

湘潭大学

分布式框架搭建

讲课：季俊豪

专 业：数据科学与大数据技术

内容纲要

- 1 几个问题
- 2 初识 hadoop
- 3 两个核心
 - HDFS 文件系统
 - MapReduce
- 4 Hadoop 模式
- 5 环境搭建
- 6 小任务

几个问题

- 虚拟机安装时, 选择.iso 镜像再开机, 安装选项一定选择**开发及生产工作站!!** 不要选最小化安装
- 安装之后, 虽然装的是中文环境, 但不带中文输入法, 有兴趣的话自己安装搜狗拼音。
- u 盘文件系统 FAT32, 最大只能支持 32GB 分区, 单个文件也只能支持最大 4GB
- 608 的 ubuntu 没有设置还原, 不要用 `sudo apt upgrade`, 会把系统引导破坏。

内容纲要

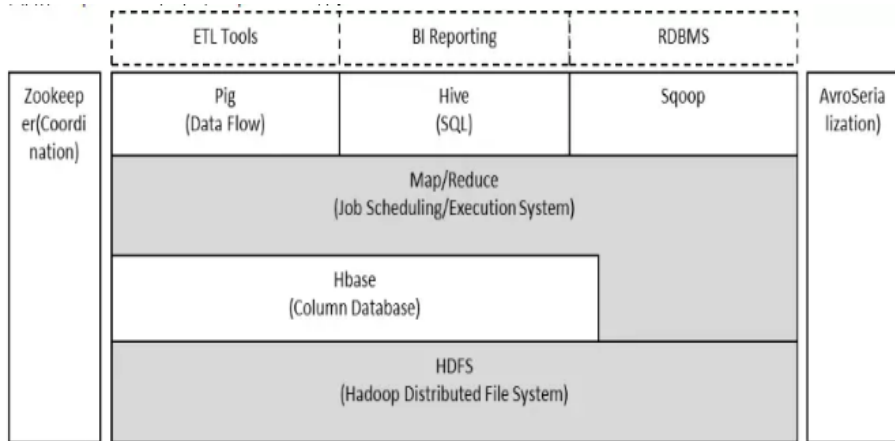
- 1 几个问题
- 2 初识 hadoop
- 3 两个核心
 - HDFS 文件系统
 - MapReduce
- 4 Hadoop 模式
- 5 环境搭建
- 6 小任务

hadoop 简单来说就是处理大数据的框架，主要思想是“分组合并”思想。

- 分组：比如有一个大型数据，那么他就会将这个数据按照算法分成多份，每份存储在从属主机上，并且在从属主机上进行计算，主节点主要负责 Hadoop 两个关键功能模块 HDFS、Map Reduce 的监督
- 合并：将每个机器上的计算结果合并起来，再在一台机器上计算，得到最终结果。这就是 mapreduce 算法。

整体框架

Hadoop 由 HDFS、MapReduce、HBase、Hive 和 ZooKeeper 等成员组成，其中最基础最重要元素为底层用于存储集群中所有存储节点文件的文件系统 HDFS（Hadoop Distributed File System）来执行 MapReduce 程序的 MapReduce 引擎。



