

Linux Hadoop

理论、实战培训

技术文档

内部资料

上海交通大学

周绪宏

zxuhong@sjtu.edu.cn

13917930750

目录

第一章 Linux Hadoop	1
1.1 Hadoop 概述	1
1.1.1 Hadoop 起源	1
1.1.2 Hadoop 核心架构	3
1.1.3 Hadoop 名词解释	5
1.2 Hadoop 安装	7
1.2.1 安装 JDK	7
1.2.2 安装 hadoop	8
1.2.3 Hadoop 目录文件	9
1.3 Hadoop 容器环境	9
1.3.1 安装容器	10
1.3.2 下载导入容器镜像	10
1.3.3 运行容器	11
1.3.4 连接容器	11
1.3.5 SSH 无密码验证	12
1.4 Hadoop 伪分布式操作	14
1.4.1 配置 Hadoop	14
1.4.2 HDFS 操作	17
1.4.3 MapReduce 操作	21
1.4.4 YARN 操作	23
1.5 Hadoop 完全分布式操作	27
1.5.1 准备容器	27
1.5.2 配置 Hadoop	28
1.5.3 运行 Hadoop 服务	33
1.5.4 网页 Hadoop	36
1.5.5 HDFS 操作	37
1.5.6 MapReduce 操作	40
1.5.7 停止 Hadoop 服务	41
1.6 Hadoop 集群快速部署	42
1.6.1 运行容器	42
1.6.2 连接容器	43

1.6.3 快速部署 hadoop.....	43
1.7 问题.....	45
1.7.1 重启机器后容器处理.....	45

第一章 LINUX HADOOP

1.1 Hadoop 概述

Hadoop 是一个由 Apache 基金会所开发的分布式系统基础架构。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进行高速运算和存储。

Hadoop 框架最核心的设计就是：HDFS（Hadoop Distributed File System）和 MapReduce。HDFS 为海量的数据提供了存储，MapReduce 为海量的数据提供了计算。

Hadoop 实现了一个分布式文件系统（Hadoop Distributed File System），简称 HDFS。

Hadoop Map/Reduce 是一个使用简易的软件框架，基于它写出来的应用程序能够运行在由上千个商用机器组成的大型集群上，并以一种可靠容错的方式并行处理上 T 级别的数据集。

Hadoop 典型应用有：搜索、日志处理、推荐系统、数据分析、视频图像分析、数据保存等。

官方网站 <http://hadoop.apache.org>

文档 <http://hadoop.apache.org/docs/current/>

注意：Hadoop 各种书籍、文档比较多，Hadoop 版本发展也比较快，建议参考官方的最新文档。

1.1.1 Hadoop 起源

Google 技术有三宝：GFS、MapReduce 和 BigTable。Google 在 2003 到 2006 年间连续发表了三篇很有影响力的 Paper，分别是 03 年 SOSP 的 GFS，04 年 OSDI 的 MapReduce，06 年 OSDI 的 BigTable。（SOSP 和 OSDI 都是操作系统领域的顶级会

议，在计算机学会推荐会议里属于 A 类。SOSP 在单数年举办，而 OSDI 在双数年举办。)

2003 年 Google 发表了一篇技术学术论文“The Google File System (GFS)”。Google 公司为了存储海量搜索数据而设计的专用文件系统。

<http://research.google.com/archive/gfs.html>

2004 年 Google 发表了一篇技术学术论文“MapReduce: Simplified Data Processing on Large Clusters”。MapReduce 是一种并行计算的编程模型，用于作业调度，用于大规模数据集（大于 1TB）的并行分析运算。

<http://research.google.com/archive/mapreduce.html>

2006 年 Google 发表了一篇学术论文“Bigtable: A Distributed Storage System for Structured Data”，一个结构化数据的分布式存储系统，BigTable 提供结构化数据服务的分布式数据库。

<http://research.google.com/archive/bigtable.html>

Hadoop 实际上就是谷歌三宝的开源实现，HDFS 对应 GFS、Hadoop MapReduce 对应 Google MapReduce，HBase 对应 BigTable。HDFS（或 GFS）为上层提供高效的非结构化存储服务，HBase（或 BigTable）是提供结构化数据服务的分布式数据库，Hadoop MapReduce（Google MapReduce）是一种并行计算的编程模型，用于作业调度。

Hadoop 雏形开始于 2002 年的 Apache 的 Nutch，Nutch 是一个开源 Java 实现的搜索引擎，提供了运行自己的搜索引擎所需的全部工具，包括全文搜索和 Web 爬虫。

2004 年 Nutch 创始人 Doug Cutting 基于 Google 的 GFS 论文实现了分布式文件存储系统名为 NDFS（Nutch Distributed File System）。

2005 年 Doug Cutting 又基于 MapReduce，在 Nutch 搜索引擎实现了该功能。

HBase（Hadoop Database）是一个高可靠性、高性能、面向列、可伸缩的分布式存储系统，利用 HBase 技术可在廉价 PC Server 上搭建起大规模结构化存储集群。

2006 年, Yahoo 雇用了 Doug Cutting, Doug Cutting 将 NDFS 和 MapReduce 升级命名为 Hadoop, Yahoo 开建了一个独立的团队给 Doug Cutting 专门研究发展 Hadoop。

Hadoop 由 Apache Software Foundation 公司于 2005 年秋天作为 Lucene 的子项目 Nutch 的一部分正式引入, 受到 Google 开发的 MapReduce 和 GFS (Google File System) 的启发。2006 年 3 月 MapReduce 和 NDFS (Nutch Distributed File System) 分别被纳入 Hadoop 项目。

目前有很多公司开始提供基于 Hadoop 的商业软件、支持、服务以及培训。Cloudera 是一家美国的企业软件公司, 该公司在 2008 年开始提供基于 Hadoop 的软件和服务。GoGrid 是一家云计算基础设施公司, 在 2012 年, 该公司与 Cloudera 合作加速了企业采纳基于 Hadoop 应用的步伐。Dataguise 公司是一家数据安全公司, 同样在 2012 年该公司推出了一款针对 Hadoop 的数据保护和风险评估。

1.1.2 Hadoop 核心架构

Hadoop 的核心就是 HDFS 和 MapReduce, 而两者只是理论基础, 不是具体可使用的高级应用, Hadoop 旗下有很多经典子项目, 比如 HBase、Hive 等, 这些都是基于 HDFS 和 MapReduce 发展出来的。

Hadoop 由许多元素构成, 其最底部是 Hadoop Distributed File System (HDFS), 它存储 Hadoop 集群中所有存储节点上的文件。HDFS 的上一层是 MapReduce 引擎, 该引擎由 JobTrackers 和 TaskTrackers 组成。通过对 Hadoop 分布式计算平台最核心的分布式文件系统 HDFS、MapReduce 处理过程, 以及数据仓库工具 Hive 和分布式数据库 Hbase 的介绍, 基本涵盖了 Hadoop 分布式平台的所有技术核心。

HDFS (Hadoop Distributed File System) 分布式文件系统是一个高度容错性的系统, 适合部署在廉价的机器上。HDFS 能提供高吞吐量的数据访问, 适合那些有着超大数据集的应用程序。

MapReduce 是一套从海量源数据提取分析元素最后返回结果集的编程模型，将文件分布式存储到文件系统是第一步，而从海量数据中提取分析我们需要的内容就是 MapReduce 做的事了。

MapReduce 的基本原理就是：将大的数据分析分成小块逐个分析，最后再将提取出来的数据汇总分析，最终获得我们想要的内容。当然怎么分块分析，怎么做 Reduce 操作非常复杂，Hadoop 已经提供了数据分析的实现，我们只需要编写简单的需求命令即可达成我们想要的数据库。

