

Linux安装部署spark



2016-11-16

湖北中医药大学信息工程学院

目录

[Linux安装部署spark 0](#_Toc467015674)

[前期准备 2](#_Toc467015675)

[说明： 2](#_Toc467015676)

[更换计算机名称（便于ssh操作） 2](#_Toc467015677)

[集群SSH免密码登录 3](#_Toc467015678)

[开始部署 4](#_Toc467015679)

[Jdk的安装配置 4](#_Toc467015680)

[准备 4](#_Toc467015681)

[环境配置 4](#_Toc467015682)

[Hadoop的安装配置 4](#_Toc467015683)

[准备 4](#_Toc467015684)

[**安装Scala、Spark和IDEA** 4](#_Toc467015685)

[环境配置 6](#_Toc467015686)

[文件配置 7](#_Toc467015687)

[Hadoop和Spark集群测试 10](#_Toc467015688)

[注： 12](#_Toc467015689)

[启动说明 12](#_Toc467015690)

[文件夹说明 12](#_Toc467015691)

[环境配置说明 12](#_Toc467015692)

[Spark配置文件说明 13](#_Toc467015693)

[集群测试 13](#_Toc467015694)

[附录 14](#_Toc467015695)

[附录一 14](#_Toc467015696)

[方法一、图形化界面安装： 14](#_Toc467015697)

[方法二、命令行安装方式 17](#_Toc467015698)

[附录二 18](#_Toc467015699)

[附录三 19](#_Toc467015700)

[附录四 19](#_Toc467015701)

[附录五 20](#_Toc467015702)

# 前期准备

## 说明：

环境说明：

使用系统为：Ubuntu 16.04

系统配置环境：LAMP（linux Apache MySQL PHP）且 装有ssh vim

Spark版本以及相关基础软件版本

Jdk：jdk-8u45-linux-x64

Scala：scala-2.11.8

Hadoop：hadoop-2.7.3

Spark：spark-2.0.0-bin-hadoop2.7

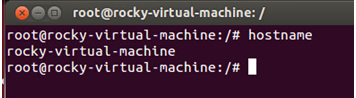
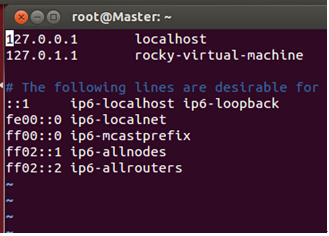
此处略去LAMP的安装步骤（比较简单）

## 更换计算机名称（便于ssh操作）

用命令：vim /etc/hostname

讲计算机名称修改为简单易记的，我主要是以Master ， Slave1 ， Slave2对三台机器命名

修改后保存重启，然后用图一命令进行检查

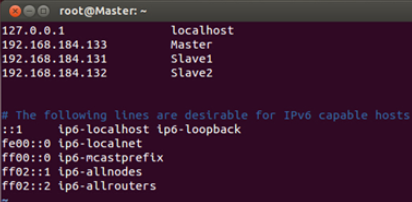
 图一 图二

用命令：vim /etc/hosts

配置主机以及子机器的ip

进入后如图二所示

修改配置后如图三所示（根据自己情况修改）

 图三

注意：必须对每一台主机进行配置，此处给出Master配置截图，建议对每一台机器都进Master主机一样的配置

## 集群SSH免密码登录

首先必须安装有ssh，如果没有安装，请安装，命令为： apt-get install ssh

启动服务，命令为 ：/etc/init.d/ssh start

测试服务，命令为 ：ps -e |grep ssh

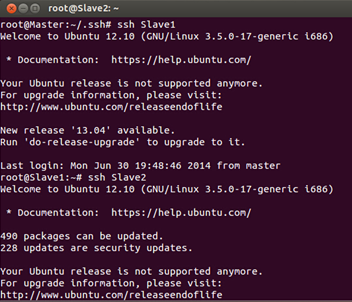
设置免密码登录： ssh-keygen -t rsa -P ""

cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys

第一条命令产生key，第二条是将public key 映射到authorized\_keys文件里面，并创建authorized\_keys文件，此文件用于免密码主机访问。

