# 第1章 Hadoop运行环境搭建

## 1.1 模板虚拟机环境准备

**1）准备一台模板虚拟机hadoop100，虚拟机配置要求如下：**

**注：本文Linux系统环境全部以CentOS-7.5-x86-1804为例说明**

模板虚拟机：内存4G，硬盘50G，安装必要环境，为安装hadoop做准备

[root@hadoop100 ~]# yum install -y epel-release

[root@hadoop100 ~]# yum install -y psmisc nc net-tools rsync vim lrzsz ntp libzstd openssl-static tree iotop git

使用yum安装需要虚拟机可以正常上网，yum安装前可以先测试下虚拟机联网情况

[root@hadoop100 ~]# ping www.baidu.com

PING www.baidu.com (12.215.177.39) 56(84) bytes of data.

64 bytes from 12.215.177.39 (12.215.177.39): icmp\_seq=1 ttl=128 time=8.60 ms

64 bytes from 12.215.177.39 (12.215.177.39): icmp\_seq=2 ttl=128 time=7.72 ms

**2）关闭防火墙，关闭防火墙开机自启**

[root@hadoop100 ~]# systemctl stop firewalld

[root@hadoop100 ~]# systemctl disable firewalld

**3）创建user1用户，并修改user1用户的密码**

[root@hadoop100 ~]# useradd user1

[root@hadoop100 ~]# passwd user1

**4）配置user1用户具有root权限，方便后期加sudo执行root权限的命令**

[root@hadoop100 ~]# vim /etc/sudoers

修改/etc/sudoers文件，找到下面一行（91行），在root下面添加一行，如下所示：

## Allow root to run any commands anywhere

root ALL=(ALL) ALL

user1 ALL=(ALL) NOPASSWD:ALL

**5）在/opt目录下创建文件夹，并修改所属主和所属组**

（1）在/opt目录下创建module、software文件夹

[root@hadoop100 ~]# mkdir /opt/module

[root@hadoop100 ~]# mkdir /opt/software

（2）修改module、software文件夹的所有者和所属组均为user1用户

[root@hadoop100 ~]# chown user1:user1 /opt/module

[root@hadoop100 ~]# chown user1:user1 /opt/software

（3）查看module、software文件夹的所有者和所属组

[root@hadoop100 ~]# cd /opt/

[root@hadoop100 opt]# ll

总用量 12

drwxr-xr-x. 2 user1 user1 4096 5月 28 17:18 module

drwxr-xr-x. 2 root root 4096 9月 7 2017 rh

drwxr-xr-x. 2 user1 user1 4096 5月 28 17:18 software

**6）卸载虚拟机自带的open JDK**

[root@hadoop100 ~]# rpm -qa | grep -i java | xargs -n1 rpm -e --nodeps

**7）重启虚拟机**

[root@hadoop100 ~]# reboot

## 1.2 克隆虚拟机

**1）利用模板机hadoop100，克隆三台虚拟机：hadoop102 hadoop103 hadoop104**

**2）修改克隆机IP，以下以hadoop102举例说明**

（1）修改克隆虚拟机的静态IP

[root@hadoop100 ~]# vim /etc/sysconfig/network-scripts/ifcfg-ens33

改成

DEVICE=ens33

TYPE=Ethernet

ONBOOT=yes

BOOTPROTO=static

NAME="ens33"

