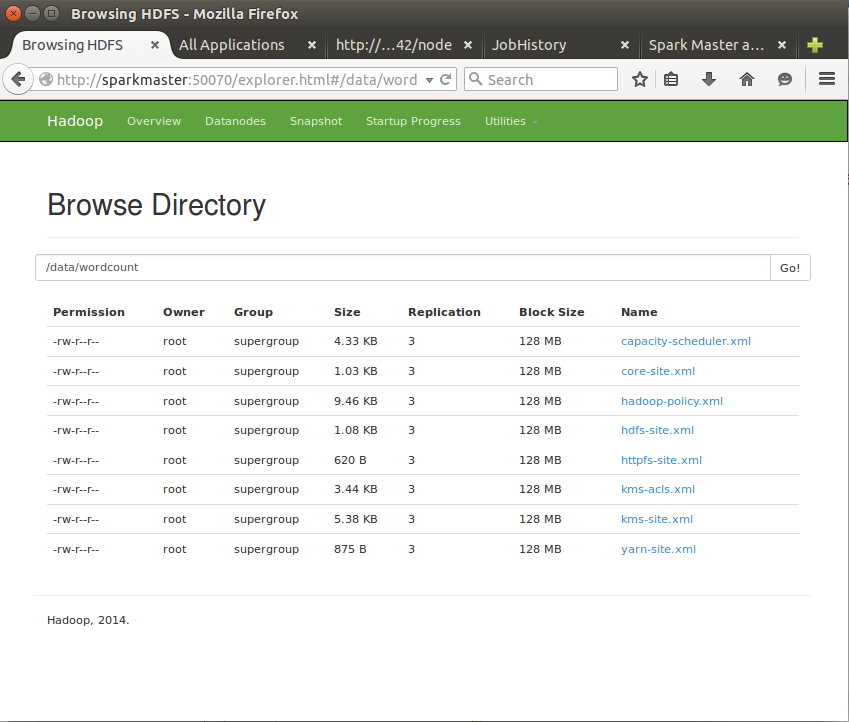




* **向分布式文件目录中拷贝文件**

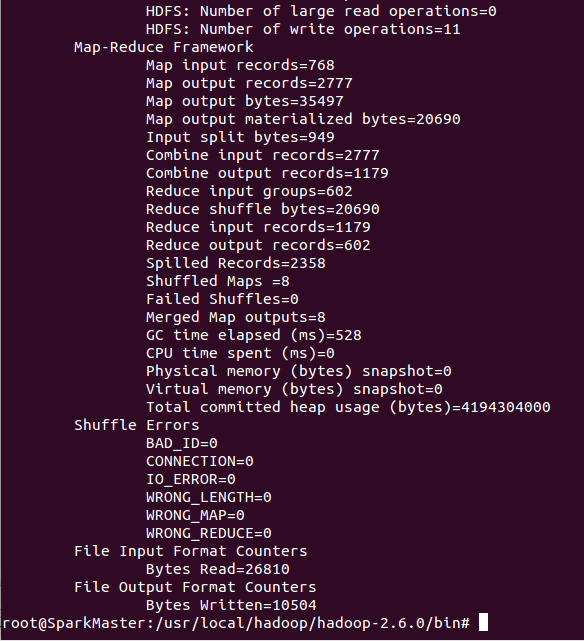
把hadoop的etc/hadoop目录下的所有xml文件放到wordcount中。

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin# hadoop fs -put ../etc/hadoop/\*.xml /data/wordcount/