需要解释的是：

先进入 /root/.ssh 文件夹，里面会出现三个文件夹（如图四），分别是authorized\_keys，id\_rsa.pub ,Id\_rsa，第一个文件是用于免密码访问，第二个文件public key，第三个文件是private key，我们需要修改的主要是authorized\_keys ，将各个主机的public key 写入其中，即可实现免密码登录访问各台主机（建议将本机的public key写在第一个）

免密码登录如图五

 图四 图五

从其他节点复制public key方法众多，在此不再累赘。

主机之间转换命令为：ssh 主机名（即刚刚修改的名称）

若需要密码，则不是没有配置好，请检查你的key是否一致，或者是否正确，主要错误是对应问题，请注意修改。

# 开始部署

## Jdk的安装配置

## 准备

创建java安装目录 mkdir /usr/spark/jdk

将下载的jdk文件放入此文件夹里面

## 环境配置

配置环境变量 vim ~/.bashrc

具体如下：

export JAVA\_HOME=/usr/spark/jdk

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

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

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

## Hadoop的安装配置

## 准备

创建Hadoop安装目录 mkdir /usr/spark/hadoop

将下载的Hadoop文件放入此文件夹里面

查jdk路径 ：进入/usr/spark/hgadoop/etc/hadoop 输入${JAVA\_HOME}

若出现 bash: /usr/lib/java/jdk1.7.0\_79: Is a directory 表示jdk配置成功 则进入下一步

**安装Scala、Spark和IDEA**

创建安装目录 mkdir /usr/spark/scala

将安装文件放入其中

环境变量配置 vim ~/.bashrc

export JAVA\_HOME=/usr/spark/jdk

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

export SCALA\_HOME=/usr/spark/scala

export SPARK\_HOME=/usr/spark/spark

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/spark/hadoop

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**

编辑文件 vim /usr/spark/spark/conf/spark-env.sh

在最后输入

export JAVA\_HOME=/usr/**spark**/jdk

export SCALA\_HOME=/usr/**spark**/scala/scala

export HADOOP\_HOME=/usr/**spark**/hadoop

export HADOOP\_CONF\_DIR=/usr/**spark**/hadoop/ etc/hadoop

export SPARK\_MASTER\_IP=Master

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

如果没有slaves文件，copy一份slaves.tempalte，将其改名为slaves

需要说明的是此处加入主机后，主机将成为worker，成为计算结点

建议不要将主机作为数据处理的结点

vim /usrspark/spark/conf/slaves

Master

Slave1

Slave2

Idea IDEA安装

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

安装路径：/usr/spark/idea

：

vim ~/.bashrc

# for examples

export JAVA\_HOME=/usr/spark/jdk

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

export SCALA\_HOME=/usr/spark/scala

export SPARK\_HOME=/usr/spark/spark

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

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

#HADOOP VARIABLES START

export HADOOP\_INSTALL=/usr/spark/Hadoop

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！

## 环境配置

1. **hadoop-env.sh**

**进入**/usr/spark/hadoop/etc/hadoop文件夹

**输入vim hadoop-env.sh**

**将export JAVA\_HOME=${JAVA\_HOME}**

**改为export JAVA\_HOME=/usr/spark/jdk**

**保存退出**

**输入source hadoop-env.sh使更改生效**

1. **yarn-env.sh**

**进入**/usr/spark/hadoop/etc/hadoop文件夹

**输入vim yarn-env.sh**

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

**export JAVA\_HOME=/usr/spark/jdk**

**保存退出**

**输入source yarn-env.sh使更改生效**

1. **mapred-env.sh**

**进入**/usr/spark/hadoop/etc/hadoop文件夹

**输入vim mapred -env.sh**

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

**export JAVA\_HOME=/usr/spark/jdk**

**保存退出**

**输入source yarn-env.sh使更改生效**

1. **修改~/.bashrc文件中的环境变量**

