

# 手动安装 Cloudera CDH4. 2

## 系统环境:

Linux master01 2.6.32-358.el6.i686 #1 SMP Thu Feb 21 21:50:49 UTC 2013 i686 i686 i386 GNU/Linux

## 安装版本:

hadoop-2.0.0-cdh4.2.0 [129M]

zookeeper-3.4.5-cdh4.2.0.tar.gz [16.1M]

hive-0.10.0-cdh4.2.0 [43.2M]

jdk-7-linux-i586.rpm [77.2M]

mysql-connector-java-5.1.18.tar.gz [3.65M]

Thrift.zip [71.7K]

sqoop-1.4.2-cdh4.2.0.tar.gz [6M]

## Hadoop 相关下载地址:

<http://archive.cloudera.com/cdh4/cdh/4/>

熊建邦整理

2015 年 4 月 10 日

第 1 页

# 目录

第 1 章 概要说明	4
1.1 Hadoop 是什么?	4
1.2 为什么选择 CDH 版本?	4
1.3 集群配置环境	4
1.4 网络结构图	5
第 2 章 安装 hadoop 环境	6
2.1 准备安装包	6
2.2 默认用户组 root:root	6
2.3 卸载自带的 jdk	6
2.4 安装和配置 jdk 环境	6
2.5 配置/etc/hosts	6
2.6 配置 ssh 无密码登陆	7
2.7 处理防火墙	7
2.8 将 hadoop-2.0.0-cdh4.2.0.zip 上传到/opt, 并解压缩	9
2.9 编辑 core-site.xml 文件	9
2.10 编辑 hdfs-site.xml 文件	9
2.11 编辑 slaves 文件	10
2.12 编辑 mapred-site.xml 文件	10
2.13 编辑 yarn-site.xml 文件	11
2.14 编辑.bashrc 文件	13
2.15 将 master01 机上的/opt/hadoop 拷贝到其他机器上	14
2.16 第一次启动 hadoop 需要先格式化 NameNode	14
2.17 在 master01 机上启动 hdfs:	14
2.18 在 master01 机上启动 mapreduce,historyserver	14
2.19 查看 master01 机的 MapReduce	15
2.20 查看 slave01,slave02 的节点	15
2.21 检查各台机器的集群进程	15
2.22 关闭服务	15
第 3 章 Zookeeper 安装	16
3.1 准备安装包	16
3.2 解压	16
3.3 修改 zoo.cfg 文件	16
3.4 修改环境变量	17
3.5 创建 data 文件夹及修改 myid 文件	17
3.6 将文件复制至其他机器	17
3.7 启动	18
3.8 检查是否成功	18
3.9 停止服务	18
3.10 参考文档	18
第 4 章 Hive 的安装	19
4.1 准备安装包	19
4.2 准备机器	19
4.3 访问 mysql	19
4.4 配置 hive-site.xml 文件, 将 meta 信息保存在 mysql 里	19
4.5 将 mysql-connector-java-5.1.18.tar.gz 解压	22
4.6 Mysql 的一些操作	22
4.7 查看日志记录	22

4.8 Hive 导入本地数据命令-----	22
第 5 章 Hive+Thrift+PHP 整合-----	23
5.1 准备安装包-----	23
5.2 编辑代码-----	23
5.3 启动 hiveserver-----	24
5.4 查看默认开启的 10000 端口-----	24
5.5 测试-----	24
5.6 出错提示及解决办法-----	24
第 6 章 sqoop 安装使用-----	25
6.1 准备安装包-----	25
6.2 前提工作-----	25
6.3 安装-----	25
6.4 放置 mysql 驱动包-----	25
6.5 修改 configure-sqoop 文件-----	25
6.6 将路径加入 PATH-----	25
6.7 使用测试-----	26
6.8 出错提示及解决办法-----	27
6.9 参考-----	27

# 第 1 章 概要说明

## 1.1 Hadoop 是什么？

Hadoop 一个分布式系统基础架构，由 Apache 基金会开发。用户可以在不了解分布式底层细节的情况下，开发分布式程序。充分利用集群的威力高速运算和存储。Hadoop 实现了一个分布式文件系统（Hadoop Distributed File System），简称 HDFS。HDFS 有着高容错性的特点，并且设计用来部署在低廉的（low-cost）硬件上。而且它提供高传输率（high throughput）来访问应用程序的数据，适合那些有着超大数据集（large data set）的应用程序。HDFS 放宽了（relax）POSIX 的要求（requirements）这样可以流的形式访问（streaming access）文件系统中的数据。

