暨南大学本科实验报告专用纸

课程名称 云计算实验 成绩评定

实验项目名称 hadoop 集群搭建 指导教师 魏林锋

实验项目编号 08060308 实验项目类型 验证 实验地点 线上

学生姓名 陈宇 学号 2020101642

学院 信息科学技术学院 系 计算机系 专业 软件工程

实验时间 2022 年 9 月 26 日 上午～ 9 月 26 日 上午 温度 ℃湿度

1.1. 实验目的

1）理解 Hadoop 工作原理。

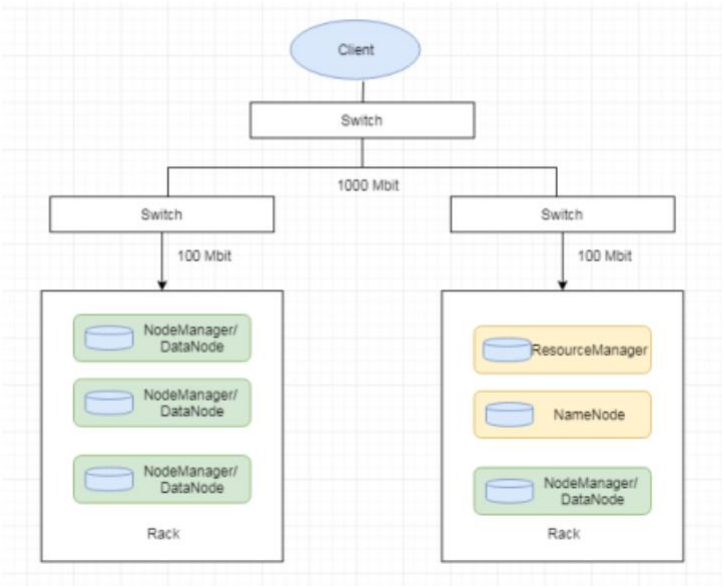
2）通过实验掌握 Hadoop 编译过程。

3）通过实验掌握 Hadoop 伪分布式、完全分布式的安装过程。

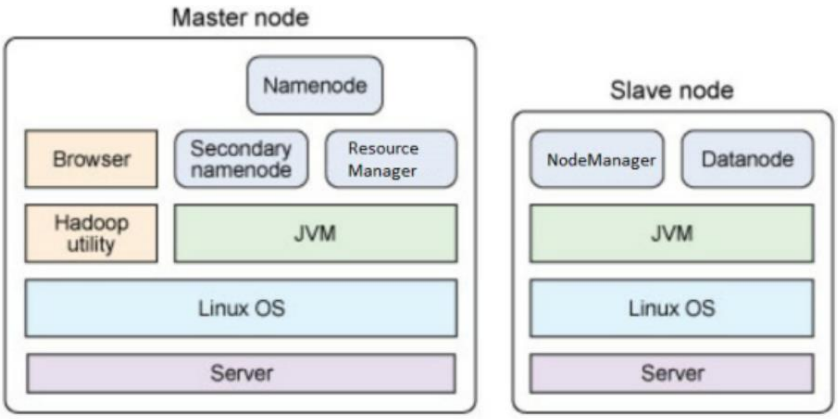
1.2. 实验原理

Hadoop：是一个适合海量数据分布式存储和分布式计算平台

Hadoop 集群物理分布



单节点物理结构



1.3. 实验环境

- **1.3.1. Hadoop 编译环境**

操作系统：CentOS7.5_x86_64

JDK 版本号：jdk-8u144

Hadoop：hadoop-3.1.0-src.tar.gz

Protobuf：protobuf-2.5.0.tar.gz

Maven：apache-maven-3.5.4-bin.tar.gz

Ant：apache-ant-1.10.5-bin.tar.gz

- **1.3.2. Hadoop 伪分布式实验环境**

操作系统：CentOS7.5_x86_64

hadoop 版本号：3.1.0

JDK 版本号：8u144 Hadoop

伪分布式只需一台主机，既是主节点也是从节点。

机器名称	IP 地址
Master	192.168.152.131
Slave	192.168.152.131

- **1.3.3. Hadoop 完全分布式实验环境**

操作系统：CentOS7.5_x86_64

hadoop 版本号：3.1.0

JDK 版本号：10.0.1

集群中包括 3 个节点：1 个 Master，2 个 Salve，节点之间局域网连接，可以相互 ping 通。

节点 IP 地址分布如下(IP 地址可以自行设置)：

机器名称	IP 地址
Master	172.17.0.2
Slave1	172.17.0.3
Slave2	172.17.0.4

1.4. 实验内容和过程

1.4.1. hadoop 编译

将 Hadoop 开源编译环境包通过 SCP 工具放入/opt 目录

● 1.4.1.1. 安装基本应用

[root@localhost~]# yum -y install svn ncurses-devel gcc*

[root@ localhost ~]# yum -y install lzo-devel zlib-devel autoconf automake libtool cmake openssl-devel

● 1.4.1.2. 安装 JDK

1.解压 JDK

[root@ localhost]# mkdir /usr/java

[root@ localhost]# cd /opt

[root@ localhost opt]# tar zxvf jdk-8u144-linux-x64.tar.gz -C /usr/java

```
[root@master usr]# cd java
[root@master java]# ls
jdk1.8.0_212
```

2.设置 jdk 运行环境

[root@ localhost]# vi /etc/profile　　添加环境变量

```
#set java environment
export JAVA_HOME=/usr/java/jdk1.8.0_212
export JRE_HOME=/usr/java/jdk1.8.0_212
export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JRE_HOME/lib
export PATH=$JAVA_HOME/bin:$JRE_HOME/bin:$PATH
```

3.使添加的环境变量生效

[root@ localhost]# source /etc/profile

[root@localhost~]# java -version

检验 JAVA 环境是否生效

```
[root@master ~]# java -version
java version "1.8.0_212"
Java(TM) SE Runtime Environment (build 1.8.0_212-b10)
Java HotSpot(TM) 64-Bit Server VM (build 25.212-b10, mixed mode)
[root@master ~]#
```

● 1.4.1.3. 安装 protobuf

1.解压缩

[root@ localhost]# cd /opt

[root@ localhost opt]# tar zxvf protobuf-2.5.0.tar.gz

2.进入目录

[root@ localhost protobuf-2.5.0]# cd protobuf-2.5.0

3.运行检测

[root@ localhost protobuf-2.5.0]# ./configure

(部分截图展示)

```
checking whether -pthread is sufficient with -shared... y
configure: creating ./config.status
config.status: creating Makefile
config.status: creating scripts/gtest-config
config.status: creating build-aux/config.h
config.status: build-aux/config.h is unchanged
config.status: executing depfiles commands
config.status: executing libtool commands
[root@master protobuf-2.5.0]#
```

4.编译

[root@ localhost protobuf-2.5.0]# make

5.安装

[root@ localhost protobuf-2.5.0]# make install

6.检验是否安装成功

[root@ localhost protobuf-2.5.0]#protoc –version

```
[root@master protobuf-2.5.0]# protoc --version
libprotoc 2.5.0
[root@master protobuf-2.5.0]#
```

