

\* 软件联盟协会大数据项目组专用 \*

# 大数据指南 (第一版)

## ——环境搭建篇

制作人：张绍磊、郁宗扬、郑泽毅

软件联盟协会大数据编写组

## 软件联盟协会大数据编写组

张绍磊.....	信管 13-3
郁宗扬.....	信管 14-1
郑泽毅.....	信 管 14-2
尚博超.....	计算机 15-3
崔 哲.....	计算机 15-3
张 凯.....	计算机 15-3
刘思远.....	信 管 15-2
李若诗.....	信 管 15-2
王文瑾.....	计算机 15-4
郑文青.....	信 管 15-1
周 丹.....	计算机 15-3
田丰收.....	计算机 15-1
张浩智.....	计算机 16-1
吴杰.....	计算机 16-1

## 内容简介

本指南是根据软件联盟协会 2011 级至 2015 级环境搭建经验总结而成，主要针对于软件联盟协会大数据项目组环境搭建使用。

本指南主要阐述了 Hadoop 所需要的开发环境，主要包括操作系统 Ubuntu 搭建过程、基础设置 ( JDK、IP 配置 )、SSH 无密钥登录配置、Hadoop 环境搭建过程、HBase 搭建过程、Zookeeper 搭建过程、Pig 搭建过程、Maven 搭建过程、Hive 搭建过程、Spark 搭建过程。

本指南包含最基本的搭建过程，但是考虑到 Hadoop 版本升级、操作系统不同等各方面因素，仅供参考，如有错误，敬请批评指正。

# 目录

1	Ubuntu环境搭建.....	1
11	Ubuntu简介.....	1
12	Ubuntu启动盘过程.....	1
13	Ubuntu环境搭建过程.....	4
2	基础设置.....	8
21	内容简介.....	8
22	IP配置.....	8
23	SSH配置.....	9
3	Hadoop基础环境搭建.....	13
31	Hadoop简介.....	13
32	Hadoop基础环境搭建过程.....	14
4	HBase搭建.....	23
41	HBase简介.....	23
42	HBase搭建过程.....	23
5	Zookeeper搭建.....	27
51	Zookeeper简介.....	27
52	Zookeeper搭建过程.....	27
6	Pig搭建.....	30
61	Pig简介.....	30
62	Pig搭建过程.....	30
7	Maven搭建.....	34
71	Maven简介.....	34
72	Maven搭建过程.....	34
8	Hive搭建.....	36
81	Hive简介.....	36
82	Hive搭建过程.....	36
83	MySQL安装.....	40

9	Spark搭建.....	43
91	Spark简介.....	43
92	Spark搭建过程.....	43
	参考文献.....	45

# 1 Ubuntu 环境搭建

## 1.1 Ubuntu 简介

Ubuntu 是一个以桌面应用为主的开源 GNU/Linux 操作系统，Ubuntu 是基于 DebianGNU/Linux，支持 x86、amd64（即 x64）和 ppc 架构，由全球化的专业开发团队（Canonical Ltd）打造的。Ubuntu 的目标在于为一般用户提供一个最新的、同时又相当稳定的主要由自由软件构建而成的操作系统。

选择 Ubuntu 有以下十种原因<sup>[1]</sup>：易于安装，默认的外观感觉良好，易于定制主题以及可供选择的大量主题，出色的社区，顺畅的学习曲线，Ubuntu 是适用于诸多发行版的标准，免费的开源工具，安全可靠，Ubuntu 很灵活，定期的更新和支持。

## 1.2 Ubuntu 启动盘过程

Ubuntu 系统的安装、搭建需要制作一个 ubuntu 启动盘，准备以下内容：一个 8G 以上的 U 盘、UltraISO 软件和 Ubuntu 的 iso 镜像。Ubuntu14.10 的下载地址是 <http://old-releases.ubuntu.com/releases/14.10/>。

第一步是打开 UltraISO 软件，然后点击工具栏的“打开”按钮，如图 1.1 红色方框标注所示。

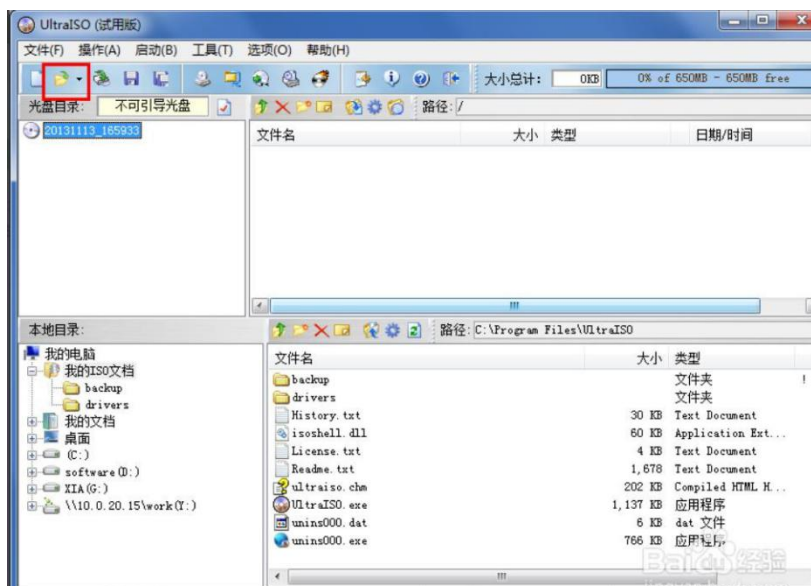


图 1.1 UltraISO 界面图

第二步是在“打开 ISO 文件”对话框中选择我们下载的 Ubuntu 系统镜像文件，然后点击右下方的“打开”按钮，iso 导入图如图 1.2 所示。

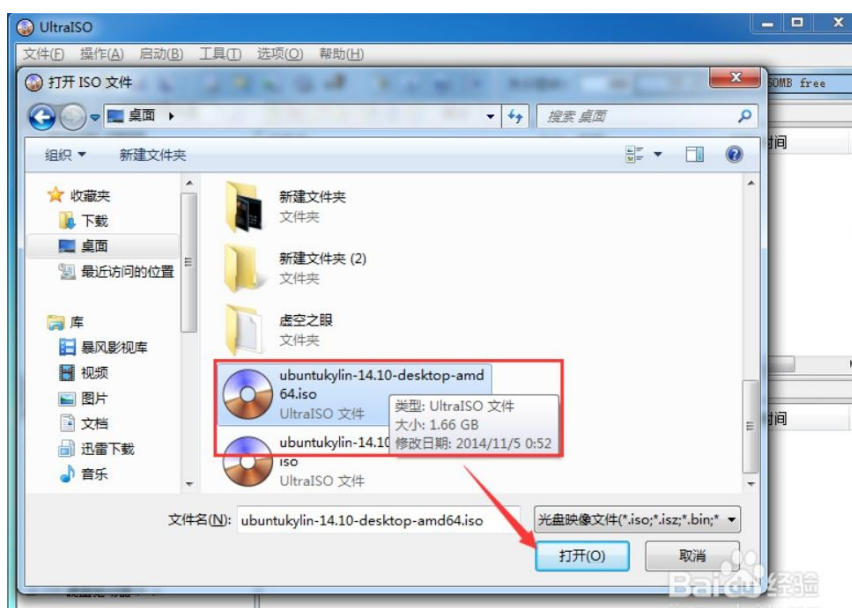


图 1.2 iso 导入图

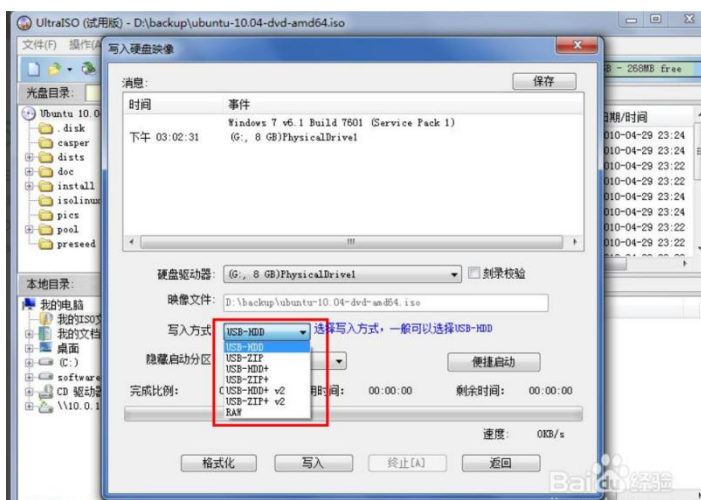
第三步是制作启动盘，点击菜单栏的“启动”，然后选择“写入硬盘映像...”，“启动”图如图 1.3 所示。



图 1.3 “启动”图

第四步是打开“写入硬盘映像”对话框，在硬盘驱动器中选择相对应的 U 盘（即需要制作启动盘的 U 盘），写入方式可以保持默认的 USB-HDD，也可以选择 USB-HDD+，选择好这些后，点击下面的“写入”按钮（注：点击“写入”后，就会弹出警告信息框，若是之前没有备份 U 盘内的资料，那么选择“否”；如果已经备份好资料，所以直接点击按钮“是”）。“写入硬盘映像”图如图 1.4 所示。

图 1.4 “写入硬盘映像”图





完成上述内容后即成功制作了 Ubuntu 的启动盘。（注意：如果权限不够不能写入的话，尝试用管理员身份打开 UltraISO）。

### 1.3 Ubuntu 环境搭建过程

首先在集群的每一台电脑上安装分区助手并启动。选择 D 盘（可根据自己实际需要进行变更）进行操作，在此处的 D 盘是一个全新的未被占用的，从中切割出至少 100G 为 F 盘。将这 100G 的空闲区域给删除。然后点击“提交”重启电脑，插入 U 盘准备安装 Ubuntu。（注意：尽管删除了这部分，但是在物理上这部分仍然是存在的。）分区图如图 1.5-1.7 所示。