整体框架

Pig 是一个基于 Hadoop 的大规模数据分析平台，Pig 为复杂的海量数据并行计算提供了一个简单的操作和编程接口

Hive 是基于 Hadoop 的一个工具，提供完整的 SQL 查询，可以将 sql 语句转换为 MapReduce 任务进行运行

ZooKeeper 高效的，可拓展的协调系统，存储和协调关键共享状态

HBase 是一个开源的，基于列存储模型的分布式数据库

HDFS 是一个分布式文件系统，有着高容错性的特点，适合那些超大数据集的应用程序

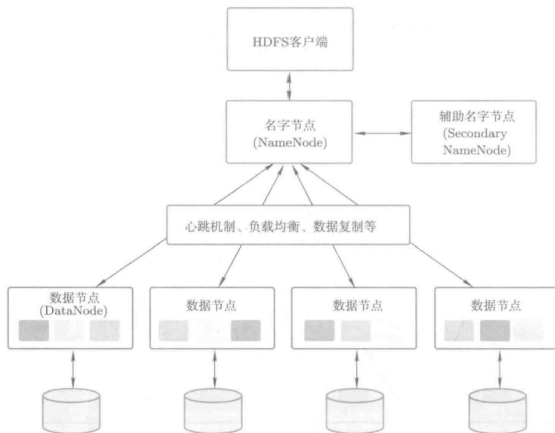
MapReduce 是一种编程模型，用于大规模数据集（大于 1TB）的并行运算

内容纲要

- 1 几个问题
- 2 初识 hadoop
- 3 两个核心
 - HDFS 文件系统
 - MapReduce
- 4 Hadoop 模式
- 5 环境搭建
- 6 小任务

HDFS 文件系统

一个典型的 HDFS 集群包含一个 NameNode 节点 (存位置信息等) 和多个 DataNode 节点 (存数据)。



HDFS 文件系统

- NameNode** 可以看作是分布式文件系统的管理者，存储文件系统的 meta-data，主要负责管理文件系统的命名空间，集群配置信息，存储块的复制。
- DataNode** 是文件存储的基本单元。它存储文件块在本地文件系统中，保存了文件块的 meta-data，同时周期性的发送所有存在的文件块的报告给 NameNode。
- Client** 需要获取分布式文件系统文件的应用程序

文件写入 I

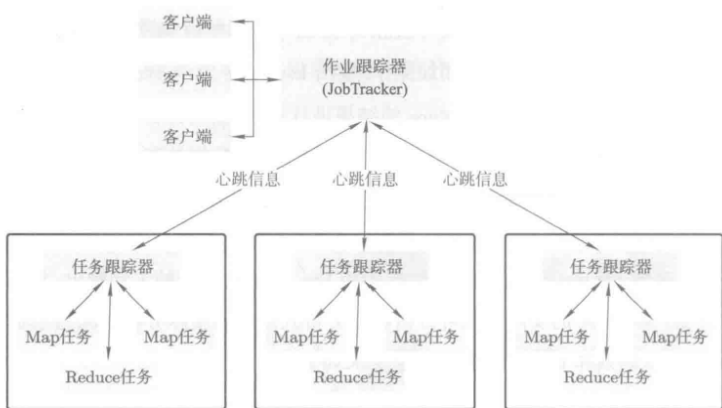
- Client 向 NameNode 发起文件读取的请求
- NameNode 返回文件存储的 DataNode 的信息
- Client 读取文件信息

文件输出 O

- Client 向 NameNode 发起文件写入的请求
- NameNode 根据文件大小和文件块配置情况，返回给 Client 它所管理部分 DataNode 的信息
- Client 将文件划分为多个文件块，根据 DataNode 的地址信息，按顺序写入到每一个 DataNode 块中

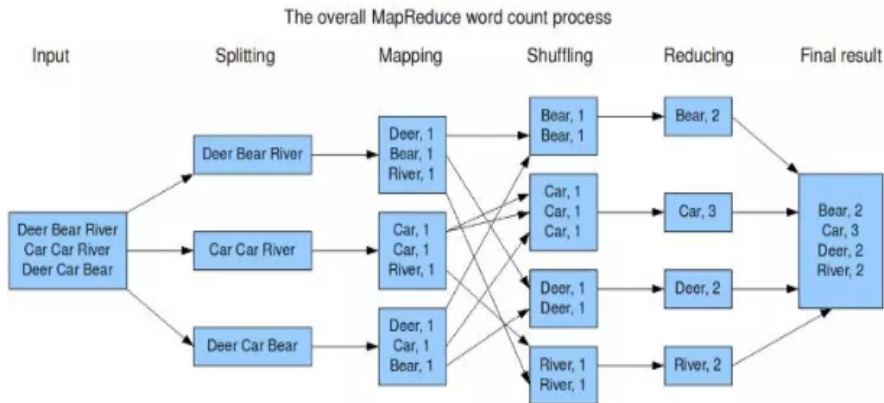
MapReduce

Map（映射）和 Reduce（化简），采用分而治之思想，先把任务分发到集群多个节点上，并行计算，然后再把计算结果合并，从而得到最终计算结果。



例子

Hadoop 的核心是 MapReduce，而 MapReduce 的核心又在于 map 和 reduce 函数。它们是交给用户实现的，这两个函数定义了任务本身。