HDFS 详细介绍请参考 Linux_HDFS.doc 文档。

MapReduce 详细介绍请参考 Linux_MapReduce.doc 文档。

整个 Hadoop 家族由以下几个子项目组成：

(1) Hadoop Common, Hadoop 体系最底层的一个模块，为 Hadoop 各子项目提供各种工具，如配置文件和日志操作等。

(2) HDFS, Hadoop 分布式文件系统 (Hadoop Distributed File System)。

(3) MapReduce, 并行计算框架，实现了 MapReduce 编程框架。

(4) HBase, 基于 Hadoop Distributed File System, 是一个开源的，基于列存储模型的分布式数据库。类似 Google BigTable 的分布式 NoSQL 列数据库。(HBase 和 Avro 已于 2010 年 5 月成为顶级 Apache 项目)。

(5) Avro, Avro 是 doug cutting 主持的 RPC 项目，有点类似 Google 的 protobuf 和 Facebook 的 thrift。Avro 用来做以后 hadoop 的 RPC，使 hadoop 的 RPC 模块通信速度更快、数据结构更紧凑。

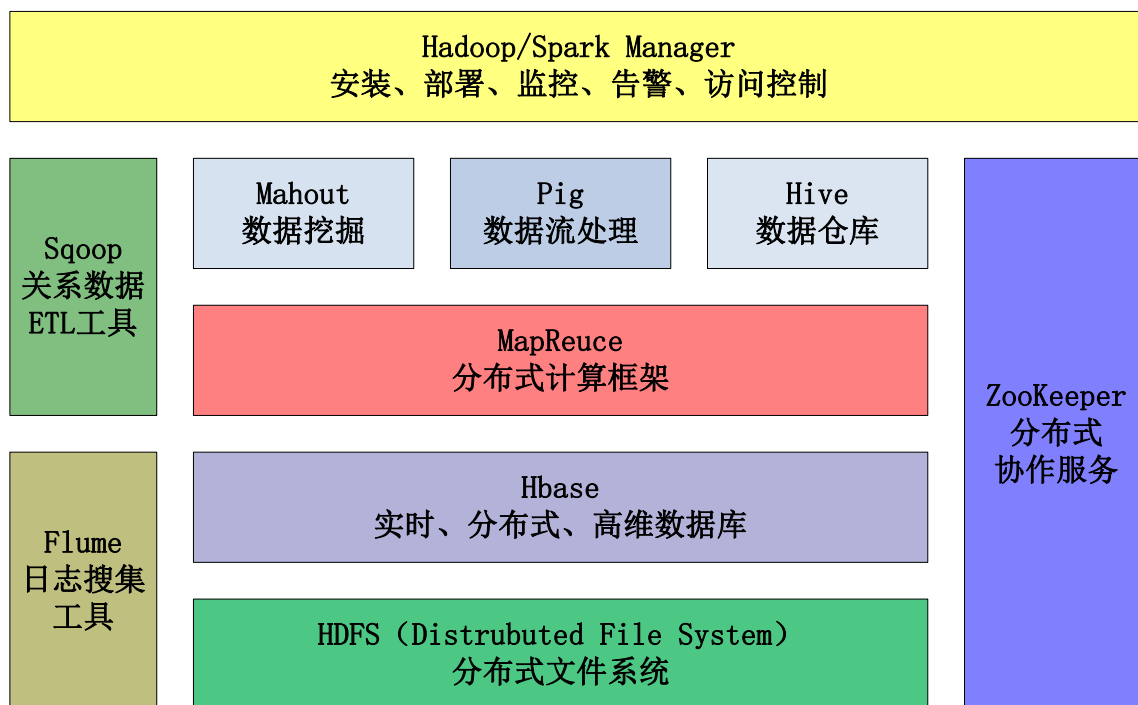
(6) Chukwa, Chukwa 是基于 Hadoop 的大集群监控系统，由 yahoo 贡献。

(7) Hive, hive 类似 CloudBase, 也是基于 hadoop 分布式计算平台上的提供 data warehouse 的 sql 功能的一套软件。使得存储在 hadoop 里面的海量数据的汇总，即席查询简单化。hive 提供了一套 QL 的查询语言，以 sql 为基础，使用起来很方便。

(8) Pig Pig, 是 SQL-like 语言，是在 MapReduce 上构建的一种高级查询语言，把一些运算编译进 MapReduce 模型的 Map 和 Reduce 中，并且用户可以定义自己的功能。Yahoo 网络运算部门开发的又一个克隆 Google 的项目 Sawzall。

(9) ZooKeeper, Zookeeper 是 Google 的 Chubby 一个开源的实现。它是一个针对大型分布式系统的可靠协调系统，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。



Hadoop 架构

1.1.3 Hadoop 名词解释

(1) 名字节点 NameNode

HDFS 守护程序，记录文件是如何分割成数据块的，以及这些数据块被存储到哪些节点上。

(2) 备用名字节点 SecondaryNameNode

监控 HDFS 状态的辅助后台程序，每个集群都有，与名字节点 NameNode 进行通讯，定期保存 HDFS 元数据快照；当名字节点 NameNode 故障可以作为备用名字节点 NameNode 使用。

(3) 数据节点 DataNode

Hadoop 集群的每台从服务器都运行数据节点 `DataNode`，负责把 HDFS 数据块读写到本地文件系统。

（4）JobTracker

MapReduce v1 用户处理作业（用户提交代码）的后台程序，决定有哪些文件参与处理，然后切割 task 并分配节点，每个集群只有一个 JobTracker，位于 Master 节点。MapReduce v2（YARN）中不再使用。

（5）TaskTracker

位于 slave 节点上，与 `DataNode` 结合（代码与数据一起的原则）管理各自节点上的 task（由 JobTracker 分配），每个节点只有一个 TaskTracker，但一个 TaskTracker 可以启动多个 JVM，用户并发执行 Map 或 Reduce 任务。与 JobTracker 交互。MapReduce v2（YARN）中不再使用。

（6）资源管理器 ResourceManager

Hadoop 集群主节点 master 上，YARN 节点的控制模块，负责统一规划资源的使用。资源管理器 ResourceManager 简称 RM，MapReduce v2（YARN）中使用。

（7）资源节点管理器 NodeManager

Hadoop 集群从节点 slave 上，YARN 的资源节点模块，负责启动管理容器 Container。资源节点管理器 NodeManager 简称 NM，MapReduce v2（YARN）中使用。

（8）应用管理 ApplicationMaster

YARN 中的每个应用都会启动一个应用管理 ApplicationMaster，负责向资源管理器 ResourceManager 申请资源，请求资源节点管理器 NodeManager 启动资源容器 Container，并告诉资源容器 Container 做什么事情。应用管理 ApplicationMaster 简称 AM，MapReduce v2（YARN）中使用。

（9）资源容器 Container

YARN 中所有的应用都是在资源容器 Container 中运行的。应用管理 ApplicationMaster 也是在资源容器 Container 中运行的，不过应用管理 ApplicationMaster（AM）的资源容器 Container 是资源管理器 ResourceManager（RM）申请的，MapReduce v2（YARN）中使用。

Hadoop 集群主节点 Master 上运行的程序：名字节点 NameNode、备用名字节点 SecondaryNameNode、资源管理器 ResourceManager。

Hadoop 集群从节点 Slave 上运行的程序：数据节点 DataNode、资源节点管理器 NodeManager、应用管理 ApplicationMaster 、资源容器 Container。

1.2 Hadoop 安装

Hadoop 集群可以部署在物理服务器上，也可以部署在虚拟机里，也可以部署在容器里。

注意：容器镜像里已经安装了 Hadoop，不需要再安装部署 Hadoop 软件。

1.2.1 安装 JDK

Hadoop 是使用 java 开发的，java 是必须安装的软件之一。在 CentOS 环境中可以通过 yum 进行安装或者去官网下载最新版本的 JDK。

容器在编译时已经安装了 JDK，不需要再安装 JDK 了。如果其他容器，还是需要安装 JDK。

源码安装

从官方网站下载 Linux x64 的 jdk-8u131-linux-x64.tar.gz 或更高版本。

下载地址：

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

所有机器上操作

(1) 从交大服务器上下载 jdk

```
wget http://202.120.32.244/bigdata/software/jdk-8u131-linux-x64.tar.gz
```