**输入vim ~/.bashrc**

**插入环境变量配置，如下所示**

**export JAVA\_HOME=/usr/spark/jdk**

**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/spark/hadoop**

**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**

**保存退出**

**输入source yarn-env.sh使更改生效**

## ****文件配置****

创建分别是文件系统所需目录

进入/usr/spark/hadoop文件夹

输入mkdir tmp dfs dfs/name dfs/data

配置文件

**1）core-site.xml**

进入/usr/local/hadoop//etc/Hadoop

输入 vim core-site.xml

* 分布式

<configuration>

<property>

<name> fs.defaultFS </name>

<value>hdfs://Master:9000</value>

</property>

<property>

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

<value>file:/usr/spark/hadoop/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**

进入/usr/spark/hadoop/etc/hadoop

输入 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/spark/hadoop/dfs/name</value>

</property>

<property>

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

<value>file:/usr/spark/hadoop/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Master也充当slave参与工作。

**3）mapred-site.xml**

进入/usr/spark/hadoop/ etc/Hadoop

输入 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>Master:9001</value>

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

</property>

</configuration>

**4）yarn-site.xml**

进入/usr/spark/hadoop/etc/Hadoop

输入vim yarn-site.xml

* 分布式

<configuration>

<property>

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

<value>Master</value>

</property>

<property>

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

<value>mapreduce\_shuffle</value>

</property>

</configuration>

对于上面xml的配置，我们主要也是参考网上相关信息copy的，具体可以查阅资料

**编辑masters和slaves文件**

进入 /usr/spark/hadoop/etc/Hadoop

输入vim masters

分布式：

Master

进入 /usr/spark/hadoop/etc/Hadoop

输入 vim slaves

分布式：

Master

Slave1

Slave2

备注：本例把master也当作slave来用，所以把Master也加到了slaves文件里了。加入的主机即将成为数据结点

## Hadoop和Spark集群测试

1. 启动hadoop分布式集群

用jps可以看集群的运行状态

 格式化集群文件系统

root@Master:/usr/spark/hadoop/bin#./hdfs namenode -format

或root@Master:/# hdfs namenode -format

 启动hadoop服务

root@Master:/usr/ spark /hadoop/sbin# ./start-dfs.sh

root@Master:/usr/ spark /hadoop/sbin# jps

3218 DataNode

4758 Jps

3512 SecondaryNameNode

4265 NodeManager

3102 NameNode

root@Master:/usr/ spark /hadoop/sbin# ./start-yarn.sh

root@Master:/usr/ spark /hadoop/sbin# jps

3218 DataNode

4758 Jps

3512 SecondaryNameNode

4265 NodeManager

3102 NameNode

4143 ResourceManager

root@Master:/usr/spark/hadoop/sbin# ./mr-jobhistory-daemon.sh start historyserver

root@Master:/usr/ spark /hadoop/sbin# jps

4658 JobHistoryServer

3218 DataNode

4758 Jps

3512 SecondaryNameNode

4265 NodeManager

3102 NameNode

4143 ResourceManager

 格式化和启动集群

root@Master:/usr/ spark /hadoop/bin# hadoop namenode –format

 重新启动hadoop服务

root@Master:/usr/spark/hadoop/sbin#./start-dfs.sh

root@Master:/usr/spark/hadoop/sbin#./start-yarn.sh

root@Master:/usr/spark/hadoop/sbin#./mr-jobhistory-daemon.sh start historyserver

 看一下各节点的运行状况：

root@Master:~# hdfs dfsadmin -report

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

（二） 启动Spark分布式集群

进入spark-shell两种方式

1. /usr/spark/spark/sbin# ./start-all.sh 此方式只能单机运行
2. /usr/spark/spark/sbin# MASTER=spark://Master://7077 spark-shell