例子说明

Split 将文本数据按行拆分，输出 key-value, 此时 key 是行号，value 是一行文本。

Map 对每个传入的 value，即一行文本进行分词，对于每个词，输出 key-value, 此时 key 是单词，value 是 1

Shuffle 将相同的 key 数据分配到集群中的相同节点。

Reduce 定义相加函数，相同 key 的 value 相加。输出 key-value。

内容纲要

- 1 几个问题
- 2 初识 hadoop
- 3 两个核心
 - HDFS 文件系统
 - MapReduce
- 4 Hadoop 模式
- 5 环境搭建
- 6 小任务

三种模式

- * 本地模式
- * 伪分布模式
- * 全分布模式

本地模式

- 默认模式
- 不对配置文件进行修改
- 使用本地文件系统，而不是 HDFS，即存在本地
- Hadoop 不会启动 NameNode、DataNode、ResourceManager、NodeManager 等守护进程，Map() 和 Reduce() 任务作为同一个进程的不同部分来执行的
- 用于对 MapReduce 程序的逻辑进行调试，确保程序的正确

伪分布式模式

- 在一台主机模拟多主机
- Hadoop 启动 NameNode、DataNode、ResourceManager、NodeManager 这些守护进程都在同一台机器上运行，是相互独立的 Java 进程
- 使用本地文件系统，而不是 HDFS，即存在本地
- 在这种模式下，Hadoop 使用的是分布式文件系统，各个作业也是由 ResourceManager 服务，来管理的独立进程。在单机模式之上增加了代码调试功能，允许检查内存使用情况，HDFS 输入输出，以及其他的守护进程交互。类似于完全分布式模式，因此，这种模式常用来开发测试 Hadoop 程序的执行是否正确。
- 修改 3 个配置文件：core-site.xml (Hadoop 集群的特性，作用于全部进程及客户端)、hdfs-site.xml (配置 HDFS 集群的工作属性)、mapred-site.xml (配置 MapReduce 集群的属性)

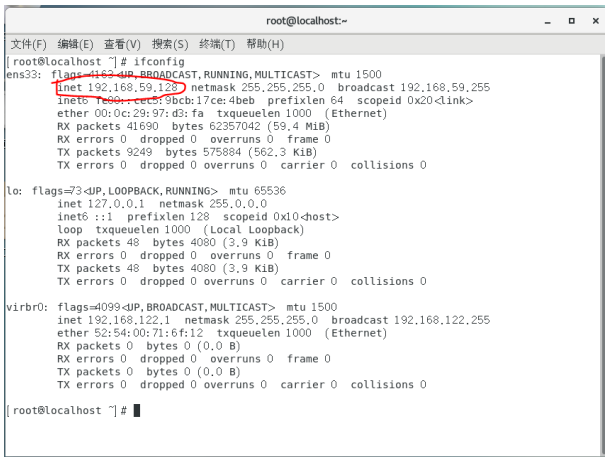
完全分布式模式

- Hadoop 的守护进程运行在由多台主机搭建的集群上，是真正的生产环境
- 在所有的主机上安装 JDK 和 Hadoop，组成相互连通的网络
- 在主机间设置 SSH 免密码登录，把各从节点生成的公钥添加到主节点的信任列表
- 修改 3 个配置文件：core-site.xml、hdfs-site.xml、mapred-site.xml，指定 NameNode 和 ResourceManager 的位置和端口，设置文件的副本等参数

内容纲要

- 1 几个问题
- 2 初识 hadoop
- 3 两个核心
 - HDFS 文件系统
 - MapReduce
- 4 Hadoop 模式
- 5 环境搭建
- 6 小任务