(2) 解压安装

```
tar zxvf jdk-8u131-linux-x64.tar.gz -C /usr/local/  
ls /usr/local/jdk1.8.0_131/
```

(3) 设置环境变量

```
vim /etc/profile                                添加以下内容  
#JAVA  
export JAVA_HOME=/usr/local/jdk1.8.0_131  
export JRE_HOME=/usr/local/jdk1.8.0_131/jre  
export CLASSPATH=$JAVA_HOME/lib:$JRE_HOME/lib:$CLASSPATH:.  
export PATH=$JAVA_HOME/bin:$JRE_HOME/bin:$PATH
```

(4) 查看版本

```
source /etc/profile  
java -version
```

source /etc/profile 使环境变量生效，查看 java 版本，如能看到版本号，说明 JAVA 环境配置成功。

1.2.2 安装 hadoop

可以从官方网站下载编译好的 hadoop (<http://hadoop.apache.org/releases.html#Download>)；但是官方提供的已经编译好的 hadoop，在 CentOS 64 位机器上使用是有问题的，我们需要自己编译的 hadoop。

容器在编译时已经安装了 hadoop，不需要再安装 hadoop 了。如果其他容器，还是需要安装 hadoop。

所有机器上操作

(1) 从交大服务器上下载编译好的 hadoop

```
wget http://202.120.32.244/bigdata/software/hadoop-2.7.3-hongdy.tar.gz
```

(2) 解压安装

```
tar zxvf hadoop-2.7.3-hongdy.tar.gz -C /usr/local/
```

（3）设置环境变量

<pre>vim /etc/profile # hadoop export HADOOP_HOME=/usr/local/hadoop-2.7.3 export HADOOP_PREFIX=\$HADOOP_HOME export HADOOP_COMMON_HOME=\$HADOOP_PREFIX export HADOOP_COMMON_LIB_NATIVE_DIR=\$HADOOP_PREFIX/lib/native export HADOOP_CONF_DIR=\$HADOOP_PREFIX/etc/hadoop export HADOOP_HDFS_HOME=\$HADOOP_PREFIX export HADOOP_MAPRED_HOME=\$HADOOP_PREFIX export HADOOP_YARN_HOME=\$HADOOP_PREFIX export LD_LIBRARY_PATH=\$HADOOP_PREFIX/lib/native:\$LD_LIBRARY_PATH export PATH=\$HADOOP_HOME/bin:\$PATH</pre>	最后添加以下内容
--	----------

（4）查看版本

<pre>source /etc/profile /usr/local/hadoop-2.7.3/bin/hadoop version</pre>

source /etc/profile 使环境变量生效；查看 hadoop 版本，如能看到版本号，说明 hadoop 环境配置成功。

1.2.3 Hadoop 目录文件

Hadoop 安装好后，代码在/usr/local/hadoop-2.7.3 目录下。

<pre>ls /usr/local/hadoop-2.7.3/ bin etc include lib libexec LICENSE.txt NOTICE.txt README.txt sbin share</pre>

1.3 Hadoop 容器环境

准备五台容器。

IP 地址	机器名	说明
-------	-----	----

172.17.0.2	hadoop20	伪分布式单节点
172.17.0.3	hadoop21	完全分布式集群主节点
172.17.0.4	hadoop22	完全分布式集群从节点
172.17.0.5	hadoop23	完全分布式集群从节点
172.17.0.6	hadoop24	完全分布式集群从节点

注意：容器 IP 地址和机器名，根据实际情况修改。

1.3.1 安装容器

（1）设置容器安装仓库源

```
vim /etc/yum.repos.d/sjtu-docker.repo          内容如下所示：
[sjtu-docker-repo]
name=Docker Repository
#baseurl=https://yum.dockerproject.org/repo/main/centos/7/
baseurl=http://202.120.32.244/repos/docker/centos/7/
enabled=1
gpgcheck=0
gpgkey=https://yum.dockerproject.org/gpg
```

（2）安装 docker-engine

```
yum -y install docker-engine
```

（3）启动 docker

```
systemctl restart docker.service
systemctl enable docker.service
```

1.3.2 下载导入容器镜像

（1）从交大服务器上下载编译好的容器镜像。

```
mkdir -p /root/docker/images
cd /root/docker/images
wget http://202.120.32.244/docker/images/docker-centos6.10-hadoop-spark.tar
```

(2) 导入容器镜像

```
docker load < docker-centos6.10-dekstop-hadoop-spark.tar
```

1.3.3 运行容器

(1) 删除已经运行的所有容器

```
docker kill $(docker ps -a -q)  
docker rm $(docker ps -a -q)
```

(2) 运行容器

伪分布式集群 (hadoop20)

```
docker run -d --name='hadoop20' --hostname='hadoop20' docker-centos6.10-hadoop-spark
```

完全分布式集群 (hadoop21、hadoop22、hadoop23、hadoop24)

```
docker run -d --name='hadoop21' --hostname='hadoop21' docker-centos6.10-hadoop-spark  
docker run -d --name='hadoop22' --hostname='hadoop22' docker-centos6.10-hadoop-spark  
docker run -d --name='hadoop23' --hostname='hadoop23' docker-centos6.10-hadoop-spark  
docker run -d --name='hadoop24' --hostname='hadoop24' docker-centos6.10-hadoop-spark
```

1.3.4 连接容器

(1) 查看容器

伪分布式集群 (hadoop20)

```
docker inspect hadoop20
```

查看容器 IP 地址。

完全分布式集群（hadoop21、hadoop22、hadoop23、hadoop24）

```
docker inspect hadoop21
docker inspect hadoop22
docker inspect hadoop23
docker inspect hadoop24
```

查看容器 IP 地址。

（2）SSH 连接容器

伪分布式集群（hadoop20）

```
rm -f /root/.ssh/known_hosts
ssh 172.17.0.2
```

输入密码 123456

完全分布式集群（hadoop21、hadoop22、hadoop23、hadoop24）

```
ssh 172.17.0.3
ssh 172.17.0.4
ssh 172.17.0.5
ssh 172.17.0.6
```

1.3.5 SSH 无密码验证

Hadoop 需要使用 SSH 协议，名字节点 namenode 将使用 SSH 协议启动名字节点 namenode 和数据节点 datanode 进程；伪分布式模式名字节点和数据节点均是本身；需要配置 SSH 无密码验证。

使用 scp 命令在两台机器之间拷贝文件，需要输入密码；使用 SSH 连接到其他机器运行程序，也需要输入密码。为了方便，配置 SSH 无密码验证。

伪分布式集群（hadoop20）机器（172.17.0.2）

完全分布式集群（hadoop21、hadoop22、hadoop23、hadoop24）机器（172.17.0.3）

（1）产生密钥

```
rm -fr /root/.ssh
ssh-keygen -t rsa -P "" -f ~/.ssh/id_rsa
```

会在/root/.ssh/目录下生成两个文件：id_rsa（私钥，产生私钥的机器，即主动访问的机器拥有）、id_rsa.pub（公钥，发给被访问机器）。

（2）查看密钥文件

```
ls /root/.ssh/  
id_rsa id_rsa.pub
```

（3）产生授权文件

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

（4）测试

此时 SSH 连接本机，不需要输入密码

```
ssh 172.17.0.2
```

（5）将密钥文件拷贝到其他机器上

```
scp -r /root/.ssh root@172.17.0.3:/root/  
scp -r /root/.ssh root@172.17.0.4:/root/  
scp -r /root/.ssh root@172.17.0.5:/root/  
scp -r /root/.ssh root@172.17.0.6:/root/
```