## 1.2 为什么选择 CDH 版本？

- CDH 基于稳定版 Apache Hadoop，并应用了最新 Bug 修复或者 Feature 的 Patch。  
Cloudera 常年坚持季度发行 Update 版本，年度发行 Release 版本，更新速度比 Apache 官方快，而且在实际使用过程中 CDH 表现无比稳定，并没有引入新的问题。
- Cloudera 官方网站上安装、升级文档详细，省去 Google 时间。
- CDH 支持 Yum/Apt 包，Tar 包，RPM 包，Cloudera Manager 四种方式安装
- 获取最新特性和最新 Bug 修复；安装维护方便，节省运维时间

## 1.3 集群配置环境

```
[root@master01 ~]# ls_b_release -a

LSBVersion:      :base-4.0-ia32:base-4.0-noarch:core-4.0-ia32:core-4.0-noarch:graphics-4.0-ia32:graphics-4.0-noarch:printing-4.0-ia32:printing-4.0-noarch

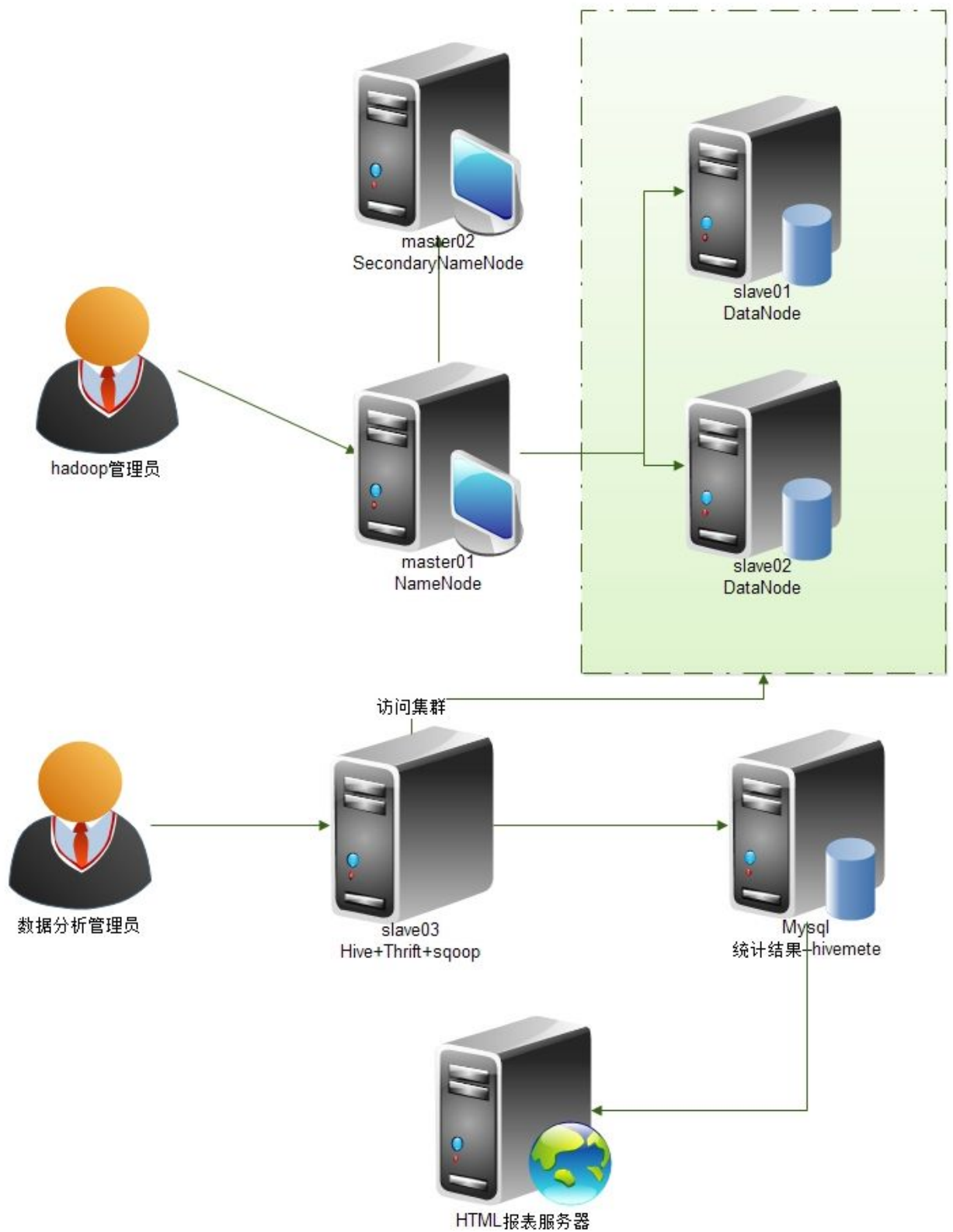
Distributor ID: CentOS

Description:     CentOS release 6.4 (Final)

Release:         6.4

Codename:        Final
```