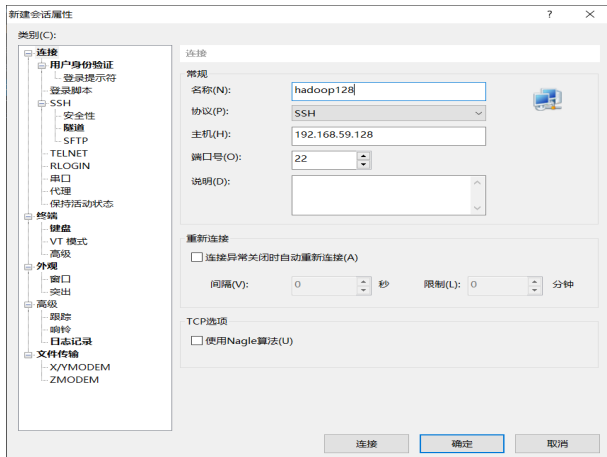
打开虚拟机，在虚拟机终端输入 ifconfig，找到自己的 ip 地址。



```
root@localhost:~  
[root@localhost ~]# ifconfig  
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.59.128 netmask 255.255.255.0 broadcast 192.168.59.255  
    inet6 fe80::ccc5:9bcb:17ce:4beb prefixlen 64 scopeid 0x20<link>  
    ether 00:0c:29:97:d3:fa txqueuelen 1000 (Ethernet)  
    RX packets 41690 bytes 62357042 (59.4 MiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 9249 bytes 575884 (562.3 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 48 bytes 4080 (3.9 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 48 bytes 4080 (3.9 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500  
    inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255  
    ether 52:54:00:71:6f:12 txqueuelen 1000 (Ethernet)  
    RX packets 0 bytes 0 (0.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 0 bytes 0 (0.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
[root@localhost ~]#
```

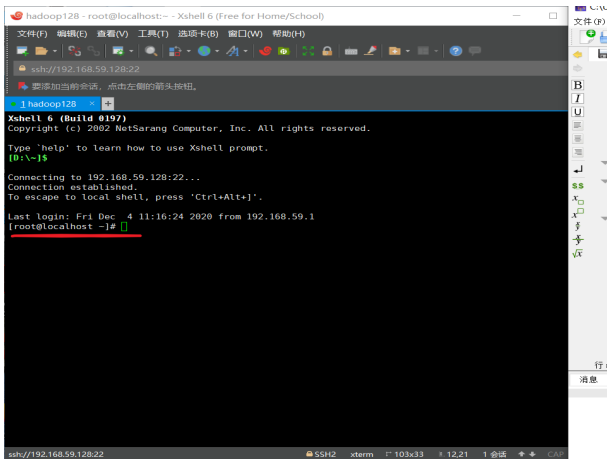
xshell 连接虚拟机

打开 xshell, 选择【文件】-【新建】, 在【主机】里面输入自己的 ip, 【名称】任意, 例如输入 hadoop128, 选择【连接】。



xshell 连接虚拟机

输入用户名，密码，都选择记住，出现下图，即成功连接。



修改主机名和 ip

修改主机名

- * vi /etc/hostname
- * 删除里面内容
- * 写入 hadoop128, 保存
- * 重启

修改 ip 映射

- * vi /etc/hosts
- * 删除里面内容
- * 写入 192.168.59.128 hadoop128, 保存

关闭防火墙

注意 Ubuntu 和 CentOS 的命令不一样，这个只适用于 CentOS
查看防火墙状态

- * `systemctl status firewalld.service`

关闭防火墙

- * `systemctl stop firewalld.service`

永久关闭防火墙

- * `systemctl disable firewalld.service`

补充内容 ssh and scp

- ssh (SSH 客户端软件) 是一个终端仿真程序, 用于注册远程主机, 访问远程系统, 执行其中的命令
- scp (安全的远程文件复制) 利用 scp 命令, 可以在网络主机之间复制文件。