注意：容器修改相应 IP 地址。

无论 Hadoop 集群部署在物理服务器、虚拟机、容器，都需要修改/etc/hosts 文件。

伪分布式集群（hadoop20）机器（172.17.0.2）

完全分布式集群（hadoop21、hadoop22、hadoop23、hadoop24）机器（172.17.0.3）

（1）修改文件 /etc/hosts

```
vim /etc/hosts
```

内容如下

```
127.0.0.1 localhost  
172.17.0.2 hadoop20  
172.17.0.3 hadoop21  
172.17.0.4 hadoop22  
172.17.0.5 hadoop23
```



```
172.17.0.6  hadoop24
```

(2) 将文件拷贝到其他服务器

```
scp /etc/hosts root@hadoop21:/etc/hosts  
scp /etc/hosts root@hadoop22:/etc/hosts  
scp /etc/hosts root@hadoop23:/etc/hosts  
scp /etc/hosts root@hadoop24:/etc/hosts
```

(3) 验证 ssh

```
ssh hadoop20  
ssh hadoop21  
ssh hadoop22  
ssh hadoop23  
ssh hadoop24
```

注意：如果服务器重启，请参看问题：重启机器后容器处理一节。

1.4 Hadoop 伪分布式操作

伪分布式集群（hadoop20）

在机器 hadoop20（172.17.0.2）上操作

1.4.1 配置 Hadoop

编辑 Hadoop 配置文件

1.4.1.1 Hadoop-env.sh

配置 hadoop 环境变量文件

```
vim /usr/local/hadoop-2.7.3/etc/hadoop/hadoop-env.sh 添加或修改以下行  
export JAVA_HOME=/usr/local/jdk1.8.0_131  
export HADOOP_PREFIX=/usr/local/hadoop-2.7.3
```

注意：容器镜像已经配置好了，不需要修改！

1.4.1.2 core-site.xml

编辑配置文件 core-site.xml

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/core-site.xml
vim /usr/local/hadoop-2.7.3/etc/hadoop/core-site.xml 内容如下
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://172.17.0.2:9000</value>
  </property>
</configuration>
```

注意：修改实际的 IP 地址。

fs.default.name 是一个描述集群中名字节点 NameNode 的 URI（包括协议、主机名称、端口号）；集群里面的每一台机器都需要知道名字节点 NameNode 的地址。数据节点 DataNode 结点会先在名字节点 NameNode 上注册，这样它们的数据才可以被使用。独立的客户端程序通过这个 URI 跟数据节点 DataNode 交互，以取得文件的块列表。

1.4.1.3 hdfs-site.xml

编辑配置文件 hdfs-site.xml

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/hdfs-site.xml
vim /usr/local/hadoop-2.7.3/etc/hadoop/hdfs-site.xml 内容如下
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

1.4.1.4 mapred-site.xml

使用 yarn 方式才需要配置此文件。

编辑配置文件 mapred-site.xml

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/mapred-site.xml
vim /usr/local/hadoop-2.7.3/etc/hadoop/mapred-site.xml    内容如下
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

1.4.1.5 yarn-site.xml

使用 yarn 方式才需要配置此文件。

编辑配置文件 yarn-site.xml

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/yarn-site.xml
vim /usr/local/hadoop-2.7.3/etc/hadoop/yarn-site.xml    内容如下
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
</configuration>
```

1.4.1.6 masters

编辑 hadoop 集群主节点 masters 文件

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/masters
vim /usr/local/hadoop-2.7.3/etc/hadoop/masters
172.17.0.2
```

注意：容器修改实际的 IP 地址。

1.4.1.7 slaves

编辑 hadoop 集群从节点 slaves 文件

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/slaves
vim /usr/local/hadoop-2.7.3/etc/hadoop/slaves
172.17.0.2
```

1.4.2 HDFS 操作

<http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html>

1.4.2.1 格式化文件系统

在首次启动 Hadoop 之前，需要格式化 HDFS 文件系统。

格式化 HDFS 分布式文件系统

```
rm -fr /tmp/hadoop-root
/usr/local/hadoop-2.7.3/bin/hdfs namenode -format
```

输出信息中看到以下内容，存储目录/tmp/hadoop-root/dfs/name 被成功格式化。

```
INFO common.Storage: Storage directory /tmp/hadoop-root/dfs/name has been successfully formatted.
```

hadoop.tmp.dir 默认值为/tmp；一些版本的 Linux 系统会在每次重新启动时删除/tmp 的内容；所以指定 hadoop.tmp.dir 数据留存的位置更为安全。

1.4.2.2 启动 HDFS 服务

(1) 启动 HDFS 服务

包括管理文件系统的名字节点 NameNode 和保存数据的数据节点 DataNode 服务。

```
/usr/local/hadoop-2.7.3/sbin/start-dfs.sh
```

启动信息如下：

```
Starting namenodes on [hadoop20]
```

```
hadoop20: Warning: Permanently added 'hadoop20' (ECDSA) to the list of known hosts.
hadoop20: starting namenode, logging to /usr/local/hadoop-2.7.3/logs/hadoop-root-namenode-hadoop20.out
172.17.0.2: starting datanode, logging to /usr/local/hadoop-2.7.3/logs/hadoop-root-datanode-hadoop20.out
Starting secondary namenodes [0.0.0.0]
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 /usr/local/hadoop-2.7.3/logs/hadoop-root-secondarynamenode-hadoop20.out
```

（2）使用 jps 查看 java 进程

启动这些组件后，使用 JDK 的 jps 工具查看哪个 Java 进程正在运行。

```
jps
```

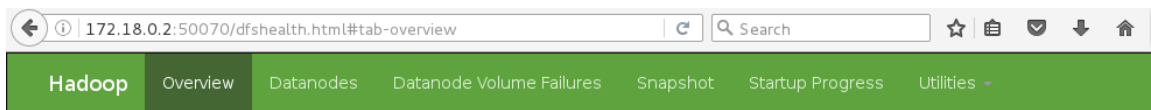
会看到如下进程

```
9800 NameNode
9911 DataNode
10168 Jps
10054 SecondaryNameNode
```

如果看不到这些进程，说明配置不正确；请检查配置文件，重新格式化 HDFS 文件系统，启动服务。

1.4.2.3 查看 HDFS 网页接口

打开浏览器，输入 <http://172.17.0.2:50070>，可看到 HDFS 页面，如下图所示：



Overview 'hadoop20:9000' (active)

Started:	Tue Aug 21 10:40:21 UTC 2018
Version:	2.7.3, rUnknown
Compiled:	2017-05-24T07:06Z by root from Unknown
Cluster ID:	CID-8204b7ee-e0bf-45db-87cc-8736d4d7a430
Block Pool ID:	BP-1911529232-172.18.0.2-1534847768132

Summary

1.4.2.4 HDFS 文件操作

HDFS 允许用户数据组织成文件和文件夹的方式，它提供一个叫 DFSShell 的接口，使用户可以和 HDFS 中的数据交互。

hadoop 命令集的语法跟其他用户熟悉的 shells（bash、csh）类似。

操作	命令
创建目录 /foodir	hadoop dfs -mkdir /foodir
查看文件 /foodir/myfile.txt	hadoop dfs -cat /foodir/myfile.txt
删除文件/foodir/myfile.txt	hadoop dfs -rm /foodir myfile.txt

DFSAdmin 命令集是用于管理 dfs 集群的，这些命令只由 HDFS 管理员使用。

操作	命令
将集群设置成安全模式	hadoop dfsadmin -safemode enter hadoop dfsadmin -safemode leave
产生一个数据节点的列表	hadoop dfsadmin -report
去掉一个数据节点	hadoop dfsadmin -decommission datanodename