## 1.4 网络结构图



## 第 2 章 安装 hadoop 环境

### 2.1 准备安装包

jdk-7-linux-i586.rpm [77.2M]

hadoop-2.0.0-cdh4.2.0 [129M] 此安装包URL下载:

<http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>

### 2.2 默认用户组 root:root

### 2.3 卸载自带的 jdk

```
[root@master01 local]# rpm -qa | grep jdk
```

```
java-1.7.0-openjdk-1.7.0.9-2.3.4.1.el6_3.i686
```

```
yum -y remove java-1.7.0-openjdk-1.7.0.9-2.3.4.1.el6_3.i686
```

```
yum -y remove java-1.6.0-openjdk-1.6.0.0-1.50.1.11.5.el6_3.i686
```

### 2.4 安装和配置 jdk 环境

```
[root@master01 local]# rpm -ivh jdk-7-linux-i586.rpm
```

```
Preparing... ##### [100%]
```

```
1:jdk ##### [100%]
```

---

#### 注意

下面有设置 JAVA\_HOME 环境的清单，写在 ~/.bashrc.sh 文件里

另外请注意：生产环境下一一般为 64 位机，请下载相应的 64 位 JDK 包进行安装

---

### 2.5 配置/etc/hosts

```
vi /etc/hosts
```

```
192.168.2.18 master01
```

```
192.168.2.19 master02
```

```
192.168.2.163 slave01
```

```
192.168.2.38 slave02
```

192.168.2.212 slave03

---

 注意：其他机器也要修改

```
rsync -vzrtopgu --progress /etc/hosts 192.168.2.38:/etc/hosts
```

---


## 2.6 配置 ssh 无密码登陆

```
ssh-keygen -t rsa
```

```
ssh-copy-id -i ~/.ssh/id_rsa.pub root@slave01
```

```
ssh-copy-id -i ~/.ssh/id_rsa.pub root@slave02
```

---

 注意

Master01 机本身也要设置一下哦！

```
cd ~
```


```
cat id_rsa.pub >>authorized_keys
```

---

## 2.7 处理防火墙

```
service iptables stop
```

---

 说明

如果不关闭防火墙，让 datanode 通过 namenode 机的访问，请配置 slave01,slave02 等相关机器的 iptables 表，各台机器都要能互相访问

```
vi /etc/sysconfig/iptables
```

添加：

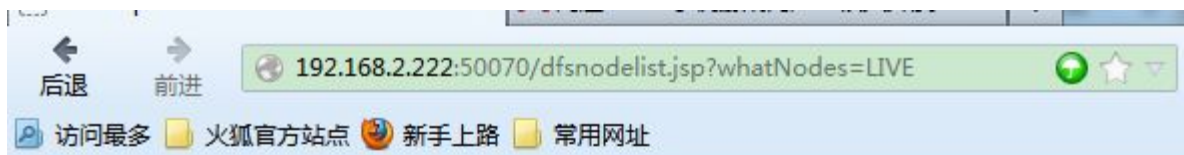
```
-I INPUT -s 192.168.2.18 -j ACCEPT
```

```
-I INPUT -s 192.168.2.38 -j ACCEPT
```

```
-I INPUT -s 192.168.2.87 -j ACCEPT
```