产生秘钥对

- * `ssh-keygen -t rsa` (之后连续按回车键)

把自己的公钥给需要免密登录对象

- * `ssh-copy-id -i .ssh/id_rsa.pub root@hadoop128`

验证是否能够实现免密登录

- * `ssh hadoop128`(第一次使用可能需要输入密码)

注意：秘钥对默认生成在用户 home 目录的.ssh 目录下

ssh 免密登录 (续)

```
Xshell 6 (Build 0197)
Copyright (c) 2002 NetSarang Computer, Inc. All rights reserved.

Type 'help' to learn how to use Xshell prompt.
[0:\~]$

Connecting to 192.168.59.128:22...
Connection established.
To escape to local shell, press 'Ctrl+Alt+J'.

Last login: Fri Dec  4 11:57:31 2020
[root@hadoop128 ~]# ls
anaconda-ks.cfg  initial-setup-ks.cfg  公共  模板  视频  图片  文档  下载  音乐  桌面
[root@hadoop128 ~]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
/root/.ssh/id_rsa already exists.
Overwrite (y/n)?
[root@hadoop128 ~]# ssh-copy-id -i .ssh/id_rsa.pub root@hadoop128
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ".ssh/id_rsa.pub"
The authenticity of host 'hadoop128 (192.168.59.128)' can't be established.
ECDSA key fingerprint is SHA256:y4FMqZVKHjCqHEb1a0JH7YKrHanKjY9XpTyYfTuR0lw.
ECDSA key fingerprint is MD5:de:6f:fa:3f:1c:9e:8f:1a:0d:8f:30:b8:d6:1f:1d:0b.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
root@hadoop128's password:

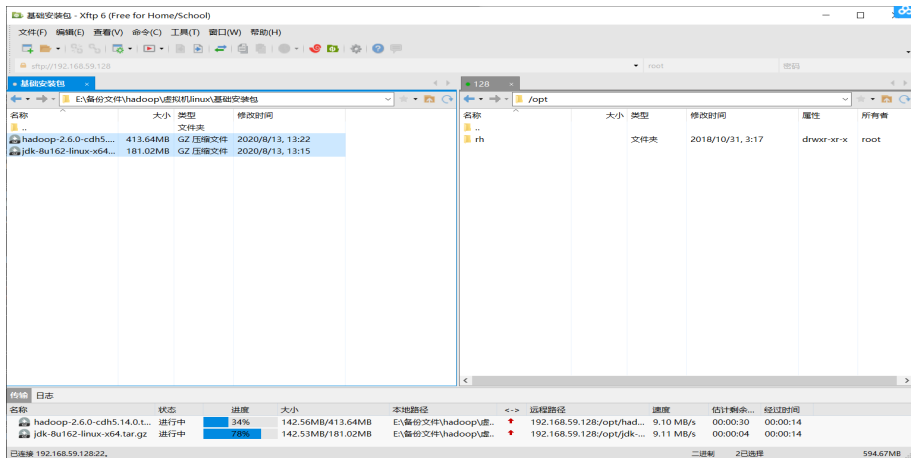
Number of key(s) added: 1

Now try logging into the machine, with:  "ssh 'root@hadoop128'"
and check to make sure that only the key(s) you wanted were added.

[root@hadoop128 ~]# ssh hadoop128
Last login: Fri Dec  4 12:01:35 2020 from 192.168.59.1
[root@hadoop128 ~]#
```

FTP 上传文件

选择绿色的 FTP 工具，将两个安装包，拖入 linux 的/opt 目录下



安装 JDK 和 Hadoop

- ① 解压
- ② 配置全局变量 (主要配置 bin 里面的脚本命令)
- ③ 刷新全局变量
- ④ 验证是否安装成功

在 Linux 里面安装软件，并没有.exe 文件，不要直接用 windows 下的软件安装。

常见安装文件

- * .rpm
- * .deb
- * tar.gz 源代码包

常见安装命令

- * `sudo rpm -ivh rpm` 的软件包名
- * `sudo dpkg -i deb` 的软件包名
- * `sudo apt install 软件`

