

Hadoop 搭建

未使用 openstack

VMware + centos7(*3)

本地模式—伪分布式—完全分布式

有案例~ 

目录

Hadoop 搭建.....	1
一、基础环境.....	4
1. 虚拟机: centos7.....	4
2. 配置 IP.....	5
3. 关闭防火墙、创建用户、修改主机名、修改用户权限.....	6
I. 关闭防火墙.....	6
II. 创建用户（本教程使用 hadoop 用户操作）.....	6
III. 修改主机名.....	7
IV. 修改 hadoop 用户权限.....	7
4. 安装 JDK1.8.....	8
5. 部分删除命令及注意事项.....	9
I. 删除命令.....	9
II. 注意事项.....	9
6. 任务介绍: 搭建 Hadoop2.7.3.....	9
二、本地（独立）模式.....	10
1. 安装 hadoop.....	11
2. 配置环境变量.....	11
3. 测试及两个案例.....	12
I. 测试: 查看版本.....	12
II. 案例 1: grep.....	12
III. 案例 2: 统计单词个数.....	13
三、伪分布式.....	14
1. 伪分布式 hdfs.....	14
I. 修改 core-site.xml.....	14
II. 修改 hdfs-site.xml.....	15
2. 启动伪分布式 hdfs.....	15
I. 格式化 namenode.....	15
II. 启动 namenode 和 DataNode.....	16
3. 伪分布式-yarn.....	18
4. 启动伪分布式 yarn.....	19
5. 伪分布式案例: wordcount.....	20
I. 对比本地模式与伪分布式.....	20
II. Wordcount 案例.....	20
四、完全分布式.....	22
1. 准备三台机器.....	22
2. 三台机器配置表.....	24
3. 配置 SSH 免密码登录.....	24
I. 生成公钥私钥对（三台机器）.....	24
II. 将公钥加入到授权列表.....	24
4. 完全分布式搭建.....	25
I. Hdfs.....	25
II. yarn.....	25

III. 同步修改文件	26
V. 格式化	26
VII. 分别启动	26
VI.同时启动	28
5. Wordcount 案例.....	29

一、基础环境

1. 虚拟机：centos7

① 本次搭建使用的是 VMWare（VBox 也可，没测试过）

镜像使用的是 centos 7 （这个大家应该都有吧~ 🤖）

- 随便找个靠谱的教程：[CentOS 7 安装教程（图文详解）招牌 DBA-](#)

[CSDN 博客 centos7 安装教程](#)




图 1. centos 大家都搭过了，就不赘述啦，当然还是建议找个教程来~

- （ I 建议网络使用 NAT 模式，当然不换局域网的话桥接也可~）
II 建议命名：master、slave1、slave2 **or** hadoop1、hadoop2、hadoop3
等具有比较明显意义的 name
III 一个镜像可以创建多个虚拟机哦~
IV 如果选择**克隆**虚拟机，建议在**搭建完本地模式后克隆**）
● 若打开虚拟机时出现：license not accepted，可参考[centos7 安装出现 license information\(license not accepted\)解决办法 首席撩妹指导 官的博客-CSDN 博客](#)

完全分布式需要三台虚拟机 （物理机建议 12g 或者 16g，有 SSD… 没有也

没办法呀 ps:创建完第一个后，可以**克隆** → **真香** 🤖 当然新手可以配三台练练手~）

创建好虚拟机后，需要依次：配置 ip、创建 hadoop 用户（可选）、修改主机名（可选）、安装 JDK1.8、安装 hadoop 等等…… **注：以下搭建过程会以**

hadoop 用户的身份操作（主机名为 master）! 

2. 配置 IP

执行以下命令（**eno 后，按一下 tab，自动补充**）进 vim 后，前十几行是默认有的，打开没有说明进错文件了！（可通过图 1 找虚拟机的子网掩码和网关，是“**编辑**”选项卡不是“查看”选项卡） vim 大家都会用的吧~

`vim /etc/sysconfig/network-scripts/ifcfg-eno16777728`

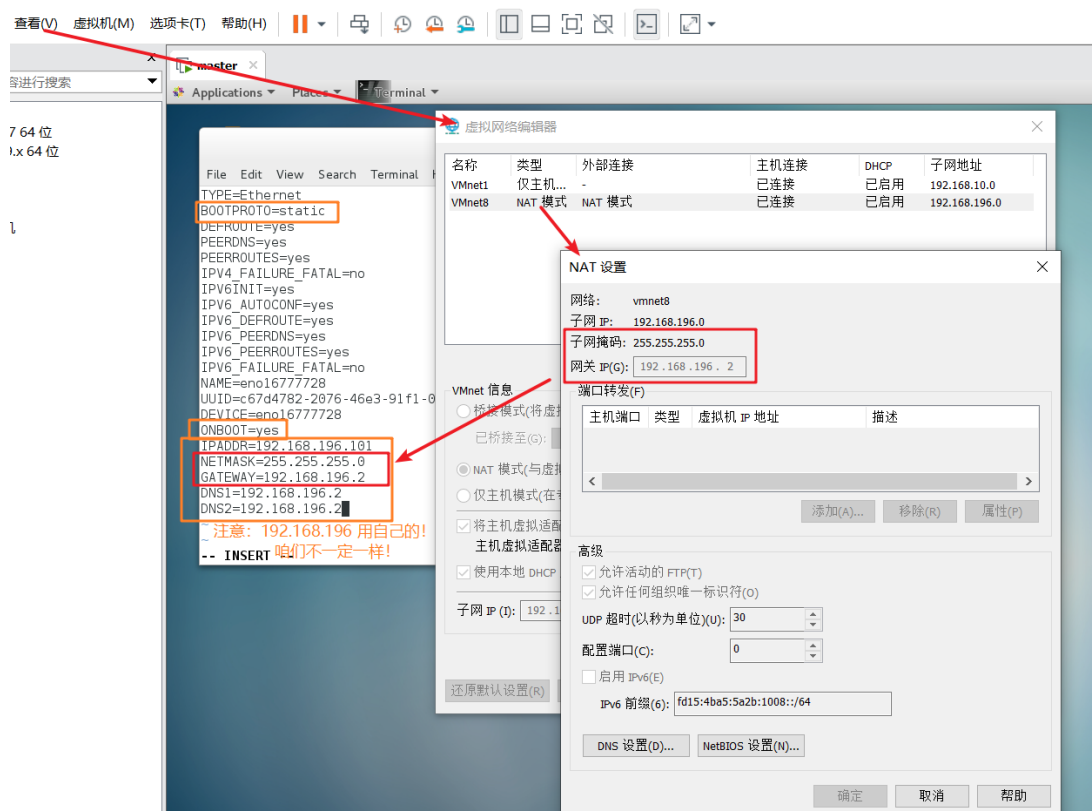




图 2.配置 IP，不是“查看”是“编辑”！

三个虚拟机都修改完 ip 配置后，重启 ping IP 地址，成功！**ctrl+c 停止**

（如果同时安装三个虚拟机，这一步三个虚拟机的 IP 不能一样哈~ xdm 都在

学计网  都知道的哈 （这里 ip 最后部分写固定值，我写的 101、102、103）

3. 关闭防火墙、创建用户、修改主机名、修改用户权限

目前都可选，连不上网，用不了 xshell 可以过来关防火墙 。

I. 关闭防火墙

- ① 关闭 centos7 的防火墙（慎重！~~个人感觉问题不大~~）

```
systemctl status firewalld.service
systemctl stop firewalld.service
systemctl disable firewalld.service
systemctl status firewalld.service
```

```
root@slave2 ~# systemctl status firewalld.service
■ firewalld.service - firewalld - dynamic firewall daemon
  Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
  Active: active (running) since Sun 2021-12-05 03:06:19 CST; 10min ago
    Main PID: 756 (firewalld)
      CGroup: /system.slice/firewalld.service
              └─756 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid

Dec 05 03:06:11 slave2 systemd[1]: Starting firewalld - dynamic firewall daemon...
Dec 05 03:06:19 slave2 systemd[1]: Started firewalld - dynamic firewall daemon.
root@slave2 ~# systemctl stop firewalld.service
root@slave2 ~# systemctl disable firewalld.service
Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
Removed symlink /etc/systemd/system/basic.target.wants/firewalld.service.
root@slave2 ~# systemctl status firewalld.service
■ firewalld.service - firewalld - dynamic firewall daemon
  Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
  Active: inactive (dead)

Dec 05 03:06:11 slave2 systemd[1]: Starting firewalld - dynamic firewall daemon...
Dec 05 03:06:19 slave2 systemd[1]: Started firewalld - dynamic firewall daemon.
Dec 05 03:17:09 slave2 systemd[1]: Stopping firewalld - dynamic firewall daemon...
Dec 05 03:17:10 slave2 systemd[1]: Stopped firewalld - dynamic firewall daemon.
root@slave2 ~#
```