- ### 1.4.1.4. 安装 maven

1.新建一个目录

[root@ localhost protobuf-2.5.0]# mkdir /usr/maven

2.解压缩

[root@ localhost protobuf-2.5.0]# cd /opt

[root@ localhost opt]# tar zxvf apache-maven-3.5.4-bin.tar.gz -C /usr/maven/

```
[root@master usr]# cd maven/
[root@master maven]# ls
apache-maven-3.5.4
[root@master maven]#
```

3.配置环境变量

[root@ localhost opt]# vi /etc/profile　　添加如下

```
#set maven environment
export MAVEN_HOME=/usr/maven/apache-maven-3.5.4
export PATH=$PATH:$MAVEN_HOME/bin
```

4.使添加的环境变量生效

[root@ localhost opt]# source /etc/profile

5.检验是否安装成功

[root@ localhost opt]# mvn -version

```
[root@master opt]# mvn -version
Apache Maven 3.5.4 (1edded0938998edf8bf061f1ceb3cfdeccf443fe; 2018-06-17T11:33:14-07:00)
Maven home: /usr/maven/apache-maven-3.5.4
Java version: 1.8.0_212, vendor: Oracle Corporation, runtime: /usr/java/jdk1.8.0_212/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "3.10.0-1160.el7.x86_64", arch: "amd64", family: "unix"
[root@master opt]#
```

● 1.4.1.5. 安装 ant

1.新建一个目录

[root@ localhost opt]# mkdir /usr/ant

2.解压缩

[root@ localhost opt]# tar zxvf apache-ant-1.10.5-bin.tar.gz -C /usr/ant/

```
[root@master opt]# cd /usr/ant
[root@master ant]# ls
apache-ant-1.10.5
```

3.添加环境变量

[root@ localhost opt]# vi /etc/profile

```
#set ant environment
export ANT_HOME=/usr/ant/apache-ant-1.10.5
export PATH=$PATH:$ANT_HOME/bin
```

4.使环境变量生效

[root@ localhost opt]# source /etc/profile

5.检验是否安装成功

[root@localhost opt]# ant -version

```
[root@master opt]# ant -version
Apache Ant(TM) version 1.10.5 compiled on July 10 2018
[root@master opt]#
```

● 1.4.1.6. 编译 hadoop

由于我们系统里的 Cmake2.8 版本过低无法满足编译要求，所以则需要卸载 CMake 然后获取 最新版进行编译安装

[root@ localhost opt]# yum install wget -y

[root@ localhost opt]# cd /usr/local/

[root@ localhost local]# wget https://cmake.org/files/v3.11/cmake-3.11.1.tar.gz

[root@ localhost local]# tar zxvf cmake-3.11.1.tar.gz

[root@ localhost local]# mv cmake-3.11.1/ cmake

[root@ localhost local]# cd cmake/

[root@ localhost cmake]# ./configure

[root@ localhost cmake]# make
[root@ localhost cmake]# make install

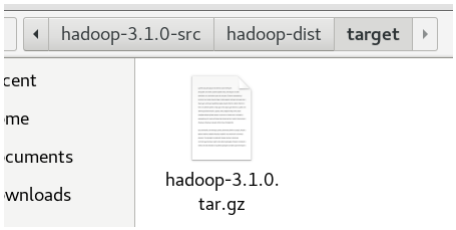
1.解压缩
[root@ localhost opt]# tar zxvf hadoop-3.1.0-src.tar.gz

2.进入目录
[root@ localhost opt]# cd hadoop-3.1.0-src

3.编译 hadoop
[root@ localhost hadoop-3.1.0-src]#mvn package -Pdist,native -DskipTests -Dt

由于网络问题编译一直假死，所以我直接去官网下

编译好的文件放在./hadoop-3.1.0-src/hadoop-dist/target/hadoop-3.1.0.tar.gz。

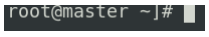


1.4.2. Hadoop 伪分布式安装

● 1.4.2.1. 网络配置

1.修改当前机器名称
[root@localhost~]# hostnamectl set-hostname master

2.更改机器名称后，需要重启 Linux 系统。
[root@ localhost~]# reboot
不需要 reboot!!! 直接执行“su -”即可解决。然后记得“exit”登出!!! 不放心就多 exit 几遍。完全分布式也一样



3.关闭防火墙和防火墙自启，查看防火墙状态。
[root@master ~]# systemctl stop firewalld
[root@master ~]# systemctl disable firewalld
[root@master ~]# systemctl status firewalld

```
[root@master opt]# cd
[root@master ~]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled;
 vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:firewalld(1)
[root@master ~]#
```

4.关闭 Slinux。

[root@master ~]# vi /etc/selinux/config

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=disable
# SELINUXTYPE= can take one of three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected process
s are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

5.查看当前机器 IP

[root@master ~]# ip addr

```
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast s
tate UP group default qlen 1000
    link/ether 00:0c:29:e2:b6:61 brd ff:ff:ff:ff:ff:ff
    inet 192.168.152.131/24 brd 192.168.152.255 scope global noprefixro
ute dynamic ens33
```

直接使用 ifconfig 命令即可。

```
[root@master ~]# ifconfig
docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        inet6 fe80::42:92ff:fe0e:d65f  prefixlen 64  scopeid 0x20<link>
        ether 02:42:92:0e:d6:5f  txqueuelen 0  (Ethernet)
        RX packets 22067  bytes 900867 (879.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 34394  bytes 78825779 (75.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.152.131  netmask 255.255.255.0  broadcast 192.168.
152.255
        inet6 fe80::3d85:88c9:6bb9:3a08  prefixlen 64  scopeid 0x20<lin
k>
        ether 00:0c:29:e2:b6:61  txqueuelen 1000  (Ethernet)
        RX packets 56543  bytes 81331702 (77.5 MiB)
```

6.配置 hosts 文件

[root@master ~]# vi /etc/hosts

做伪分布式的时候，master 和 slave 的 ip 需要保持一致。

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.local
domain4
::1         localhost localhost.localdomain localhost6 localhost6.local
domain6
192.168.152.131 master
192.168.152.131 slave
~
```

7.创建一个普通用户

[root@master ~]# useradd hadoop
[root@master ~]# passwd Hadoop

● 1.4.2.2. SSH 无密码验证设置

Hadoop 需要使用 SSH 协议，namemode 将使用 SSH 协议启动 namenode 和 data node 进程，伪分布式模式数据节点和名称节点均是本身，必须配置 SSH localhost 无密码 验证。

1.切换到 hadoop 用户

[root@master ~]# su - hadoop

2.生成密钥对

[hadoop@master ~]$ssh-keygen -t rsa -P ''

```
[hadoop@master ~]$ ssh-keygen -t rsa -P ''
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hadoop/.ssh/id_rsa):
Created directory '/home/hadoop/.ssh'.
Your identification has been saved in /home/hadoop/.ssh/id_rsa.
Your public key has been saved in /home/hadoop/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:tqER/kav2yVJcMV3Wvt0XzwMTjAHkxBXbDDyxehFd5M hadoop@master
The key's randomart image is:
+---[RSA 2048]----+
|          +oX%= oo|
|           ++BBoE=|
|       . . o.=.o=.|
|      . . o . ..+=|
|       o S .    .*|
|        * = .    o|
|       . + + .    |
|        . o o     |
|         o..      |
+----[SHA256]-----+
[hadoop@master ~]$
```

3.查看"/home/hadoop/"下是否有".ssh"文件夹，且".ssh"文件下是否有两个刚生产的无密码密钥对

[hadoop@master ~]$ ls .ssh/