图1.5 分区图（1）

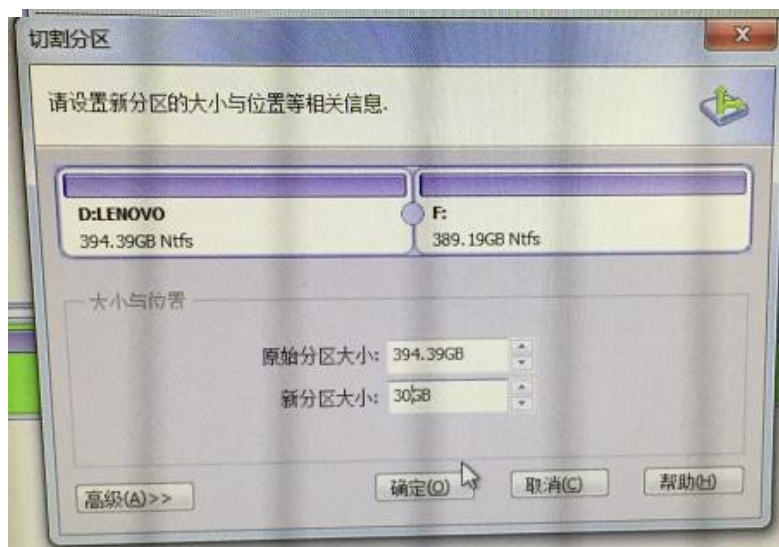


图1.6 分区图 ( 2 )

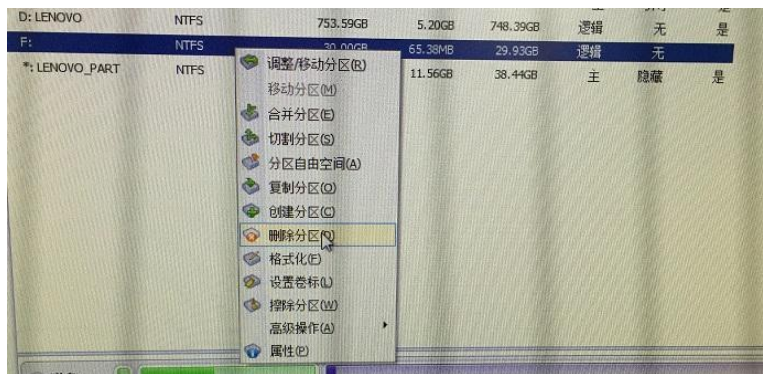


图1.7 分区图 ( 3 )

然后使用 USB 安装 Ubuntu 操作系统。开机时，长按 F2 或 F12 修改引导方式（具体的方式，各品牌电脑略有不同）。修改完毕后，保存并重启（注：插入 u 盘（启动引导盘）），会出现下列界面，具体安装过程如下。

第一步是在左边的引导条中选择“简体中文”，然后点击“安装”。安装启动图如图 1.8 所示。



图 1.8 安装启动图

第二步是选择“其他选项”，然后点击“确定”即可为 Ubuntu 分区。

Ubuntu 分区选择图如图 1.9 所示。

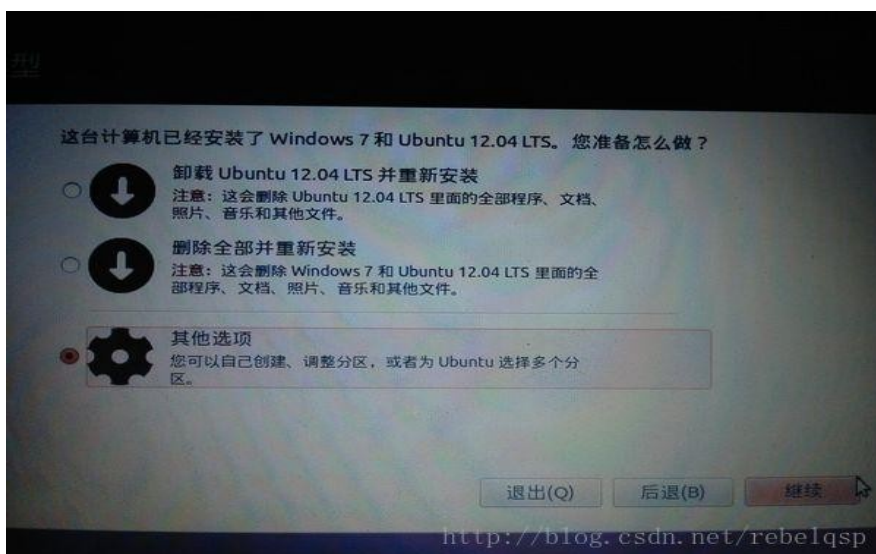


图 1.9 Ubuntu 分区选择图

第三步是选择“空闲”的区域进行分区，在这个过程中要重复四个略有不同的操作，将空间分为四个区，分区图如图 1.10 所示。设置无误后，一直点击“继续”直至要求设置用户信息的界面出现。设置的用户信息为：姓名：Hadoop，密码：000000。



图1.10 分区图

表 1-1 分区作用表

分区	新建分区 容量	新分区 的位置	挂载点	用于
逻辑分区（启动 引导区）	200MB	起始	/boot	Ext4 日志文件系统
逻辑分区（类似 于我的文档下的 东西）	7000MB	起始	/home	Ext4 日志文件系统
逻辑分区（缓存）	2048MB	起始		交换空间
主分区（系统盘）	剩余的所 有容量	起始	/	Ext4 日志文件系统

## 2 基础设置

### 2.1 内容简介

在本章中描述 Ubuntu 系统下 JDK 的安装配置和 SSH 无密钥登录。

JDK 是 Java 语言的软件开发工具包，主要用于移动设备、嵌入式设备上的 Java 应用程序。JDK 是整个 Java 开发的核心，它包含了 Java 的运行环境，Java 工具和 Java 基础类库。

SSH ( Secure Shell ) 是一种远程登录安全协议，它提供了一条安全的远程登录通道，可以简单理解为一种逻辑上的“通道”。SSH 是最通用的系统管理工具之一，允许登录远程系统并在其上执行命令。它利用强大的加密技术和主机密钥来防止网络嗅探。它是默认启用的唯一网络服务，并且接受远程访问。

### 2.2 IP 配置

第一步是打开“网络”，在“网络代理”中选择“手动”方式，配置“有线连接”。网络配置图如图 2.1 所示。



图 2.1 网络配置图

第二步是点击“添加”，将IP 地址，子网掩码，默认网关，首选 DNS 服务器的地址填入。IP 配置图如图 2.2 所示。



图 2.2 IP 配置图

## 2.3 SSH 配置

首先明确结点信息(分布式集群架构：master 为主节点，其余为从节点)。本次以四台主机搭建集群，具体信息如表 2-1 所示。(注：为使在启动 hadoop 集群时不用输入 datanode 的密码，最好将所有机器的用户名设为同一个名字)

表 2-1 结点信息表

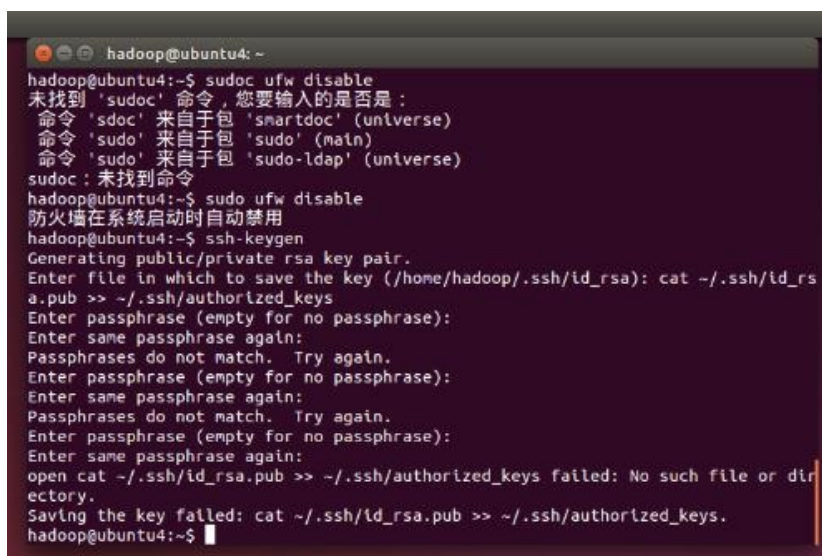
机器名	IP	作用
Master(ubuntu1)	172.31.55.32	NameNode and JobTracker
slave1(ubuntu2)	172.31.55.26	DataNode and TaskTracker
slave2(ubuntu3)	172.31.55.27	
slave3(ubuntu4)	172.31.55.33	

步骤一：SSH 服务的安装程序自动安装，在终端输入：`sudo apt-get install openssh-server`

步骤二：关闭防火墙，在终端输入：`sudo ufw disable`

步骤三：生成公钥和私钥，在终端输入：`ssh-keygen` 回车后显示“Enter file which to save the key”不用输入密码，一直回车等到新内容的出现。完成之后，在/home/Hadoop(“username”是登录名)目录下会有生成一个

“.ssh”目录，（.ssh 文件夹是隐藏的，Ctrl+H 查看隐藏的文件）“ls .ssh”之后会发现里面有两个文件，一个是 id\_rsa（私钥），另一个是 id\_rsa.pub（公钥）。密钥生成图如图 2.3 所示。

A terminal window titled 'hadoop@ubuntu4: ~' showing the execution of several commands. The first command is 'sudo ufw disable', which outputs a message about 'sudoc' not being found. The second command is 'sudo ufw disable', which outputs '防火墙在系统启动时自动禁用'. The third command is 'ssh-keygen', which starts generating a public/private rsa key pair. It prompts for a file to save the key, and the user enters '~/.ssh/id\_rsa'. It then prompts for a passphrase, and the user enters an empty one. The process repeats for the passphrase three times. Finally, it prompts to open the public key file '~/.ssh/id\_rsa.pub' and append its contents to '~/.ssh/authorized\_keys'. The process fails with the message 'No such file or directory'. The user then manually runs 'cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys' to complete the setup. The terminal output is as follows:

```
hadoop@ubuntu4:~$ sudo ufw disable
未找到 'sudoc' 命令，您要输入的是否是：
命令 'sdoc' 来自于包 'smartdoc' (universe)
命令 'sudo' 来自于包 'sudo' (main)
命令 'sudo' 来自于包 'sudo-ldap' (universe)
sudoc: 未找到命令
hadoop@ubuntu4:~$ sudo ufw disable
防火墙在系统启动时自动禁用
hadoop@ubuntu4:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hadoop/.ssh/id_rsa): cat ~/.ssh/id_rsa
a.pub >> ~/.ssh/authorized_keys
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match. Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match. Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
open cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys failed: No such file or dir
ectory.
Saving the key failed: cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys.
hadoop@ubuntu4:~$
```