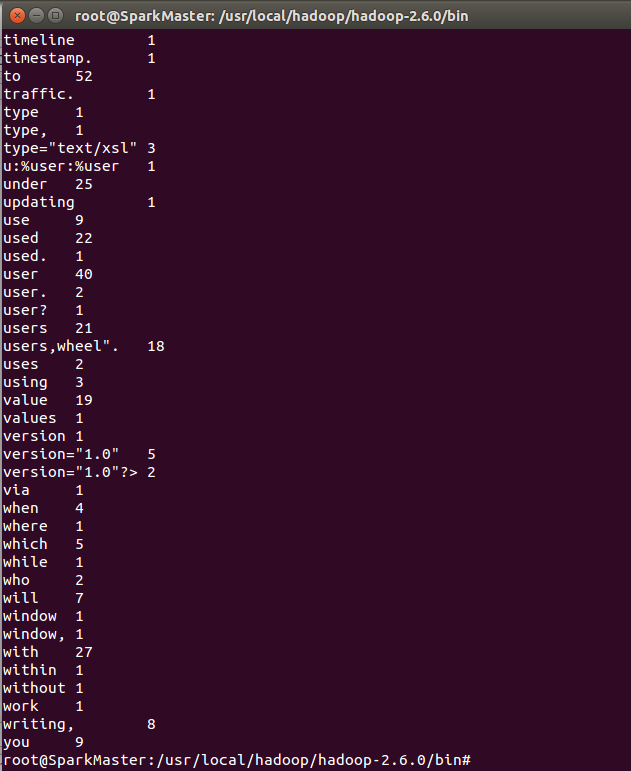
* **执行wordcount算例**

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin# hadoop jar ../share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar wordcount /data/wordcount /output/wordcount



* **输出结果**

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin# hadoop fs -cat /output/wordcount/\*



* **重新执行算例**

马上重新运行该示例会报错，删掉output下的wordcount目录即可，具体如下：

* 查看hdfs根目录：

注：新版Hadoop建议用hdfs dfs ……代替hadoop fs ……

由于配置的路径环境变量，以下命令可以在任何路径下直接使用。

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin#./hdfs dfs -ls /

或

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin#./hadoop fs -ls /

Found 3 items

drwxr-xr-x - root supergroup 0 2015-05-01 19:45 /data

drwxr-xr-x - root supergroup 0 2015-05-01 20:24 /output

drwxrwx--- - root supergroup 0 2015-05-01 18:51 /tmp

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin# hdfs fs -ls /output

Found 1 items

drwxr-xr-x - root supergroup 0 2015-05-01 20:47 /output/wordcount

* **先删掉/output/wordcount目录**

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin# hdfs fs -rm -r /output/wordcount

* **再次运行示例**

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/bin# hadoop jar ../share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar wordcount /data/wordcount /output/wordcount

关闭hadoop

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0/bin# stop-all.sh

备注：如需重新运行应用需删除output目录及文件

root@lolo-virtual-machine:/usr/local/hadoop/hadoop-2.6.0#hadoop dfs -rm output

## 在Spark分布式集群中运行wordcount示例

root@SparkMaster:/usr/local/spark/spark-1.3.1-bin-hadoop2.6# hadoop fs -put README.md /data/

scala> val file = sc.textFile("hdfs://SparkMaster:9000/data/README.md")

15/05/01 21:23:28 INFO storage.MemoryStore: ensureFreeSpace(182921) called with curMem=0, maxMem=278302556

15/05/01 21:23:28 INFO storage.MemoryStore: Block broadcast\_0 stored as values in memory (estimated size 178.6 KB, free 265.2 MB)

15/05/01 21:23:28 INFO storage.MemoryStore: ensureFreeSpace(25373) called with curMem=182921, maxMem=278302556

15/05/01 21:23:28 INFO storage.MemoryStore: Block broadcast\_0\_piece0 stored as bytes in memory (estimated size 24.8 KB, free 265.2 MB)

15/05/01 21:23:28 INFO storage.BlockManagerInfo: Added broadcast\_0\_piece0 in memory on localhost:42086 (size: 24.8 KB, free: 265.4 MB)

15/05/01 21:23:28 INFO storage.BlockManagerMaster: Updated info of block broadcast\_0\_piece0

15/05/01 21:23:28 INFO spark.SparkContext: Created broadcast 0 from textFile at <console>:21

file: org.apache.spark.rdd.RDD[String] = hdfs://SparkMaster:9000/data/README.md MapPartitionsRDD[1] at textFile at <console>:21

scala> val count = file.flatMap(line => line.split(" ")).map(word => (word,1)).reduceByKey(\_+\_)