1、创建目录

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -mkdir /user  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -mkdir /user/zxuhong
```

hadoop 为每个用户保留一个主目录，这些主目录位于 HDFS 文件系统的/user 路径下；如果主目录不存在，需要创建主目录。

此时在浏览器中输入 <http://172.17.0.2:50070/explorer.html#/> 就可以看到新建的/user 目录。

2、查看目录

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -ls /user/  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -ls /user/zxuhong/
```

3、上传文件

```
echo "hello" > /tmp/hello  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -put /tmp/hello /user/zxuhong/
```

创建文件、上传文件。

4、查看上传文件

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -ls /user/zxuhong  
相当于  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -ls hdfs://172.17.0.2:9000/user/zxuhong
```

5、下载文件

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -get /user/zxuhong/hello /root/hello  
cat /root/hello
```

下载文件、查看文件内容

6、在线查看文件内容

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -cat /user/zxuhong/hello
```

7、删除文件

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm /user/zxuhong/hello
```

8、删除目录

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rmdir /user/zxuhong  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r /user/
```

9、帮助

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -help
```

注意：上面所有的操作命令 hdfs 换成 hadoop 也是一样的；中间参数选项 dfs 换成 fs 也是一样的。

1.4.2.5 停止名字节点和数据节点服务

```
/usr/local/hadoop-2.7.3/sbin/stop-dfs.sh
```

输出信息如下：

```
Stopping namenodes on [hadoop20]  
hadoop20: stopping namenode  
172.17.0.2: stopping datanode  
Stopping secondary namenodes [0.0.0.0]  
0.0.0.0: stopping secondarynamenode
```

1.4.3 MapReduce 操作

如果 dfs 服务已经停止，请再次启动

```
/usr/local/hadoop-2.7.3/sbin/start-dfs.sh
```


1.4.3.1 上传作业

上传测试作业

```
cd /usr/local/hadoop-2.7.3/  
bin/hdfs dfs -mkdir /user  
bin/hdfs dfs -mkdir /user/root  
bin/hdfs dfs -put etc/hadoop input  
bin/hdfs dfs -ls /user/root/input
```

第一条命令创建/user 命令，相当于根目录。

第二条命令创建用户目录，跟当前登录用户有关。

第三条命令把 etc/hadoop 目录下文件拷贝到 HDFS 文件系统的/user/root/input 目录下。

第四条命令查看用户目录刚上传的文件。

此时可在浏览器中 <http://172.17.0.2:50070/explorer.html#/user/root/input> 查看文件。

1.4.3.2 运行作业

请先删除 etc/hadoop/mapred-site.xml 和 etc/hadoop/yarn-site.xml 文件，现在先不用 yarn 方式处理。

```
rm -fr /usr/local/hadoop-2.7.3/etc/hadoop/mapred-site.xml  
rm -fr /usr/local/hadoop-2.7.3/etc/hadoop/yarn-site.xml
```

查询比较作业

```
cd /usr/local/hadoop-2.7.3/  
bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar grep input output 'dfs[a-z.]+'
```

此时在 HDFS 文件系统中生成输出目录 /user/root/output。

1.4.3.3 查看作业

查看输出文件

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -get /user/root/output /tmp/output  
cat /tmp/output/*
```

从 HDFS 文件系统下载 output 目录，此时/tmp 目录下有个 output 目录，查看输出文件信息。

或直接在 HDFS 文件系统上查看

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -cat output/*
```

如果再次做作业，需要删除 input、output 目录，重新上传作业。

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r input  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r output
```

1.4.4 YARN 操作

可以运行 MapReduce 作业在 YARN

1.4.4.1 编辑配置文件

编辑配置文件 etc/hadoop/mapred-site.xml、etc/hadoop/yarn-site.xml

(1) 编辑配置文件 mapred-site.xml:

```
vim /usr/local/hadoop-2.7.3/etc/hadoop/mapred-site.xml    内容如下  
<?xml version="1.0"?>  
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>  
<configuration>  
  <property>  
    <name>mapreduce.framework.name</name>  
    <value>yarn</value>  
  </property>  
</configuration>
```

(2) 编辑配置文件 yarn-site.xml

`vim /usr/local/hadoop-2.7.3/etc/hadoop/yarn-site.xml` 内容如下

```
<?xml version="1.0"?>
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
</configuration>
```

1.4.4.2 启动 YARN 服务

(1) 启动 YARN 服务

包括 ResourceManager 和 NodeManager 服务。

`/usr/local/hadoop-2.7.3/sbin/start-yarn.sh`

输出信息如下所示：

```
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop-2.7.3/logs/yarn-root-resourcemanager-hadoop20.out
172.17.0.2: starting nodemanager, logging to /usr/local/hadoop-2.7.3/logs/yarn-root-nodemanager-hadoop20.out
```

(2) 使用 jps 查看 java 进程

启动这些组件后，使用 JDK 的 jps 工具查看哪个 Java 进程正在运行。

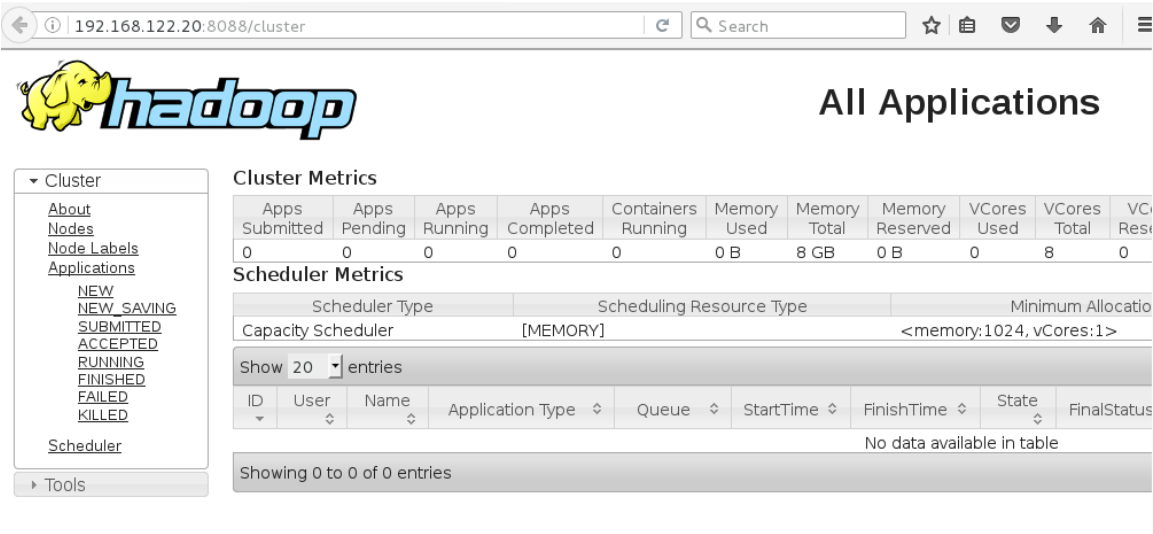
`jps`

可以看到如下 java 进程

```
30179 Jps
28508 SecondaryNameNode
28235 NameNode
28354 DataNode
28667 ResourceManager
28765 NodeManager
```

1.4.4.3 查看 YARN 网页接口

打开浏览器，输入 <http://172.17.0.2:8088> 可看到各种应用，如下图所示：



1.4.4.4 上传作业

清理删除原有 input、output 目录所有文件。

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r /user/root/input
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r /user/root/output
```

上传作业到/user/root/input 目录

```
cd /usr/local/hadoop-2.7.3/
bin/hdfs dfs -mkdir /user/root
bin/hdfs dfs -put etc/hadoop input
bin/hdfs dfs -ls /user/root/input
```