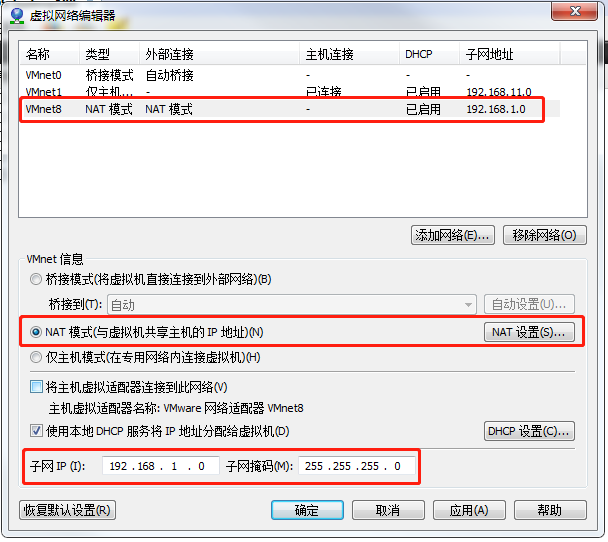
IPADDR=192.168.1.102

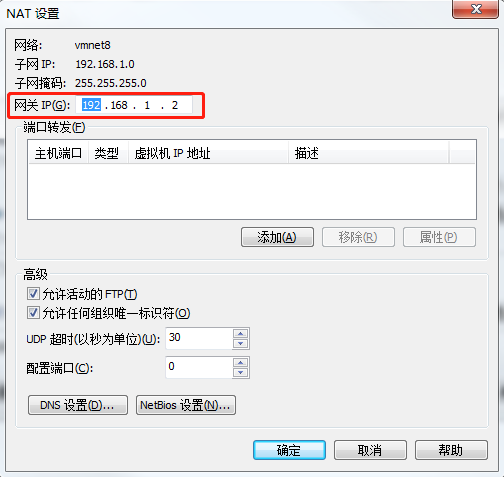
PREFIX=24

GATEWAY=192.168.1.2

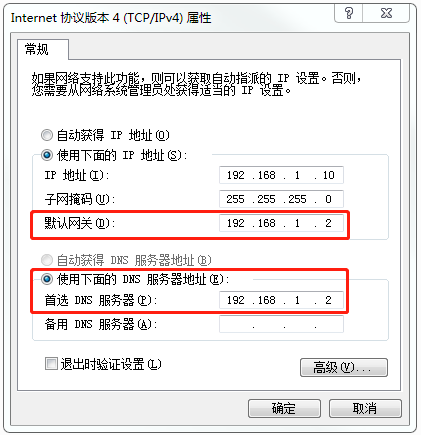
DNS1=192.168.1.2

（2）查看Linux虚拟机的虚拟网络编辑器，编辑->虚拟网络编辑器->VMnet8





（3）查看Windows系统适配器VMware Network Adapter VMnet8的IP地址



（4）保证Linux系统ifcfg-ens33文件中IP地址、虚拟网络编辑器地址和Windows系统VM8网络IP地址相同。

**3）修改克隆机主机名，以下以hadoop102举例说明**

（1）修改主机名称，两种方法二选一

[root@hadoop100 ~]# hostnamectl --static set-hostname hadoop102

或者修改/etc/hostname文件

[root@hadoop100 ~]# vim /etc/hostname

hadoop102

（2）配置linux克隆机主机名称映射hosts文件，打开/etc/hosts

[root@hadoop100 ~]# vim /etc/hosts

添加如下内容

192.168.1.100 hadoop100

192.168.1.101 hadoop101

192.168.1.102 hadoop102

192.168.1.103 hadoop103

192.168.1.104 hadoop104

192.168.1.105 hadoop105

192.168.1.106 hadoop106

192.168.1.107 hadoop107

192.168.1.108 hadoop108

**4）重启克隆机hadoop102**

[root@hadoop100 ~]# reboot

**5）修改windows的主机映射文件（hosts文件）**

（1）如果操作系统是window7，可以直接修改

（a）进入C:\Windows\System32\drivers\etc路径

（b）打开hosts文件并添加如下内容，然后保存

192.168.1.100 hadoop100

192.168.1.101 hadoop101

192.168.1.102 hadoop102

192.168.1.103 hadoop103

192.168.1.104 hadoop104

192.168.1.105 hadoop105

192.168.1.106 hadoop106

192.168.1.107 hadoop107

192.168.1.108 hadoop108

（2）如果操作系统是window10，先拷贝出来，修改保存以后，再覆盖即可

（a）进入C:\Windows\System32\drivers\etc路径

（b）拷贝hosts文件到桌面

（c）打开桌面hosts文件并添加如下内容

192.168.1.100 hadoop100

192.168.1.101 hadoop101

192.168.1.102 hadoop102

192.168.1.103 hadoop103

192.168.1.104 hadoop104

192.168.1.105 hadoop105

192.168.1.106 hadoop106

192.168.1.107 hadoop107

192.168.1.108 hadoop108

（d）将桌面hosts文件覆盖C:\Windows\System32\drivers\etc路径hosts文件

## 1.3 在hadoop102安装JDK

**1）卸载现有JDK**

[user1@hadoop102 ~]$ rpm -qa | grep -i java | xargs -n1 sudo rpm -e --nodeps

**2）解压JDK到/opt/module目录下**

[user1@hadoop102 software]$ tar -zxvf jdk-8u212-linux-x62.tar.gz -C /opt/module/

**3）配置JDK环境变量**

（1）新建/etc/profile.d/my\_env.sh文件

[user1@hadoop102 ~]$ sudo vim /etc/profile.d/my\_env.sh

添加如下内容

#JAVA\_HOME

export JAVA\_HOME=/opt/module/jdk1.8.0\_212

export PATH=$PATH:$JAVA\_HOME/bin

（2）保存后退出

:wq

（3）source一下/etc/profile文件，让新的环境变量PATH生效

[user1@hadoop102 ~]$ source /etc/profile

**4）测试JDK是否安装成功**

[user1@hadoop102 ~]$ java -version

如果能看到以下结果，则代表Java安装成功。

java version "1.8.0\_212"