15/05/01 21:23:45 INFO mapred.FileInputFormat: Total input paths to process : 1

count: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <console>:23

scala> count.collect

15/05/01 21:24:25 INFO spark.SparkContext: Starting job: collect at <console>:26

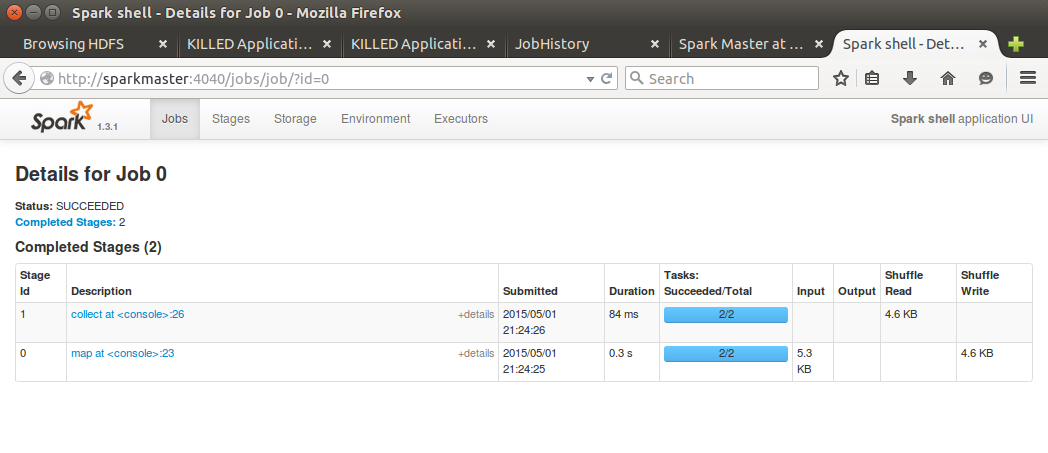
15/05/01 21:24:25 INFO scheduler.DAGScheduler: Registering RDD 3 (map at <console>:23)

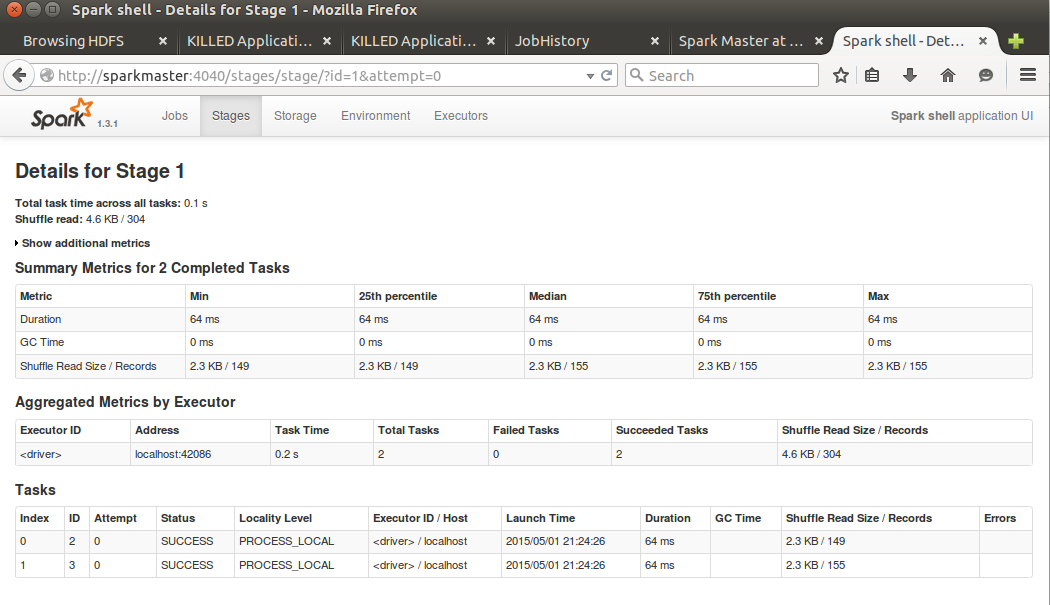
15/05/01 21:24:25 INFO scheduler.DAGScheduler: Got job 0 (collect at <console>:26) with 2 output partitions (allowLocal=false)