图 3.关闭 centos7 的防火墙

- ② 关闭 windows 的防火墙：

控制面板\系统和安全\Windows Defender 防火墙\自定义设置

II. 创建用户（本教程使用 hadoop 用户操作）

- ① 搭建 hadoop 环境的时候，创建一个 hadoop 用户（可选，root 用户权

限过高，新手（大概率）易出错，比如配置环境变量~）

useradd 用户名

passwd 用户名

设置密码

su 用户名 切换用户

```

[root@slave1 ~]# useradd hadoop
[root@slave1 ~]# passwd hadoop
Changing password for user hadoop.
New password:
BAD PASSWORD: The password contains the user name in some form
Retype new password:
passwd: all authentication tokens updated successfully.
[root@slave1 ~]# su hadoop
[hadoop@slave1 root]$ _

```

图 4.添加用户

III. 修改主机名

- ① 修改主机名（切换到 root 用户，使用 vim 进入 hostname 文件直接修改，保存退出即可，用使用 hostname 查看当前主机名，如主机名没变，重启一下即可~ 以下是 xshell 视图）

```

[hadoop@master ~]$ su root
Password:
[root@master hadoop]# vim /etc/hostname
[root@master hadoop]# hostname
master
如果主机名没变，重启一下即可
[root@master hadoop]#

```

图 5.修改主机名

- ② 修改主机名后， 在 /etc/hosts 里加上 IP 主机名 （root 用户—vim /etc/hosts 或 修改权限后的 hadoop 用户—sudo vim /etc/hosts）

```

127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.196.101 master

```

图 6. 在/etc/hosts 里添加 “IP 主机名”

或者直接将 master 加在 第一行最后（127.0.0.1 表示本地 192.168.196.101 也是本地）

IV. 修改 hadoop 用户权限

有一些文件普通用户只读，想修改只能切换到 root 用户，比较麻烦。修改一下普通用户 hadoop 的权限可以方便地修改一些重要配置文件。

```
[hadoop@master ~]$ vim /etc/sudoers 普通用户看不到该文件
[hadoop@master ~]$ su root
Password: 切换到root用户后, 修改此文件
[root@master hadoop]# vim /etc/sudoers
[root@master hadoop]# su hadoop
[hadoop@master ~]$ sudo vim /etc/sudoers 此时可以通过sudo命令, 编辑此文件 (需要输入密码)

## Allow root to run any commands anywhere
root    ALL=(ALL)    ALL
hadoop  ALL=(ALL)    ALL  添加hadoop用户权限为ALL, 然后用:wq! 强制保存退出
## Allows member to run any commands
member  ALL=(ALL)    ALL

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

#1) Respect the privacy of others.
#2) Think before you type.
#3) With great power comes great responsibility.

[sudo] password for hadoop: 输入正确密码后, 可进入编辑该文件
[hadoop@master ~]$
```

图 7.修改 hadoop 权限

4. 安装 JDK1.8

Hadoop 用 java 开发的, 需要安装 JDK [Java Downloads | Oracle](#) 不想登录的童鞋, 可以在这里[免费下载](#): [jdk1.8 linux 版.rar-Java 文档类资源-CSDN 文库](#)

① 上传 JDK 到虚拟机:

方法一: 使用 xshell (下一个, 一劳永逸哈哈哈)

```
[hadoop@master ~]$ pwd
/home/hadoop
[hadoop@master ~]$ mkdir software  创建一个专门放软件的目录
[hadoop@master ~]$ ls
software
[hadoop@master ~]$ ls
jdk-8u201-linux-x64.tar.gz software
[hadoop@master ~]$ ls
software
[hadoop@master ~]$ cd software/  进入该目录
[hadoop@master software]$ ls
jdk-8u201-linux-x64.tar.gz  通过xshell, 将jdk1.8放至该目录
```

图 8. xshell 使用如图, 先创建一个 software 目录专门放软件~

方法二: 使用 vmtools 的共享文件夹

[VMware 设置 centos7 共享文件夹_nesxiaogu 的博客-CSDN 博客_centos 共享文件夹](#)

方法三: 其它工具, 例如 Notepad++ (这软件居然也可以上传, 绝!)

<https://jingyan.baidu.com/article/d8072ac4791d0cec94cefd7c.html>

② 解压 JDK 的压缩文件 (*.tar.gz)

```
tar -zxvf jdk-8u201-linux-x64.tar.gz
rm -rf jdk-8u201-linux-x64.tar.gz  rm -rf 慎用
```


`ln -s jdk1.8.0_201/ jdk` 创建软连接

③ 配置环境变量

I 在 root 用户下配置环境变量 `/etc/profile`

II 配置当前用户的环境变量 `~/.bash_profile` （以普通用户为例，如下）

`vim ~/.bash_profile` 进入配置文件添加环境变量

`export JAVA_HOME=/home/hadoop/software/jdk`

`export PATH=$PATH:$JAVA_HOME/bin`

`source ~/.bash_profile` 需要执行该文件

注：在普通用户配置的环境变量，root 用户可用，其它普通用户无法使用

④ 验证

`java -version` 查看 java 版本，能查到说明配置成功

5. 部分删除命令及注意事项


I. 删除命令

误创建文件，删除：`rm 文件名`；

误创建文件夹，删除 `rm -rf 文件夹`

误创建软连接，删除 `rm 链接名` *不建议使用 `rm -rf*`

II. 注意事项

- a) 一定要及时备份！（vm 拍摄快照 ）
- b) 在完成本地模式后，克隆机器出两个机器，选择完整克隆！路径一定是空目录不要跟被克隆机器放在一个地方!!! 会出问题!!!（克隆机需要修改……）建议小白自己创建两个哟，这样就是自己搭建了三遍，配置了六次环境变量 hhh
- c) 未完待续……

6. 任务介绍：搭建 Hadoop2.7.3

本次搭建需要完成以下任务（参考文档：[Apache Hadoop 2.7.3 – Hadoop:](#)