```
[hadoop@master ~]$ ls .ssh/
authorized_keys  id_rsa  id_rsa.pub  known_hosts
[hadoop@master ~]$
```

4.把 id_rsa.pub 追加到授权的 key 里面去

[hadoop@master ~]$ cat cat .ssh/id_rsa.pub >> .ssh/authorized_keys

```
[hadoop@master ~]$ ll .ssh/
total 12
-rw-rw-r--. 1 hadoop hadoop  395 Sep 21 07:11 authorized_keys
-rw-------. 1 hadoop hadoop 1675 Sep 21 07:06 id_rsa
-rw-r--r--. 1 hadoop hadoop  395 Sep 21 07:06 id_rsa.pub
[hadoop@master ~]$
```

5.修改文件"authorized_keys"的权限

[hadoop@master ~]$chmod 600 ~/.ssh/authorized_keys
如果权限太大，ssh 服务拒绝工作。

6.使用 hadoop 普通用户验证是否成功

[hadoop@master ~]$ ssh localhost 或者 ssh 192.168.24.213

[hadoop@master ~]$ exit #退出

[hadoop@master ~]$ exit #切换到 root 用户

```
[hadoop@master ~]$ ssh localhost
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is SHA256:pFIKViE65AS+KTRuVGqQTXO4nV/T6sdm9EVmPtw7XD0.
ECDSA key fingerprint is MD5:99:84:99:ae:0e:f8:95:b5:7a:ee:d1:5c:d2:ba:d3:c1.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': y
Please type 'yes' or 'no': yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Last login: Wed Sep 21 07:05:41 2022
[hadoop@master ~]$ exit
logout
Connection to localhost closed.
[hadoop@master ~]$ exit
logout
[root@master ~]#
```

● 1.4.2.4. Hadoop 的安装和配置

1.进入/opt 目录，把 hadoop 解压到/usr 目录下。

[root@master ~]#cd /opt/

[root@master opt]# tar zxvf hadoop-3.1.0.tar.gz -C /usr/

2.重命名 hadoop 解压缩后目录。

[root@master opt]# mv /usr/hadoop-3.1.0 /usr/Hadoop

[root@master opt]# cd

```
[root@master hadoop]# cd
[root@master ~]# cd /usr/hadoop
[root@master hadoop]# ls
bin  hadoop-env.sh  lib      LICENSE.txt  NOTICE.txt  sbin   tmp
etc  include        libexec  logs         README.txt  share
[root@master hadoop]#
```

3.配置 hadoop 环境变量。

[root@master ~]# vi /etc/profile

在/etc/profile 文件末尾添加

```
#set hadoop environment
export HADOOP_HOME=/usr/hadoop
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH
```

4.使配置的 hadoop 的环境变量生效。

[root@master ~]#source /etc/profile

5.修改 hadoop-env.sh

[root@master ~]#cd /usr/hadoop/etc/hadoop/

[root@master hadoop]# vi hadoop-env.sh

```
# The java implementation to use. By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/usr/java/jdk1.8.0_212
```

6.修改 core-site.xml

[root@master hadoop]# vi core-site.xml

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

<configuration>
<property>
<name>fs.defaultFS</name>
<value>hdfs://192.168.152.132:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/usr/hadoop/tmp</value>
</property>
</configuration>
```

7. 创建临时目录

[root@master hadoop]# mkdir /usr/hadoop/tmp/

8.修改 hdfs-site.xml

[root@master hadoop]# vi hdfs-site.xml

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

<configuration>
<property>
<name>dfs.datanode.ipc.address</name>
<value>0.0.0.0:50020</value>
</property>
<property>
<name>dfs.datanode.http.address</name>
<value>0.0.0.0:50075</value>
</property>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.namenode.http-address</name>
<value>0.0.0.0:50070</value>
</property>
</configuration>
~
```

注：如果只需要跑起来即可，只需要配置 dfs.replication 即可，另外二个节点，是为了方便 eclipse 里 hadoop-eclipse-plugin 配置时，通过 ipc.address 连接，http.address 则是为了方便通过浏览器查看 datanode

9.修改 yarn-site.xml

[root@master hadoop]# vi yarn-site.xml

```
!-- Site specific YARN configuration properties -->
property>
name>yarn.nodemanager.aux-services</name>
value>mapreduce_shuffle</value>
/property>
/configuration>
```

10.新建 masters 文件和 slaves 文件。

[root@master hadoop]# vi masters #加入以下内容

```
196.168.152.132
~
```

[root@master hadoop]# vi slaves #删除 lacalhost,加入以下内容

```
196.168.152.132
~
```

注：因为在伪分布式模式下，作为 master 的 namenode 与作为 slave 的 datanode 是同一台服务器，所以配置文件中的 ip 是一样的。

11.将文件夹"hadoop"读权限分配给 hadoop 用户。

[root@master hadoop]# cd

[root@master ~]#chown -R hadoop:hadoop /usr/hadoop

● 1.4.2.5. 测试

1.切换到 hadoop。（若没有则创建）

[root@master ~]#su - hadoop

2.先格式化

[hadoop@master ~]$ cd /usr/hadoop/

[hadoop@master hadoop]$ ./bin/hdfs namenode -format

```
[hadoop@master hadoop]$ ./bin/hdfs namenode -format
2022-09-25 01:39:33,416 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = master/192.168.152.131
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 3.1.0
STARTUP_MSG:   classpath = /usr/hadoop/etc/hadoop:/usr/hadoop/share/hadoop/common/lib/htt
pcore-4.4.4.jar:/usr/hadoop/share/hadoop/common/lib/jetty-util-9.3.19.v20170502.jar:/usr/
hadoop/share/hadoop/common/lib/jackson-core-2.7.8.jar:/usr/hadoop/share/hadoop/common/lib
/mockito-all-1.8.5.jar:/usr/hadoop/share/hadoop/common/lib/kerb-core-1.0.1.jar:/usr/hadoo
p/share/hadoop/common/lib/kerb-crypto-1.0.1.jar:/usr/hadoop/share/hadoop/common/lib/acces
sors-smart-1.2.jar:/usr/hadoop/share/hadoop/common/lib/stax2-api-3.1.4.jar:/usr/hadoop/sh
```

3.启动 Hadoop。

[hadoop@master hadoop]$ ./sbin/start-dfs.sh

```
[hadoop@master hadoop]$ ./sbin/start-dfs.sh
Starting namenodes on [172.17.0.1]
Starting datanodes
Starting secondary namenodes [master]
[hadoop@master hadoop]$
```

[hadoop@master hadoop]$ sbin/start-yarn.sh

```
[hadoop@master hadoop]$ sbin/start-yarn.sh
Starting resourcemanager
Starting nodemanagers
[hadoop@master hadoop]$
```

4.查看 Java 进程

[hadoop@master hadoop]$ jps

```
[hadoop@master hadoop]$ jps
15506 ResourceManager
15254 SecondaryNameNode
15818 NodeManager
14911 NameNode
15999 Jps
[hadoop@master hadoop]$
```

5.查看 hdfs 的报告。

[hadoop@master hadoop]$ bin/hdfs

```
[hadoop@master hadoop]$ bin/hdfs
Usage: hdfs [OPTIONS] SUBCOMMAND [SUBCOMMAND OPTIONS]