……………

res0: Array[(String, Int)] = Array((package,1), (this,1), (Because,1), (Python,2), (cluster.,1), (its,1), ([run,1), (general,2), (YARN,,1), (have,1), (pre-built,1), (locally.,1), (changed,1), (locally,2), (sc.parallelize(1,1), (only,1), (several,1), (This,2), (basic,1), (first,1), (documentation,3), (Configuration,1), (learning,,1), (graph,1), (Hive,2), (["Specifying,1), ("yarn-client",1), (page](http://spark.apache.org/documentation.html),1), ([params]`.,1), (application,1), ([project,2), (prefer,1), (SparkPi,2), (<http://spark.apache.org/>,1), (engine,1), (version,1), (file,1), (documentation,,1), (MASTER,1), (example,3), (are,1), (systems.,1), (params,1), (scala>,1), (provides,1), (refer,2), (configure,1), (Interactive,2), (distribution.,1), (can,6), (build,3), (when,1), (Apache,1), ...

scala>





# 附录

## 附录一：64位Ubuntu Linux虚拟机中手动安装或升级 VMware Tools

对于 Linux 虚拟机，您可以使用命令行工具手动安装或升级 VMware Tools。在升级 VMware Tools 前，请考察运行虚拟机的环境，并权衡不同升级策略的利弊。例如，您可以安装最新版本的 VMware Tools 以增强虚拟机的客户机操作系统的性能并改进虚拟机管理，也可以继续使用现有版本以在所处环境中提供更大的灵活性。