[设置单节点集群。](#) 注，最新版是 3.3.1，本次搭建 2.7.3）

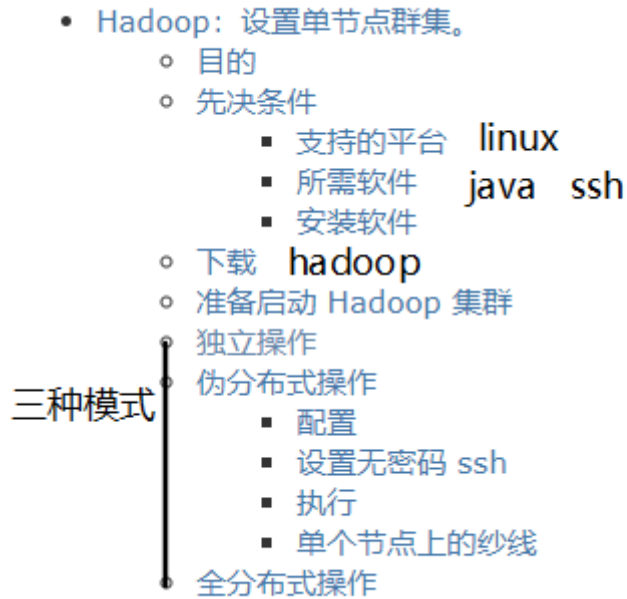


图 9. 官网教程截图~

可以直接跟着官方教程来，也可以继续走下去（建议同步看 [centOS7 搭建 hadoop 环境（非常详细！非常适合新手！）](#) [yajuan110 的博客-CSDN 博客](#) [_centos7 搭建 hadoop](#) 里面详细介绍了三台虚拟机如何配置）

我的三台虚拟机的主机名及 IP:

主机名: master:
··· 用户: hadoop ··· hadoop
··· IPADDR: ··· 192.168.196.101

主机名: slave1:
··· 用户: hadoop ··· hadoop
··· IPADDR: ··· 192.168.196.102

主机名: slave2:
··· 用户: hadoop ··· hadoop
··· IPADDR: ··· 192.168.196.103

二、本地（独立）模式

只需要一台机器。

安装 hadoop 流程同 java: 上传、解压、配置环境变量、验证+（运行官方案例）

1. 安装 hadoop

① [Apache Hadoop](#) 选择绿色按钮

Release 2.7.3 available

Please see the [Hadoop 2.7.3 Release Notes](#) for the list of 221 bug fixes and patches since the previous release 2.7.2.

2016 Aug 26

linux用的安装包

Download tar.gz

(checksum signature)

一般是源码

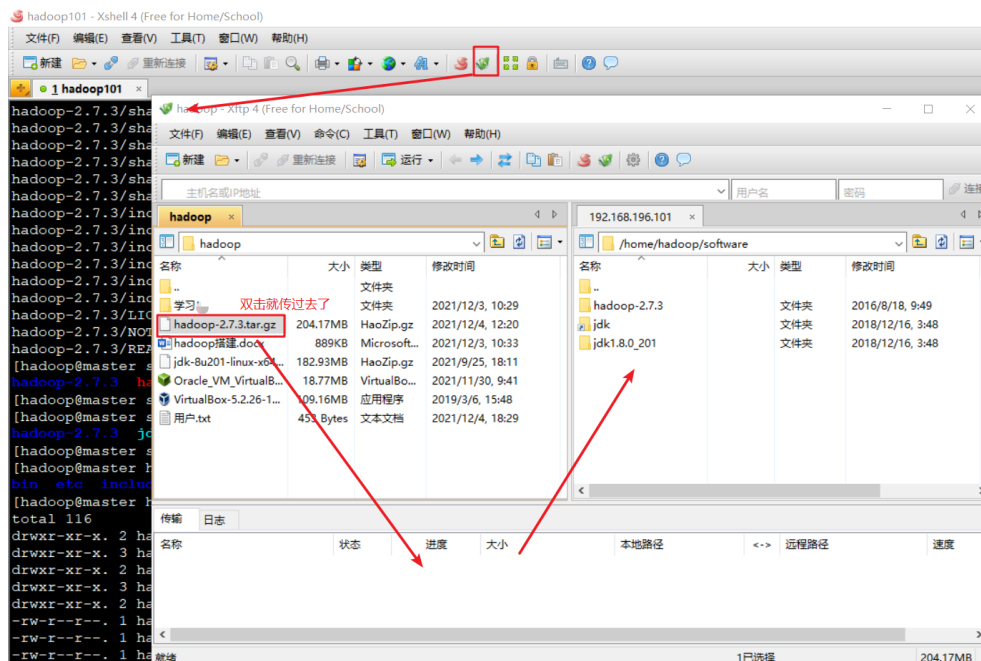
Download src

(checksum signature)

帮助文档

Documentation

将压缩包上传到虚拟机，我用的 xshell，灰常方便！



② 解压: tar -zxvf Hadoop-2.7.3.tar.gz

2. 配置环境变量

① （可选）创建软连接 ln -s hadoop-2.7.3/ Hadoop



② 配置环境变量（应该都会了吧？）：

```

#java的环境变量
export JAVA_HOME=/home/hadoop/software/jdk
export PATH=$PATH:$JAVA_HOME/bin

#hadoop的环境变量
export HADOOP_HOME=/home/hadoop/software/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin

:wq

```

此处使用了软连接

此处使用了软连接

bin和sbin两个都要配!

图 10. 配置内容如图，自己来一波？~

```

vim ~/.bash_profile    进入普通用户 Hadoop 的配置文件，添加环境变量
#hadoop 的环境变量    #是注释
export HADOOP_HOME=/home/hadoop/software/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
source ~/.bash_profile  执行配置文件

```

3. 测试及两个案例

I. 测试：查看版本

查看 hadoop 版本 `hadoop version`

能看到版本信息说明配置成功，出现 `bash: hadoop: command not found...` 说明配置失败，请检查之前步骤

II. 案例 1: grep

默认情况下，Hadoop 配置为在非分布式模式下作为单个 Java 进程运行。这对于调试非常有用。

下面的示例复制解压缩的 conf 目录以用作输入，然后查找并显示给定正则表达式的每个匹配项。输出将写入给定的输出目录。

```

$ mkdir input
$ cp etc/hadoop/*.xml input
$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar grep input output 'dfs[a-z.]+'
$ cat output/*

```

图 11. 官网案例：grep

看不懂？没关系一步一步来~

`mkdir input` 先进入 hadoop 创建目录 `input`

`cp etc/hadoop/*.xml input` 将 etc/hadoop 下的 xml 文件拷贝到 input

hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar grep input output 'dfs[a-z.]+' 运行 hadoop 的（在 share 下的 hadoop 下的 mr 下的 example）jar 包，输入是 grep 过滤后的 input 中的 dfsxxx(正则表达式)，jar 包运行的结果放入 output（如果报错未知的名称或服务，是因为/etc/hosts 下没有添加 “IP 主机名”

cat output/* 查看 hadoop 运行结果

```
[hadoop@master hadoop]$ cat output/*
1      dfsadmin
[hadoop@master hadoop]$ cd output
[hadoop@master output]$ ll
total 4
-rw-r--r--. 1 hadoop hadoop 11 Dec  5 12:38 part-r-00000
-rw-r--r--. 1 hadoop hadoop  0 Dec  5 12:38 _SUCCESS
```

测试成功！可删掉 input 和 output rm -rf input/ output/

III. 案例 2: 统计单词个数

统计 input 目录下，所有文件中某个单词的个数

先在 hadoop 下创建一个目录 wc_in；然后在 wc_in 里创建一个 a.txt，单词内容随意（此处用 hadoop hive hbase spark hadoop hbase Hadoop）；复制 a.txt 为 b.txt，使用 hadoop 的 jar 包统计单词个数

```
[hadoop@master hadoop]$ mkdir wc_in
[hadoop@master hadoop]$ cd wc_in
[hadoop@master wc_in]$ vim a.txt
[hadoop@master wc_in]$ cp a.txt b.txt
[hadoop@master wc_in]$ ll
total 8
-rw-rw-r--. 1 hadoop hadoop 44 Dec  5 17:06 a.txt
-rw-rw-r--. 1 hadoop hadoop 44 Dec  5 17:06 b.txt
[hadoop@master wc_in]$ cd ../
[hadoop@master hadoop]$ hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar wordcount wc_in/ wc_out
```