此处以第一种为例

root@Master:/usr/spark/spark/sbin# ./start-all.sh

root@Master:/usr/spark/spark/sbin# jps

13018 Master

11938 NameNode

12464 ResourceManager

13238 Worker

13362 Jps

12601 NodeManager

12296 SecondaryNameNode

12101 DataNode

10423 JobHistoryServer

root@Master:/usr/spark/spark/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>

（三） 服务启动后，可以启动webUI，具体端口如下：

<http://master:50070>

<http://master:8088>

<http://master:8080>

<http://master:4040>

# 注：

## 启动说明

部署完之后，再次启动不需要那么麻烦

先进入Hadoop文件夹里面的sbin文件夹，输入./start-all.sh

再进入spark文件夹里面的sbin，输入./start-all.sh

需要说明的是，这两个文件夹里面的stop-all.sh是关掉应用的命令，运行即可

## 文件夹说明

将所有安装文件解压到/usr/spark 文件夹中，spark文件夹为新创建文件夹，

解压后的jdk命名为jdk

解压后的Hadoop命名为hadoop

解压后的idea命名为idea

解压后的scala命名为scale

解压后的spark命名为spark

## 环境配置说明

#jdk spark idea cala

export JAVA\_HOME=/usr/spark/jdk

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

export SCALA\_HOME=/usr/spark/scala

export SPARK\_HOME=/usr/spark/spark

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

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

#HADOOP VARIABLES START

export HADOOP\_INSTALL=/usr/spark/Hadoop

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

## Spark配置文件说明

在spark文件夹里面的conf配置文件夹里面，主要是关于配置文件的文件，但初始条件下该文件无法生效，也就是带有template，如果需要该功能，需要将对应的文件copy一份在此文件夹里面，并且将后缀template去掉。配置文件分别为：

docker.properties.template 关于docker的配置

fairscheduler.xml.template

log4j.properties.template 监控日志生成的配置文件

metrics.properties.template 生成文件的配置 监控

slaves.template worker丛机的配置文件

spark-defaults.conf.template

spark-env.sh.template

需要说明的是，各个文件里面的配置众多，可以按照需要修改，文件中注释写的比较清楚，可以按照注释进行修改。

## 集群测试

集群搭建完成，进行实例测试

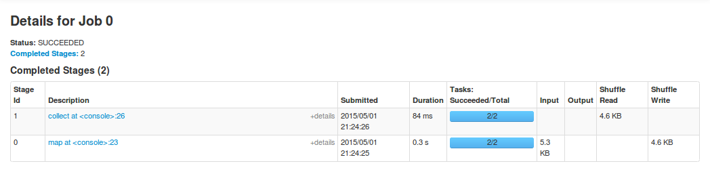
用命令 hadoop fs -put README.md /data/ 将README.md上传