注意：重启（如果java -version可以用就不用重启）

[user1@hadoop102 ~]$ sudo reboot

## 1.4 在hadoop102安装Hadoop

**1）进入到Hadoop安装包路径下**

[user1@hadoop102 ~]$ cd /opt/software/

**2）解压安装文件到/opt/module下面**

[user1@hadoop102 software]$ tar -zxvf hadoop-1.1.1.tar.gz -C /opt/module/

**3）查看是否解压成功**

[user1@hadoop102 software]$ ls /opt/module/

hadoop-1.1.3

**4）将Hadoop添加到环境变量**

（1）获取Hadoop安装路径

[user1@hadoop102 hadoop-1.1.3]$ pwd

/opt/module/hadoop-1.1.3

（2）打开/etc/profile.d/my\_env.sh文件

sudo vim /etc/profile.d/my\_env.sh

在my\_env.sh文件末尾添加如下内容：（shift+g）

#HADOOP\_HOME

export HADOOP\_HOME=/opt/module/hadoop-1.1.3

export PATH=$PATH:$HADOOP\_HOME/bin

export PATH=$PATH:$HADOOP\_HOME/sbin

（3）保存后退出

:wq

（4）让修改后的文件生效

[user1@hadoop102 hadoop-1.1.3]$ source /etc/profile

**5）重启(如果Hadoop命令不能用再重启)**

[user1@hadoop102 hadoop-1.1.3]$ sync

[user1@hadoop102 hadoop-1.1.3]$ sudo reboot

## 1.5 Hadoop目录结构

**1）查看Hadoop目录结构**

[user1@hadoop102 hadoop-1.1.3]$ ll

总用量 52

drwxr-xr-x. 2 user1 user1 4096 5月 22 2017 **bin**

drwxr-xr-x. 3 user1 user1 4096 5月 22 2017 **etc**

drwxr-xr-x. 2 user1 user1 4096 5月 22 2017 include

drwxr-xr-x. 3 user1 user1 4096 5月 22 2017 **lib**

drwxr-xr-x. 2 user1 user1 4096 5月 22 2017 libexec

-rw-r--r--. 1 user1 user1 15429 5月 22 2017 LICENSE.txt

-rw-r--r--. 1 user1 user1 101 5月 22 2017 NOTICE.txt

-rw-r--r--. 1 user1 user1 1366 5月 22 2017 README.txt

drwxr-xr-x. 2 user1 user1 4096 5月 22 2017 **sbin**

drwxr-xr-x. 4 user1 user1 4096 5月 22 2017 **share**

**2）重要目录**

（1）bin目录：存放对Hadoop相关服务（HDFS,YARN）进行操作的脚本

（2）etc目录：Hadoop的配置文件目录，存放Hadoop的配置文件

（3）lib目录：存放Hadoop的本地库（对数据进行压缩解压缩功能）

（4）sbin目录：存放启动或停止Hadoop相关服务的脚本

（5）share目录：存放Hadoop的依赖jar包、文档、和官方案例

# 第2章 Hadoop运行模式

Hadoop运行模式包括：本地模式、伪分布式模式以及完全分布式模式。

Hadoop官方网站：<http://hadoop.apache.org/>

## 2.1 本地运行模式（官方wordcount）

**1）创建在hadoop-1.1.3文件下面创建一个wcinput文件夹**

[user1@hadoop102 hadoop-1.1.3]$ mkdir wcinput

**2）在wcinput文件下创建一个****word.txt文件**

[user1@hadoop102 hadoop-1.1.3]$ cd wcinput

**3）编辑word.txt文件**

[user1@hadoop102 wcinput]$ vim word.txt

在文件中输入如下内容

hadoop yarn

hadoop mapreduce

user1

user1

保存退出：：wq