（输入参数 wordcount 输入路径 wc_in 输出路径 wc_out）

```
[hadoop@master hadoop]$ cat wc_out/* 查看输出内容
hadoop 6
hbase 4
hive 2
spark2
```

测试成功！

三、伪分布式

使用一台机器 master 模拟分布式（一下子装三台的 xdm，这里只操作一台哦~）
需要修改 hadoop-2.7.3/etc/hadoop 下的配置文件（/etc 是用来存放主要的配置文件）

环境准备及需要的配置：

- ① java 环境
- ② Hadoop 环境
- ③ 配置主机名与 ip 地址的映射（修改 core-site.xml 会用到）
- ④ 修改配置文件

core-site.xml （指定 namenode 在哪台机器，设置临时目录的路径）

hdfs-site.xml （修改 block 块的副本数）

1. 伪分布式 hdfs

I. 修改 core-site.xml

配置文件路径：/home/hadoop/software/hadoop-2.7.3/etc/Hadoop

使用 vim 进入 core-site.xml 文件：添加 hdfs 的 namenode 节点地址配置（必选）和 临时文件的路径配置（推荐）

① Hdfs 的 namenode 节点地址

在<configuration></configuration>之间添加

```
<property>
  <!--指定 hdfs 的 namenode 节点的地址-->
  <name>fs.defaultFS</name>
  <value>hdfs://master:9000</value>
</property>
```

这里的 **master**，是主机名（使用前提：步骤二中修改主机名并且在 /etc/hosts 里加上“IP 主机名”）

② 修改临时文件地址

在<configuration></configuration>之间添加

```
<property>
  <!--hadoop 运行产生的文件目录-->
  <name>hadoop.tmp.dir</name>
```

```
<value>/home/hadoop/software/hadoop/data/tmp</value>
</property>
```

配置如图:

```
<configuration>
  <property>
    <!--指定hdfs的namenode节点的地址-->
    <name>fs.defaultFS</name>
    <value>hdfs://master:9000</value>
  </property>
  <property>
    <!--hadoop运行产生的文件目录-->
    <name>hadoop.tmp.dir</name>
    <value>/home/hadoop/software/hadoop/data/tmp</value>
  </property>
</configuration>
```

图 12. core-site.xml 配置修改如图

II. 修改 hdfs-site.xml

```
<configuration>
  <property>
    <!--伪分布式，需要指定副本数为1-->
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

修改配置文件如图:

```
<configuration>
  <property>
    <!--伪分布式，需要指定副本数为1-->
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
~
:wq
```

图 13. hdfs-site.xml 配置修改如图

2. 启动伪分布式 hdfs

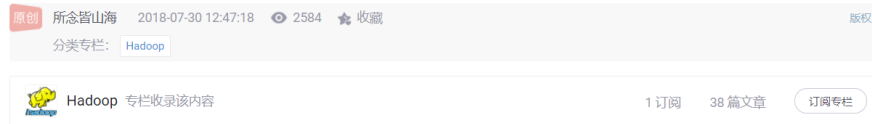
I. 格式化 namenode

① 注意：先格式化，仅在第一次使用的时候才格式化！

```
[hadoop@master ~]$ hdfs namenode -format
```

 按照 hdfs 格式化

hadoop中的namenode进行格式化是什么意思？



只有namenode需要format,
secondarynamenode和datanode不需要format。
类似于硬盘分区后以及新买的u盘需要格式化的问题。
因为硬盘,u盘以及hdfs一样都是一个文件系统,
所以使用前要格式化。

图 14.什么是格式化 namenode

②据说格式化成功会出现 successful format（可参阅博客）：

[安装伪分布的 Hadoop 时 SHUTDOWN_MSG: Shutting down NameNode at xxx 并不一定是 namenode 格式化失败 weixin_42069087 的博客-CSDN 博客](#)

如果格式化失败(类似下图)，请仔细看报错提示，仔细检查配置文件！

```
at org.apache.hadoop.conf.Configuration.set(Configuration.java:1115)
at org.apache.hadoop.conf.Configuration.setBoolean(Configuration.java:1451)
at org.apache.hadoop.util.GenericOptionsParser.processGeneralOptions(GenericOptionsParser.java:170)
at org.apache.hadoop.util.GenericOptionsParser.parseGeneralOptions(GenericOptionsParser.java:48)
at org.apache.hadoop.util.GenericOptionsParser.<init>(GenericOptionsParser.java:170)
at org.apache.hadoop.util.GenericOptionsParser.<init>(GenericOptionsParser.java:153)
at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(NameNode.java:1422)
at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1559)
21/12/05 21:18:55 ERROR namenode.NameNode: Failed to start namenode.
java.lang.RuntimeException: org.xml.sax.SAXParseException; systemId: file:/home/hadoop/software/hadoop-2.7.3/etc/hadoop/core-site.xml; lineNumber: 1; column: 1; The root element must be well-formed.
at org.apache.hadoop.conf.Configuration.loadResource(Configuration.java:2645)
at org.apache.hadoop.conf.Configuration.loadResources(Configuration.java:2492)
at org.apache.hadoop.conf.Configuration.getProps(Configuration.java:2405)
at org.apache.hadoop.conf.Configuration.set(Configuration.java:1143)
at org.apache.hadoop.conf.Configuration.set(Configuration.java:1115)
at org.apache.hadoop.conf.Configuration.setBoolean(Configuration.java:1451)
at org.apache.hadoop.util.GenericOptionsParser.processGeneralOptions(GenericOptionsParser.java:170)
at org.apache.hadoop.util.GenericOptionsParser.parseGeneralOptions(GenericOptionsParser.java:48)
at org.apache.hadoop.util.GenericOptionsParser.<init>(GenericOptionsParser.java:170)
at org.apache.hadoop.util.GenericOptionsParser.<init>(GenericOptionsParser.java:153)
at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(NameNode.java:1422)
at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1559)
Caused by: org.xml.sax.SAXParseException; systemId: file:/home/hadoop/software/hadoop-2.7.3/etc/hadoop/core-site.xml; lineNumber: 1; column: 1; The root element must be well-formed.
at org.apache.xerces.parsers.DOMParser.parse(Unknown Source)
at org.apache.xerces.jaxp.DocumentBuilderImpl.parse(Unknown Source)
at javax.xml.parsers.DocumentBuilder.parse(DocumentBuilder.java:150)
at org.apache.hadoop.conf.Configuration.parse(Configuration.java:2480)
at org.apache.hadoop.conf.Configuration.parse(Configuration.java:2468)
at org.apache.hadoop.conf.Configuration.loadResource(Configuration.java:2539)
```

图 15.出现这个“形状”的，大概率是出错了。。。

③格式化完成后，会发现 hadoop 下出现了 data，因为在 core-site.xml 配置过。

II. 启动 namenode 和 DataNode

①启动 NameNode daemon

hadoop-daemon.sh start namenode