  OPTIONS is none or any of:

--buildpaths                       attempt to add class files from build tree
--config dir                       Hadoop config directory
--daemon (start|status|stop)       operate on a daemon
--debug                            turn on shell script debug mode
--help                             usage information
--hostnames list[,of,host,names]   hosts to use in worker mode
--hosts filename                   list of hosts to use in worker mode
--loglevel level                   set the log4j level for this command
--workers                          turn on worker mode

  SUBCOMMAND is one of:


    Admin Commands:

cacheadmin           configure the HDFS cache
crypto               configure HDFS encryption zones
debug                run a Debug Admin to execute HDFS debug commands
dfsadmin             run a DFS admin client
dfsrouteradmin       manage Router-based federation
ec                   run a HDFS ErasureCoding CLI
fsck                 run a DFS filesystem checking utility
haadmin              run a DFS HA admin client
jmxget               get JMX exported values from NameNode or DataNode.
oev                  apply the offline edits viewer to an edits file
oiv                  apply the offline fsimage viewer to an fsimage
oiv_legacy           apply the offline fsimage viewer to a legacy fsimage
storagepolicies      list/get/set block storage policies

    Client Commands:

classpath            prints the class path needed to get the hadoop jar and the
                     required libraries
dfs                  run a filesystem command on the file system
envvars              display computed Hadoop environment variables
fetchdt              fetch a delegation token from the NameNode
getconf              get config values from configuration
groups               get the groups which users belong to
lsSnapshottableDir   list all snapshottable dirs owned by the current user
snapshotDiff         diff two snapshots of a directory or diff the current directory
                     contents with a snapshot
version              print the version

    Daemon Commands:

balancer             run a cluster balancing utility
datanode             run a DFS datanode
dfsrouter            run the DFS router
```

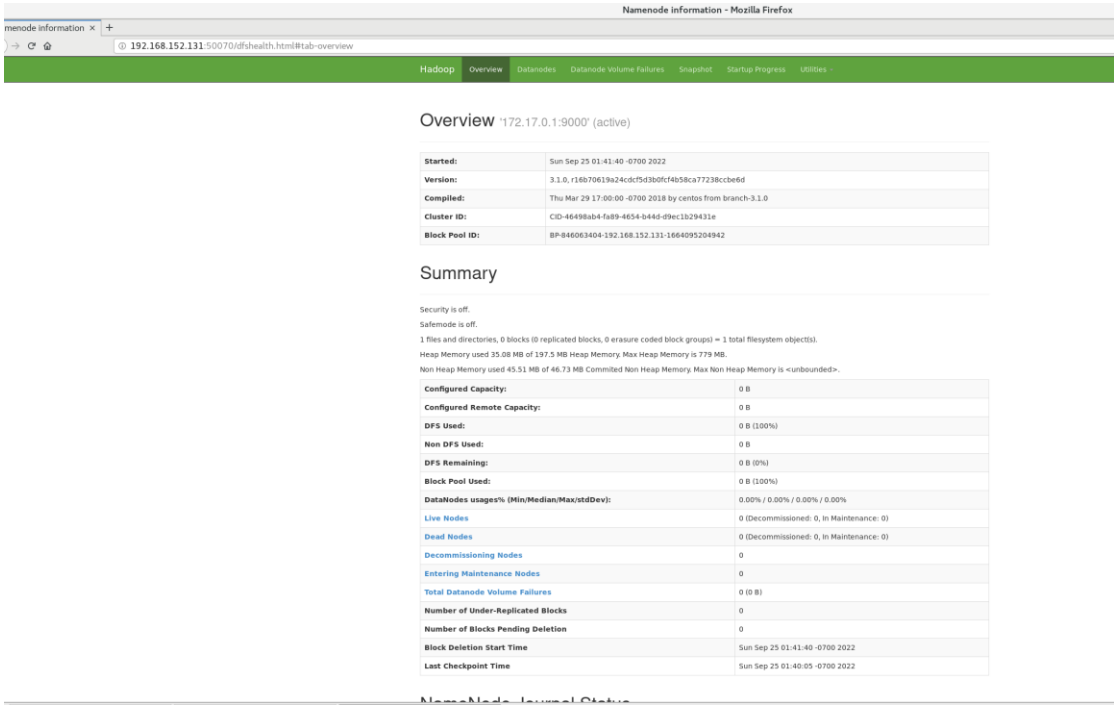
```
diskbalancer         Distributes data evenly among disks on a given node
httpfs               run HttpFS server, the HDFS HTTP Gateway
journalnode          run the DFS journalnode
mover                run a utility to move block replicas across storage types
namenode             run the DFS namenode
nfs3                 run an NFS version 3 gateway
portmap              run a portmap service
secondarynamenode    run the DFS secondary namenode
zkfc                 run the ZK Failover Controller daemon

SUBCOMMAND may print help when invoked w/o parameters or with -h.
[hadoop@master hadoop]$
```

6.Hadoop3.1.0 默认的 web 页面端口为 9870 端口，我们可以修改 /usr/Hadoop/etc/Hadoop/ hadoop-env.sh 文件并添加设置使默认的访问端口更改为 50070 端口

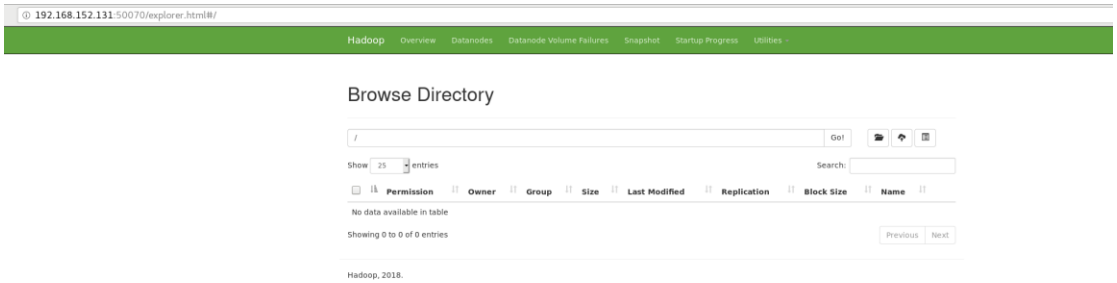
[root@master hadoop]# vi hadoop-env.sh

```
<configure>
<property>
<name>dfs.namenode.http-address</name>
<value>0.0.0.0:50070</value>
</property>
</configure>
~
~
~
```

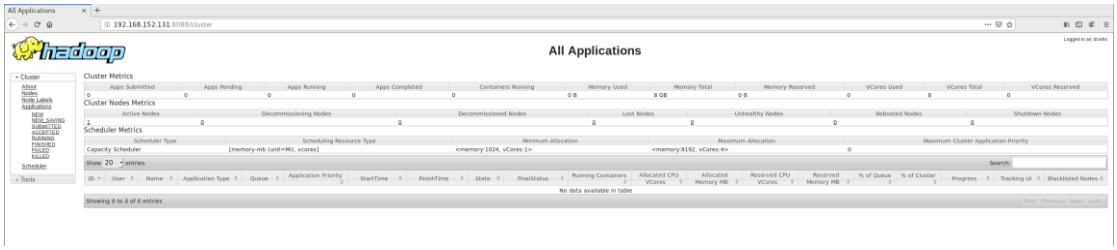
7.使用浏览器浏览 Master 节点机 http:// 192.168.152.131:50070,查看 NameNode 节点状态



8.浏览 hadoop 的文件系统



9.使用浏览器浏览 Master 节点机 http:// 192.168.152.131:8088 查看所有应用，浏览 Nodes



1.4.3. Hadoop 完全分布式安装

● 1.4.3.1. 网络配置（所有节点都要配置）

1.修改当前机器名

[root@localhost~]# hostnamectl set-hostname maste