安装 JDK

- ❶ `tar -zxvf jdk-8u162-linux-x64.tar.gz`
- ❷ `vi .bash_profile`
- ❸ 配置全局环境变量，把 bin 配置进去，即在最后输入
`export JAVA_HOME=/opt/jdk1.8.0_162`
`export CLASSPATH=$JAVA_HOME/lib`
`export PATH=$JAVA_HOME/bin:$PATH`
- ❹ `source .bash_profile`
- ❺ 验证是否成功
`java -version`
`java`
`javac`

安装 JDK，修改环境变量

```

$ .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin

export PATH

export JAVA_HOME=/opt/jdk1.8.0_162
export CLASSPATH=${JAVA_HOME}/lib
export PATH=${JAVA_HOME}/bin:$PATH

```

~~~~~

```
".bash profile" 18L, 283C
```

# 安装 JDK，验证

出现下图，即配置成功。

```
[root@hadoop128 ~]# vi .bash_profile
[root@hadoop128 ~]# source .bash_profile
[root@hadoop128 ~]# java
用法: java [-options] class [args...]
      (执行类)
或 java [-options] -jar jarfile [args...]
      (执行 jar 文件)

其中选项包括:
  -d32          使用 32 位数据模型 (如果可用)
  -d64          使用 64 位数据模型 (如果可用)
  -server       选择 "server" VM
                 默认 VM 是 server,
                 因为您是在服务器类计算机上运行。

  -cp <目录和 zip/jar 文件的类搜索路径>
  -classpath <目录和 zip/jar 文件的类搜索路径>
                 用 : 分隔的目录, JAR 档案
                 和 ZIP 档案列表, 用于搜索类文件。
  -D<名称>=<值> 设置系统属性
  -verbose:[class|gc|jni] 启用详细输出
  -version      输出产品版本并退出
  -version:<值> 警告: 此功能已过时, 将在
                 未来发行版中删除。
                 需要指定的版本才能运行
  -showversion  输出产品版本并继续
  -jre-restrict-search | -no-jre-restrict-search
                 警告: 此功能已过时, 将在
                 未来发行版中删除。
                 在版本搜索中包括/排除用户专用 JRE
```

# 安装 Hadoop

hadoop 安装包的目录结构，方便后面查看，和修改配置文件。



# 安装 Hadoop

- ① 解压（自己试试）
- ② `cd /opt/hadoop-2.6.0-cdh5.14.0/etc/hadoop`
- ③ `vi hadoop-env.sh`(修改 hadoop 所要用的 java 程序)
- ④ `JAVA_HOME=JAVA_HOME=/opt/jdk1.8.0_162`
- ⑤ 增加 hadoop 的全局环境变量 (两个地方 bin、sbin), 自己试试
- ⑥ 验证输入  
`start-all.sh`  
`jps`

# 安装 Hadoop, 修改 java 路径

```
[root@hadoop128 hadoop]# pwd
/opt/hadoop-2.6.0-cdh5.14.0/etc/hadoop
[root@hadoop128 hadoop]# vi hadoop-env.sh

# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements.  See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership.  The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License.  You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# Set Hadoop-specific environment variables here.

# The only required environment variable is JAVA_HOME.  All others are
# optional.  When running a distributed configuration it is best to
# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The java implementation to use.
export JAVA_HOME=/opt/jdk1.8.0_162

# The jsvc implementation to use. Jsvc is required to run secure datanodes
# that bind to privileged ports to provide authentication of data transfer
```

## 安装 Hadoop, 配置环境

```
.bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin

export PATH

export JAVA_HOME=/opt/jdk1.8.0_162
export CLASSPATH=${JAVA_HOME}/lib
export PATH=${JAVA_HOME}/bin:$PATH

HADOOP_HOME=/opt/hadoop-2.6.0-cdh5.14.0
export HADOOP_HOME
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH

~
~
~
~
~
~
~
~
~
~
~/bash profile" 22L, 396C
```

# 安装 Hadoop, 验证安装

可以看到没有 namenode, 没有 datanode。