```
[hadoop@master ~]$ hadoop-daemon.sh start namenode
starting namenode, logging to /home/hadoop/software/hadoop-2.7.3/logs/hadoop-hadoop-namenode-master.out
[hadoop@master ~]$ jps
22289 NameNode
22377 Jps
```

图 16.启动 namenode，jps 可以查看当前运行的 java 进程

此时可以访问 50070 端口的页面(在 window 中浏览器输入 虚拟机 IP:端口号)

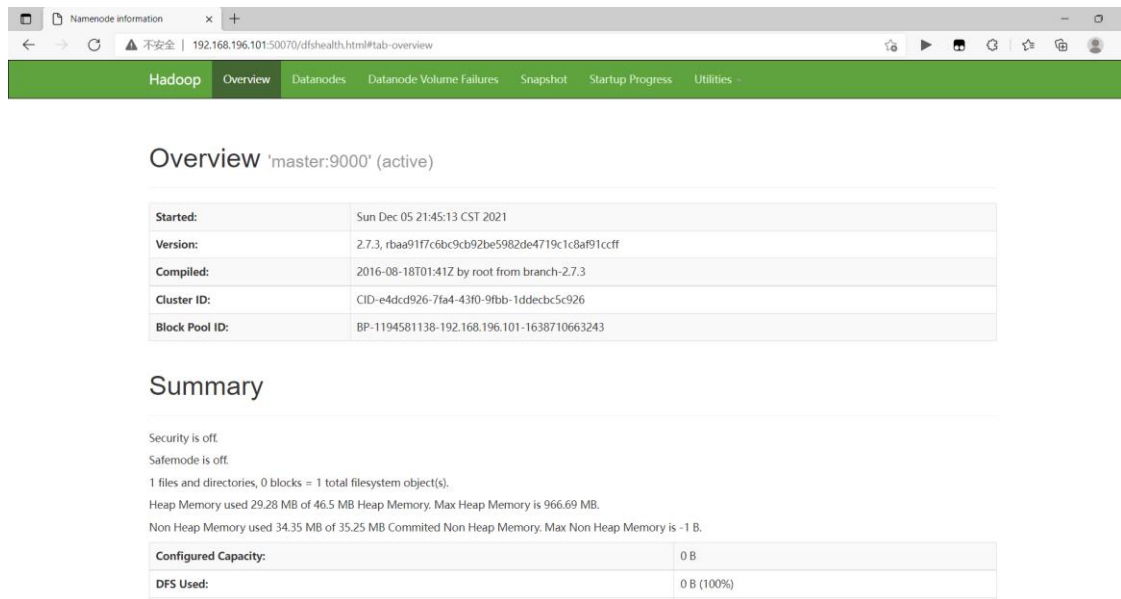


图 17.hdfs 的前端页面

②启动 DataNode daemon

`hadoop-daemon.sh start datanode`

```
[hadoop@master ~]$ jps
22289 NameNode
22747 DataNode
22846 Jps
```

③Linux 上传文件到 hadoop 系统

`hdfs dfs -put test1.txt /` 上传本地的 test.txt（随便写一个~）到 hadoop 的根目录（存储是存在 datanode 哦~, 而不是 namenode）

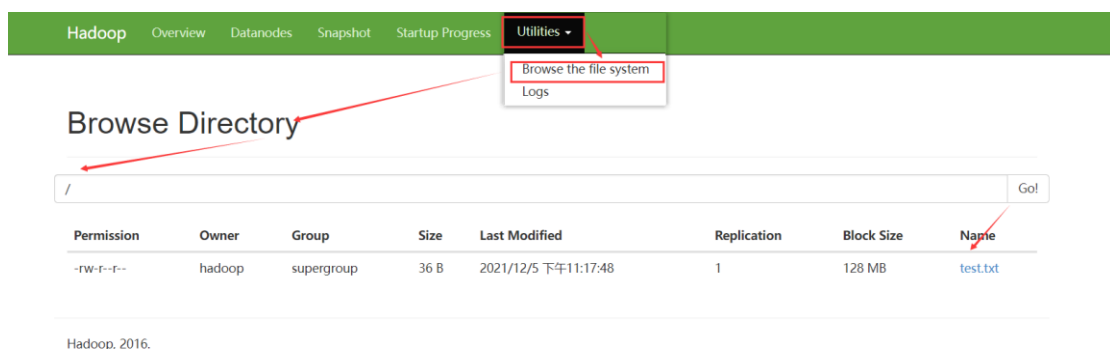


图 18.查看 hdfs 里的文件

④如果格式化多次, 造成 datanode 无法启动, 删除 temp 目录 `rm -rf temp/`, 重新格式化即可。

(需要关闭 NameNode `hadoop-daemon.sh stop namenode`
Datanode `hadoop-daemon.sh stop datanode`)

3. 伪分布式-yarn

Yarn:资源与任务调度

也有两个进程：resourcemanager（调度作用） nodemanager（跑任务）

待完成任务：

1> yarn-site.xml 文件配置

Shuffle 流程 resourcemanager 的运行机器

2> Hadoop-env.sh mapred-env.sh yarn-env.sh

3> 改名 mapred-site

4> 修改文件：mapred-site.xml 指定运行 mr 程序的时候使用 yarn 做调度

① 修改 yarn-site.xml

```
<configuration>
<!-- Site specific YARN configuration properties -->
  <property>
    <!--设置 shuffle 流程-->
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <!--设置 resourcemanager 的是哪个节点-->
    <name>yarn.resourcemanager.hostname</name>
    <value>master</value>
  </property>
</configuration>

<configuration>
<!-- Site specific YARN configuration properties -->
  <property>
    <!--设置 shuffle流程 -->
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <!--设置 resourcemanager的是哪个节点 -->
    <name>yarn.resourcemanager.hostname</name>
    <value>master</value>
  </property>
</configuration>
:wq
```

图 19.修改如图

② 配置环境文件 (3 处): JAVA_HOME export JAVA_HOME=/home/hadoop/software/jdk

```
[hadoop@master hadoop]$ vim hadoop-env.sh
#export JAVA_HOME=${JAVA_HOME}
export JAVA_HOME=/home/hadoop/software/jdk
[hadoop@ master hadoop]$ vim yarn-env.sh
```

```
# export JAVA_HOME=/home/y/libexec/jdk1.6.0/
export JAVA_HOME=/home/hadoop/software/jdk
```

```
[hadoop@ master hadoop]$ vim mapred-env.sh
```

```
# export JAVA_HOME=/home/y/libexec/jdk1.6.0/
export JAVA_HOME=/home/hadoop/software/jdk
```

③ 将 mapred-site.xml.template 修改为 mapred-site.xml

```
[hadoop@master hadoop]$ mv mapred-site.xml.template mapred-site.xml
```

④ 修改 mapreduce 核心文件 vim mapred-site.xml

```
<configuration>
  <!--指定 mr 运行在 yarn-->
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

如图

```
<configuration>
  <!--指定mr运行在 yarn-->
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
:wq
```

图 20.修改如图

4. 启动伪分布式 yarn

注意：启动 yarn 的时候必须保证 namenode 和 datanode 已经启动

①启动 resourcemanager 会有一个 web 页面，端口号为 8088

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

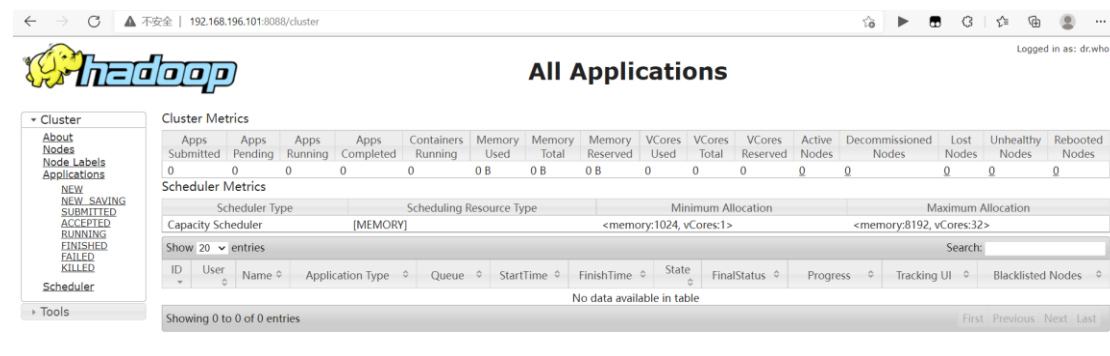


图 21.在 windows 浏览器地址栏输入 IP:8088

② 启动 nodemanager

```
[hadoop@master ~]$ yarn-daemon.sh start nodemanager  
  
[hadoop@master hadoop]$ jps  
24513 DataNode  
26499 NodeManager  
26213 ResourceManager  
26534 Jps  
24410 NameNode
```