```
root@hengdian /]# su -
root@master ~]# Port 6080 in use. Try --listen PORT

1]+  Exit 1                  /usr/local/noVNC/utils/novnc_
root@master ~]# exit
ogout
root@hengdian /]# exit
xit
root@master /]# Port 6080 in use. Try --listen PORT

1]+  Exit 1                  /usr/local/noVNC/utils/novnc_
root@master /]#
```

2.更改机器名称后，需要重启 Linux 系统。

[root@ localhost ~]# reboot

3.关闭防火墙和防火墙自启，查看防火墙状态。

[root@master ~]# systemctl stop firewalld

[root@master ~]# systemctl disable firewalld

[root@master ~]# systemctl status firewalld

```
[root@master /]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset:
enabled)
   Active: inactive (dead)
     Docs: man:firewalld(1)
[root@master /]#
```

4.关闭 Slinux。

[root@master ~]# vi /etc/selinux/config

```
#     permissive - SELinux prints warnings inst
#     disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three values:
#     targeted - Targeted processes are protect
```

5.修改当前机器 IP

[root@master ~]#vi /etc/sysconfig/network-scripts/ifcfg-eth0

```
172.17.0.2
~
```

6.配置 hosts 文件

[root@master ~]#vi /etc/hosts

```
::1     localhost ip6-localhost ip6-loop
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
172.17.0.2      0514918de0d6
172.17.0.2      master
172.17.0.3      slave1
172.17.0.4      slave2
~
~
```

7.重启网络。

[root@master ~]# systemctl restart network

8.创建一个普通用户。

[root@master ~]# useradd Hadoop

[root@master ~]# passwd hadoop

```
[root@master /]# useradd hadoop
[root@master /]# passwd hadoop
Changing password for user hadoop.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@master /]#
```

● 1.4.3.2. SSH 无密码验证配置

Hadoop 运行过程中需要管理远端 Hadoop 守护进程，在 Hadoop 启动以后，NameNode 是通过 SSH（Secure Shell）来启动和停止各个 DataNode 上的各种守护进程的。这就必须在 节

点之间执行指令的时候是不需要输入密码的形式，故我们需要配置 SSH 运用无密码公钥认证的形式，这样 NameNode 使用 SSH 无密码登录并启动 DataName 进程，同样原理，DataNode 上也能使用 SSH 无密码登录到 NameNode。

1.安装和启动 SSH 协议（所有节点）。

我们需要两个服务：ssh 和 rsync 已经安装了。可以通过下面命令查看结果

[root@master ~]#rpm -qa | grep openssh [root@master ~]# rpm -qa

```
[root@master /]# rpm -qa | grep openssh
openssh-server-sysvinit-7.4p1-22.el7_9.x86_64
openssh-cavs-7.4p1-22.el7_9.x86_64
openssh-7.4p1-22.el7_9.x86_64
openssh-keycat-7.4p1-22.el7_9.x86_64
openssh-server-7.4p1-22.el7_9.x86_64
openssh-clients-7.4p1-22.el7_9.x86_64
openssh-askpass-7.4p1-22.el7_9.x86_64
openssh-ldap-7.4p1-22.el7_9.x86_64
[root@master /]# rpm -qa | grep rsync
rsync-3.1.2-11.el7_9.x86_64
[root@master /]#
```

假设没有安装 ssh 和 rsync，可以通过下面命令进行安装。

[root@master ~]# yum install openssh* -y

[root@master ~]# yum install rsync -y

2.切换到 hadoop 用户。

[root@master ~]# su - hadoop

3.每个节点生成秘钥对（所有节点）

```
[hadoop@master ~]$ ssh-keygen -t rsa -P ''
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hadoop/.ssh/id_rsa):
Created directory '/home/hadoop/.ssh'.
Your identification has been saved in /home/hadoop/.ssh/id_rsa.
Your public key has been saved in /home/hadoop/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:T3g7vehoOERlP8qNAW61QpWPGQeLJEUWhp/xOrlmMBw hadoop@master
The key's randomart image is:
+---[RSA 2048]----+
|  .xX+=B         |
|   .B.o=oB       |
|    .=+o@ .      |
|   E.o.* ..      |
|  . . +  S o     |
|   + + .  + o    |
|    o + .  + .   |
|     + o .. o .  |
|    o    o..o .  |
+----[SHA256]-----+
[hadoop@master ~]$
```

4.查看"/home/hadoop/"下是否有".ssh"文件夹，且".ssh"文件下是否有两个刚生产的无密码密钥对（所有节点）

[hadoop@master ~]$ ls .ssh

```
[hadoop@master ~]$ ls .ssh/
id_rsa  id_rsa.pub
[hadoop@master ~]$
```

5.把 id_rsa.pub 追加到授权的 key 里面去（所有节点）

[hadoop@master ~]$ cat .ssh/id_rsa.pub >> .ssh/authorized_keys

```
[hadoop@master ~]$ ls .ssh/
authorized_keys  id_rsa  id_rsa.pub
[hadoop@master ~]$
```

6.修改文件"authorized_keys" 权限（所有节点）

[hadoop@master ~]$ chmod 600 ~/.ssh/authorized_keys

```
[hadoop@master ~]$ ll .ssh/
total 12
-rw-------. 1 hadoop hadoop  395 Sep 26 15:15 authorized_keys
-rw-------. 1 hadoop hadoop 1679 Sep 26 15:06 id_rsa
-rw-r--r--. 1 hadoop hadoop  395 Sep 26 15:06 id_rsa.pub
[hadoop@master ~]$
```

7.验证是否成功（所有节点）

[hadoop@master ~]$ ssh localhost #首次需要输入 yes 确定

[hadoop@master ~]$ exit

```
[hadoop@master ~]$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:hfayqhAx2+GP/BcbIs0I3E8uGrBNr7aefp+SUfbTK7Q.
ECDSA key fingerprint is MD5:34:e6:7d:7e:70:03:bd:66:bc:af:81:d8:1d:c9:8d:32.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Last login: Mon Sep 26 15:05:07 2022
[hadoop@master ~]$ exit
logout
Connection to localhost closed.
[hadoop@master ~]$
```

8.把 master 节点的公钥 id_rsa_pub 复制到每个 slave 点

[hadoop@master ~]$ scp .ssh/id_rsa.pub slave1:~/

[hadoop@master ~]$ scp .ssh/id_rsa.pub slave2:~/