```
[root@hadoop128 ~]# start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
2012/04 13:17:22 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform...
  using builtin-java classes where applicable
Incorrect configuration: namenode address dfs.namenode.servicerpc-address or dfs.namenode.rpc-address 1
s not configured.
Starting namenodes on [ ]
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:y4FMqZVKHjCqHEb1a0JH7YKrHanKjY9XpTyYfTuR0lw.
ECDSA key fingerprint is MD5:de:6f:fa:3f:1c:9e:8f:1a:0d:8f:30:b8:d6:1f:1d:0b.
Are you sure you want to continue connecting (yes/no)? yes
localhost: Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
localhost: starting namenode, logging to /opt/hadoop-2.6.0-cdh5.14.0/logs/hadoop-root-namenode-hadoop12
8.out
localhost: starting datanode, logging to /opt/hadoop-2.6.0-cdh5.14.0/logs/hadoop-root-datanode-hadoop12
8.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 SHA256:y4FMqZVKHjCqHEb1a0JH7YKrHanKjY9XpTyYfTuR0lw.
ECDSA key fingerprint is MD5:de:6f:fa:3f:1c:9e:8f:1a:0d:8f:30:b8:d6:1f:1d:0b.
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 /opt/hadoop-2.6.0-cdh5.14.0/logs/hadoop-root-secondaryn
amenode-hadoop128.out
2012/04 13:17:50 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform...
  using builtin-java classes where applicable
starting yarn daemons
starting resourcemanager, logging to /opt/hadoop-2.6.0-cdh5.14.0/logs/yarn-root-resourcemanager-hadoop1
28.out
localhost: starting nodemanager, logging to /opt/hadoop-2.6.0-cdh5.14.0/logs/yarn-root-nodemanager-hado
op128.out
[root@hadoop128 ~]# ssh hadoop128
Last login: Fri Dec 4 13:04:22 2020 from 192.168.59.1
[root@hadoop128 ~]# jps
58866 ResourceManager
59141 NodeManager
59302 Jps
```



# 实例尝试

旨在证明，本地模式可以使用 mapreduce。

- 在 /root 下创建一个文件夹 data，再创建一个 data 的子文件夹 input
- 在 input 里面创建一个 data.txt
- 输入  
I am a student  
I am XTU student  
I like python
- `cd /opt/hadoop-2.6.0-cdh5.14.0/share/hadoop/mapreduce`(目的是用这个目录下自带的 wordcount 程序)
- `hadoop jar hadoop-mapreduce-examples-2.6.0-cdh5.14.0.jar wordcount /root/data/input/data.txt /root/data/output/wordcount`  
让 hadoop 调用 wordcount 程序对 data 分析，输出到 output 文件夹下
- 查看结果，`cat /root/data/output/wordcount/part-r-000000`

# 结果展示

```
FILE: Number of bytes written=1220138
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
Map-Reduce Framework
  Map input records=4
  Map output records=11
  Map output bytes=90
  Map output materialized bytes=79
  Input split bytes=95
  Combine input records=11
  Combine output records=7
  Reduce input groups=7
  Reduce shuffle bytes=79
  Reduce input records=7
  Reduce output records=7
  Spilled Records=14
  Shuffled Maps =1
  Failed Shuffles=0
  Merged Map outputs=1
  GC time elapsed (ms)=0
  Total committed heap usage (bytes)=425721856
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=47
File Output Format Counters
  Bytes Written=57
[root@hadoop128 mapreduce]# cat /root/data/output/wordcount/part-r-00000
I      3
XTU    1
a      1
am     2
like   1
python 1
student 2
[root@hadoop128 mapreduce]#
```

# 内容纲要

- 1 几个问题
- 2 初识 hadoop
- 3 两个核心
  - HDFS 文件系统
  - MapReduce
- 4 Hadoop 模式
- 5 环境搭建
- 6 小任务

# 小任务

- 尝试 scp 命令
- 尝试本地模式的文本分析

谢 谢 大 家!