图 2.3 密钥生成图

步骤四：建立信任连接列表，在终端输入：

`cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys`

步骤五：无密钥登录，在终端输入：`ssh localhost`，然后需要我们输入本机登录密码，正确的输入即可。在终端输入“ssh localhost”后即可无密登录了。无密钥登录图如图 2.4 所示。



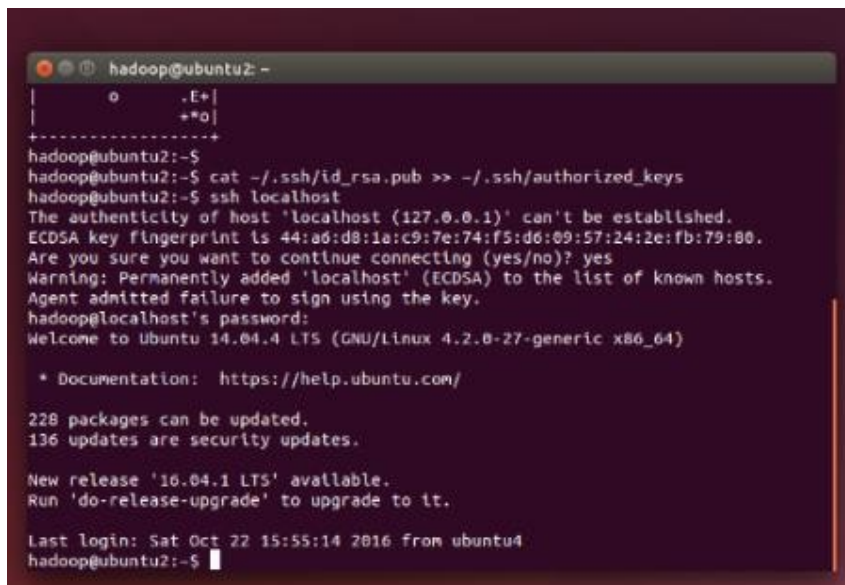


图 2.4 无密钥登录图

步骤六：无密码登录另几台机器(即 172.31.55.32 (主)无密码登录 172.31.55.26 以及 172.31.55.27 和 172.31.55.33)。在 IP 为 172.31.55.32 主机终端中输入: `sudo gedit /etc/hosts` 。在弹出的文件中，在文件最后面输入 172.31.55.32 hadoop（注意：此处的 hadoop 为机器名称），保存后退出。hosts 配置图如图 2.5 所示。

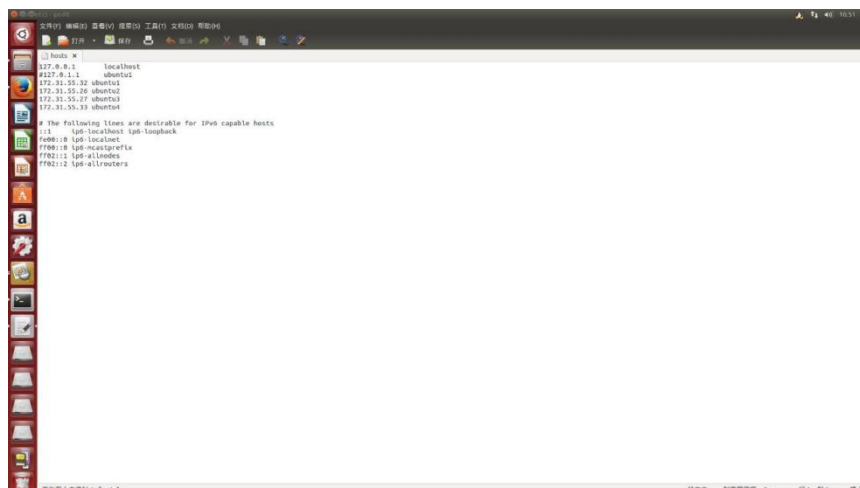


图 2.5 hosts 配置图



步骤七：在主机的主终端中分别键入三个分机的 IP, 这里可能会让输入 hadoop@172.31.55.26 的机器密码，输入就可以了。在终端输入：`scp ~/.ssh/id_rsa.pub hadoop@172.31.55.26:/home/hadoop/`（注意：此处的 IP 都为分机的 IP。）拷贝主节点至各子节点如图 2.6 所示。



```
hadoop@ubuntu3: -
hadoop@ubuntu2:~$ sudo gedit /etc/hosts
[sudo] password for hadoop:

(gedit:4489): IBUS-WARNING **: The owner of /home/hadoop/.config/ibus/bus is not root!

(gedit:4489): Gtk-WARNING **: Calling Inhibit failed: GDBus.Error:org.freedesktop.DBus.Error.ServiceUnknown: The name org.gnome.SessionManager was not provided by any .service files

(gedit:4489): Gtk-WARNING **: Calling Inhibit failed: GDBus.Error:org.freedesktop.DBus.Error.ServiceUnknown: The name org.gnome.SessionManager was not provided by any .service files
hadoop@ubuntu2:~$ scp ~/.ssh/id_rsa.pub hadoop@172.31.55.27:/home/hadoop/
hadoop@172.31.55.27's password:
id_rsa.pub                                100% 396      0.4KB/s   00:00
hadoop@ubuntu2:~$ ssh hadoop@172.31.55.27
Agent admitted failure to sign using the key.
hadoop@172.31.55.27's password:
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

228 packages can be updated.
```

图 2.6 拷贝图

步骤八：建立信任列表，在 IP 为 172.31.55.26 的机器终端（从机）中键入：`cat ~/id_rsa.pub >> ~/.ssh/authorized_keys`。

步骤九：回到 172.31.55.32(主机)的机器上，在终端键入 `ssh hadoop@172.31.55.26`。如果有输入密码的提示，直接输入 ip 为 172.31.55.26 的机器密码就可以了。（补充：如果你键入 `ssh 172.31.55.26`，程序也可能会提示你输入 `hadoop@172.31.55.26` 密码，输入即可。作用是相同的）。

## 3 Hadoop 基础环境搭建

### 3.1 Hadoop 简介

Hadoop 起源于开源网络搜索引擎 Apache Nutch 项目，2004 年，Google 公开发表了题为“MapReduce：Simplified Data Processing on Large Clusters”的论文，Nutch 开发人员受到启发实现 MapReduce 计算框架并与 Nutch 的文件系统 NDFS 结合。2006 年 3 月，NDFS 和 MapReduce 从 Nutch 独立出来成为 Hadoop。到 2008 年初，Hadoop 成为 Apache 的顶级项目，应用到诸多互联网公司。

Hadoop 的核心由两个子项目组成：HDFS 分布式文件系统和 MapReduce 计算框架。

Hadoop 分布式文件系统（HDFS）是 Hadoop 所使用的主要存储系统。HDFS 是一个主从结构，一个 HDFS 集群是由一个名字节点，它是一个管理文件命名空间和调节客户端访问文件的主服务器，当然还有一些数据节点，通常是一个节点一个机器，它来管理对应节点的存储。HDFS 对外开放文件命名空间并允许用户数据以文件形式存储。HDFS 有着高容错性的特点，并且设计用来部署在低廉的硬件上。而且它提供高吞吐量来访问应用程序的数据，适合那些有着超大数据集的应用程序。

MapReduce 是一种编程模型，用于大规模数据集（大于 1TB）的并行运算。它极大地方便了编程人员在不会分布式并行编程的情况下，将自己的程序运行在分布式系统上。当前的软件实现是指定一个 Map（映射）函数，用来把一组键值对映射成一组新的键值对，指定并发的 Reduce（归约）函数，用来保证所有映射的键值对中的每一个共享相同的键组。



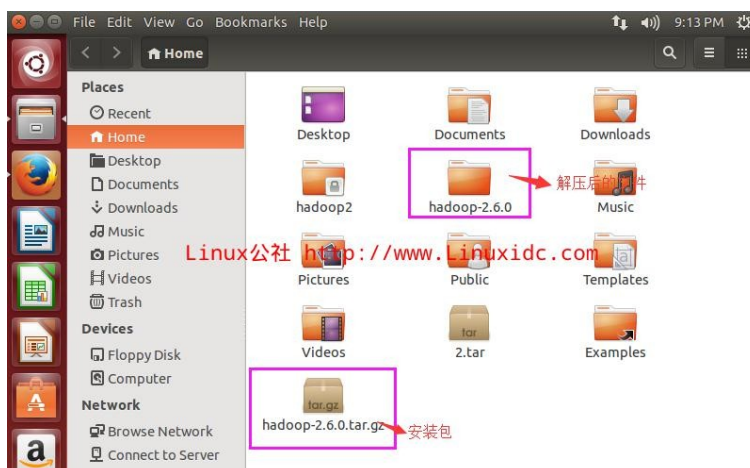


图 3.2 hadoop 存储位置图

步骤二：修改 hadoop-2.6.0/etc/hadoop/hadoop-env.sh，添加 JDK 支持，在打开的文件中输入：`export JAVA_HOME=/usr/local/java/jdk1.8.0_91`。Hadoop-env.sh 配置图如图 3.3 所示。