在spark-shell 中运行word count例子

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

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

3. count.collect

spark监控如图。

在<http://spark.apache.org/examples.html> 官网有不少例子，可以根据要求运行实例

# 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

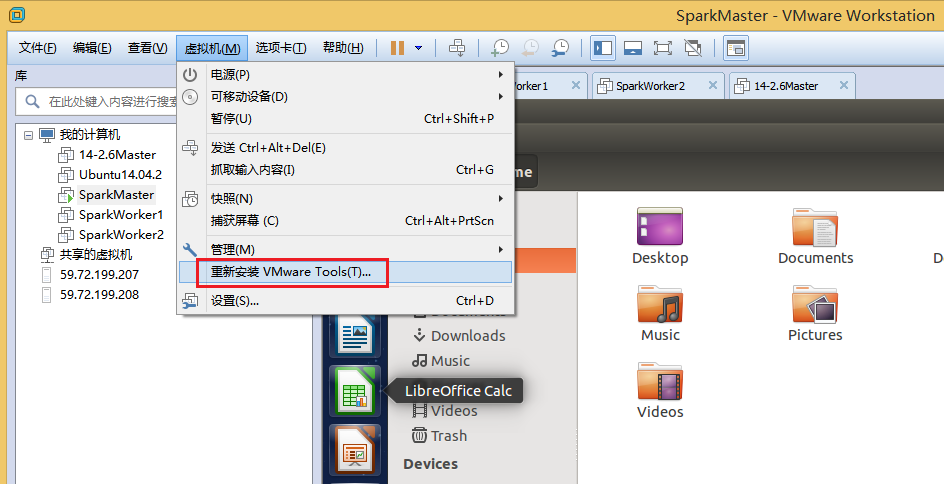
■ 打开虚拟机电源。

■ 确认客户机操作系统正在运行。

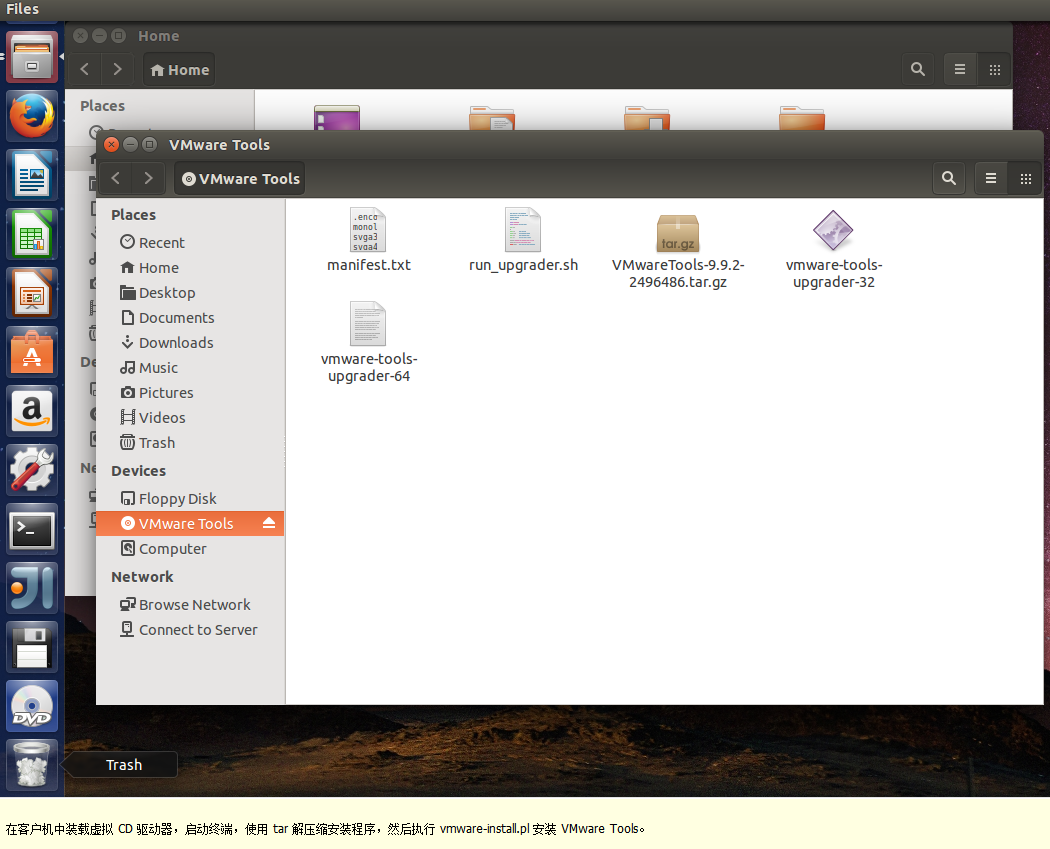
■ 由于 VMware Tools 安装程序是采用 Perl 语言编写的，因此请确认客户机操作系统中已安装 Perl