图 22. jps 显示当前所有 java 进程

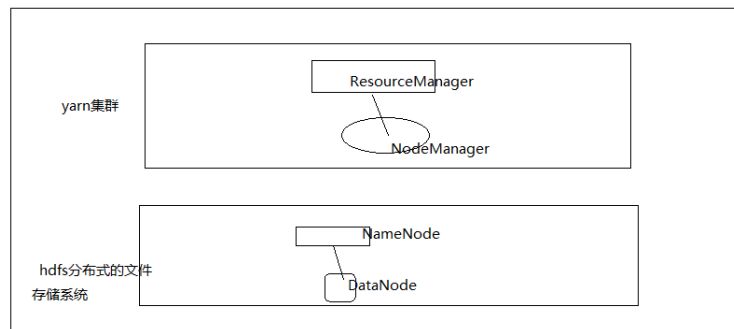


图 23. master 伪分布式集群

5. 伪分布式案例：wordcount

I. 对比本地模式与伪分布式

本地模式:

运行 mr 程序 输出和输入都是用的 linux 系统的文件

伪分布式:

运行 mr 程序 输出和输入都是用的 hdfs 上的文件

So hdfs 上应该有文件，需要将本地文件上传到 hdfs

II. Wordcount 案例

- ① 在 hdfs 根目录创建一个目录 wc_in `hdfs dfs -mkdir /wc_in`
- ② 创建一个 a.txt （内容随意），复制为 b.txt，将这两个文件上传到/wc_in
`vim a.txt`
`cp a.txt b.txt`
`hdfs dfs -put a.txt b.txt /wc_in`

Browse Directory

/wc_in

Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	hadoop	supergroup	36 B	2021/12/6 下午4:03:11	1	128 MB	a.txt
-rw-r--r--	hadoop	supergroup	36 B	2021/12/6 下午4:03:11	1	128 MB	b.txt

图 24.查看上传到 hdfs 的 wc_in 目录下的文件

(上述两个文件其实在如下目录 (注: BP 文件名一般不一样):
 /home/hadoop/software/hadoop/data/tmp/dfs/data/current/BP-1194581138-192.168.196.101-1638710663243/current/finalized/subdir0/subdir0)

③ 运行官方案例 wordcount

现在写的路径, 是 hdfs 的路径!

```
[hadoop@master hadoop]$ hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar wordcount /wc_in /wc_out
```

The screenshot shows the Hadoop web interface. On the left, a sidebar contains links like 'Cluster', 'About Nodes', 'Node Labels', and 'Applications' (which is highlighted). The main area is titled 'All Applications' and shows 'Cluster Metrics' with various resource usage statistics. Below this, 'Scheduler Metrics' are displayed, including 'Capacity Scheduler' and 'Scheduling Resource Type'. A table lists applications, with one entry for 'word count' in the 'MAPREDUCE' queue, showing a progress bar at 50.0%.

图 25.yarn 页面, 可以看当前进行的任务

注: 可能有些慢, 我们是模拟环境, 用的虚拟机, 且 hadoop 针对大数据更有优势

④ 查看统计结果 [hadoop@master hadoop]\$ hdfs dfs -cat /wc_out/*

四、完全分布式

1. 准备三台机器

默认泥萌选择克隆了哈，一下子装三台的 xdm 可以简单看一下

① 修改 ip 地址

执行以下命令（eno 后，按一下 tab，自动补充）

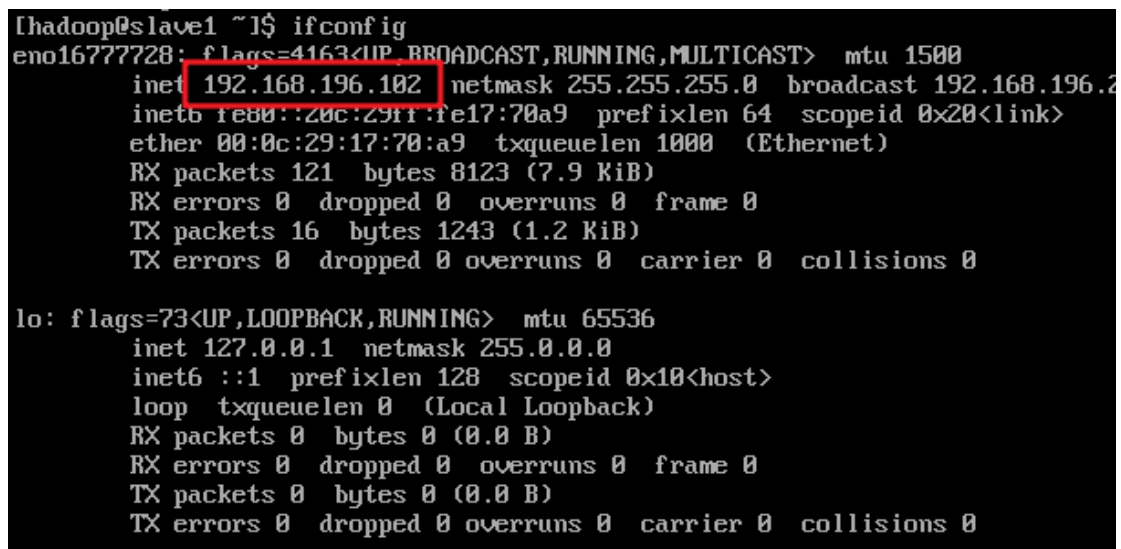
```
vim /etc/sysconfig/network-scripts/ifcfg-eno16777728
```

只需改 IPADDR，三台主机不能一样！

```
service network restart
```

 重启网卡，修改生效！

```
ifconfig
```

 显示目前网络设置

```
hadoop@slave1 ~1$ ifconfig
eno16777728: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.196.102 netmask 255.255.255.0 broadcast 192.168.196.255
    inet6 fe80::20c:29ff:fe17:70a9 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:17:70:a9 txqueuelen 1000 (Ethernet)
    RX packets 121 bytes 8123 (7.9 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 16 bytes 1243 (1.2 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 0 (Local Loopback)
    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
```

图 26.查看当前 IP

② 关闭防火墙，关闭虚拟机+windows

关闭虚拟机（权限不足命令前加 sudo →root 用户）：

```
systemctl status firewalld.service
```

如果是 dead 就不用执行以下命令了

```
systemctl stop firewalld.service
```

```
systemctl disable firewalld.service
```

```
systemctl status firewalld.service
```

关闭 windows：控制面板\系统和安全\Windows Defender 防火墙\自定义设置

③ 修改 hadoop 用户权限

有一些文件普通用户只读，想修改只能切换到 root 用户，比较麻烦。修改一下普通用户 hadoop 的权限可以方便地修改一些重要配置文件。

```
[hadoop@master ~]$ vim /etc/sudoers 普通用户看不到该文件
[hadoop@master ~]$ su root
Password:
[root@master hadoop]# vim /etc/sudoers
[root@master hadoop]# vim /etc/sudoers
[root@master hadoop]# su hadoop
[hadoop@master ~]$ sudo vim /etc/sudoers

## Allow root to run any commands anywhere
root    ALL=(ALL)    ALL
hadoop  ALL=(ALL)    ALL
## Allows member to run any commands
#member  ALL=(ALL)    ALL

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

#1) Respect the privacy of others.
#2) Think before you type.
#3) With great power comes great responsibility.

[sudo] password for hadoop: 输入正确密码后, 可进入编辑该文件
[hadoop@master ~]$
```

