

# hadoop configuration

关于 *hadoop* 的配置笔记

屈庆磊

quqinglei@icloud.com

2013 年 7 月

## 目录

<b>1 配置 JAVA 环境</b>	<b>1</b>
1.1 配置 JAVA_HOME . . . . .	1
<b>2 Hadoop 安装包</b>	<b>1</b>
<b>3 Hadoop 集群安装实例</b>	<b>1</b>
3.1 说明 . . . . .	1
3.2 第二步, 配置 hosts . . . . .	2
3.3 第三步, 配置 JAVA_HOME 见上述章节 . . . . .	2
3.4 第四步, 建立新用户, 并解压安装包 . . . . .	2
3.5 第四步, 配置 ssh 无密码访问 . . . . .	2
3.6 第五步, 更改 Hadoop 配置文件 . . . . .	3
3.7 第六步, 把 hadoop 打包, 复制到其他两台机器 . . . . .	4
3.8 完毕 . . . . .	4
<b>4 编译 Hadoop 的 FUSE 模块, 机器一定要能上网</b>	<b>4</b>
4.1 编译前的软件依赖安装 . . . . .	5
4.2 下载 ant 包, 编译需要它 . . . . .	5
4.3 设置环境变量 . . . . .	5
4.3.1 更新环境变量 . . . . .	5
4.3.2 编译 HDFS . . . . .	5
4.4 编译 fuse_dfs . . . . .	5

## 1 配置 JAVA 环境

### 1.1 配置 JAVA\_HOME

需要根据系统位数下载不同的安装包，如：32 位的 `jdk-7u25-linux-i586.gz`，或者 64 位的 `jdk-7u25-linux-x64.tar.gz` 建议去甲骨文官网下载。

例子：

```
tar xzv jdk-7u25-linux-i586.gz -C /opt
cd /opt
ln -s jdk1.7.0_25 jdk
echo "export JAVA_HOME=/opt/jdk" >> /etc/profile
source /etc/profile
```

## 2 Hadoop 安装包

由于在 *Linux* 下有很多安装方式，为了统一，本次采用解压包介绍。去 *Hadoop* 的官网，找到 *Hadoop* 的下载地址，在稳定版本里挑一个下载，本例中采用的包为：`hadoop-1.1.2.tar.gz`

## 3 Hadoop 集群安装实例

本例中有三台机器参与，一台 *master*，两台 *slave*，配置比较简单

### 3.1 说明

192.168.1.5 此台机器作为 *master*

192.168.1.6 此台机器作为 *slave01*

192.168.1.7 此台机器作为 *slave02*

### 3.2 第二步，配置 hosts

配置 *Hosts*：编辑 *master* 机器的 `/etc/hosts` 文件，添加内容为如下所示：

```
192.168.1.5 master
192.168.1.6 slave01
192.168.1.7 slave02
```

同步此配置文件到其它两台机器，保持一致即可。

### 3.3 第三步，配置 JAVA\_HOME 见上述章节

### 3.4 第四步，建立新用户，并解压安装包

```
useradd -m hadoop
passwd hadoop # 设置密码
su hadoop
cd /home/hadoop
tar xzf hadoop-1.1.2.tar.gz
ln -s hadoop-1.1.2 hadoop
#hadoop 的家路径此时为：/home/hadoop/hadoop
```

### 3.5 第四步，配置 ssh 无密码访问

其实就是交换公匙

```
# machine master, 使用 hadoop 用户
ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys
scp ~/.ssh/id_dsa.pub hadoop@192.168.1.6:/home/hadoop/.ssh/authorized_keys_master
scp ~/.ssh/id_dsa.pub hadoop@192.168.1.6:/home/hadoop/.ssh/authorized_keys_master

# slave01, 使用 hadoop 用户
ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys
cat ~/.ssh/authorized_keys_master >> ~/.ssh/authorized_keys
scp ~/.ssh/id_dsa.pub hadoop@192.168.1.5:/home/hadoop/.ssh/authorized_keys_slave01

# slave02, 使用 hadoop 用户
ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys
cat ~/.ssh/authorized_keys_master >> ~/.ssh/authorized_keys
scp ~/.ssh/id_dsa.pub hadoop@192.168.1.5:/home/hadoop/.ssh/authorized_keys_slave02

# master, 使用 hadoop 用户
cat ~/.ssh/authorized_keys_slave01 >> ~/.ssh/authorized_keys
cat ~/.ssh/authorized_keys_slave02 >> ~/.ssh/authorized_keys
```

注意：在 CentOS 下做的同学需要注意，一定要看看 `~/.ssh/authorized_keys` 的权限是不是 644，如果不是 644，请 `chmod 644 authorized_keys`

### 3.6 第五步，更改 Hadoop 配置文件

```
su hadoop
cd /home/hadoop/hadoop/conf/
# 需要更改：masters, slaves, hadoop-env.sh, core-site.xml, hdfs-site.xml, mapred-site.xml
```