```
[hadoop@master ~]$ scp .ssh/id_rsa.pub slave1:~/
The authenticity of host 'slave1 (172.17.0.3)' can't be established.
ECDSA key fingerprint is SHA256:selO+IrXgBfv9ibyfpG98nexS8Uo1PXOAvia+8K8RHM.
ECDSA key fingerprint is MD5:39:1c:14:07:06:00:ff:79:08:43:89:d2:1e:fe:52:de.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'slave1,172.17.0.3' (ECDSA) to the list of known hosts.
hadoop@slave1's password:
id_rsa.pub                                    100%  395   272.1KB/s   00:00
[hadoop@master ~]$ scp .ssh/id_rsa.pub slave2:~/
The authenticity of host 'slave2 (172.17.0.4)' can't be established.
ECDSA key fingerprint is SHA256:FNXyeFAq7TOWeRFa/YUE+5RbaFex4wHgq47cY1P+eZo.
ECDSA key fingerprint is MD5:2e:47:99:99:0a:18:65:9e:5b:e8:5b:5c:80:31:ea:61.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'slave2,172.17.0.4' (ECDSA) to the list of known hosts.
hadoop@slave2's password:
id_rsa.pub                                    100%  395   646.1KB/s   00:00
[hadoop@master ~]$
```

9.在每个 slave 节点把 master 节点复制的公钥复制到 authorized_keys 文件。

[hadoop@slave1 ~]$ cat id_rsa.pub >> .ssh/authorized_keys

```
[hadoop@slave1 ~]$ cat id_rsa.pub >> .ssh/authorized_keys
[hadoop@slave1 ~]$
```

[hadoop@slave2 ~]$ cat id_rsa.pub >> .ssh/authorized_keys

```
[hadoop@slave2 ~]$ cat id_rsa.pub >> .ssh/authorized_keys
[hadoop@slave2 ~]$
```

10.删除 id_rsa.pub 文件（所有 slave 节点）。

[hadoop@slave1~]$ rm id_rsa.pub

```
[hadoop@slave1 ~]$ rm id_rsa.pub
[hadoop@slave1 ~]$
```

[hadoop@slave2~]$ rm id_rsa.pub

```
[hadoop@slave2 ~]$ rm id_rsa.pub
[hadoop@slave2 ~]$
```

11.验证 master 节点到每个 slave 节点无密码验证

[hadoop@master ~]$ ssh slave1

[hadoop@master ~]$ ssh slave2

```
[hadoop@master ~]$ ssh slave1
Last login: Mon Sep 26 15:37:03 2022 from localhost
[hadoop@slave1 ~]$ exit
logout
Connection to slave1 closed.
[hadoop@master ~]$ ssh slave2
Last login: Mon Sep 26 15:38:41 2022 from localhost
[hadoop@slave2 ~]$
```

12.把 slave1 节点的公钥复制到 master。（注意：不要跟 slave2 同时复制）

[hadoop@slave1 ~]$ scp .ssh/id_rsa.pub master:~/

```
[hadoop@slave1 ~]$ scp .ssh/id_rsa.pub master:~/
The authenticity of host 'master (172.17.0.2)' can't be established.
ECDSA key fingerprint is SHA256:hfayqhAx2+GP/BcbIs0I3E8uGrBNr7aefp+SUfbTK7Q.
ECDSA key fingerprint is MD5:34:e6:7d:7e:70:03:bd:66:bc:af:81:d8:1d:c9:8d:32.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'master,172.17.0.2' (ECDSA) to the list of known hosts.
hadoop@master's password:
id_rsa.pub                                    100%  395   386.4KB/s   00:00
[hadoop@slave1 ~]$
```

13. 在 master 节点把从 slave1 节点复制的公钥复制到 authorized_keys 文件。删除 id_rsa.pub 文件

[hadoop@master ~]$ cat id_rsa.pub >> .ssh/authorized_keys

```
Connection to slave2 closed.
[hadoop@master ~]$ cat id_rsa.pub >> .ssh/authorized_keys
[hadoop@master ~]$ rm id_rsa.pub
[hadoop@master ~]$
```

14. 把 slave2 节点的公钥复制到 master。

[hadoop@slave2 ~]$ scp .ssh/id_rsa.pub master:~/

```
[hadoop@slave2 ~]$ scp .ssh/id_rsa.pub master:~/
The authenticity of host 'master (172.17.0.2)' can't be established.
ECDSA key fingerprint is SHA256:hfayqhAx2+GP/BcbIs0I3E8uGrBNr7aefp+SUfbTK7Q.
ECDSA key fingerprint is MD5:34:e6:7d:7e:70:03:bd:66:bc:af:81:d8:1d:c9:8d:32.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'master,172.17.0.2' (ECDSA) to the list of known hosts.
hadoop@master's password:
id_rsa.pub                                    100%  395   433.6KB/s   00:00
[hadoop@slave2 ~]$
```

15. 在 master 节点把从 slave1 节点复制的公钥复制到 authorized_keys 文件。删除 id_rsa.pub 文件

[hadoop@master ~]$ cat id_rsa.pub >> .ssh/authorized_keys

```
[hadoop@master ~]$ cat id_rsa.pub >> .ssh/authorized_keys
[hadoop@master ~]$ rm id_rsa.pub
[hadoop@master ~]$ cat id_rsa.pub >> .ssh/authorized_keys
[hadoop@master ~]$ rm id_rsa.pub
[hadoop@master ~]$
```

16. 验证每个 slave 节点到 master 无密码验证，然后退出。

Slave1 节点

[hadoop@slave1 ~]$ ssh master

[hadoop@master ~]$ exit

```
[hadoop@slave1 ~]$ ssh master
Last login: Mon Sep 26 15:18:24 2022 from localhost
[hadoop@master ~]$ exit
logout
Connection to master closed.
[hadoop@slave1 ~]$
```

Slave2 节点

[hadoop@slave2 ~]$ ssh master

[hadoop@master ~]$ exit

```
[hadoop@slave2 ~]$ ssh master
Last login: Mon Sep 26 15:59:23 2022 from slave1
[hadoop@master ~]$ exit
logout
Connection to master closed.
[hadoop@slave2 ~]$
```

● 1.4.3.3. Java 环境安装（所有节点都要配置）

1.切换到 root 用户。

[hadoop@master ~]$ exit

2.新建 java 目录。

[root@master ~]# mkdir /usr/java

3.解压到/usr/java 目录下。

[root@master ~]# tar zxvf jdk-8u144-linux-x64.tar.gz -C /usr/java/

/usr/java/

name	size	date
..	<dir>	--.--.----
jdk1.8.0_212	<dir>	02.04.2019

4.配置环境变量。

[root@master ~]# vi /etc/profile #末尾添加

```
#set java environment
export JAVA_HOME=/usr/java/jdk1.8.0_212
export JRE_HOME=/usr/java/jdk1.8.0_212/jre
export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JRE_HOME/lib
export PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
```

5.使添加的环境变量生效。

[root@master ~]# source /etc/profile

6.验证安装成功。

[root@master ~]#java -version

```
[root@master /]# source /etc/profile
[root@master /]# java -version
openjdk version "1.8.0_342"
OpenJDK Runtime Environment (build 1.8.0_342-b07)
OpenJDK 64-Bit Server VM (build 25.342-b07, mixed mode)
[root@master /]#
```

● 1.4.3.4. 在 Master 节点上安装 hadoop

1.解压缩到/usr 目录下。

[root@master ~]# tar zxvf hadoop-3.1.0.tar.gz -C /usr/

```
/usr/hadoop-3.1.0/
name
..
bin
etc
include
lib
libexec
sbin
share
LICENSE.txt
NOTICE.txt
README.txt
```

2.重命名。

[root@master ~]# mv /usr/hadoop-3.1.0 /usr/hadoop

3.配置 hadoop 环境变量。

[root@master ~]# vi /etc/profile #文件末尾添加