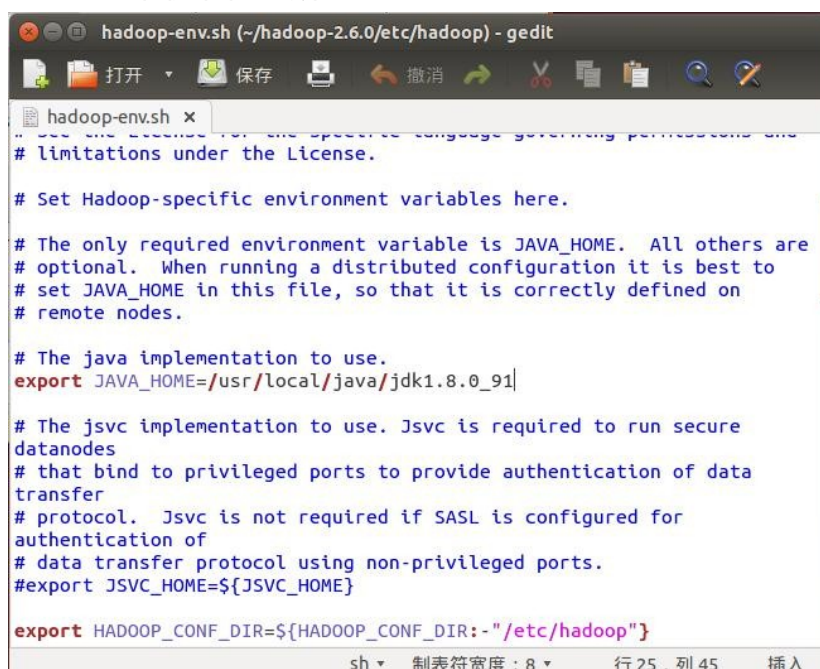


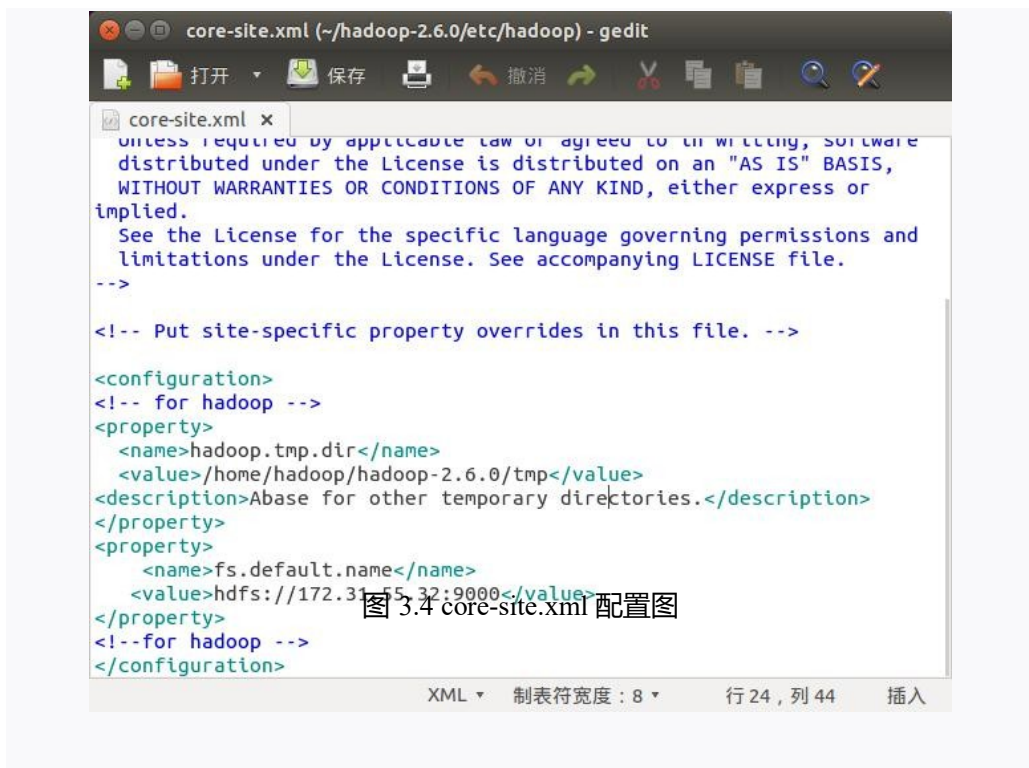
图 3.3 Hadoop-env.sh 配置图

步骤三：修改 hadoop-2.6.0/etc/hadoop/core-site.xml，core-site.xml 配置

图如图 3.4 所示 ( 注意：必须加在<configuration></configuration>节点内 )。

```
<configuration>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/home/hadoop/hadoop-2.6.0/tmp</value>
    <description>Abase for other temporary directories.</description>
  </property>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://172.31.55.32:9000</value>
  </property>
</configuration>
```

注：hadoop.tmp.dir 指定了所有上传到 Hadoop 的文件的存放目录，所以要确保这个目录是足够大的。Fs.default.name 指定 NameNode 的 IP 地址和端口号，默认值是 [file:///](#)，表示使用本地文件系统，用于单机非分布式模式。



步骤四：修改 hadoop-2.6.0/etc/hadoop/hdfs-site.xml(注意：必须加在 <configuration></configuration>节点内)。

```

<property>
  <name>dfs.name.dir</name>
  <value>/home/hadoop/hadoop-2.6.0/dfs/name</value>
  <description>Path on the local filesystem where the NameNode stores
the namespace and transactions logs persistently.</description>
</property>
<property>
  <name>dfs.data.dir</name>
  <value>/home/hadoop/hadoop-2.6.0/dfs/data</value>
  <description>Comma separated list of paths on the local filesystem of
a DataNode where it should store its blocks.</description>

```

```

</property>
<property>
    <name>dfs.replication</name>
    <value>1</value>
</property>

```

注：hadoop-site.xml 文件中设置与 HDFS 相关的设定，如文件的副本个数、块大小及是否使用强制权限等，此中的参数定义会覆盖 hdfs-default.xml 文件中的默认配置。

步骤五：修改 hadoop-2.6.0/etc/hadoop/mapred-site.xml（注：配置 MapReduce 的相关设定，如 reduce 任务的默认个数、任务所能够使用内存的默认上下限等，此种参数会覆盖 mapred-default.xml 文件中的默认配置）。

```

<property>
    <name>mapred.job.tracker</name>
    <value>172.31.55.32:9001</value>
    <description>Host or IP and port of JobTracker.</description>
</property>

```

步骤六：配置 hadoop 环境变量，打开/etc/profile，加入以下内容，配置完成后，可以使用 hadoop version 来查看版本。

```

export HADOOP_HOME=/home/hadoop/hadoop-2.6.0
export PATH=$PATH:$HADOOP_HOME/bin

```

步骤七：修改 hadoop-2.6.0/etc/hadoop/masters，这个 masters 文件是需要自己建立的。列出所有的 master 节点，即为所有的主机名称：IP 地址 Master 名称。masters 配置图如图 3.5 所示。

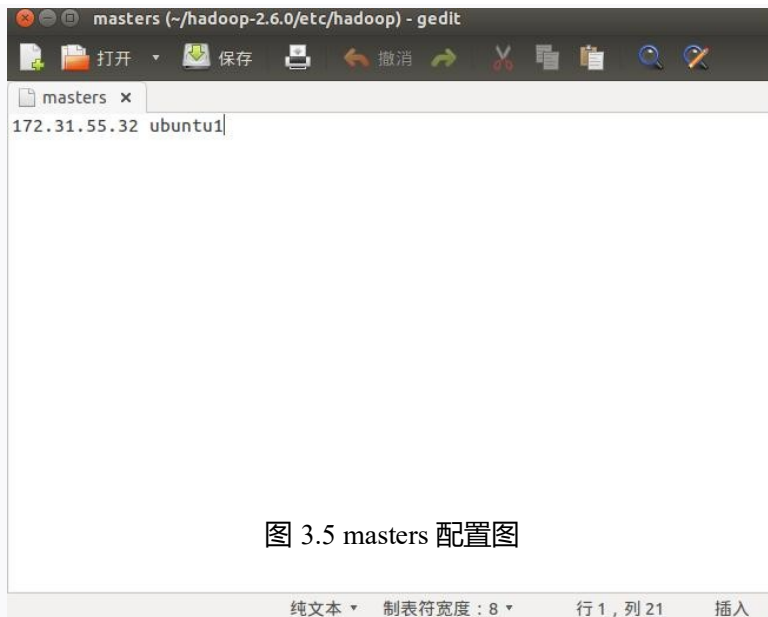


图 3.5 masters 配置图

步骤八：修改 `hadoop-2.6.0/etc/hadoop/slaves`, 这个是所有 datanode 的机器，即为所有的分机。slaves 配置图如图 3.6 所示。