更改结果如下所示：

#masters 文件内容

master

#slaves 文件内容

slave01

slave02

#hadoop-env.sh

# 仅仅需要更改 JAVA\_HOME 那一行，更改为 JAVA\_HOME 的实际路径，本例中我们采用的 JAVA\_HOME 为：

export JAVA\_HOME=/opt/jdk

#core-site.xml 文件内容

<?xml version="1.0"?>

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

<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>

<name>fs.default.name</name>

<value>hdfs://master:9000</value>

</property>

</configuration>

#hdfs-site.xml 文件内容

<?xml version="1.0"?>

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

<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>

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

<value>/home/hadoop/hadoopfs/name1,/home/hadoop/hadoopfs/name2</value>

</property>

<property>

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

<value>/home/hadoop/hadoopfs/data1,/home/hadoop/hadoopfs/data2</value>

</property>

<property>

<name>dfs.replication</name>

<value>2</value>

</property>

</configuration>

#mapred-site.xml 文件内容

<?xml version="1.0"?>

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

```
<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
<name>mapred.job.tracker</name>
<value>master:9200</value>
</property>
</configuration>
```

### 3.7 第六步，把 hadoop 打包，复制到其他两台机器

在配置完一台机器后，直接可以把配置好的 *hadoop* 复制到其他机器上，当然在其他的机器上的位置应该是一样的

### 3.8 完毕

基本配置完毕，在主机 `/home/hadoop/hadoop/bin` 下执行：`./hadoop fs namenode -format` 格式化文件系统，然后就可以启动 *hadoop* 集群了，`./start-all.sh` 即可启动 *hadoop* 集群

```
cd /home/hadoop/hadoop
./hadoop fs namenode -format
./start-all.sh
```

如果成功，我们可以在其他机器上都能看到 *hadoop* 的进程，并且在 `/home/hadoop/` 会看到 `/home/hadoop/hadoopfs` 的路径出现。

## 4 编译 Hadoop 的 FUSE 模块，机器一定要能上网

如果想在其他机器上挂载 *HDFS*，则需要编译 *fuse\_dfs* 模块

### 4.1 编译前的软件依赖安装

```
# 如果系统是基于 Redhat 的，请使用 yum 安装，如果基于 debian，使用 apt-get 安装，具体如下：
yum install install automake autoconf m4 libtool pkgconfig fuse fuse-devel fuse-libs

apt-get install automake autoconf m4 libtool libextutils-pkgconfig-perl \
    fuse libfuse-dev lrzsz build-essential
```

### 4.2 下载 ant 包，编译需要它

```
tar xzf apache-ant-1.9.1-bin.tar.gz -C /opt
cd /opt
ln -s apache-ant-1.9.1 ant
```

### 4.3 设置环境变量

把下面环境变量写到 `/etc/profile` 文件里

```
# add below to /etc/profile
export JAVA_HOME=/opt/jdk
export HADOOP_HOME=/home/hadoop/hadoop
export OS_ARCH=i386 #or amd64
export OS_BIT=32 #or 64
export ANT_HOME=/opt/ant
export PATH=$PATH:$ANT_HOME/bin
export LD_LIBRARY_PATH=$JAVA_HOME/jre/lib/$OS_ARCH/server:\
    $HADOOP_HOME/build/c++/Linux-$OS_ARCH-$OS_BIT/lib:/usr/local/lib:/usr/lib
```

#### 4.3.1 更新环境变量

```
source /etc/profile
```

#### 4.3.2 编译 HDFS

```
cd /home/hadoop/hadoop # 进入 hadoop 的家路径
ant compile-c++-libhdfs -Dlibhdfs=1 -Dcompile.c++=1 # 编译 libhdfs，机器一定要能上网，否则无法编译
```

### 4.4 编译 fuse\_dfs

```
# 进入 hadoop 家路径，执行
ln -s c++/Linux-$OS_ARCH-$OS_BIT/lib build/libhdfs
ant compile-contrib -Dlibhdfs=1 -Dfusedfs=1
```

如果以上不出错，则说明问题解决 编译完成后挂载就很简单了，但应该注意以下问题：

- (1) 编辑 `/etc/ld.so.conf` 添加 `libhdfs.so` 文件的路径，或者可以直接把：`/home/hadoop/hadoop/c++/Linux-i386-32/lib/` 里的所有文件拷贝到 `/usr/lib` 路径，记得执行 `ldconfig`
- (2) 可以把 `/home/hadoop/build/contrib/fuse-dfs` 路径里的两个文件拷贝到 `/usr/local/bin/` 但必须更改 `fuse_dfs_wrapper.sh` 的最后一行 `./fuse_dfs` 为：`/usr/local/bin/fuse_dfs`，`fuse_dfs_wrapper.sh` 是个脚本，请给予执行权限，并可以更改里面的环境变量为真正的值，如果环境变量已设为全局，则不需要更改。