前提条件

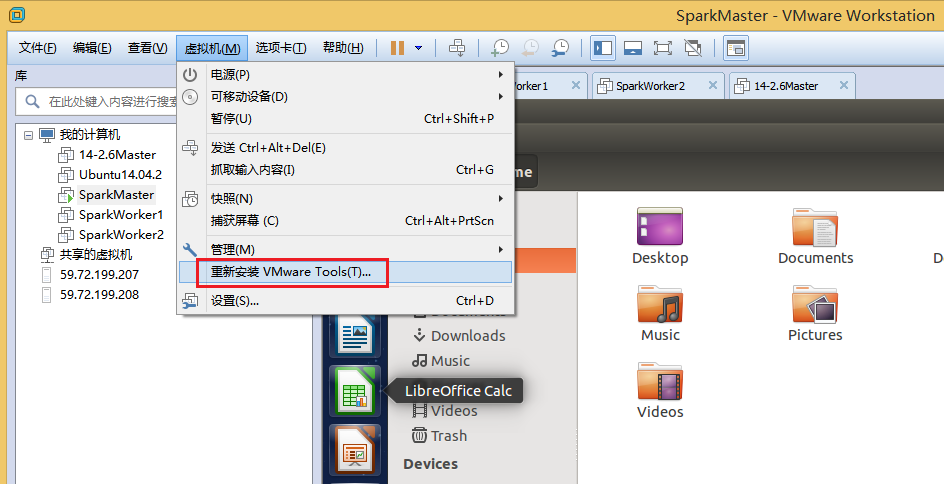
■ 打开虚拟机电源。

■ 确认客户机操作系统正在运行。

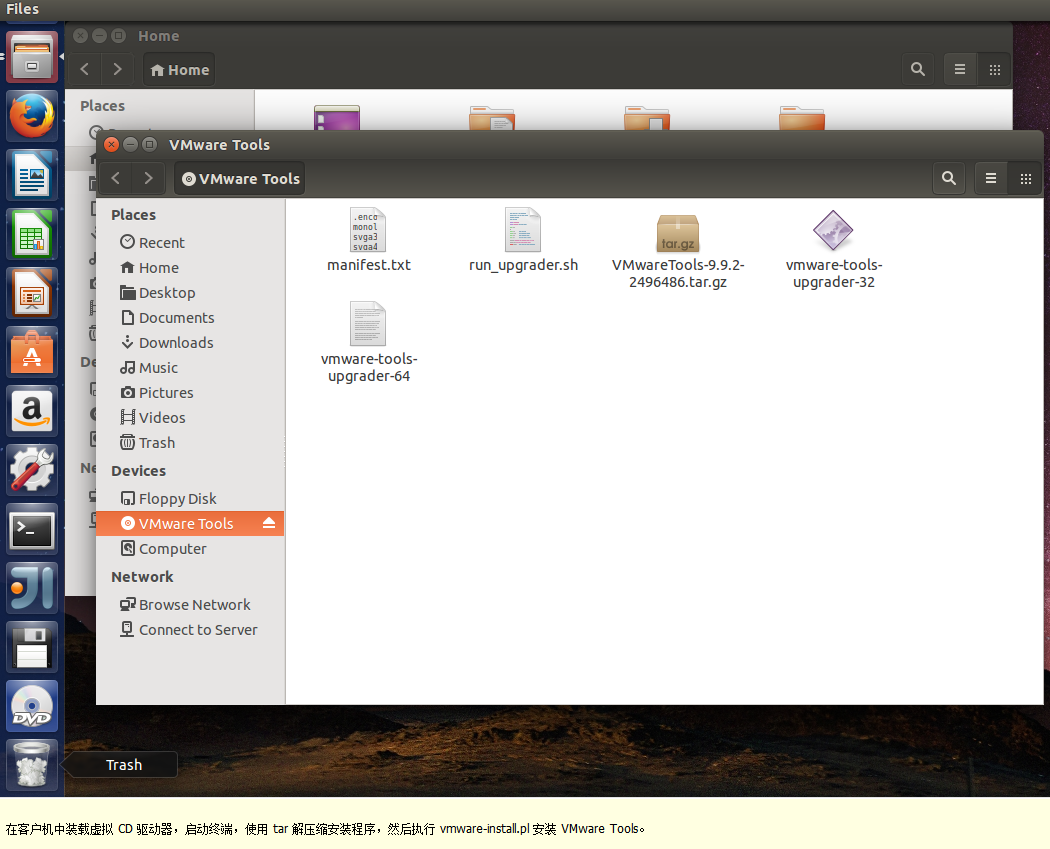
■ 由于 VMware Tools 安装程序是采用 Perl 语言编写的，因此请确认客户机操作系统中已安装 Perl