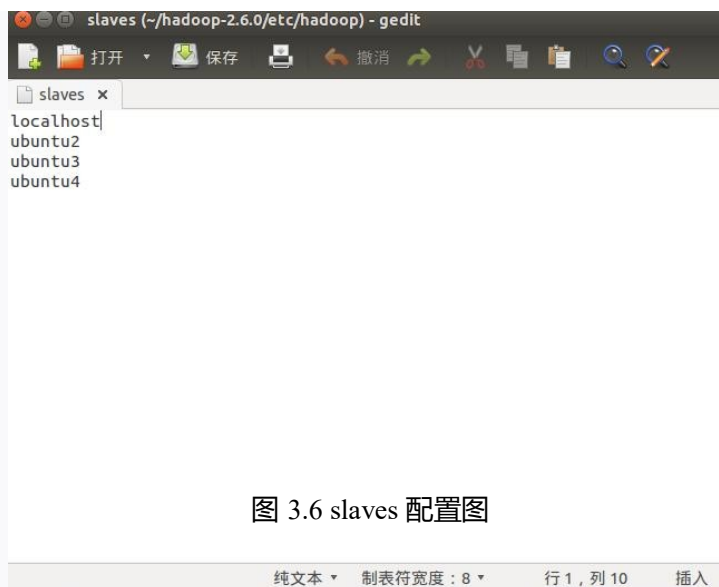


图 3.6 slaves 配置图

步骤九：将 master 结点上配置好的 hadoop 文件夹拷贝到所有的 slave

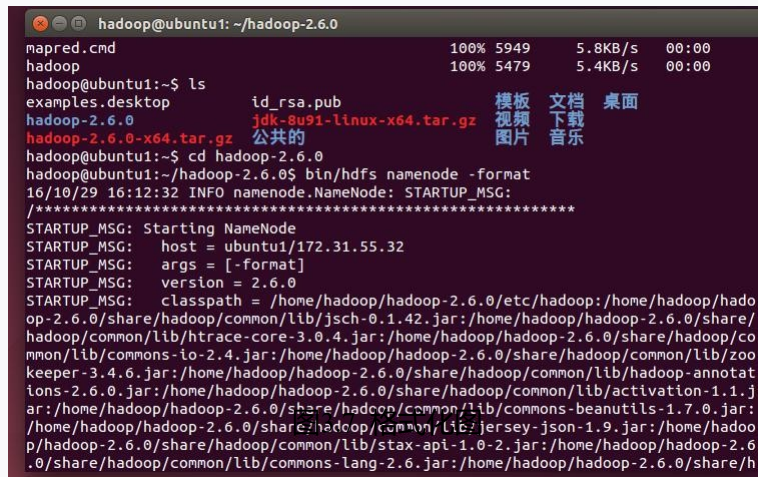


结点上，在终端输入如下命令：scp -r ~/hadoop-2.6.0 hadoop@slave:~/（注意：slave 处用 IP 地址或者机器名）。

步骤十：安装完成后，进入 hadoop 目录并格式化 HDFS，在终端输入如下命令。格式化图如图 4.7 所示。

cd \$HADOOP\_HOME（注意：这里要进入 hadoop-2.6.0 目录来格式化好些）

bin/hdfs namenode -format（注意：此处只格式化主机，切记不可反复格式化）



```
hadoop@ubuntu1: ~/hadoop-2.6.0
mapred.cmd                               100% 5949    5.8KB/s   00:00
hadoop                                   100% 5479    5.4KB/s   00:00
hadoop@ubuntu1:~$ ls
examples.desktop      id_rsa.pub          模板 文档 桌面
hadoop-2.6.0         jdk-8u91-linux-x64.tar.gz 视频
hadoop-2.6.0-x64.tar.gz 公共的              图片
hadoop@ubuntu1:~$ cd hadoop-2.6.0
hadoop@ubuntu1:~/hadoop-2.6.0$ bin/hdfs namenode -format
16/10/29 16:12:32 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = ubuntu1/172.31.55.32
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 2.6.0
STARTUP_MSG:   classpath = /home/hadoop/hadoop-2.6.0/etc/hadoop:/home/hadoop/hado
op-2.6.0/share/hadoop/common/lib/jsch-0.1.42.jar:/home/hadoop/hadoop-2.6.0/share/
hadoop/common/lib/htrace-core-3.0.4.jar:/home/hadoop/hadoop-2.6.0/share/hadoop/co
mmon/lib/commons-io-2.4.jar:/home/hadoop/hadoop-2.6.0/share/hadoop/common/lib/zoo
keeper-3.4.6.jar:/home/hadoop/hadoop-2.6.0/share/hadoop/common/lib/hadoop-annotat
ions-2.6.0.jar:/home/hadoop/hadoop-2.6.0/share/hadoop/common/lib/activation-1.1.j
ar:/home/hadoop/hadoop-2.6.0/share/hadoop/common/lib/commons-beanutils-1.7.0.jar:
/home/hadoop/hadoop-2.6.0/share/hadoop/common/lib/jersey-json-1.9.jar:/home/hadoo
p/hadoop-2.6.0/share/hadoop/common/lib/stax-api-1.0-2.jar:/home/hadoop/hadoop-2.6
.0/share/hadoop/common/lib/commons-lang-2.6.jar:/home/hadoop/hadoop-2.6.0/share/h
```

步骤十一：启动 Hadoop 集群。如果同时启动 HDFS 和 Map/Reduce，可在 hadoop-2.6.0/sbin 目录下有 start-all.sh 文件(Hadoop 的启动文件)，在终端输入：sbin/start-all.sh。如果不同时启动 HDFS 和 Map/Reduce，在终端分别输入：sbin/start-dfs.sh（单独启动 HDFS 集群）和sbin/start-mapred.sh（单独启动 Map/Reduce）。启动图如图 3.8 所示。

```
hadoop@ubuntu1: ~/hadoop-2.6.0
es with txid >= 0
16/10/29 16:12:33 INFO util.ExitUtil: Exiting with status 0
16/10/29 16:12:33 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at ubuntu1/172.31.55.32
*****/
hadoop@ubuntu1:~/hadoop-2.6.0$ sbin/start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [localhost]
localhost: namenode running as process 3130. Stop it first.
localhost: datanode running as process 3592. Stop it first.
ubuntu3: starting datanode, logging to /home/hadoop/hadoop-2.6.0/logs/hadoop-hado
op-datanode-ubuntu3.out
ubuntu2: starting datanode, logging to /home/hadoop/hadoop-2.6.0/logs/hadoop-hado
op-datanode-ubuntu2.out
ubuntu4: starting datanode, logging to /home/hadoop/hadoop-2.6.0/logs/hadoop-hado
op-datanode-ubuntu4.out
Starting secondary namenodes [0.0.0.0]
The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.
ECDSA key fingerprint is db:d8:ae:c5:7d:ad:67:94:78:96:2b:65:cb:3d:7f:cd.
Are you sure you want to continue connecting (yes/no)? yes
0.0.0.0: Warning: Permanently added '0.0.0.0' (ECDSA) to the list of known hosts.
0.0.0.0: starting secondarynamenode, logging to /home/hadoop/hadoop-2.6.0/logs/ha
dop-hadoop-secondarynamenode-ubuntu1.out

localhost: starting nodemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-h
adoop-nodemanager-ubuntu1.out
ubuntu4: starting nodemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-had
oop-nodemanager-ubuntu4.out
ubuntu3: starting nodemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-had
oop-nodemanager-ubuntu3.out
ubuntu2: starting nodemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-had
oop-nodemanager-ubuntu2.out
```

图3.8 启动图

步骤十二：用命令行来查看 Hadoop 的运行状态：在终端输入 jps，测试图如图 3.9 所示；也可用 web 查看 Hadoop 的运行状态：localhost:50070。网络测试图如图 3.10 所示。

```
hadoop@ubuntu1:~/hadoop-2.6.0$ jps
824 Jps
3592 DataNode
5714 NodeManager
3130 NameNode
5234 SecondaryNameNode
5398 ResourceManager
hadoop@ubuntu1:~/hadoop-2.6.0$
```

图3.9 测试图

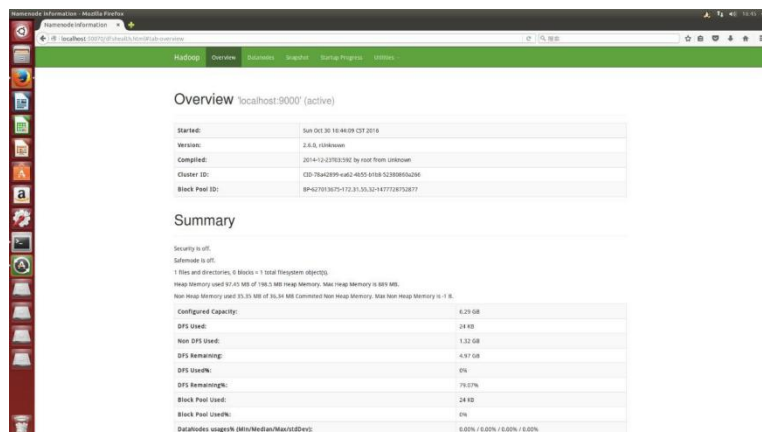


图 3.10 网络测试图

## 4 HBase 搭建

### 4.1 HBase 简介

HBase 是一个分布式的、面向列的开源数据库。HBase 是 Apache 的 Hadoop 项目的子项目。HBase 不同于一般的关系数据库，它是一个适合于非结构化数据存储的数据库，而且 HBase 采用基于列的而不是基于行的模式。

HBase 适用于半结构化或非结构化数据，记录非常稀疏，多版本数据以及超大数据量<sup>[2]</sup>。

相对于 Oracle 等关系型数据库而言，HBase 适合大量数据插入同时伴随读取的情况。Hbase 的瓶颈是硬盘传输速度，Oracle 的瓶颈是硬盘寻道时间。Hbase 本质上只有一种操作——插入，其更新操作是插入一个带有新的时间戳的行，而删除是插入一个带有插入标记的行。其主要操作是收集内存中一批数据，然后批量的写入硬盘，所以其写入的速度主要取决于硬盘传输的速度。Oracle 则不同，因为它经常要随机读写，这样硬盘磁头需要不断的寻找数据所在，所以瓶颈在于硬盘寻道时间<sup>[3]</sup>。

Hbase 同样存在局限，只能做简单的 Key value 查询，复杂的 sql 统计做不到；只能在 row key 上做快速查询。

### 4.2 HBase 搭建过程

步骤一：下载并解压 HBase。把下载好的 hbase-1.2.5-bin.tar.gz，复制到 hadoop-2.6.0 文件夹里面，在终端输入如下内容。

```
cd $HADOOP_HOME
```

```
tar xzvf hbase-1.2.5-bin.tar.gz
```

步骤二：配置环境变量，在终端输入如下内容。

```
sudo gedit ~/.bashrc(sudo gedit /etc/profile)
```

步骤三：在打开的文件里（ ~/.bashrc 或/etc/profile ）最后面加上：

```
export HBASE_HOME=/home/hadoop/hadoop-2.7.2/hbase-1.2.5
```

```
export PATH=$HBASE_HOME/bin:$PATH
```

步骤四：在终端生效/etc/profile。

```
source /etc/profile
```

步骤五：配置 HBase。进入到/home/hadoop/hadoop-2.7.2/hbase-1.2.5/conf文件夹下，修改 hbase-env.sh 和hbase-site.xml。

步骤六：配置 hbase-env.sh。找到# export

JAVA\_HOME=/usr/java/jdk1.6.0/这一行，在下面加上如下内容。

```
export JAVA_HOME=/usr/local/java/jdk1.8.0_91
```

步骤七：配置 hbase-site.xml（注：在<configuration></configuration>里）加上如下内容。

```
<property>
```

```
  <name>hbase.rootdir</name>
```

```
  <value>hdfs://172.31.55.32:9000/hbase</value>
```

```
</property>
```

```
<property>
```

```
  <name>hbase.cluster.distributed</name>
```

```
  <value>true</value>
```

```
</property>
```

```
<property>
```

```
  <name>dfs.replication</name>
```

```
<value>1</value>
</property>
<property>
  <name>hbase.master</name>
  <value>172.31.55.32:60000</value>
</property>
<property>
  <name>hbase.master.info.port</name>
  <value>60010</value>
</property>
<property>
  <name>hbase.zookeeper.quorum</name>
  <value>172.31.55.32</value>
</property>
```

步骤八：HBase 启动（前提是 Hadoop 集群）。在终端输入如下命令。

```
cd ./hbase-1.2.5/bin
./start-hbase.sh
```