开启 master01 的 8088 和 50070 端口，方便 WEB 访问 namenode 和 mapreduce

---



## NameNode 'master01:8020'

**Started:** Fri Jul 19 09:28:24 CST 2013  
**Version:** 2.0.0-cdh4.2.0, 8bce4bd28a464e0a92950c50ba01a9deb1d85  
**Compiled:** Fri Feb 15 10:42:32 PST 2013 by jenkins from Unknown  
**Upgrades:** There are no upgrades in progress.  
**Cluster ID:** CID-55ab35e0-4072-43a1-9dde-0e3d863fb8da  
**Block Pool ID:** BP-626960086-192.168.2.190-1373680798549

[Browse the filesystem](#)

[NameNode Logs](#)

[Go back to DFS home](#)

Live Datanodes : 2

Node	Last Contact	Admin State	Configured Capacity (GB)	Used (GB)	Non DFS Used (GB)	R
slave01	1	In Service	36.95	0.19	6.89	
slave02	2	In Service	36.95	0.19	6.87	

图1 Hadoop, 2013.

**Nodes of the cluster**

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Active Nodes	Decommissioned
7	0	0	7	0	0 KB	16 GB	0 KB	2	0

Show 20 entries

Rack	Node State	Node Address	Node HTTP Address	Health-status	Last health-update	Health
/default-rack	RUNNING	slave02:40813	slave02:8042	Healthy	2-Jul-2013 15:22:47	
/default-rack	RUNNING	slave01:42501	slave01:8042	Healthy	2-Jul-2013 15:23:25	

Showing 1 to 2 of 2 entries

图2



## 2.8 将 **hadoop-2.0.0-cdh4.2.0.zip** 上传到/opt, 并解压缩

```
tar xzvf hadoop-2.0.0-cdh4.2.0.tar.gz
mv hadoop-2.0.0-cdh4.2.0 hadoop
cd hadoop/etc/hadoop/
```

## 2.9 编辑 **core-site.xml** 文件

```
vi core-site.xml

<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://master01</value>
  </property>
  <property>
    <name>fs.trash.interval</name>
    <value>10080</value>
  </property>
  <property>
    <name>fs.trash.checkpoint.interval</name>
    <value>10080</value>
  </property>
</configuration>
```

## 2.10 编辑 **hdfs-site.xml** 文件

```
vi hdfs-site.xml

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

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

        <value>/opt/data/hadoop-${user.name}</value>

    </property>

    <property>

        <name>dfs.namenode.http-address</name>

        <value>master01:50070</value>

    </property>

    <property>

        <name>dfs.secondary.http.address</name>

        <value>master02:50090</value>

    </property>

    <property>

        <name>dfs.webhdfs.enabled</name>

        <value>true</value>

    </property>

</configuration>
```

## 2.11 编辑 slaves 文件

```
vi slaves
```

```
slave01
```

```
slave02
```

## 2.12 编辑 mapred-site.xml 文件

```
cp mapred-site.xml.template mapred-site.xml
```

```
vi mapred-site.xml
```

```
<configuration>

<property>

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

    <value>yarn</value>

</property>
```

```
<property>
    <name>mapreduce.jobhistory.address</name>
    <value>master01:10020</value>
</property>
<property>
    <name>mapreduce.jobhistory.webapp.address</name>
    <value>master01:19888</value>
</property>
</configuration>
```

## 2.13 编辑 yarn-site.xml 文件

`vi yarn-site.xml`

```
<configuration>
<!-- Site specific YARN configuration properties -->
<property>
    <name>yarn.resourcemanager.resource-tracker.address</name>
    <value>master01:8031</value>
</property>
<property>
    <name>yarn.resourcemanager.address</name>
    <value>master01:8032</value>
</property>
<property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>master01:8030</value>
</property>
<property>
    <name>yarn.resourcemanager.admin.address</name>
    <value>master01:8033</value>
</property>
```

```

<property>
  <name>yarn.resourcemanager.webapp.address</name>
  <value>master01:8088</value>
</property>
<property>
  <description>Classpath for typical applications.</description>
  <name>yarn.application.classpath</name>
  <value>$HADOOP_CONF_DIR,$HADOOP_COMMON_HOME/share/hadoop/common/*,
    $HADOOP_COMMON_HOME/share/hadoop/common/lib/*,
    $HADOOP_HDFS_HOME/share/hadoop/hdfs/*,$HADOOP_HDFS_HOME/share/hadoop/hdfs/lib/
    *,
    $YARN_HOME/share/hadoop/yarn/*,$YARN_HOME/share/hadoop/yarn/lib/*,
    $YARN_HOME/share/hadoop/mapreduce/*,$YARN_HOME/share/hadoop/mapreduce/lib/*</valu
e>
</property>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce.shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
  <name>yarn.nodemanager.local-dirs</name>
  <value>/opt/data/yarn/local</value>
</property>
<property>
  <name>yarn.nodemanager.log-dirs</name>
  <value>/opt/data/yarn/logs</value>