图 27 进入 sudoers 文件以修改 hadoop 权限

④ 修改主机名

修改主机名（可切换到 root 用户，可使用 sudo）

`sudo vim hostname` 直接修改，保存退出

`hostname` 查看当前主机名,如主机名没变，重启一下即可

（重启可以点按钮重启，也可在 root 用户下使用 `init 6` 重启）

⑤ 配置 ip 地址与主机名的映射

在 `/etc/hosts` 里加上三个 IP 主机名，方便三台主机使用主机名 ping 通

(hosts 是不是计网上课刚讲过! 激动)

`sudo vim /etc/hosts`

```
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.196.101 master
192.168.196.102 slave1
192.168.196.103 slave2
```

图 28.在 hosts 里添加 IP 主机名

三台都加上下面三行后，互相之间可以用主机名 ping 通啦~

⑥ 按照 java 并配置 java 环境变量、安装 hadoop 及 Hadoop 环境变量（参考之前教程）

⑦ 设置三台机器的时间一致！（2>可能经常用哦~） `date` 命令查看当前时间

1>安装 ntpdate 工具: `sudo yum -y install ntp ntpdate`

2>设置系统时间与网络时间同步 `sudo ntpdate cn.pool.ntp.org`

3>再次查看时间 `date`

⑧ 删除（三台机器）hadoop 下的 data 文件和 logs （创建全新的环境）

2. 三台机器配置表

name	master	slave1	slave2
hdfs	namenode		secondarynamenode
	datanode	datanode	datanode
yarn		resourcemanager	
	nodemanager	nodemanager	nodemanager

注：此处设置三个 datanode 是为了测试副本为 3 的情况

3. 配置 SSH 免密码登录

I .生成公钥私钥对（三台机器）

[hadoop@master hadoop]\$ ssh-keygen -t rsa 连敲三个回车

公钥和私钥存放在家目录/home/hadoop 的隐藏目录.ssh 内 ll -a

II .将公钥加入到授权列表

每台主机要将自己的公钥发给另外两台主机，也要发给自己（方便 root 用户），所有三台主机分别需要三台机器的公钥，授权列表相同。

在三台机器上（.ssh 目录下）执行以下命令，意思是将本机的公钥发给 maser

[hadoop@master .ssh]\$ ssh-copy-id master 需要输入 master 主机的 hadoop 用户的密码

[hadoop@slave1 .ssh]\$ ssh-copy-id master 需要输入 master 主机的 hadoop 用户的密码

[hadoop@slave2 .ssh]\$ ssh-copy-id master 需要输入 master 主机的 hadoop 用户的密码

```
[hadoop@master .ssh]$ ll
total 16
-rw-----. 1 hadoop hadoop 1185 Dec  7 18:58 authorized_keys
-rw-----. 1 hadoop hadoop 1679 Dec  7 18:40 id_rsa
-rw-r--r--. 1 hadoop hadoop  395 Dec  7 18:40 id_rsa.pub
-rw-r--r--. 1 hadoop hadoop  355 Dec  5 21:44 known_hosts
[hadoop@master .ssh]$ cat authorized_keys
rUMq4uqp+rbJbrud4effSlC6sh+V22P2FiGYP2iGE3d6EZJ3cw8p hadoop@master
1ZB8mhwRx3XX5pGR2em8c3YISmsSyMO1p8BpPo0jAbQjSnK3hwbl hadoop@slave1
NZgshqIBILOmo5/rD3wjIMzWnzp0AK7qkIeKuQcLTP1OIcxa8Ygx hadoop@slave2
```

图 29.查看.ssh 内容及授权列表的内容

分别在三台主机上执行 ssh-copy-id slave1 和 ssh-copy-id slave2

现在可以使用免密登录+操作其它两台主机啦

4. 完全分布式搭建

在一台机器上搭建，然后远程发给其他机器即可~ 基于 master 主机

基于伪分布式的配置改动：

I . Hdfs

- ① 保留 core-site.xml 修改（三-1- I above）
- ② 将 hdfs-site.xml 的副本数修改为 3，并添加指定 secondarynode 节点（修改原因：四-2 above）

```
[hadoop@master hadoop]$ vim hdfs-site.xml
<!--指定 secondarynamenode 节点-->
    <property>
        <name>dfs.namenode.secondary.http-address</name>
        <value>slave2:50090</value>
    </property>

<configuration>
  <property>
    <!--分布式，（默认）副本数为3-->
    <name>dfs.replication</name>
    <value>3</value>
  </property>
  <!--指定secondarynamenode节点-->
  <property>
    <name>dfs.namenode.secondary.http-address</name>
    <value>slave2:50090</value>
  </property>
</configuration>
```

图 30.修改了两部分~

II . yarn

- ① 修改 yarn-site.xml 的 resourcemanager 为 slave1 节点

```
<property>
    <!--设置 resourcemanager 的是哪个节点-->
    <name>yarn.resourcemanager.hostname</name>
    <value>slave1</value>
</property>

<property>
  <!--设置resourcemanager的是哪个节点-->
  <name>yarn.resourcemanager.hostname</name>
  <value>slave1</value>
</property>
```

② 保留三个 JAVA_HOME 配置（三-3-② above）

③ 保留 mapred-site.xml 修改

III. 同步修改文件

可以直接将 master 上的/etc/hadoop 分发到 slave1 和 slave2

```
[hadoop@master etc]$ scp -r hadoop/ slave1:/home/hadoop/software/hadoop/etc
```

```
[hadoop@master etc]$ scp -r hadoop/ slave2:/home/hadoop/software/hadoop/etc
```

V. 格式化

第一次使用的时候一定要格式化，是在 namenode 上格式化

（如果一不小心第二次格式化，需要将三台机器上的 data logs 全部删除）

```
[hadoop@master software]$ hadoop namenode -format
```

```
21/12/07 20:31:33 INFO util.GSet: capacity = 2^20 = 1048576 entries
21/12/07 20:31:33 INFO namenode.FSDirectory: ACLs enabled? false
21/12/07 20:31:33 INFO namenode.FSDirectory: XAttrs enabled? true
21/12/07 20:31:33 INFO namenode.FSDirectory: Maximum size of an xattr: 16384
21/12/07 20:31:33 INFO namenode.NameNode: Caching file names occurring more than 10 times
21/12/07 20:31:34 INFO util.GSet: Computing capacity for map cachedBlocks
21/12/07 20:31:34 INFO util.GSet: VM type = 64-bit
21/12/07 20:31:34 INFO util.GSet: 0.25% max memory 966.7 MB = 2.4 MB
21/12/07 20:31:34 INFO util.GSet: capacity = 2^18 = 262144 entries
21/12/07 20:31:34 INFO namenode.FSNamesystem: dfs.namenode.safemode.threshold-pct = 0.9990000128746033
21/12/07 20:31:34 INFO namenode.FSNamesystem: dfs.namenode.safemode.min.datanodes = 0
21/12/07 20:31:34 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension = 30000
21/12/07 20:31:34 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
21/12/07 20:31:34 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
21/12/07 20:31:34 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
21/12/07 20:31:34 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
21/12/07 20:31:34 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
21/12/07 20:31:34 INFO util.GSet: Computing capacity for map NameNodeRetryCache
21/12/07 20:31:34 INFO util.GSet: VM type = 64-bit
21/12/07 20:31:34 INFO util.GSet: 0.0299999999329447746% max memory 966.7 MB = 297.0 KB
21/12/07 20:31:34 INFO util.GSet: capacity = 2^15 = 32768 entries
21/12/07 20:31:34 INFO namenode.FSImage: Allocated new BlockPoolId: BP-979052689-192.168.196.101-1639880295016
21/12/07 20:31:35 INFO common.Storage: Storage directory /home/hadoop/software/hadoop/data/tmp/dfs/name has been successfully formatted.
21/12/07 20:31:35 INFO namenode.FSImageFormatProtobuf: Saving image file /home/hadoop/software/hadoop/data/tmp/dfs/name/current/fsimage.ckpt_00000000 using no compression
21/12/07 20:31:36 INFO namenode.FSImageFormatProtobuf: Image file /home/hadoop/software/hadoop/data/tmp/dfs/name/current/fsimage.ckpt_00000000 of size 353 bytes saved in 1 seconds.
21/12/07 20:31:37 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
21/12/07 20:31:37 INFO util.ExitUtil: Exiting with status 0
21/12/07 20:31:37 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at master/192.168.196.101
*****/
[hadoop@master software]$
```