步骤九：用命令行来查看 HBase 的运行状态：在终端输入 `jps`，测试图如图 4.1 所示；也可用 web 查看 HBase 的运行状态：`localhost:60010`。网络测试图如图 4.2 所示。

```
hadoop@hadoop-QiTianM6500-D684: ~/hadoop-2.7.2/hbase-1.2.5/bin
hbase-cleanup.sh      local-regionervers.sh  stop-hbase.cmd
hbase.cmd              master-backup.sh       stop-hbase.sh
hbase-common.sh        region_mover.rb        test
hbase-config.cmd       regionservers.sh       thread-pool.rb
hbase-config.sh        region_status.rb       zookeepers.sh
hbase-daemon.sh        replication
hadoop@hadoop-QiTianM6500-D684:~/hadoop-2.7.2/hbase-1.2.5/bin$ ./start-hbase.sh
172.31.55.38: starting zookeeper, logging to /home/hadoop/hadoop-2.7.2/hbase-1.2.5/bin/./logs/hbase-hadoop-zookeeper-hadoop-QiTianM6500-D684.out
starting master, logging to /home/hadoop/hadoop-2.7.2/hbase-1.2.5/logs/hbase-hadoop-master-hadoop-QiTianM6500-D684.out
starting regionserver, logging to /home/hadoop/hadoop-2.7.2/hbase-1.2.5/logs/hbase-hadoop-1-regionserver-hadoop-QiTianM6500-D684.out
hadoop@hadoop-QiTianM6500-D684:~/hadoop-2.7.2/hbase-1.2.5/bin$ jps
3812 ResourceManager
4998 HQuorumPeer
3254 NameNode
5559 Jps
5081 HMaster
3628 SecondaryNameNode
4124 NodeManager
3421 DataNode
5230 HRegionServer
hadoop@hadoop-QiTianM6500-D684:~/hadoop-2.7.2/hbase-1.2.5/bin$
```

图 4.1 测试图

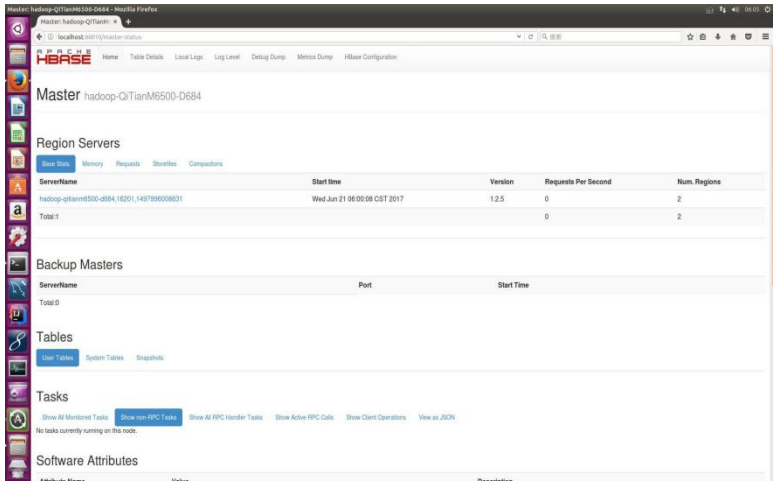


图 4.2 网络测试图

## 5 Zookeeper 搭建

### 5.1 Zookeeper 简介

Zookeeper 是一个分布式的，开放源码的分布式应用程序协调服务，是 Google 的 Chubby 一个开源的实现，是 Hadoop 和 Hbase 的重要组件。它是一个为分布式应用提供一致性服务的软件，提供的功能包括：配置维护、名字服务、分布式同步、组服务等。ZooKeeper 的目标就是封装好复杂易出错的关键服务，将简单易用的接口和性能高效、功能稳定的系统提供给用户。

Zookeeper 主要应用于数据发布与订阅（配置中心）、负载均衡、命名服务(Naming Service)、分布式通知/协调、集群管理与 Master 选举、分布式锁、分布式队列<sup>[4]</sup>。

### 5.2 Zookeeper 搭建过程

步骤一：下载并解压 Zookeeper。把下载好的 zookeeper-3.4.9.tar.gz，复制到 hadoop-2.6.0 文件夹里面，在终端输入如下内容。

```
cd $HADOOP_HOME  
tar xzvf zookeeper-3.4.9.tar.gz
```

步骤二：配置环境变量，在终端输入如下内容。

```
sudo gedit ~/.bashrc(sudo gedit /etc/profile)
```

步骤三：在打开的文件里（~/.bashrc 或/etc/profile）最后面加上：

```
export ZOOKEEPER_HOME=/usr/zookeeper
```



```
export PATH=$PATH:$ZOOKEEPER_HOME/bin
```

步骤四：在终端生效/etc/profile。

```
source /etc/profile
```

步骤五：在终端输入如下内容，修改 zoo\_sample.cfg 为 zoo.cfg。

```
cd /usr/zookeeper/conf zoo
```

```
cp zoo_sample.cfg zoo.cfg
```

步骤六：配置 zoo.cfg（注：在这里需要注意 zoo.cfg 的配置文件中的 dataDir 和 dataLogDir 的路径当中的文件夹必须要已存在，否则后面启动 zkServer 服务的时候会失败）。

```
# The number of milliseconds of each tick
```

```
tickTime=2000
```

```
# The number of ticks that the initial
```

```
# synchronization phase can take
```

```
initLimit=10
```

```
# The number of ticks that can pass between
```

```
# sending a request and getting an acknowledgement
```

```
syncLimit=5
```

```
# the directory where the snapshot is stored.
```

```
# do not use /tmp for storage, /tmp here is just
```

```
# example sakes.
```

```
dataDir=/home/hadoop/zookeeper/data
```

```
dataLogDir=/home/hadoop/zookeeper/log
```

```
# the port at which the clients will connect
```

```
clientPort=2181
```

```
# the maximum number of client connections.
```

```
# increase this if you need to handle more clients
#maxClientCnxns=60
# Be sure to read the maintenance section of the
# administrator guide before turning on autopurge.
```

步骤七：zkServer 启动，在终端输入如下命令。

```
cd /home/hadoop/zookeeper-3.4.9/
bin/zkServer.sh start
```

步骤八：在终端输入 jps 可以看到启动了 QuorumPeerMain 进程。

```
[hadoop@hadoop-QiTianM6500-D684]# jps
9904 QuorumPeerMain
9987 Jps
6950 NodeManager
3943 DataNode
9657 NameNode
6847 ResourceManager
```

## 6 Pig 搭建

### 6.1 Pig 简介

Pig 是一种数据流语言和运行环境，用于检索非常大的数据集。为大型数据集的处理提供了一个更高层次的抽象。Pig 包括两部分：一是用于描述数据流的语言，称为 Pig Latin；二是用于运行 Pig Latin 程序的执行环境。

Apache Pig 是一个高级过程语言，适合于使用 Hadoop 和 MapReduce 平台来查询大型半结构化数据集。通过允许对分布式数据集进行类似 SQL 的查询，Pig 可以简化 Hadoop 的使用。

用 MapReduce 进行数据分析。当业务比较复杂的时候，使用 MapReduce 将会是一个很复杂的事情，比如你需要对数据进行很多预处理或转换，以便能够适应 MapReduce 的处理模式。另一方面，编写 MapReduce 程序，发布及运行作业都将是一个比较耗时的事情。Pig 的出现很好的弥补了这一不足。Pig 能够让你专心于数据及业务本身，而不是纠结于数据的格式转换以及 MapReduce 程序的编写。本质是上来说，使用 Pig 进行处理时，Pig 本身会在后台生成一系列的 MapReduce 操作来执行任务，但是这个过程对用户来说是透明的。

### 6.2 Pig 搭建过程

步骤一：下载并解压 Pig。把下载好的 pig-0.15.0.tar.gz，复制到 hadoop-2.6.0 文件夹里面解压，在终端输入如下内容。

```
cd $HADOOP_HOME
```

```
tar -zxvf pig-0.15.0.tar.gz
```

步骤二：移动位置，在终端输入如下内容。

`mv /home/hadoop-2.6.0/pig-0.15.0 /usr/local/hadoop/` (此时并没有 hadoop 文件夹，所以会生成一个 hadoop 文件夹)

步骤三：改变 Pig 的所有者，在终端输入如下内容。

`Chown -R hadoop:hadoop /usr/local/Hadoop/pig-0.15.0`

步骤四：配置环境变量，在终端输入如下内容。

`sudo gedit ~/.bashrc(sudo gedit /etc/profile)`

步骤五：在打开的文件里 ( `~/.bashrc` 或 `/etc/profile` ) 最后面加上：

`export PIG_HOME=/usr/local/hadoop/pig-0.15.0`

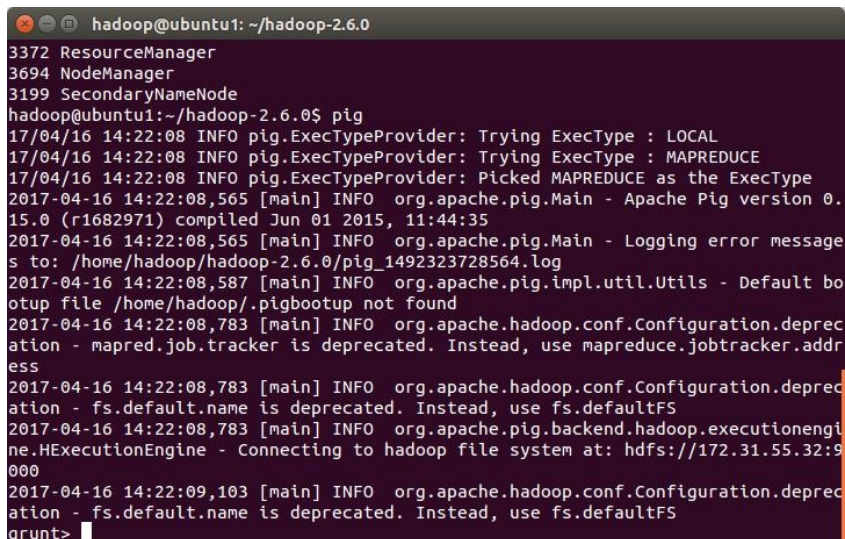
`export PATH=$PATH:${PIG_HOME}/bin`

步骤六：在终端生效 `/etc/profile`。

`source /etc/profile`

步骤七：验证是否安装成功。

输入 `pig`，若看到 `“grunt>”` 提示符，表明 `pig` 已经安装成功。成功启动图如图 6.1 所示。



```
hadoop@ubuntu1: ~/hadoop-2.6.0
3372 ResourceManager
3694 NodeManager
3199 SecondaryNameNode
hadoop@ubuntu1:~/hadoop-2.6.0$ pig
17/04/16 14:22:08 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
17/04/16 14:22:08 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
17/04/16 14:22:08 INFO pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2017-04-16 14:22:08,565 [main] INFO org.apache.pig.Main - Apache Pig version 0.
15.0 (r1682971) compiled Jun 01 2015, 11:44:35
2017-04-16 14:22:08,565 [main] INFO org.apache.pig.Main - Logging error message
s to: /home/hadoop/hadoop-2.6.0/pig_1492323728564.log
2017-04-16 14:22:08,587 [main] INFO org.apache.pig.impl.util.Utils - Default bo
otup file /home/hadoop/.pigbootup not found
2017-04-16 14:22:08,783 [main] INFO org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
2017-04-16 14:22:08,783 [main] INFO org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-04-16 14:22:08,783 [main] INFO org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to hadoop file system at: hdfs://172.31.55.32:9
000
2017-04-16 14:22:09,103 [main] INFO org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt>
```