```

```
</property>

<property>

  <description>Where to aggregate logs</description>

  <name>yarn.nodemanager.remote-app-log-dir</name>

  <value>/opt/data/yarn/logs</value>

</property>

<property>

  <name>yarn.app.mapreduce.am.staging-dir</name>

  <value>/user</value>

</property>

</configuration>
```

## 2.14 编辑.bashrc 文件

```
cd ~

vi .bashrc

#export LANG=zh_CN.utf8

export JAVA_HOME=/usr/java/jdk1.7.0

export JRE_HOME=$JAVA_HOME/jre

export CLASSPATH=.:$JAVA_HOME/lib:$JRE_HOME/lib:$JRE_HOME/lib/tools.jar

export HADOOP_HOME=/opt/hadoop

export HIVE_HOME=/opt/hive

export HBASE_HOME=/opt/hbase

export HADOOP_MAPRED_HOME=${HADOOP_HOME}

export HADOOP_COMMON_HOME=${HADOOP_HOME}

export HADOOP_HDFS_HOME=${HADOOP_HOME}

export YARN_HOME=${HADOOP_HOME}

export HADOOP_YARN_HOME=${HADOOP_HOME}

export HADOOP_CONF_DIR=${HADOOP_HOME}/etc/hadoop

export HDFS_CONF_DIR=${HADOOP_HOME}/etc/hadoop

export YARN_CONF_DIR=${HADOOP_HOME}/etc/hadoop
```

```
export
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin:$HADOOP_HOME/sbin:$HBASE_HOME/bin:$HIVE_HOME/bin
source .bashrc
```

## 2.15 将 master01 机上的/opt/hadoop 拷贝到其他机器上

```
rsync -vzrtopgu --progress hadoop slave01:/opt/
```

```
rsync -vzrtopgu --progress hadoop slave02:/opt/
```

或者

```
rsync -vzrtopgu --progress hadoop 192.168.2.38:/opt/
```

```
rsync -vzrtopgu --progress hadoop 192.168.2.163:/opt/
```

---

### rsync 命令参数解释

- v, --verbose 详细模式输出
- z, --compress 对备份的文件在传输时进行压缩处理
- r, --recursive 对子目录以递归模式处理
- t, --times 保持文件时间信息
- o, --owner 保持文件属主信息
- p, --perms 保持文件权限
- g, --group 保持文件属组信息
- u, --update 仅仅进行更新，也就是跳过所有已经存在于 DST，并且文件时间晚于要备份的文件。(不覆盖更新的文件)