### 方法一、图形化界面安装：

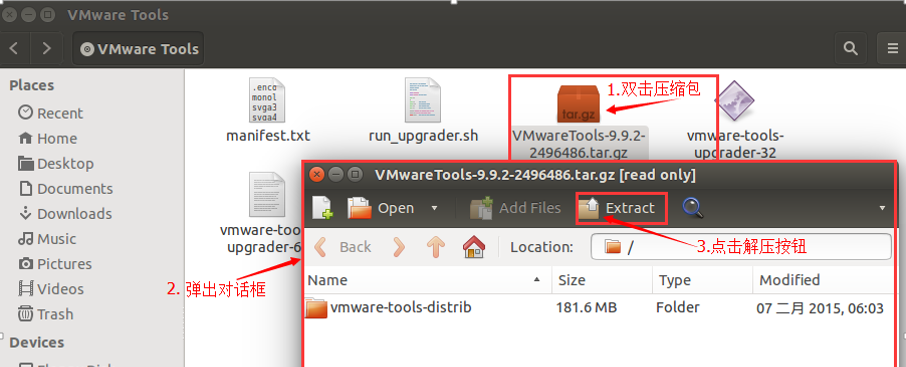
**1.载入vmware tools光盘镜像**

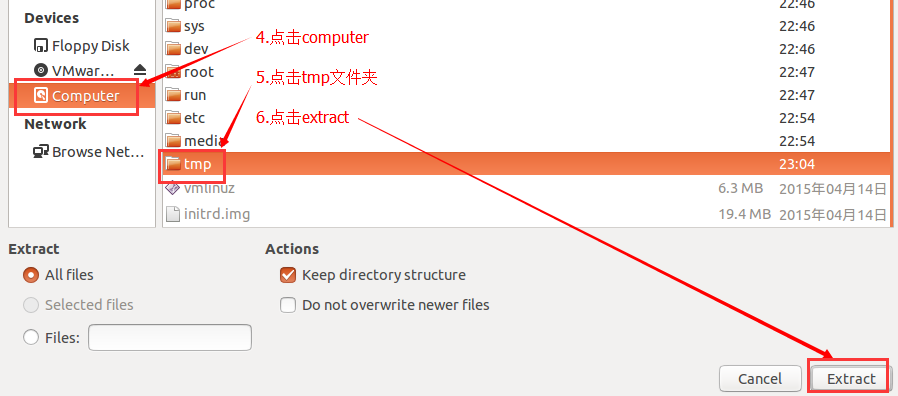


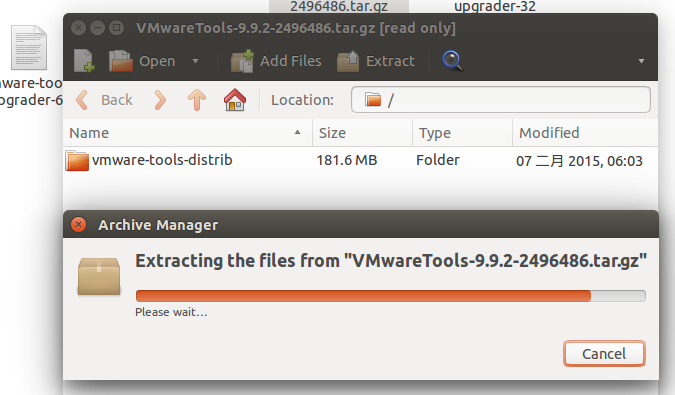
系统自动装载vmware tools光盘，并弹出窗口。

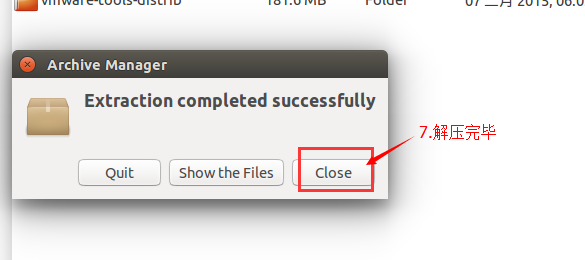


**2.解压安装包**









**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。**