**方法一、图形化界面安装：**

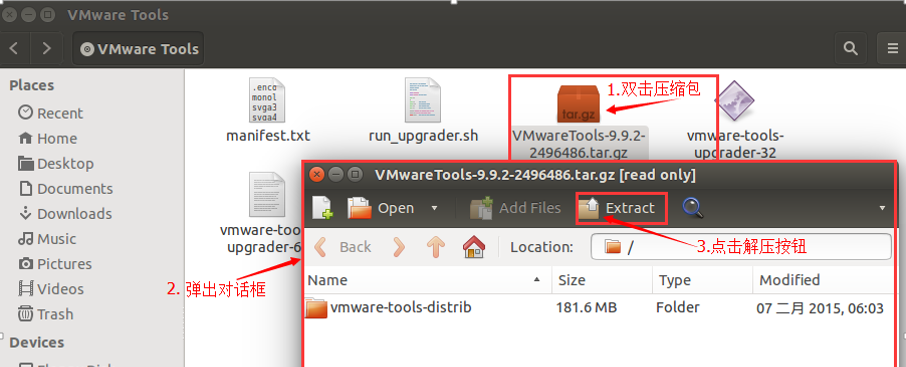
**1.载入vmware tools光盘镜像**

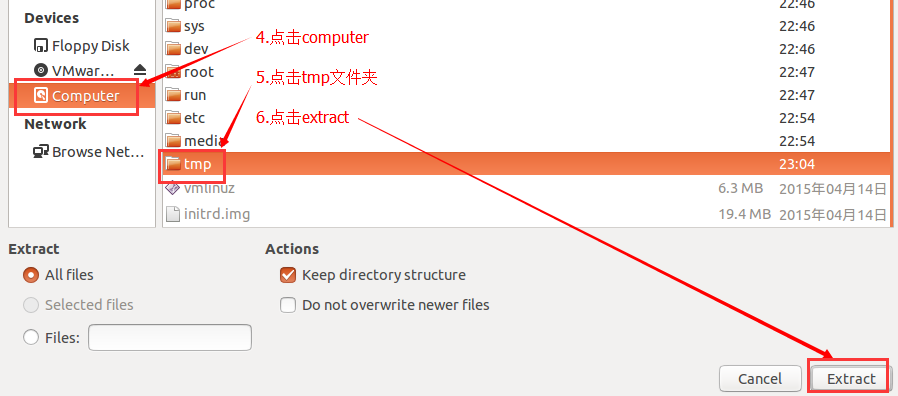


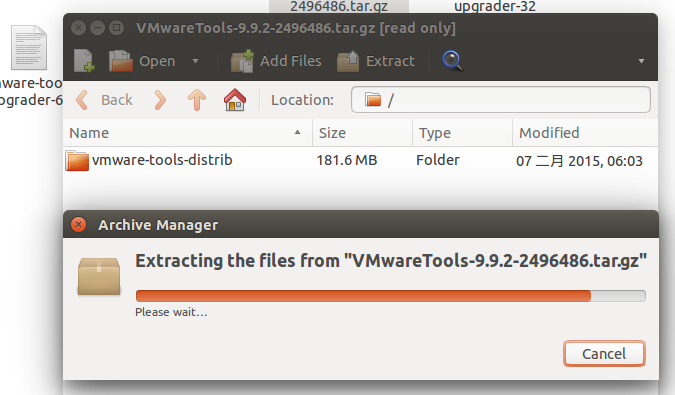
系统自动装载vmware tools光盘，并弹出窗口。

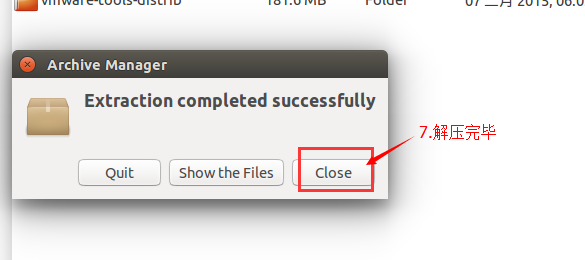


**2.解压安装包**









**3.安装vmware tools软件**

执行如下命令：

sudo /tmp/vmware-tools-distrib/vmware-install.pl

一路默认，就OK了！

**方法二、命令行安装方式**

步骤

1 在主机上，从 Workstation 菜单栏中选择虚拟机 > 安装 VMware Tools。如果安装了早期版本的 VMware Tools，则菜单项为更新 VMware Tools。

2 在虚拟机中，以 root 身份登录客户机操作系统，然后打开终端窗口。

3 不带参数运行 mount 命令，以确定 Linux 发行版是否自动装载 VMware Tools 虚拟 CD-ROM 映像。

如果装载了 CD-ROM 设备，将按如下方式列出 CD-ROM 设备及其装载点：/dev/cdrom on /mnt/cdrom type iso9660 (ro,nosuid,nodev)

4 如果未装载 VMware Tools 虚拟 CD-ROM 映像，请装载 CD-ROM 驱动器。

a 如果装载点目录尚不存在，请创建该目录。

mkdir /mnt/cdrom

某些 Linux 发行版使用不同的装载点名称。例如，某些发行版上的装载点是 /media/VMware Tools 而不是 /mnt/cdrom。请修改该命令以反映您的发行版使用的约定。

b 装载 CD-ROM 驱动器。

mount /dev/cdrom /mnt/cdrom

某些 Linux 发行版使用不同的设备名称，或者以不同的方式组织 /dev 目录。如果 CD-ROM 驱动器不是 /dev/cdrom 或 CD-ROM 装载点不是 /mnt/cdrom，则必须修改该命令以反映您的发行版使用的约定。