---

## 2.16 第一次启动 hadoop 需要先格式化 NameNode

```
/opt/hadoop/bin/hadoop namenode -format
```

---

### 说明：

该操作只做一次。当修改了配置文件时，需要重新格式化

---

## 2.17 在 master01 机上启动 hdfs:

```
/opt/hadoop/sbin/start-dfs.sh
```

## 2.18 在 master01 机上启动 mapreduce,historyserver

```
/opt/hadoop/sbin/start-yarn.sh
```

`/opt/hadoop/sbin/mr-jobhistory-daemon.sh start historyserver`

## 2.19 查看 master01 机的 MapReduce

<http://192.168.2.18:8088/cluster>

## 2.20 查看 slave01,slave02 的节点

<http://192.168.2.163:8042/node/node>

## 2.21 检查各台机器的集群进程

```
[root@master01 ~]# jps
```

5389 NameNode

5980 Jps

5710 ResourceManager

7032 JobHistoryServer

```
[root@slave01 ~]# jps
```

3187 Jps

3124 SecondaryNameNode

```
[root@slave02~]# jps
```

3187 Jps

3124 DataNode

5711 NodeManager

## 2.22 关闭服务

`/opt/hadoop/sbin/stop-all.sh`

## 第 3 章 Zookeeper 安装

### 3.1 准备安装包

```
zookeeper-3.4.5-cdh4.2.0.tar.gz
```

### 3.2 解压

```
tar xzvf zookeeper-3.4.5-cdh4.2.0.tar.gz  
mv zookeeper-3.4.5-cdh4.2.0 zookeeper
```

### 3.3 修改 zoo.cfg 文件

```
cd conf/  
cp zoo_sample.cfg zoo.cfg  
vi zoo.cfg  
  
# 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=/opt/zookeeper/data  
#dataLogDir=/opt/zookeeper/log  
# the port at which the clients will connect  
clientPort=2181  
  
#  
  
# Be sure to read the maintenance section of the  
# administrator guide before turning on autopurge.
```



```
#
# http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc_maintenance
#
# The number of snapshots to retain in dataDir
#autopurge.snapRetainCount=3
# Purge task interval in hours
# Set to "0" to disable auto purge feature
#autopurge.purgeInterval=1
server.1=master01:2888:3888
server.2=master02:2888:3888
server.3=slave01:2888:3888
server.4=slave02:2888:3888
```

### 3.4 修改环境变量

```
vi ~/.bashrc
export ZOOKEEPER_HOME=/opt/zookeeper
export PATH=$PATH:$ZOOKEEPER_HOME/bin
```

### 3.5 创建 data 文件夹及修改 myid 文件

```
mkdir /opt/zookeeper/data
touch myid
vi myid
第一台机器写入数字 1
第二台机器写入数字 2
依此类推
```

### 3.6 将文件复制至其他机器

```
rsync -vzrtopgu --progress zookeeper master02:/opt/
rsync -vzrtopgu --progress zookeeper slave01:/opt/
rsync -vzrtopgu --progress zookeeper slave02:/opt/
```

## 3.7 启动

```
sh /opt/zookeeper/bin/zkServer.sh start
```

```
[root@master01 zookeeper]# jps
```

```
3459 JobHistoryServer
```

```
6259 Jps
```

```
2906 NameNode
```

```
3171 ResourceManager
```

```
6075 QuorumPeerMain
```

## 3.8 检查是否成功

```
/opt/zookeeper/bin/zkCli.sh -server master01:2181
```

或者

```
sh /opt/zookeeper/bin/zkServer.sh stop
```

## 3.9 停止服务

```
sh /opt/zookeeper/bin/zkServer.sh stop
```

## 3.10 参考文档

<http://archive.cloudera.com/cdh4/cdh/4/zookeeper-3.4.5-cdh4.2.0/>

## 第 4 章 Hive 的安装

### 4.1 准备安装包

hive-0.10.0-cdh4.2.0 [43.2M]

mysql-connector-java-5.1.18.tar.gz [3.65M]

### 4.2 准备机器

**slave03** 机器，安装 **hive+thrift+sqoop**，专门作为数据分析用途。

### 4.3 访问 mysql

和mysql整合前，请务必配置好各机器间能访问Mysql服务器机

```
GRANT select, insert, update, delete ON *.* TO 'hadoop'@'slave01' IDENTIFIED BY 'hadoop';
```

```
GRANT select, insert, update, delete ON *.* TO 'hadoop'@'slave01' IDENTIFIED BY 'hadoop';
```

```
GRANT select, insert, update, delete ON *.* TO 'hadoop'@'slave01' IDENTIFIED BY 'hadoop';
```

```
flush privileges;
```

```
show grants for 'hive'@'slave03';
```

```
revoke all on *.* from 'hadoop'@'slave01';
```

```
drop user 'hive'@'slave03';
```