1.4.4.5 运行作业

删除之前作业生成的结果

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r output
```

查询比较作业

```
cd /usr/local/hadoop-2.7.3/
bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar grep input output 'dfs[a-z.]+'
```

此时打开浏览器 <http://172.17.0.2:8088/cluster>，可以看到一个 applications 在运行。

The screenshot shows the Hadoop cluster management interface. On the left is a sidebar with navigation links: Cluster, About, Nodes, Node Labels, Applications, NEW, NEW SAVING, SUBMITTED, ACCEPTED, RUNNING, FINISHED, FAILED, KILLED, Scheduler, and Tools. The main content area is titled 'All Applications' and contains two tables: 'Cluster Metrics' and 'Scheduler Metrics'.

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved
1	0	1	0	7	8 GB	8 GB	0 B	7	8	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>

Below the scheduler metrics, there is a table of applications. The first entry is shown:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State
application_1534849023349_0001	root	grep-search	MAPREDUCE	default	Tue Aug 21 07:05:55 -0400 2018	N/A	RUNNING

Showing 1 to 1 of 1 entries

如果运行中出现以下错误：

```
INFO ipc.Client: Retrying connect to server: localhost/172.17.0.2:10020. Already tried 5 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
```

这个问题是由于没有启动 historyserver 引起的，解决办法：

在 mapred-site.xml 配置文件中添加：

```
<property>
  <name>mapreduce.jobhistory.address</name>
  <value>172.17.0.2:10020</value>
</property>
```

在 namenode 上执行命令：

```
/usr/local/hadoop-2.7.3/sbin/mr-jobhistory-daemon.sh start historyserver
```

这样在 namenode 上会启动 JobHistoryServer 服务，可以在 historyserver 的日志中查看运行情况。

1.4.4.6 查看作业

下载输出文件到本地目录查看

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -get /user/root/output /tmp/output  
cat /tmp/output/*
```

从 HDFS 文件系统下载 output 目录，此时/tmp 目录下有个 output 目录，再查看输出文件信息。

直接在 HDFS 分布式文件系统上查看

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -cat output/*
```

1.4.4.7 停止 ResourceManager 和 NodeManager 服务

```
/usr/local/hadoop-2.7.3/sbin/stop-yarn.sh
```

1.5 Hadoop 完全分布式操作

1.5.1 准备容器

为了方便，使用云计算平台的容器作为 hadoop 环境。

IP 地址	运行程序	机器名	说明
172.17.0.3	NameNode、SecondaryNameNode、ResourceManager	hadoop21	负责总管分布式数据和分解任务执行
172.17.0.4	DataNode、NodeManager	hadoop22	负责分布式数据存储和任务的执行
172.17.0.5	DataNode、NodeManager	hadoop23	负责分布式数据存储和任务的执行
172.17.0.5	DataNode、NodeManager	hadoop24	负责分布式数据存储和任务的执行

1.5.2 配置 Hadoop

完全分布式集群（hadoop21、hadoop22、hadoop23、hadoop24）

在机器 hadoop21（172.17.0.3）上操作

```
cd /usr/local/hadoop-2.7.3/  
ls etc/hadoop
```

hadoop 配置文件在/usr/local/hadoop-2.7.3/etc/hadoop 目录下，主要修改 core-site.xml、hdfs-site.xml、mapred-site.xml、yarn-site.xml、masters、slaves 等配置文件。

在 hadoop 集群主节点 hadoop21（172.17.0.3）上操作

1.5.2.1 hadoop-env.sh

修改配置文件 hadoop-env.sh

```
vim /usr/local/hadoop-2.7.3/etc/hadoop/hadoop-env.sh 添加或修改以下两行  
export JAVA_HOME=/usr/local/jdk1.8.0_131  
export HADOOP_PREFIX=/usr/local/hadoop-2.7.3
```

注意：容器镜像已经配置好了，不需要修改！

1.5.2.2 core-site.xml

编辑配置文件 core-site.xml

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/core-site.xml  
vim /usr/local/hadoop-2.7.3/etc/hadoop/core-site.xml 内容如下  
<?xml version="1.0" encoding="UTF-8"?>  
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>  
<configuration>  
  <property>  
    <name>hadoop.tmp.dir</name>  
    <value>/root/hadoop/tmp</value>  
  </property>
```

```
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://172.17.0.3:9000</value>
</property>
</configuration>
```

fs.default.name 是一个描述集群中 NameNode 结点的 URI(包括协议、主机名称、端口号), 集群里面的每一台机器都需要知道 NameNode 的地址。DataNode 结点会先在 NameNode 上注册, 这样它们的数据才可以被使用。独立的客户端程序通过这个 URI 跟 DataNode 交互, 以取得文件的块列表。

hadoop.tmp.dir 是 hadoop 文件系统依赖的基础配置, 很多路径都依赖它。如果 hdfs-site.xml 中不配置 namenode 和 datanode 的存放位置, 默认就放在这个路径中。

HDFS 格式化时, hadoop.tmp.dir 指定的目录会被格式化, 如果没指定 hadoop.tmp.dir, 默认存储目录/tmp/hadoop-root/dfs/name 被格式化。

hadoop.tmp.dir 的默认值为/tmp, 一些版本的 Linux 系统会在每次重新启动时删除/tmp 的内容, 所以指定 hadoop.tmp.dir 数据留存的位置更为安全。

配置文件中指定了 hadoop.tmp.dir 路径为/root/hadoop/tmp, 需要手动创建这个目录。

```
mkdir -p /root/hadoop/tmp
```

1.5.2.3 hdfs-site.xml

编辑配置文件 hdfs-site.xml

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/hdfs-site.xml
vim /usr/local/hadoop-2.7.3/etc/hadoop/hdfs-site.xml 内容如下
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>3</value>
  </property>
```



```
<property>
  <name>dfs.namenode.name.dir</name>
  <value>/root/hadoop/namenodedir</value>
</property>

<property>
  <name>dfs.datanode.data.dir</name>
  <value>/root/hadoop/datanodedir</value>
</property>

<property>
  <name>dfs.permissions</name>
  <value>>false</value>
</property>
</configuration>
```

`dfs.namenode.name.dir` 是 NameNode 结点存储 hadoop 文件系统信息的本地系统路径，这个值只对 NameNode 有效，DataNode 并不需要使用到它。

`dfs.datanode.data.dir` 是 DataNode 结点被指定要存储数据的本地文件系统路径。DataNode 结点上的这个路径没有必要完全相同，因为每台机器的环境很可能是不一样的。但如果每台机器上的这个路径都是统一配置的话，会使工作变得简单一些。默认的情况下，它的值 `hadoop.tmp.dir` 这个路径只能用于测试的目的，因为它很可能会丢失掉一些数据。所以这个值最好还是被覆盖。

`dfs.replication` 决定着系统里面的文件块的数据备份个数。对于一个实际的应用，它应该被设为 3（这个数字并没有上限，但更多的备份可能并没有作用，而且会占用更多的空间）。少于三个的备份，可能会影响到数据的可靠性，系统故障时，也许会造成数据丢失。

配置文件中 `dfs.namenode.name.dir` 指定的路径为 `/root/hadoop/namenodedir`；`dfs.datanode.data.dir` 指定的路径 `/root/hadoop/datanodedir`，需要手动创建目录。

```
mkdir -p /root/hadoop/namenodedir
mkdir -p /root/hadoop/datanodedir
```

1.5.2.4 mapred-site.xml

编辑配置文件 mapred-site.xml

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/mapred-site.xml
vim /usr/local/hadoop-2.7.3/etc/hadoop/mapred-site.xml 内容如下
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>

  <property>
    <name>mapreduce.jobtracker.http.address</name>
    <value>172.17.0.3:50030</value>
  </property>

  <property>
    <name>mapreduce.jobhistory.address</name>
    <value>172.17.0.3:10020</value>
  </property>

  <property>
    <name>mapreduce.jobhistory.webapp.address</name>
    <value>172.17.0.3:19888</value>
  </property>
</configuration>
```

mapred.job.tracker 指定 JobTracker 的主机（或者 IP）和端口。

1.5.2.5 yarn-site.xml

编辑配置文件 yarn-site.xml

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/yarn-site.xml
```

`vim /usr/local/hadoop-2.7.3/etc/hadoop/yarn-site.xml` 内容如下