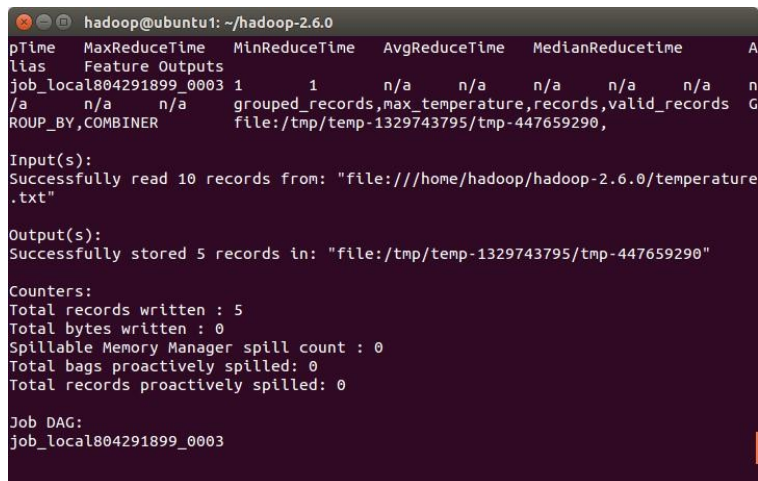
图6.1 成功启动图

步骤八：简单示例。新建一个 test.pig 的脚本文档，此文档的要和 temperature.txt 在同一文件夹中。text.pig 的内容为：

```
records = load 'temperature.txt' as (year:chararray,temperature:int);  
dump records;  
describe records;  
valid_records = filter records by temperature !=56;  
grouped_records = group valid_records by year;  
dump grouped_records;  
describe grouped_records;  
max_temperature = foreach grouped_records generate  
group,MAX(valid_records.temperature);  
dump max_temperature;
```

步骤九：在终端输入如下内容，示例运行图如图 6.2-6.3 所示。

pig -x local test.pig



```
hadoop@ubuntu1: ~/hadoop-2.6.0  
pTime  MaxReduceTime  MinReduceTime  AvgReduceTime  MedianReducetime  A  
lias    Feature  Outputs  
job_local1804291899_0003 1      1      n/a      n/a      n/a      n/a      n/a      n  
/a      n/a      n/a      grouped_records,max_temperature,records,valid_records  G  
ROUP_BY,COMBINER      file:/tmp/tmp-1329743795/tmp-447659290,  
  
Input(s):  
Successfully read 10 records from: "file:///home/hadoop/hadoop-2.6.0/temperature  
.txt"  
  
Output(s):  
Successfully stored 5 records in: "file:/tmp/tmp-1329743795/tmp-447659290"  
  
Counters:  
Total records written : 5  
Total bytes written : 0  
Spillable Memory Manager spill count : 0  
Total bags proactively spilled: 0  
Total records proactively spilled: 0  
  
Job DAG:  
job_local1804291899_0003
```

图6.2 示例运行图（1）

```

hadoop@ubuntu1: ~/hadoop-2.6.0
2017-04-16 14:57:43,964 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics -
Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already
initialized
2017-04-16 14:57:43,964 [main] INFO org.apache.hadoop.metrics.jvm.JvmMetrics -
Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already
initialized
2017-04-16 14:57:43,966 [main] INFO org.apache.pig.backend.hadoop.executionengi
ne.mapreduce.MapReduceLayer.MapReduceLauncher - Success!
2017-04-16 14:57:43,966 [main] INFO org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-04-16 14:57:43,966 [main] WARN org.apache.pig.data.SchemaTupleBackend - Sc
hemaTupleBackend has already been initialized
2017-04-16 14:57:43,969 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2017-04-16 14:57:43,970 [main] INFO org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(1989,33)
(1991,30)
(1992,35)
(1993,40)
(1994,20)
2017-04-16 14:57:43,984 [main] INFO org.apache.pig.Main - Pig script completed
in 2 seconds and 574 milliseconds (2574 ms)
hadoop@ubuntu1:~/hadoop-2.6.0$

```

图6.3 示例运行图 ( 2 )

## 7 Maven 搭建

### 7.1 Maven 简介

Maven 项目对象模型(POM)，可以通过一小段描述信息来管理项目的构建，报告和文档的软件项目管理工具。

Maven 除了以程序构建能力为特色之外，还提供高级项目管理工具。由于 Maven 的缺省构建规则有较高的可重用性，所以常常用两三行 Maven 构建脚本就可以构建简单的项目。由于 Maven 的面向项目的方法，许多 Apache Jakarta 项目发文时使用 Maven，而且公司项目采用 Maven 的比例在持续增长。

Maven 是一个项目管理工具，它包含了一个项目对象模型 (Project Object Model)，一组标准集合，一个项目生命周期(Project Lifecycle)，一个依赖管理系统(Dependency Management System)，和用来运行定义在生命周期阶段(phase)中插件(plugin)目标(goal)的逻辑。当你使用 Maven 的时候，你用一个明确定义的项目对象模型来描述你的项目，然后 Maven 可以应用横切的逻辑，这些逻辑来自一组共享的（或者自定义的）插件。

Maven 有一个生命周期，当你运行 `mvn install` 的时候被调用。这条命令告诉 Maven 执行一系列的有序的步骤，直到到达你指定的生命周期。遍历生命周期旅途中的一个影响就是，Maven 运行了许多默认的插件目标，这些目标完成了像编译和创建一个 JAR 文件这样的工作。

### 7.2 Maven 搭建过程

步骤一：下载并解压 Maven。把下载好的 `apache-maven-3.3.9-bin.tar.gz`，

复制到 hadoop-2.6.0 文件夹里面解压，在终端输入如下内容。

```
cd $HADOOP_HOME
```

```
tar zxvf apache-maven-3.3.9-bin.tar.gz
```

步骤二：配置环境变量，在终端输入如下内容。

```
sudo gedit ~/.bashrc(sudo gedit /etc/profile)
```

步骤三：在打开的文件里（ ~/.bashrc 或/etc/profile ）最后面加上：

```
export MAVEN_HOME=/home/hadoop/maven-3.3.9    // 安装目录
```

```
export PATH=${MAVEN_HOME}/bin:${PATH}
```

步骤四：在终端生效/etc/profile。

```
source /etc/profile
```

步骤五：验证是否安装成功

```
sudo apt-get install mvn
```

```
mvn -v
```



## 8 Hive 搭建

### 8.1 Hive 简介

Hive 是基于 Hadoop 的一个数据仓库工具，可以将结构化的数据文件映射为一张数据库表，并提供完整的 sql 查询功能，可以将 sql 语句转换为 MapReduce 任务进行运行。其优点是学习成本低，可以通过类 SQL 语句快速实现简单的 MapReduce 统计，不必开发专门的 MapReduce 应用，十分适合数据仓库的统计分析。

Hive 是建立在 Hadoop 上的数据仓库基础构架。它提供了一系列的工具，可以用来进行数据提取转化加载(ETL)，这是一种可以存储、查询和分析存储在 Hadoop 中的大规模数据的机制。Hive 定义了简单的类 SQL 查询语言，称为 HQL，它允许熟悉 SQL 的用户查询数据。同时，这个语言也允许熟悉 MapReduce 开发者的开发自定义的 mapper 和 reducer 来处理内建的 mapper 和 reducer 无法完成的复杂的分析工作。

### 8.2 Hive 搭建过程

步骤一：下载并解压 Hive。把下载好的 hive-1.2.1-bin.tar.gz，复制到 hadoop-2.6.0 文件夹里面解压，在终端输入如下内容（注：下载 hive 时要注意选择与 hadoop 版本兼容的）。

```
cd $HADOOP_HOME  
tar -zxvf apache-hive-1.2.1-bin.tar.gz  
cp apache-hive-1.2.1-bin hive-1.2.1
```

步骤二：配置环境变量，在终端输入如下内容。

`sudo gedit ~/.bashrc(sudo gedit /etc/profile)`

步骤三：在打开的文件里（`~/.bashrc` 或 `/etc/profile`）最后面加上：

```
export HIVE_HOME=/home/hadoop/hadoop-2.6.0/hive-1.2.1
```

```
export PATH=$PATH:$HIVE_HOME/bin
```

步骤四：在终端生效`/etc/profile`。

```
source /etc/profile
```

步骤五：修改 `hive-env.sh`（由于 Hive 使用了 Hadoop, 需要在 `hive-env.sh` 文件中指定 Hadoop 安装路径）。

```
export JAVA_HOME=/usr/java/jdk18.0_121
```

```
export HADOOP_HOME=/home/hadoop/hadoop-2.7.2
```

```
export HIVE_HOME=/home/hadoop/hadoop-2.7.2/hive-1.2.1
```

```
export
```

```
HIVE_CONF_DIR=/home/hadoop/hadoop-2.7.2/hive-1.2.1/conf
```

步骤六：将 `/home/hadoop/hadoop-2.6.0/hive-1.2.1/conf` 中的 `hive-default.xml.template` 复制到当前路径下并命名为 `hive-site.xml`，在终端输入如下内容。

```
cd /home/hadoop/hadoop-2.6.0/hive-1.2.1/conf
```

```
cp hive-default.xml.template hive-site.xml
```