5 转到工作目录，例如 /tmp。

cd /tmp

6 安装 VMware Tools 之前，删除先前的 vmware-tools-distrib 目录。

该目录的位置取决于先前安装时存储的位置。通常，该目录位于 /tmp/vmware-tools-distrib。

7 列出装载点目录的内容，并记下 VMware Tools tar 安装程序的文件名。

ls mount-point

8 解压缩安装程序。

tar zxpf /mnt/cdrom/VMwareTools-x.x.x-yyyy.tar.gz

x.x.x 值是产品版本号，yyyy 是产品发行版本的内部版本号。

如果您尝试安装 tar 安装以覆盖 RPM 安装或相反，安装程序将检测以前的安装，并且必须转换安装程序数据库格式后才能继续操作。

9 如果需要，请卸载 CD-ROM 映像。

umount /dev/cdrom

如果 Linux 发行版自动装载 CD-ROM，则不需要卸载该映像。

10 运行安装程序并配置 VMware Tools。

cd vmware-tools-distrib

./vmware-install.pl

通常情况下，运行完安装程序文件之后会运行 vmware-config-tools.pl 配置文件。

11 如果默认值符合您的配置，则请按照提示接受默认值。

12 按照脚本末尾的说明操作。

视所用的功能而定，这些说明可能包括重新启动 X 会话、重新启动网络连接、重新登录以及启动 VMware 用户进程。或者，也可以重新引导客户机操作系统以完成所有这些任务。

## 附录二：FTP工具Win-SCP

如果自己改过vsftpd配置文件，错误的配置文件会导致vsftpd无法启动。可以先尝试彻底删除vsftpd，然后重新安装，用缺省的vsftpd配置文件。

* 删除vsftpd

sudo apt-get purge vsftpd

* 重新安装

sudo apt-get install vsftpd

* 查看服务

ps -ef |grep vsftpd

最后一条命令应该可以看到这样的结果：

root@SparkWorker1:~# ps -ef |grep vsftpd

root 1312 1 0 15:34 ? 00:00:00 /usr/sbin/vsftpd    <--看到这个就说明vsftpd起来了  
root 3503 2708 0 17:43 pts/7 00:00:00 grep --color=auto vsftpd

* 修改配置文件vsftpd.conf

先备份配置文件

sudo cp /etc/vsftpd.conf /etc/vsftpd.conf.old

修改配置文件

gedit /etc/vsftpd.conf

把文件中的

# write\_enable=YES的注释去掉变为：

write\_enable=YES

允许上传文件，其他配置不变，这是最简单的ftp配置，可实现文件的上传和下载，使用win-scp工具，使用最开始安装操作系统的用户“lolo”可以查看和上传文件，这里主要用这个功能向linux里传文件。如果需要更多的安全性就需要配置其他的内容，略。

* 重新启动vsftpd

sudo /sbin/service vsftpd restart

用lolo用户登陆即可

* 使用Win-SCP连接Ftp服务器

## 附录三：SecureCRT SSH登陆管理

在Windows中使用SecureCRT的SSH2协议连接到Ubuntu Linux上进行远程管理，需要停止防火墙

* **关闭防火墙命令：**

root@SparkMaster:~# sudo ufw disable

使用安装时建立用户“lolo”登陆，root用户默认不允许使用ssh远程连接。

* **打开防火墙命令：（打开后ssh登陆不上了，除非做访问控制列表策略）**

root@SparkMaster:~# sudo ufw enable

首先添加root用户的密码，然后编辑/etc/ssh/sshd\_config。注释了这句PermitRootLogin without-password，然后在这句下面添加如下这句：PermitRootLogin yes。最后重启ssh即可实现root用户使用ssh登录。（没成功，还没细研究）

修改/etc/ssh/sshd\_config文件.