```
<?xml version="1.0"?>
<configuration>
  <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.aux-services</name>
    <value>mapreduce.shuffle</value>
  </property>

  <property>
    <name>yarn.resourcemanager.address</name>
    <value>172.17.0.3:8032</value>
  </property>

  <property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>172.17.0.3:8030</value>
  </property>

  <property>
    <name>yarn.resourcemanager.resource-tracker.address</name>
    <value>172.17.0.3:8031</value>
  </property>

  <property>
    <name>yarn.resourcemanager.admin.address</name>
    <value>172.17.0.3:8033</value>
  </property>
```

```
<property>
  <name>yarn.resourcemanager.webapp.address</name>
  <value>172.17.0.3:8088</value>
</property>
</configuration>
```

1.5.2.6 masters

编辑 hadoop 集群主节点 masters 文件

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/masters
vim /usr/local/hadoop-2.7.3/etc/hadoop/masters
172.17.0.3
```

1.5.2.7 slaves

编辑 hadoop 集群从节点 slaves 文件

```
rm -f /usr/local/hadoop-2.7.3/etc/hadoop/slaves
vim /usr/local/hadoop-2.7.3/etc/hadoop/slaves
172.17.0.4
172.17.0.5
172.17.0.6
```

1.5.2.8 拷贝文件

将 hadoop 集群主节点（172.17.0.3）上的配置文件，拷贝到从节点机器上。

```
scp -r /usr/local/hadoop-2.7.3/etc/hadoop root@hadoop22:/usr/local/hadoop-2.7.3/etc/
scp -r /usr/local/hadoop-2.7.3/etc/hadoop root@hadoop23:/usr/local/hadoop-2.7.3/etc/
scp -r /usr/local/hadoop-2.7.3/etc/hadoop root@hadoop24:/usr/local/hadoop-2.7.3/etc/
```

1.5.3 运行 Hadoop 服务

在 hadoop 集群主节点（172.17.0.3）上操作

1.5.3.1 格式化文件系统

在首次启动 Hadoop 之前，需要格式化 Hadoop 将要用到的 HDFS 文件系统，只需要在名字节点 NameNode 节点上运行。

如果重新格式化 HDFS 文件系统，有时需要删除各节点的数据。

```
rm -fr /root/hadoop/namenodedir /root/hadoop/datanodedir /root/hadoop/tmp
```

格式化 HDFS 分布式文件系统

```
/usr/local/hadoop-2.7.3/bin/hdfs namenode -format
```

根据配置文件 hdfs-site.xml，存储目录/root/hadoop/namenodedir 被成功格式化文件系统。

1.5.3.2 启动 HDFS 服务

启动 HDFS 服务，包括管理文件系统的名字节点 NameNode 和保存数据的数据节点 DataNode 服务。

```
/usr/local/hadoop-2.7.3/sbin/start-dfs.sh
```

Hadoop 集群从节点 slave 机器（172.17.0.4、172.17.0.5、172.17.0.6）上的数据节点 DataNode 服务会被自动启动。

1.5.3.3 启动 YARN 服务

启动 ResourceManager 和 NodeManager 服务

```
/usr/local/hadoop-2.7.3/sbin/start-yarn.sh
```

Hadoop 集群从节点 slave 机器（172.17.0.4、172.17.0.5、172.17.0.6）的 NodeManager 服务会被自动启动。

1.5.3.4 查看 HDFS 存储位置

机器（172.17.0.3）

查看 HDFS 名字节点 NameNode 的存储位置

```
ls /root/hadoop/namenodendir/
```

机器（172.17.0.4）

查看 HDFS 数据节点 DataNode 的存储位置

```
ls /root/hadoop/datanodendir/
```

机器（172.17.0.5）

查看 HDFS 数据节点 DataNode 的存储位置

```
ls /root/hadoop/datanodendir/
```

机器（172.17.0.6）

查看 HDFS 数据节点 DataNode 的存储位置

```
ls /root/hadoop/datanodendir/
```

1.5.3.5 查看 Java 进程

机器（172.17.0.3）

启动了 HDFS 和 YARN 服务后，使用 jps 命令可以查看启动的相关 Java 进程。

```
jps
```

已启动运行的 Java 进程如下所示：

```
12719 Jps
14645 SecondaryNameNode
11312 ResourceManager
13909 NameNode
```

机器（172.17.0.4）

查看 Java 进程

```
jps
```

已启动运行的 Java 进程如下所示：

```
15487 DataNode
16001 NodeManager
16123 Jps
```

机器（172.17.0.5）

查看 Java 进程

```
jps
```

已启动运行的 Java 进程如下所示：

```
15185 NodeManager
14902 DataNode
15297 Jps
```

机器（172.17.0.6）

查看 Java 进程

```
jps
```

已启动运行的 Java 进程如下所示：

```
14961 DataNode
15243 NodeManager
15356 Jps
```

1.5.4 网页 Hadoop

1.5.4.1 HDFS 网页接口

打开浏览器，输入 <http://172.17.0.3:50070>，可看到 HDFS 页面，如下图所示：

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities										
Datanode Information										
In operation										
Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
hadoop23:50010 (172.18.0.5:50010)	0	In Service	9.99 GB	4 KB	2.37 GB	7.62 GB	0	4 KB (0%)	0	2.7.3
hadoop22:50010 (172.18.0.4:50010)	0	In Service	9.99 GB	4 KB	2.37 GB	7.62 GB	0	4 KB (0%)	0	2.7.3
hadoop24:50010 (172.18.0.6:50010)	0	In Service	9.99 GB	4 KB	2.37 GB	7.62 GB	0	4 KB (0%)	0	2.7.3

点击上面的 Datanodes 菜单时，可以看到三个从节点 slave 的数据节点 DataNode。

1.5.4.2 YARN 网页接口

打开浏览器，输入 <http://172.17.0.3:8088> 可看到各种应用，如下图所示：

172.18.0.3:8088/cluster

Q

Search

☆


📁

🔒

⬇️

🏠

☰

 **hadoop**

All Applications

▼ Cluster

About

Nodes

Node Labels

Applications

NEW

NEW SAVING

SUBMITTED

ACCEPTED

RUNNING

FINISHED

FAILED

KILLED

Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved
0	0	0	0	0	0 B	24 GB	0 B	0	24	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>

Show 20 entries

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus
No data available in table								

Showing 0 to 0 of 0 entries

1.5.5 HDFS 操作

HDFS 允许用户数据组织成文件和文件夹的方式，它提供一个叫 DFSShell 的接口，使用户可以和 HDFS 中的数据交互。

hadoop 命令集的语法跟其他用户熟悉的 shells（bash、csh）类似。

操作	命令
创建目录 /foodir	hadoop dfs -mkdir /foodir
查看文件 /foodir/myfile.txt	hadoop dfs -cat /foodir/myfile.txt
删除文件/foodir/myfile.txt	hadoop dfs -rm /foodir myfile.txt

DFSAdmin 命令集是用于管理 dfs 集群的，这些命令只由 HDFS 管理员使用。

操作	命令
将集群设置成安全模式	hadoop dfsadmin -safemode enter
产生一个数据节点的列表	hadoop dfsadmin -report
去掉一个数据节点	hadoop dfsadmin -decommission datanodename