步骤七：对 `hive-site.xml` 文档进行修改，手动添加以下内容（注意：账户和密码是使用 `mysql` 的账户密码）。

```
<property>
```

```
<name>javax.jdo.option.ConnectionURL</name>
```

```
<value>jdbc:mysql://localhost:3306/hive?createDatabaseIfNotExist=tr
```

```
ue</value>
```

```
<description>JDBC connect string for a JDBC
```

```

metastore</description>
</property>
<property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
    <description>Driver class name for a JDBC metastore</description>
</property>
<property>
    <name>javax.jdo.option.ConnectionUserName</name>
    <value>hadoop</value>
    <description>Username to use against metastore
database</description>
</property>
<property>
    <name>javax.jdo.option.ConnectionPassword</name>
    <value>321654</value>
    <description>password to use against metastore
database</description>
</property>

```

步骤八：配置 hive-site.xml，设置路径。将`${system:java.io.tmpdir}`替换为`/home/hadoop/hadoop-2.6.0/hive/tmpir`，`${system:user.name}`替换为`hadoop`。hive-site.xml 配置图如图 8.1-8.3 所示。

```

<property>
  <name>hive.exec.local.scratchdir</name>
  <value>/home/hadoop/hadoop-2.6.0/hive-1.2.1/tmpdir/hadoop</value>
  <description>Local scratch space for Hive jobs</description>
</property>
<property>
  <name>hive.downloaded.resources.dir</name>
  <value>/home/hadoop/hadoop-2.6.0/hive-1.2.1/tmpdir/${hive.session.id}
_resources</value>
  <description>Temporary local directory for added resources in the remote
file system.</description>
</property>

```

图8.1 hive-site.xml 配置图 ( 1 )

```

<property>
  <name>hive.querylog.location</name>
  <value>/home/hadoop/hadoop-2.6.0/hive-1.2.1/tmpdir/hadoop</value>
  <description>Location of Hive run time structured log file</description>
</property>

```

图8.2 hive-site.xml 配置图 ( 2 )

```

<property>
  <name>hive.server2.logging.operation.log.location</name>
  <value>/home/hadoop/hadoop-2.6.0/hive-1.2.1/tmpdir/hadoop/operation_logs</
value>
  <description>Top level directory where operation logs are stored if
logging functionality is enabled</description>
</property>

```

图8.3 hive-site.xml 配置图 ( 3 )

步骤九：在/home/hadoop/hadoop-2.6.0/hive-1.2.1/bin/hive-config.sh 中添加如下语句。hive-config.sh 配置图如图 8.4 所示。

```

export JAVA_HOME=/usr/local/java/jdk1.8.0_91
export HADOOP_HOME=/home/hadoop/hadoop-2.6.0
export HIVE_HOME=/home/hadoop/hadoop-2.6.0/hive-1.2.1

# for hive
export JAVA_HOME=/usr/local/java/jdk1.8.0_91
export HADOOP_HOME=/home/hadoop/hadoop-2.6.0
export HIVE_HOME=/home/hadoop/hadoop-2.6.0/hive-1.2.1

```

图 8.4 hive-config.sh 配置图

## 8.3 MySQL 安装

步骤一：下载安装。sudo apt-get install mysql-server mysql-client 。Mysql 下载安装图如图 8.5 所示。

```
hadoop@ubuntu1:~$ sudo apt-get install mysql-server mysql-client
[sudo] password for hadoop:
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
将会安装下列额外的软件包：
libaio1 libdbd-mysql-perl libdbi-perl libhtml-template-perl libmysqlclient18
libperl5.18 libterm-readkey-perl mysql-client-5.5 mysql-client-core-5.5
mysql-common mysql-server-5.5 mysql-server-core-5.5 perl perl-base
perl-modules
建议安装的软件包：
libnldb-perl libnet-daemon-perl libplrpc-perl libsql-statement-perl
libipc-sharedcache-perl tinyca mailx perl-doc libterm-readline-gnu-perl
libterm-readline-perl-perl libb-lint-perl libcanplus-dist-build-perl
libcanplus-perl libfile-checktree-perl liblog-message-perl
libobject-accessor-perl
下列【新】软件包将被安装：
libaio1 libdbd-mysql-perl libdbi-perl libhtml-template-perl libmysqlclient18
libterm-readkey-perl mysql-client mysql-client-5.5 mysql-client-core-5.5
mysql-common mysql-server mysql-server-5.5 mysql-server-core-5.5
下列软件包将被升级：
libperl5.18 perl perl-base perl-modules
升级了 4 个软件包，新安装了 13 个软件包，要卸载 0 个软件包，有 328 个软件包未被
升级。
```

8.5 Mysql 下载安装图

步骤二：安装完成后，使用命令检测是否成功安装（注：此命令必须进root 权限，否则会报 “未知命令” ）。获取 root 权限图如图 8.6 所

```
hadoop@ubuntu1:~$ sudo su
[sudo] password for hadoop:
root@ubuntu1:/home/hadoop# service mysql status
mysql start/running, process 1108
```

示。

图8.6 获取root 权限图

步骤三：建立 mysql 帐号。

方式一是以 root 身份进入 mysql 并为 hive 建立 mysql 的账号。Root 权限创建 mysql 帐号图如图 8.7 所示。然后尝试读取数据，读取数据图如图 8.8 所示。

```
mysql -u root -p
```

```
mysql> create user 'hadoop' identified by '321654';
```

```
root@ubuntu1: /home/hadoop
hadoop@ubuntu1:~$ sudo su
[sudo] password for hadoop:
root@ubuntu1:/home/hadoop# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 47
Server version: 5.5.54-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create user 'hadoop' identified by '321654';
```

图 8.7 root 创建 mysql 帐号图

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'hadoop'@'localhost'
IDENTIFIED BY '321654' WITH GRANT OPTION;
```

```
mysql> flush privileges;
```

```
mysql> exit
```

```
mysql> create user 'hadoop' identified by '321654';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'hadoop'@'localhost' IDENTIFIED BY '321654'
WITH GRANT OPTION;
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> exit;
Bye
root@ubuntu1:/home/hadoop#
```

图8.8 读取数据图

方式二是使用 hadoop 账号密码登录 MySQL。Hadoop 账户创建 mysql 帐号图如图 8.9 所示。登录后建立专用的数据库 ‘hive’，创建数据库图如

图 8.10 所示。

```
mysql -u hadoop -p
```

```

root@ubuntu1:/home/hadoop# mysql -u hadoop -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 50
Server version: 5.5.54-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```

图 8.9 Hadoop 账户创建 mysql 帐号图

creat database hive;

```

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database hive;
Query OK, 1 row affected (0.01 sec)

mysql>

```

图8.10 创建数据库图

步骤四：下载 JDBC 驱动包并建立软连接到 hive 根目录下的 lib 中( 下载后复制即可 )。

ln -s /home/hadoop/mysql-connector-java-5.1.7-bin.jar

/home/hadoop/hadoop-2.6.0/hive-1.2.1/lib/mysql-connector-java-5.1.7-bin.jar

步骤五：启动 hive 并进行测试，hive 测试图如图 8.11 所示。

```

hadoop@ubuntu1:~
hadoop@ubuntu1:~$ hive

Logging initialized using configuration in jar:file:/home/hadoop/hadoop-2.6.0/hive-1.2.1/lib/hive-common-1.2.1.jar!/hive-log4j.properties
hive> show tables;
OK
test
test1
Time taken: 0.6 seconds, Fetched: 2 row(s)
hive> create table test2(id int);
OK
Time taken: 0.294 seconds
hive> show tables;
OK
test
test1
test2
Time taken: 0.01 seconds, Fetched: 3 row(s)
hive>

```

图 8.11 hive 测试图

## 9 Spark 搭建

### 9.1 Spark 简介

Spark 诞生于 2009 年，其设计目标是实现快速、通用且可扩展的大数据分析引擎，解决大数据的快速、海量、价值和多样四大难题[6]。目前，广泛应用的分析模型是 MapReduce，但作为后起之秀的 Spark 在运行效率、运行模式都有较大的技术革新。

Spark 的发展历程是从 2009 年提出 Spark 这个概念进行研究，在 2010 年正式开源，在 2013 成为 Apache 基金项目，在 2016 年发布 2.0 版本。当前最新版是 2016 年年底发布的 2.1.0 版本，这是 Spark2.X 版本的第二版，本版本对 Structured Streaming 进行了重大突破，更加注重可用性。

Spark 利用弹性分布式数据集，进行数据的运算。它与 MapReduce 等数据流模型相比，迭代效率和编程方式上更加高效，它交互式查询 1TB 数据集可以在 5-7 秒内完成。

### 9.2 Spark 搭建过程

步骤一：下载并解压 Spark。把下载好的 spark-2.1.0-bin-hadoop2.6.tgz, 复制到 hadoop-2.6.0 文件夹里面解压，在终端输入如下内容。

```
cd $HADOOP_HOME  
tar -zxf spark-2.1.0-bin-hadoop2.7.tgz
```

步骤二：安装并获取权限，在终端输入如下内容。

```
mv spark-2.1.0-bin-hadoop2.6 spark-2.1.0 //移动并改名  
sudo chown -R hadoop:hadoop ./spark-2.1.0 //获取权限并安装
```

步骤三：配置 Spark。进入 /spark-2.1.0/conf 文件夹下，修改 spark-env.sh，



在终端输入如下内容。

```
cd $HADOOP_HOME/spark-2.1.0/conf
```

```
cp spark-env.sh.template spark-env.sh
```

在 spark-env.sh 最后面加上如下内容。

```
export
```

```
SPARK_DIST_CLASSPATH=$(/home/hadoop/hadoop-2.6.0/bin/hadoop  
classpath)
```

步骤四：Spark 简单应用，在终端输入如下内容。成功时显示 “Pi is roughly 3.142915714578573” 。

```
cd ./spark-2.1.0
```

```
./bin/run-example SparkPi 2>&1 | grep “Pi is roughly”
```

## 参考文献

- [1] <http://www.linuxidc.com/Linux/2016-12/138355.htm>
- [2] <http://blog.csdn.net/lifuxiangcaohui/article/details/39894265>
- [3] <http://blog.csdn.net/allen879/article/details/40461227>
- [4] <http://blog.csdn.net/tycoon1988/article/details/38866395>