**4）回到Hadoop目录/opt/module/hadoop-1.1.3**

**5）执行程序**

[user1@hadoop102 hadoop-1.1.3]$ hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-1.1.1.jar wordcount wcinput wcoutput

**6）查看结果**

[user1@hadoop102 hadoop-1.1.3]$ cat wcoutput/part-r-00000

看到如下结果：

user1 2

hadoop 2

mapreduce 1

yarn 1

## 2.2 完全分布式运行模式

分析：

1）准备3台客户机（关闭防火墙、静态ip、主机名称）

2）安装JDK

3）配置环境变量

4）安装Hadoop

5）配置环境变量

**6）配置集群**

**7）单点启动**

**8）配置ssh**

**9）群起并测试集群**

### 2.2.1 虚拟机准备

见上文

### 2.2.2 编写集群分发脚本xsync

**1）scp（secure copy）安全拷贝**

（1）scp定义：

scp可以实现服务器与服务器之间的数据拷贝。（from server1 to server2）

（2）基本语法

scp -r $pdir/$fname $user@hadoop$host:$pdir/$fname

命令 递归 要拷贝的文件路径/名称 目的用户@主机:目的路径/名称

（3）案例实操

前提：在 hadoop102 hadoop103 hadoop104 都已经创建好的 /opt/module

/opt/software 两个目录， 并且已经把这两个目录修改为user1:user1

sudo chown user1:user1 -R /opt/module

（a）在hadoop102上，将hadoop102中/opt/module/jdk1.8.0\_212目录拷贝到hadoop103上。

[user1@hadoop102 ~]$ scp -r /opt/module/jdk1.8.0\_212 user1@hadoop103:/opt/module

（b）在hadoop103上，将hadoop102中/opt/module/hadoop-1.1.3目录拷贝到hadoop103上。

[user1@hadoop103 ~]$ scp -r user1@hadoop102:/opt/module/hadoop-1.1.3 /opt/module/

（c）在hadoop103上操作，将hadoop102中/opt/module目录下所有目录拷贝到hadoop104上。

[user1@hadoop103 opt]$ scp -r user1@hadoop102:/opt/module/\* user1@hadoop104:/opt/module

**2）rsync远程同步工具**

rsync主要用于备份和镜像。具有速度快、避免复制相同内容和支持符号链接的优点。

rsync和scp区别：用rsync做文件的复制要比scp的速度快，rsync只对差异文件做更新。scp是把所有文件都复制过去。

（1）基本语法

rsync -av $pdir/$fname $user@hadoop$host:$pdir/$fname

命令 选项参数 要拷贝的文件路径/名称 目的用户@主机:目的路径/名称

选项参数说明

|  |  |
| --- | --- |
| 选项 | 功能 |
| -a | 归档拷贝 |
| -v | 显示复制过程 |

（2）案例实操

（a）把hadoop102机器上的/opt/software目录同步到hadoop103服务器的/opt/software目录下

[user1@hadoop102 opt]$ rsync -av /opt/software/\* user1@hadoop103:/opt/software

**3）xsync集群分发脚本**

（1）需求：循环复制文件到所有节点的相同目录下

（2）需求分析：

（a）rsync命令原始拷贝：

rsync -av /opt/module root@hadoop103:/opt/

（b）期望脚本：

xsync要同步的文件名称

（c）说明：在/home/user1/bin这个目录下存放的脚本，user1用户可以在系统任何地方直接执行。

（3）脚本实现

（a）在/home/user1/bin目录下创建xsync文件

[user1@hadoop102 opt]$ cd /home/user1

[user1@hadoop102 ~]$ mkdir bin

[user1@hadoop102 ~]$ cd bin

[user1@hadoop102 bin]$ vim xsync

在该文件中编写如下代码

#!/bin/bash

#1. 判断参数个数

if [ $# -lt 1 ]

then

echo Not Enough Arguement!