将其中的PermitRootLogin no修改为yes

PubkeyAuthentication yes修改为no

AuthorizedKeysFile .ssh/authorized\_keys前面加上#屏蔽掉，

PasswordAuthentication no修改为yes就可以了。

## 附录四：Ubuntu下火狐浏览器安装Flash及书签使用相关事项

备注：百度云盘上传文件时需要用到Flash插件。

1. 下载tar包：

http://get.adobe.com/flashplayer/

下载到一个目录内，解压。会出现三个文件或目录：

libflashplayer.so

readme.txt

usr（目录）

根据readme.txt说明：

1. 安装插件

要把libflashplayer.so这个文件拷贝到浏览器插件目录下

火狐的插件目录为：/usr/lib/mozilla/plugins/

在解压后的目录下，执行命令:

sudo cp libflashplayer.so /usr/lib/mozilla/plugins/

sudo cp -r usr/\* /usr/

这样就安装好了。

1. 从其他浏览器导入书签

打开firefox浏览器，在menubars里找到Bookmarks选项，点击第一项Show all bookmarks:

找到import and backup选项，选择最后一项，import data from other browers，就可以导入书签了。

下面到了同步的阶段，在选项栏里tools选项，选择sync now，后面按步骤操作就行了。

## 附录五：Hadoop2.6.0在Ubuntu14.04.2 64位系统中使用的编译方法

备注：

判断是否都是64位的hadoop，可用“file”命令查看

root@SparkMaster:/usr/local/hadoop/hadoop-2.6.0/lib/native# file libhadoop.so.1.0.0

libhadoop.so.1.0.0: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, BuildID[sha1]=2bf804e2565fd12f70c8beba1e875f73b5ea30f9, not stripped

如上所示已经是64位的就不需要编译了，经过验证目前的官方发布的hadoop2.6.0已经是64位的，不需要编译了。下面的方法你可以略过。

如果你对hadoop进行了源码修改，那就需要进行编译，下面的方法还可以看，期待会用到O(∩\_∩)0！

**1、安装JDK，我这里使用的是OpenJDK**

**（如果你使用的是官方的jdk1.7就不用安装了）**

sudo apt-get install default-jdk

注意：如果安装了其他版本的JDK，需要修改~/.bashrc文件，修改JAVA\_HOME的路径为：

export JAVA\_HOME=/usr/lib/jvm/java-7-openjdk-amd64

java -version

显示版本信息：

java version "1.7.0\_79"

OpenJDK Runtime Environment (IcedTea 2.5.5) (7u79-2.5.5-0ubuntu0.14.04.2)

OpenJDK 64-Bit Server VM (build 24.79-b02, mixed mode)

**2、安装maven**

sudo apt-get install maven

mvn –version

或mvn --version

显示版本信息：

Apache Maven 3.0.5

Maven home: /usr/share/maven

Java version: 1.7.0\_79, vendor: Oracle Corporation

Java home: /usr/lib/jvm/java-7-openjdk-amd64/jre

Default locale: en\_US, platform encoding: UTF-8

OS name: "linux", version: "3.16.0-30-generic", arch: "amd64", family: "unix"

**3、安装openssh**

sudo apt-get install openssh-server

**4、安装依赖库**

sudo apt-get install g++ autoconf automake libtool cmake zlib1g-dev pkg-config libssl-dev

**5、安装protoc**

sudo apt-get install protobuf-compiler

protoc --version

显示版本信息:

libprotoc 2.5.0

**6、开始编译**

进入HADOOP源代码目录 hadoop-2.6.0-src，执行：

mvn clean package -Pdist,native -DskipTests -Dtar

好了，经过漫长等待， 应该就能得到编译好的结果了。

**7、编译好的文件放在：**

/usr/local/hadoop/hadoop-2.6.0-src/hadoop-dist/target/hadoop-2.6.0目录中

另外有一个编译好的压缩包**hadoop-2.6.0.tar.gz**在

/usr/local/hadoop/hadoop-2.6.0-src/hadoop-dist/target/目录中

**将该目录移动到hadoop目录下或将压缩包解压到该目录下即可。**