---

#### 说明

测试环境下，本人仍然用 **slave03** 机做 **mysql** 服务器。在实际生产环境中，建议用专门的机器做 **Mysql**。

---

### 4.4 配置 hive-site.xml 文件，将 meta 信息保存在 mysql 里

```
cd /opt/hive
```

```
vi hive-site.xml
```

```
<?xml version="1.0"?>
```

```
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
```

```
<configuration>
```

```
<property>
```

```
<name>javax.jdo.option.ConnectionURL</name>
<value>jdbc:mysql://slave03:3306/hive?createDatabaseIfNotExist=true&characterEncoding=UTF-8</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>hadoop</value>
  <description>password to use against metastore database</description>
</property>
<property>
  <name>mapred.job.tracker</name>
  <value>master01:8031</value>
</property>
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
</property>
<property>
  <name>hive.metastore.warehouse.dir</name>
  <value>/opt/data/warehouse-${user.name}</value>
```

```
<description>location of default database for the warehouse</description>
</property>
<property>
  <name>hive.exec.scratchdir</name>
  <value>/opt/data/hive-${user.name}</value>
  <description>Scratch space for Hive jobs</description>
</property>
<property>
  <name>hive.querylog.location</name>
  <value>/opt/data/querylog-${user.name}</value>
  <description>
    Location of Hive run time structured log file
  </description>
</property>
<property>
  <name>hive.support.concurrency</name>
  <description>Enable Hive's Table Lock Manager Service</description>
  <value>>false</value>
</property>
<property>
  <name>hive.hwi.listen.host</name>
  <value>master01</value>
  <description>This is the host address the Hive Web Interface will listen on</description>
</property>
<property>
  <name>hive.hwi.listen.port</name>
  <value>9999</value>
  <description>This is the port the Hive Web Interface will listen on</description>
</property>
<property>
  <name>hive.hwi.war.file</name>
```

```
<value>lib/hive-hwi-0.10.0-cdh4.2.0.war</value>

<description>This is the WAR file with the jsp content for Hive Web Interface</description>

</property>

</configuration>
```

## 4.5 将 mysql-connector-java-5.1.18.tar.gz 解压


```
tar xzvf mysql-connector-java-5.1.18.tar.gz
mv mysql-connector-java-5.1.18-bin.jar /opt/hive/lib
```

## 4.6 Mysql 的一些操作

```
create database hive;

alter database hive character set latin1;
```

---

 注意:

如果不设置上述命令，则会出现如下：

Specified key was too long; max key length is 767 bytes

---

## 4.7 查看日志记录

```
tail /tmp/root/hive.log
```

## 4.8 Hive 导入本地数据命令

- 1) CREATE TABLE mytest2(num INT, name STRING) COMMENT 'only a test'  
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' STORED AS  
TEXTFILE;
- 2) LOAD DATA LOCAL INPATH '/var/22.txt' INTO TABLE mytest2;

## 第 5 章 Hive+Thrift+PHP 整合

### 5.1 准备安装包

Thrift.zip [71.7K] 下载URL: <http://download.csdn.net/detail/jiedushi/3409880>

PHP安装, 略过

### 5.2 编辑代码

vi test.php

```
<?php
    $GLOBALS['THRIFT_ROOT'] = '/home/wwwroot/Thrift/';
    require_once $GLOBALS['THRIFT_ROOT'] .
'packages/hive_service/ThriftHive.php';
    require_once $GLOBALS['THRIFT_ROOT'] . 'transport/TSocket.php';
    require_once $GLOBALS['THRIFT_ROOT'] . 'protocol/TBinaryProtocol.php';

    $transport = new TSocket('slave03', 10000);
    $protocol = new TBinaryProtocol($transport);
    $client = new ThriftHiveClient($protocol);
    $transport->open();

    #$client->execute('add jar /opt/hive/lib/hive-contrib-0.10.0-cdh4.2.0.jar ');
    $client->execute("LOAD DATA LOCAL INPATH '/var/22.txt' INTO TABLE mytest2");
    $client->execute("SELECT COUNT(1) FROM mytest2");
    var_dump($client->fetchAll());
    $transport->close();

?>
```



说明:

/var/22.txt 文件内容为:

1	jj
2	kk

---

## 5.3 启动 hiveserver

```
/opt/hive/bin/hive --service hiveserver >/dev/null 2>/dev/null &
```

## 5.4 查看默认开启的 10000 端口

```
netstat -lntp|grep 10000
```

## 5.5 测试

```
php test.php
```

## 5.6 出错提示及解决办法

- Warning: stream\_set\_timeout(): supplied argument is not a valid stream resource in /home/wwwroot/Thrift/transport/TSocket.php on line 213

修改 php.ini 中的 disable\_functions

disable\_functions =

passthru,exec,system,chroot,scandir,chgrp,chown,shell\_exec,proc\_get\_status,ini\_alter,ini\_alter,ini\_restore,dl,openlog,syslog,readlink,symlink,popepassthru