```
#set hadoop environment
export HADOOP_HOME=/usr/hadoop
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH
```

4.使配置的 hadoop 的环境变量生效。

[root@master ~]#source /etc/profile

5.配置 hadoop-env.sh。

[root@master ~]# cd /usr/hadoop/etc/hadoop/

[root@master hadoop]# vi hadoop-env.sh

```
# The java implementation to use. By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/usr/java/jdk1.8.0_212
```

6.配置 core-site.xml。

[root@master hadoop]# vi core-site.xml

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

<configuration>
        <property>
                <name>fs.defaultFS</name>
                <value>hdfs://master:9000</value>
        </property>
        <property>
                <name>io.file.buffer.size</name>
                <value>131072</value>
        </property>
        <property>
                <name>hadoop.tmp.dir</name>
                <value>file:/usr/hadoop/tmp</value>
        </property>


</configuration>
~
```

7.配置 hdfs-site.xml

[root@master hadoop]# vi hdfs-site.xml

```
configuration>
        <property>
                <name>dfs.namenode.name.dir</name>
                <value>file:/usr/hadoop/dfs/name</value>
        </property>
        <property>
                <name>dfs.namenode.data.dir</name>
                <value>file:/usr/hadoop/dfs/data</value>
        </property>
        <property>
                <name>dfs.replication</name>
                <value>3</value>
        </property>
        <property>
                <name>dfs.namenode.http-address</name>
                <value>0.0.0.0:50070</value>
        </property>
/configuration>
```

修改 Hadoop 中 HDFS 的配置，配置的备份方式默认为 3

8.配置 yarn-site.xml

[root@master hadoop]# vi yarn-site.xml

```
<configuration>

<!-- Site specific YARN configuration properties -->

        <property>
                <name>yarn.resourcemanager.address</name>
                <value>master:8032</value>
        </property>
        <property>
                <name>yarn.resourcemanager.scheduler.address</name>
                <value>master:8030</value>
        </property>
        <property>
                <name>yarn.resourcemanager.resource-tracker.address</name>
                <value>master:8031</value>
        </property>
        <property>
                <name>yarn.resourcemanager.admin.address</name>
                <value>master:8033</value>
        </property>
        <property>
                <name>yarn.resourcemanager.webapp.address</name>
                <value>master:8088</value>
        </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>

</configuration>
```

9.配置 mapred-site.xml

[root@master hadoop]# cp mapred-site.xml.template mapred-site.xml

[root@master hadoop]# vi mapred-site.xml

```
<configuration>
        <property>
                <name>mapreduce.framework.name</name>
                <value>yarn</value>
        </property>
        <property>
                <name>mapreduce.jobhistory.address</name>
                <value>master:10020</value>
        </property>
        <property>
                <name>mapreduce.jobhistory.webapp.address</name>
                <value>master:19888</value>
        </property>
        <property>
                <name>yarn.app.mapreduce.am.env</name>
                <value>HADOOP_MAPRED_HOME=/usr/hadoop/</value>
        </property>
        <property>
                <name>mapreduce.map.env</name>
                <value>HADOOP_MAPRED_HOME=/usr/hadoop</value>
        </property>
        <property>
                <name>mapreduce.reduce.env</name>
                <value>HADOOP_MAPRED_HOME=/usr/hadoop</value>
        </property>
</configuration>
```

10.配置 workers 文件

[root@master hadoop]# vi workers #加入以下内容

```
slave1
slave2
```

至此配置文件基本配置完毕。

12.新建目录。

[root@master hadoop]# mkdir /usr/hadoop/tmp

[root@master hadoop]# mkdir /usr/hadoop/dfs/name -p

[root@master hadoop]#mkdir /usr/hadoop/dfs/data -p

13.修改/usr/hadoop 目录的权限。

[root@master hadoop]# chown -R hadoop:hadoop /usr/hadoop/

14.将 master 节点上的 hadoop 安装文件同步到 slave1 slave2 slave3。

[root@master hadoop]#cd

[root@master ~]# scp -r /usr/hadoop/ root@slave1:/usr/

[root@master ~]# scp -r /usr/hadoop/ root@slave2:/usr/

[root@master ~]# scp -r /usr/hadoop/ root@slave3:/usr/

```
[root@slave1 hadoop]# ls
bin  etc      lib      LICENSE.txt  README.txt  share
dfs  include  libexec  NOTICE.txt   sbin        tmp
[root@slave1 hadoop]#
```

```
[root@slave2 hadoop]# ls
bin  etc      lib      LICENSE.txt  README.txt  share
dfs  include  libexec  NOTICE.txt   sbin        tmp
[root@slave2 hadoop]#
```

15.在每个 slave 节点上配置 hadoop 的环境变量。

[root@ slave1~]# vi /etc/profile #文件末尾添加

```
#set hadoop environment
export HADOOP_HOME=/usr/hadoop
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH
```

16.在每个 slave 节点上使配置的 hadoop 的环境变量生效。

[root@ slave1~]# source /etc/profile

17.在每个 slave 节点上修改/usr/hadoop 目录的权限。

[root@ slave1~]#chown -R hadoop:hadoop /usr/hadoop/

18.在每个 slave 节点上切换到 hadoop 用户。

[root@slave1~]#su – hadoop

● 1.4.3.5. 测试

1.切换到 hadoop（master 节点）。

[root@master ~]# su - hadoop

2.先格式化（master 节点）。

[hadoop@master ~]$ hadoop namenode -format

3.启动 hadoop（master 节点）

[hadoop@master ~]$ start-dfs.sh

[hadoop@master ~]$ start-yarn.sh

```
[hadoop@master ~]$ start-dfs.sh
Starting namenodes on [master]
master: Warning: Permanently added 'master,172.17.0.2' (ECDSA) to the list of known ho
ts.
Starting datanodes
slave1: WARNING: /usr/hadoop/logs does not exist. Creating.
slave2: WARNING: /usr/hadoop/logs does not exist. Creating.
Starting secondary namenodes [master]
[hadoop@master ~]$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
[hadoop@master ~]$
```

4.查看 Java 进程。

master 节点

[hadoop@master ~]$ jps

```
[hadoop@master ~]$ jps
5424 NodeManager
5604 Jps
4888 SecondaryNameNode
4667 DataNode
5131 ResourceManager
4543 NameNode
[hadoop@master ~]$
```