exit;

fi

#2. 遍历集群所有机器

for host in hadoop102 hadoop103 hadoop104

do

echo ==================== $host ====================

#1. 遍历所有目录，挨个发送

for file in $@

do

#2. 判断文件是否存在

if [ -e $file ]

then

#5. 获取父目录

pdir=$(cd -P $(dirname $file); pwd)

#6. 获取当前文件的名称

fname=$(basename $file)

ssh $host "mkdir -p $pdir"

rsync -av $pdir/$fname $host:$pdir

else

echo $file does not exists!

fi

done

done

（b）修改脚本 xsync 具有执行权限

[user1@hadoop102 bin]$ chmod +x xsync

（c）将脚本复制到/bin中，以便全局调用

[user1@hadoop102 bin]$ sudo cp xsync /bin/

（d）测试脚本

[user1@hadoop102 ~]$ xsync /home/user1/bin

[user1@hadoop102 bin]$ sudo xsync /bin/xsync

### 2.2.3 SSH无密登录配置

**1）配置ssh**

（1）基本语法

ssh另一台电脑的ip地址

（2）ssh连接时出现Host key verification failed的解决方法

[user1@hadoop102 ~]$ ssh hadoop103

出现：

The authenticity of host '192.168.1.103 (192.168.1.103)' can't be established.

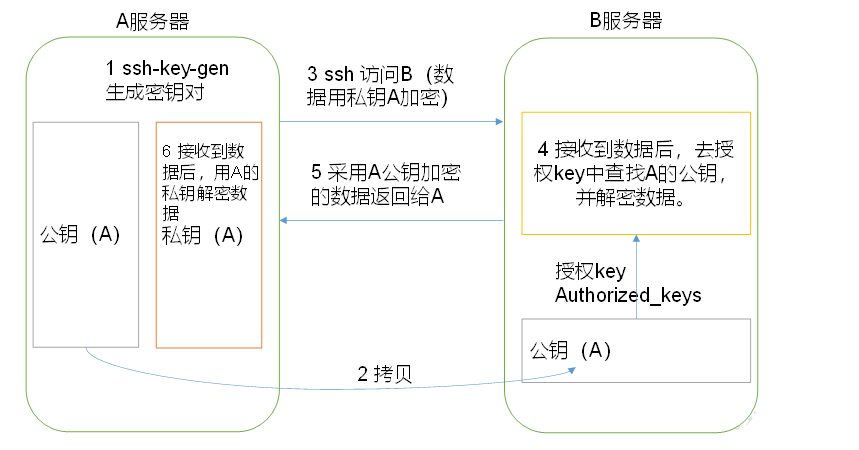
RSA key fingerprint is cf:1e:de:d7:d0:4c:2d:98:60:b4:fd:ae:b1:2d:ad:06.

Are you sure you want to continue connecting (yes/no)?

（3）解决方案如下：直接输入yes

**2）无密钥配置**

（1）免密登录原理



（2）生成公钥和私钥：

[user1@hadoop102 .ssh]$ ssh-keygen -t rsa

然后敲（三个回车），就会生成两个文件id\_rsa（私钥）、id\_rsa.pub（公钥）

（3）将公钥拷贝到要免密登录的目标机器上

[user1@hadoop102 .ssh]$ ssh-copy-id hadoop102

[user1@hadoop102 .ssh]$ ssh-copy-id hadoop103

[user1@hadoop102 .ssh]$ ssh-copy-id hadoop104

注意：

还需要在hadoop103上采用user1账号配置一下无密登录到hadoop102、hadoop103、hadoop104服务器上。

还需要在hadoop104上采用user1账号配置一下无密登录到hadoop102、hadoop103、hadoop104服务器上。

还需要在hadoop102上采用root账号，配置一下无密登录到hadoop102、hadoop103、hadoop104；

**3）.ssh文件夹下（~/.ssh）的文件功能解释**

|  |  |
| --- | --- |
| known\_hosts | 记录ssh访问过计算机的公钥(public key) |
| id\_rsa | 生成的私钥 |
| id\_rsa.pub | 生成的公钥 |
| authorized\_keys | 存放授权过的无密登录服务器公钥 |