## 第 6 章 sqoop 安装使用

### 6.1 准备安装包

sqoop-1.4.2-cdh4.2.0.tar.gz [6M]

### 6.2 前提工作

按第一章的介绍步骤配置好hadoop，环境变量HADOOP\_HOME已经设置好。

### 6.3 安装

```
cd /opt/  
tar xzvf sqoop-1.4.2-cdh4.2.0.tar  
mv sqoop-1.4.2-cdh4.2.0 sqoop
```

### 6.4 放置 mysql 驱动包

将mysql-connector-java-5.1.18-bin.jar包放至/opt/sqoop/lib下

### 6.5 修改 configure-sqoop 文件

```
vi /opt/sqoop/bin/configure-sqoop
```

因为没安装hbase，请注释

```
#!/bin/sh  
if [ ! -d "${HBASE_HOME}" ]; then  
#   echo "Warning: $HBASE_HOME does not exist! HBase imports will fail."  
#   echo 'Please set $HBASE_HOME to the root of your HBase installation.'  
#fi
```

### 6.6 将路径加入 PATH

```
vi ~/.bashrc
```

```
export  
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin:$HADOOP_HOME/sbin:$HBASE_HOME/  
bin:$HIVE_HOME/bin:$ANT_HOME/bin:/opt/sqoop/bin
```

## 6.7 使用测试

- 列出 mysql 数据库中的所有数据库命令

```
sqoop list-databases --connect jdbc:mysql://slave03:3306/ --username hadoop --password hadoop
```

- 列出表名:

```
sqoop list-tables -connect jdbc:mysql://slave03:teiren -username hadoop -password hadoop
```

- 将关系型数据的表结构复制到 hive 中

```
sqoop create-hive-table --connect jdbc:mysql://master01:3306/teiren --table 996com_area --username hadoop --password hadoop --hive-table teiren_996com_area
```

- 从关系数据库导入文件到 hive 中

```
sqoop import -connect jdbc:mysql://slave03:teiren -username hadoop -password hadoop -table sp_log_fee -hive-import --hive-table hive_log_fee --split-by id -m 4
```

---

### 参照

一般导入:

```
import \  
  --append \  
  --connect $DS_BJ_HOTBACKUP_URL \  
  --username $DS_BJ_HOTBACKUP_USER \  
  --password $DS_BJ_HOTBACKUP_PWD \  
  --table 'seven_book_sync' \  
  --where "create_date >= '${par_31days}' and create_date < '${end_date}'" \  
  --hive-import \  
  --hive-drop-import-delims \  
  --hive-table ${hive_table} \  
  --m 1
```

//可以点分法识别 schema.table

以时间作为增量条件是最好的办法

并行导入:

```
sqoop import --append --connect $CONNECTURL --username $ORACLENAME --password $ORACLEPASSWORD --target-dir $hdfsPath --m 12 --split-by CLIENTIP --table $oracleTableName --columns $columns --fields-terminated-by '\001' --where "data_desc='2011-02-26'"
```

增量导入:

```
sqoop import --connect jdbc:mysql://master01:3306/teiren --username hadoop --password hadoop --table 996com_area --columns "id,name,reid,disorder" --direct --hive-import --hive-table 996com_area --incremental append --check-column id --last-value 0
```

```
sqoop job --exec area_import
```

以上为网上找来的命令，经测试，不起作用。留着仅供参考。

---

- 将 hive 中的表数据导出到 mysql 中

```
sqoop export --connect jdbc:mysql://master01:3306/teiren --username hadoop --password hadoop --table mytest2 --export-dir /opt/data/warehouse-root/teiren_996com_area
```

---

 备注

分区保存：/user/hive/warehouse/uv/dt=2011-08-03

---

## 6.8 出错提示及解决办法

- Encountered IOException running import job:  
org.apache.hadoop.fs.FileAlreadyExistsException: Output directory  
hdfs://master01/user/root/996com\_area **already exists**  
  
*/opt/hadoop/bin/hadoop fs -rm -r /user/root/996com\_area*

## 6.9 参考

<http://archive.cloudera.com/cdh/3/sqoop/SqoopUserGuide.html>

<http://sqoop.apache.org/docs/1.4.2/SqoopUserGuide.html>