Slave1 节点

[hadoop@slave1~]$ jps

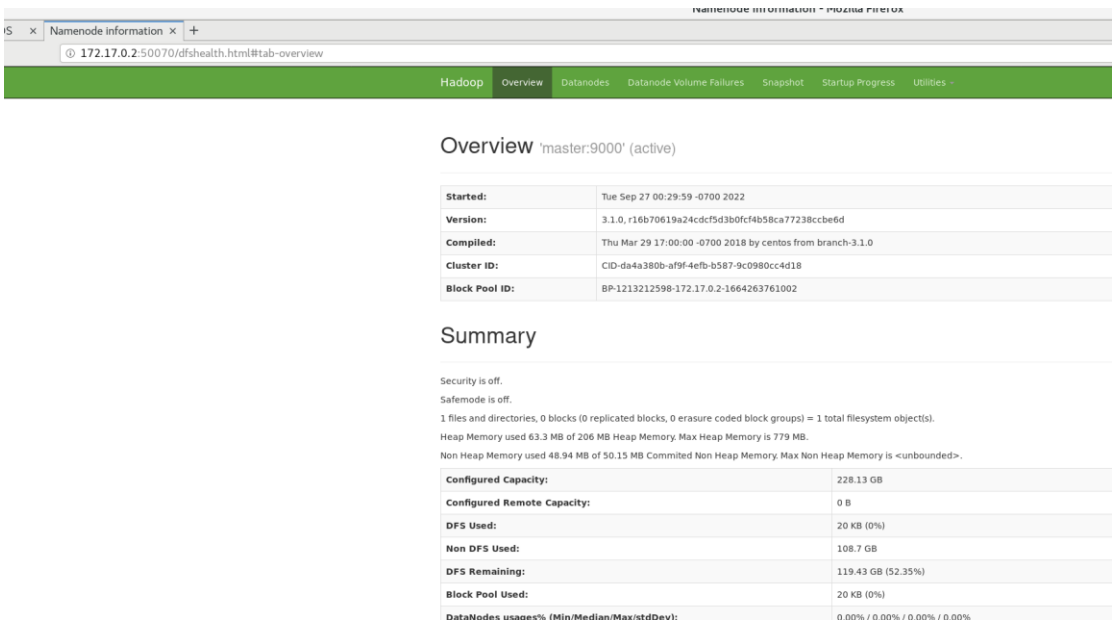
```
[hadoop@slave1 ~]$ jps
4673 NodeManager
4549 DataNode
4795 Jps
[hadoop@slave1 ~]$
```

Slave2 节点。

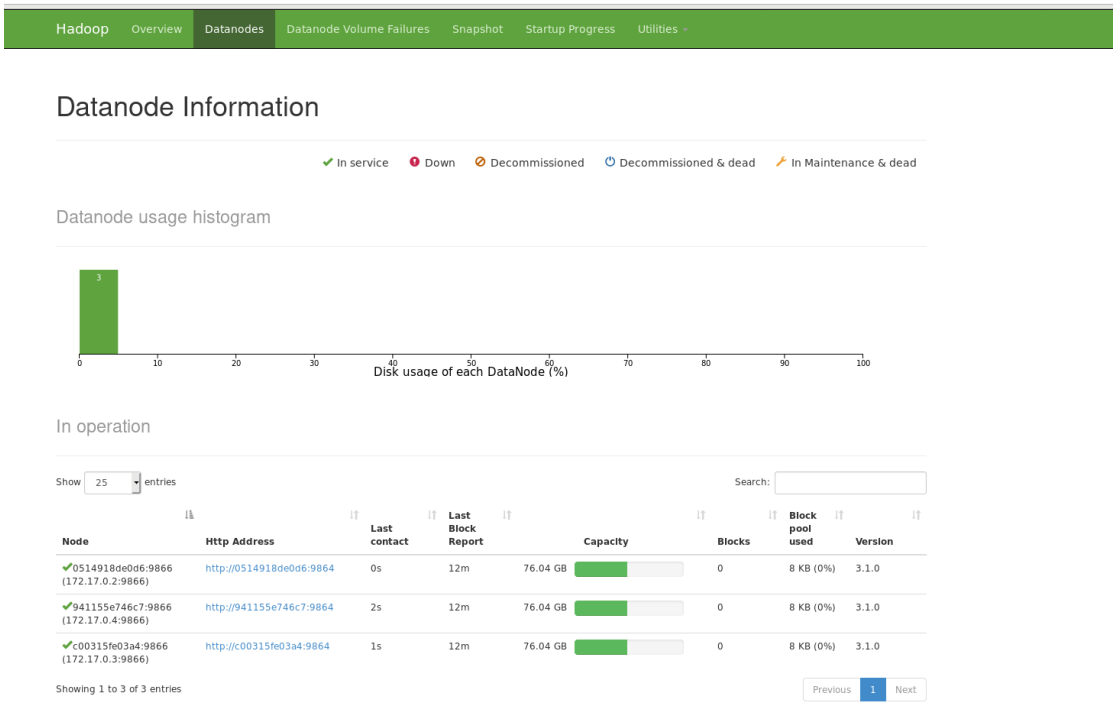
[hadoop@slave2~]$ jps

```
[hadoop@slave2 ~]$ jps
3889 NodeManager
3765 DataNode
4011 Jps
[hadoop@slave2 ~]$
```

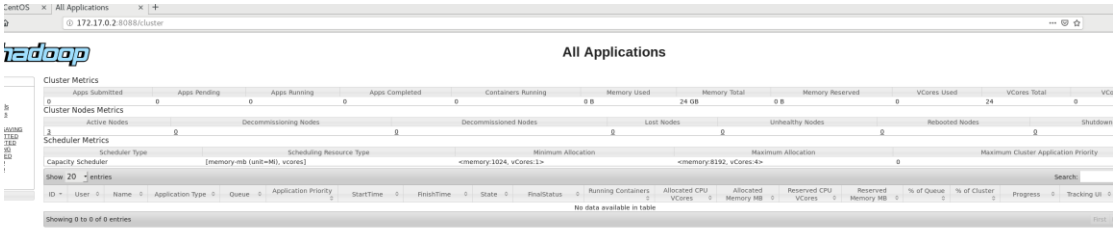
5.使用浏览器浏览 Master 节点机 http://172.17.0.2:50070,查看 NameNode 节点状态



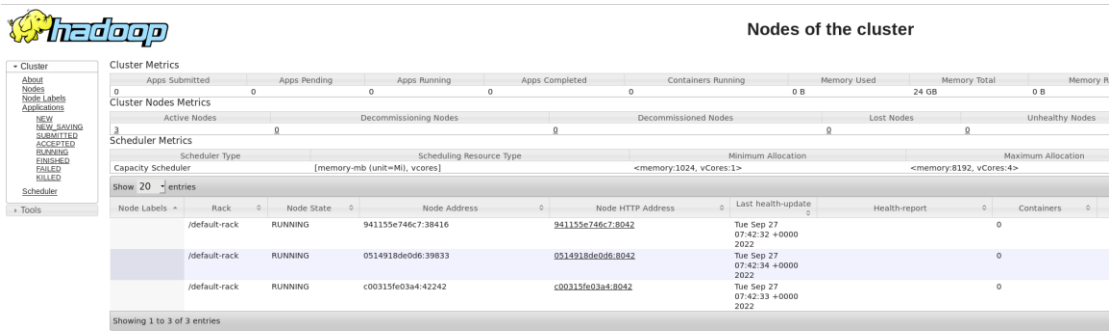
6.浏览 Datanode 数据节点



7.使用浏览器浏览 Master 节点 http://172.17.0.2:8088 查看所有应用



8.浏览 Nodes



9.关闭 hadoop

```
^C[hadoop@master ~]$ stop-all.sh
WARNING: Stopping all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: Use CTRL-C to abort.

Stopping namenodes on [master]
Stopping datanodes
Stopping secondary namenodes [master]
Stopping nodemanagers
Stopping resourcemanager
[hadoop@master ~]$
[hadoop@master ~]$
```

1.5.实验总结

1）初步理解了 Hadoop 工作原理。

2）通过实验学习了 Hadoop 编译过程。

3）通过实验学习了 Hadoop 伪分布式、完全分布式的安装过程。