图 31.出现如上信息，一般是正确格式化了~

VII. 分别启动

a.启动 dfs

① master 启动 namenode

```
[hadoop@master data]$ hadoop-daemon.sh start namenode
```

② 三台机器启动 datanode `hadoop-daemon.sh start datanode`

③ 现在可以访问 192.168.196.101:50070

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	
slave2:50010 (192.168.196.103:50010)	2	In Service	15.79 GB	4 KB	1.87 GB	13.93 GB	0	4 KB (0%)	0	2.7.3	
slave1:50010 (192.168.196.102:50010)	2	In Service	15.79 GB	4 KB	1.98 GB	13.81 GB	0	4 KB (0%)	0	2.7.3	
master:50010 (192.168.196.101:50010)	1	In Service	15.79 GB	4 KB	5.32 GB	10.47 GB	0	4 KB (0%)	0	2.7.3	

图 32.datanode 有三个!

- ④ 启动 slave2 的 secondarynamenode
`hadoop-daemon.sh start secondarynamenode`
- ⑤ 上传一个文件到 hdfs, 可以看到该文件有三个副本
`[hadoop@master ~]$ hdfs dfs -put a.txt /`

File information - a.txt

Download

文件大小小于128M只占一个块

Block information -- Block 0

Block ID: 1073741825
 Block Pool ID: BP-979052689-192.168.196.101-1638880295016
 Generation Stamp: 1001
 Size: 50
 Availability:

- master
- slave2
- slave1

三个副本

Close

图 33.查看 a.txt 的具体内容

- ⑥ 现在 master 知道 namenode 在哪、知道 secondarynamenode 在哪, 但是不知道 datanode 在哪, 通过修改 slaves 文件确定
- `[hadoop@master hadoop]$ pwd`
`/home/hadoop/software/hadoop/etc/hadoop`
- `[hadoop@master hadoop]$ vim slaves`

```

1 hadoop101
master
slave1
slave2
~

```

同步文件到 slave1,slave2

`[hadoop@master hadoop]$ scp slaves slave1:/home/hadoop/software/hadoop/etc/hadoop/`

```
[hadoop@master hadoop]$ scp slaves slave2:/home/hadoop/software/hadoop/etc/hadoop/
```

⑦ 现在可以同步启动和结束 dfs（随便哪个主机都可以）：

启动 `start-dfs.sh` 结束 `stop-dfs.sh`

```
[hadoop@master hadoop]$ start-dfs.sh
Starting namenodes on [master]
master: starting namenode, logging to /home/hadoop/software/hadoop-2.7.3/logs/hadoop-hadoop-namenode-master.out 配置文件
slave2: starting datanode, logging to /home/hadoop/software/hadoop-2.7.3/logs/hadoop-hadoop-datanode-slave2.out slaves文件
slave1: starting datanode, logging to /home/hadoop/software/hadoop-2.7.3/logs/hadoop-hadoop-datanode-slave1.out slaves文件
master: starting datanode, logging to /home/hadoop/software/hadoop-2.7.3/logs/hadoop-hadoop-datanode-master.out slaves文件
Starting secondary namenodes [slave2]
slave2: starting secondarynamenode, logging to /home/hadoop/software/hadoop-2.7.3/logs/hadoop-hadoop-secondarynamenode-slave2.out 配置文件
```

图 34. 启动集群的 dfs

b.启动 yarn

注：如果 namenode 和 resourcemanager 不在一台机器上，启动 yarn 的时候，

要在 resourcemanager 的机器上启动！ 我在 `slave1` 上配置的 resourcemanager

① 在 resourcemanager 的机器上输入 `start-yarn.sh`（类似⑦同步启动）

② 可以查看三台机器启动的 java 进程 `jps`

[hadoop@master hadoop]\$ jps	[hadoop@slave1 hadoop]\$ jps	[hadoop@slave2 hadoop]\$ jps
34804 NameNode	4465 NodeManager	4417 DataNode
35242 NodeManager	4249 DataNode	4616 NodeManager
35389 Jps	4362 ResourceManager	4524 SecondaryNameNode
34910 DataNode	4766 Jps	4735 Jps

图 35.三台主机的 jps 不一样

③ 查看 yarn 的 web 页面 `192.168.196.102:8088`

在 hadoop102 启动 resourcemanager

(修改 windows 的 hosts 文件 `C:\Windows\System32\drivers\etc`，可以通过主机名:端口号访问 权限不足参考[如何修改 Hosts 文件-百度经验 \(baidu.com\)](http://baidu.com))

#	::1	localhost
127.0.0.1	activate.navicat.com	
192.168.196.101	master	
192.168.196.102	slave1	
192.168.196.103	slave2	

(如果分别启动 yarn，使用如下命令：

启动 yarn 的 resourcemanager `yarn-daemon.sh start resourcemanager`

启动 yarn 的 nodemanager `yarn-daemon.sh start nodemanager`

结束将 start 换成 stop 即可)

VI.同时启动

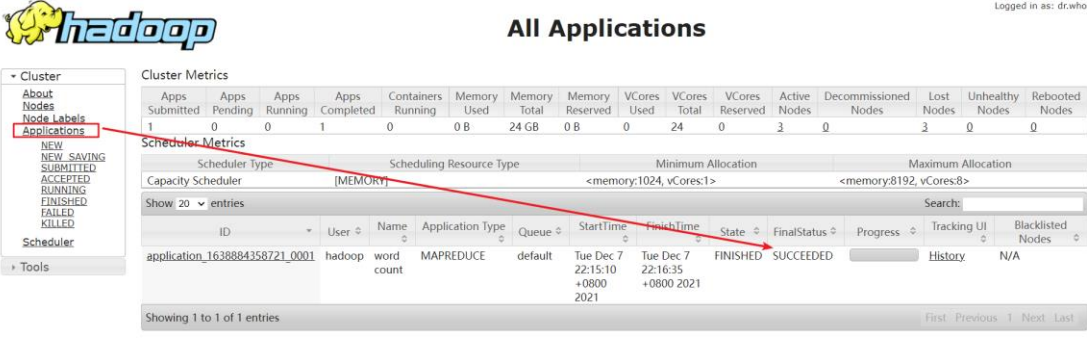
`start-all.sh` 尽量选择 resourcemanager 的机器启动

`stop-all.sh` 群关闭

5. Wordcount 案例

这次是在完全分布式的情况下（三台机器）运行 wordcount 案例

```
[hadoop@master hadoop]$ hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar wordcount /wc_in /wc_out
```



The screenshot shows the Hadoop YARN web interface. The left sidebar has a menu with 'Applications' highlighted. The main area is titled 'All Applications'. It includes a 'Cluster Metrics' table and a 'Scheduler Metrics' table. Below these, there's a table of applications. A red arrow points from the 'Applications' tab in the sidebar to the 'word count' application in the table.

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
1	0	0	1	0	0 B	24 GB	0 B	0	24	0	3	0	3	0	0

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI	Blacklisted Nodes
application_1638884358721_0001	hadoop	word count	MAPREDUCE	default	Tue Dec 7 22:15:10 +0800 2021	Tue Dec 7 22:16:35 +0800 2021	FINISHED	SUCCEEDED		History	N/A

图 36.yarn 页面，查看任务进度

完结撒花！



使用本教程包售后 hhh