1、创建目录

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -mkdir /user  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -mkdir /user/zxuhong
```

hadoop 为每个用户保留一个主目录，这些主目录位于 HDFS 文件系统的/user 路径下；如果主目录不存在，需要创建主目录。

此时在浏览器中输入 <http://172.17.0.2:50070/explorer.html#/> 就可以看到新建的/user 目录。

2、查看目录

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -ls /user/  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -ls /user/zxuhong/
```

3、上传文件

```
echo "hello" > /tmp/hello  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -put /tmp/hello /user/zxuhong/
```

创建文件、上传文件。

4、查看上传文件

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -ls /user/zxuhong  
相当于  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -ls hdfs://172.17.0.2:9000/user/zxuhong
```

5、下载文件

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -get /user/zxuhong/hello /root/hello  
cat /root/hello
```

下载文件、查看文件内容

6、在线查看文件内容

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -cat /user/zxuhong/hello
```

7、删除文件

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm /user/zxuhong/hello
```

8、删除目录

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rmdir /user/zxuhong  
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r /user/
```

9、帮助

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -help
```

注意：上面所有的操作命令 `hdfs` 换成 `hadoop` 也是一样的；中间参数选项 `dfs` 换成 `f` `s` 也是一样的。

1.5.6 MapReduce 操作

1.5.6.1 上传作业

机器（172.17.0.3）上操作

上传作业

```
cd /usr/local/hadoop-2.7.3/  
bin/hdfs dfs -mkdir /user  
bin/hdfs dfs -mkdir /user/root  
bin/hdfs dfs -put etc/hadoop input  
bin/hdfs dfs -ls /user/root/input
```

第一条命令创建 `/user` 命令，相当于根目录。

第二条命令创建用户目录，跟当前登录用户有关。

第三条命令把 `etc/hadoop` 目录下文件拷贝到 HDFS 文件系统的 `/user/root/input` 目录下。

第四条命令查看用户目录刚上传的文件。

此时可在浏览器中 <http://172.17.0.3:50070/explorer.html#/user/root/input> 查看文件。

1.5.6.2 运行作业

机器（172.17.0.3）上操作

删除之前的结果

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r output
```

查询比较作业

```
cd /usr/local/hadoop-2.7.3/  
bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar grep in  
put output 'dfs[a-z.]+'
```

此时打开浏览器 <http://172.17.0.3:8088/cluster>，可以看到一个 `applications` 在运行。

1.5.6.3 查看作业

机器（172.17.0.3）上操作

下载作业结果到本地目录查看

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -get /user/root/output /tmp/output
cat /tmp/output/*
```

从 HDFS 文件系统下载 output 目录，此时/tmp 目录下有个 output 目录，再查看输出文件内容。

也可以直接在 HDFS 文件系统上查看

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -cat output/*
```

如果再次做作业，需要删除 input、output 目录，重新上传作业。

```
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r input
/usr/local/hadoop-2.7.3/bin/hdfs dfs -rm -r output
```

1.5.7 停止 Hadoop 服务

在 hadoop 集群主节点（172.17.0.3）上操作

1.5.7.1 停止 YARN 服务

停止 YARN 服务，包括 ResourceManager 和 NodeManager 服务

```
/usr/local/hadoop-2.7.3/sbin/stop-yarn.sh
```

1.5.7.2 停止 HDFS 服务

停止 HDFS 服务，包括管理文件系统的名字节点 NameNode 和保存数据的数据节点 DataNode 服务。

```
/usr/local/hadoop-2.7.3/sbin/stop-dfs.sh
```

1.6 Hadoop 集群快速部署

Hadoop 容器镜像在编译时，已经加入了快速部署的脚本。

1.6.1 运行容器

(1) 删除已经运行的所有容器

```
docker kill $(docker ps -a -q)
docker rm $(docker ps -a -q)
```

(2) 运行容器

伪分布式集群 (hadoop20)

```
docker run -d --name='hadoop20' --hostname='hadoop20' docker-centos6.10-hadoop-spark
```

完全分布式集群 (hadoop21、hadoop22、hadoop23、hadoop24)

```
docker run -d --name='hadoop21' --hostname='hadoop21' docker-centos6.10-hadoop-spark
docker run -d --name='hadoop22' --hostname='hadoop22' docker-centos6.10-hadoop-spark
```

```
docker run -d --name='hadoop23' --hostname='hadoop23' docker-centos6.10-hadoop-spark
docker run -d --name='hadoop24' --hostname='hadoop24' docker-centos6.10-hadoop-spark
```

1.6.2 连接容器

(1) 查看容器

伪分布式集群 (hadoop20)

```
docker inspect hadoop20
```

查看容器 IP 地址。

完全分布式集群 (hadoop21、hadoop22、hadoop23、hadoop24)

```
docker inspect hadoop21
docker inspect hadoop22
docker inspect hadoop23
docker inspect hadoop24
```

查看容器 IP 地址。

(2) SSH 连接容器

伪分布式集群 (hadoop20)

```
rm -f /root/.ssh/known_hosts
ssh 172.17.0.2
```

输入密码 123456

完全分布式集群 (hadoop21、hadoop22、hadoop23、hadoop24)

```
ssh 172.17.0.3
ssh 172.17.0.4
ssh 172.17.0.5
ssh 172.17.0.6
```

1.6.3 快速部署 hadoop

在 hadoop 集群主节点上操作

伪分布式集群（hadoop），机器 hadoop20（172.17.0.2）上

完全分布式集群（hadoop21、hadoop22、hadoop23、hadoop24），机器 hadoop21（172.17.0.3）上

进入目录，查看脚本

```
cd /root/hadoop  
ls
```

（1）编辑 masters 文件 内容是 hadoop master 节点 IP，

```
sh 1-edit-file-master.sh
```

（2）编辑 slaves 文件 内容是 hadoop slave 节点 IP，可以有多个 IP

```
sh 2-edit-file-slaves.sh
```

（3）编辑 hosts 文件 内容是对应的 IP 和机器名 P，其中机器名应该添加容器时指定的容器名称，这一步很关键，否则看不到 DataNode。

```
sh 3-edit-file-hosts.sh
```

注意：对于多节点集群，需要添加其他机器 IP 地址和机器名。

（4）设置 SSH 无密码验证，在 masters 节点上产生 SSH Key，然后 ssh-copy-id 拷贝到 slaves 节点上 依次输入各个节点的密码 123456

```
sh 4-setup-sshkey.sh
```

（5）设置 Hadoop 集群 通过脚本自动修改并拷贝配置文件 core-sites.xml、hdfs-site.xml、mapred-site.xml、yarn-site.xml、masters、slaves

```
sh 5-setup-cluster.sh
```

（6）格式化 HDFS 文件系统

```
sh 6-format-hdfs.sh
```

（7）运行 Hadoop 服务，包括 HDFS、YARN 服务，使用 jps 可以查看启动的服务

```
sh 7-start-hadoop.sh
```

(8) HDFS 基本操作测试

```
sh 8-hdfs-test.sh
```

(9) MapReduce 作业测试

```
sh 9-mapreduce-test.sh
```

(10) 停止 Hadoop 服务，包括 HDFS、YARN 服务

```
sh 10-stop-hadoop.sh
```

(11) HDFS 文件系统网页接口

<http://msterip:50070/>

(12) YARN 网页接口

<http://msterip:8088/>

1.7 问题

1.7.1 重启机器后容器处理

(1) 需要重新启动容器

```
docker start hadoop20
docker start hadoop21
docker start hadoop22
docker start hadoop23
docker start hadoop24
```

(2) 需要修改 hosts 文件

ssh 连接到容器后，修改/etc/hosts 文件

修改文件 /etc/hosts

```
vim /etc/hosts
```

内容如下

```
127.0.0.1    localhost
172.17.0.2   hadoop20
172.17.0.3   hadoop21
```


172.17.0.4	hadoop22
172.17.0.5	hadoop23
172.17.0.6	hadoop24