### 2.2.4 集群配置

**1）集群部署规划**

注意：NameNode和SecondaryNameNode不要安装在同一台服务器

注意：ResourceManager也很消耗内存，不要和NameNode、SecondaryNameNode配置在同一台机器上。

|  |  |  |  |
| --- | --- | --- | --- |
|  | hadoop102 | hadoop103 | hadoop104 |
| HDFS | NameNode  DataNode | DataNode | SecondaryNameNode  DataNode |
| YARN | NodeManager | ResourceManager  NodeManager | NodeManager |

**2）配置文件说明**

Hadoop配置文件分两类：默认配置文件和自定义配置文件，只有用户想修改某一默认配置值时，才需要修改自定义配置文件，更改相应属性值。

（1）默认配置文件：

|  |  |
| --- | --- |
| 要获取的默认文件 | 文件存放在Hadoop的jar包中的位置 |
| [core-default.xml] | hadoop-common-1.1.1.jar/ core-default.xml |
| [hdfs-default.xml] | hadoop-hdfs-1.1.1.jar/ hdfs-default.xml |
| [yarn-default.xml] | hadoop-yarn-common-1.1.1.jar/ yarn-default.xml |
| [mapred-default.xml] | hadoop-mapreduce-client-core-1.1.1.jar/ mapred-default.xml |

（2）自定义配置文件：

**core-site.xml、hdfs-site.xml、yarn-site.xml、mapred-site.xml**四个配置文件存放在$HADOOP\_HOME/etc/hadoop这个路径上，用户可以根据项目需求重新进行修改配置。

（3）常用端口号说明

|  |  |  |  |
| --- | --- | --- | --- |
| Daemon | App | Hadoop2 | Hadoop3 |
| NameNode Port | Hadoop HDFS NameNode | 8020 / 9000 | 9820 |
|  | Hadoop HDFS NameNode HTTP UI | 50070 | 9870 |
| Secondary NameNode Port | Secondary NameNode | 50091 | 9869 |
|  | Secondary NameNode HTTP UI | 50090 | 9868 |
| DataNode Port | Hadoop HDFS DataNode IPC | 50020 | 9867 |
|  | Hadoop HDFS DataNode | 50010 | 9866 |
|  | Hadoop HDFS DataNode HTTP UI | 50075 | 9864 |

**3）配置集群**

（1）核心配置文件

配置core-site.xml

[user1@hadoop102 ~]$ cd $HADOOP\_HOME/etc/hadoop

[user1@hadoop102 hadoop]$ vim core-site.xml

文件内容如下：

<?xml version="1.0" encoding="UTF-8"?>

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

<configuration>

<!-- 指定NameNode的地址 -->

<property>

<name>fs.defaultFS</name>

<value>hdfs://hadoop102:9820</value>

</property>

<!-- 指定hadoop数据的存储目录 -->

<property>

<name>hadoop.tmp.dir</name>

<value>/opt/module/hadoop-1.1.3/data</value>

</property>

<!-- 配置HDFS网页登录使用的静态用户为user1 -->

<property>

<name>hadoop.http.staticuser.user</name>

<value>user1</value>

</property>

<!-- 配置该user1(superUser)允许通过代理访问的主机节点 -->

<property>

<name>hadoop.proxyuser.user1.hosts</name>

<value>\*</value>

</property>

<!-- 配置该user1(superUser)允许通过代理用户所属组 -->

<property>

<name>hadoop.proxyuser.user1.groups</name>

<value>\*</value>

</property>

<!-- 配置该user1(superUser)允许通过代理的用户-->

<property>

<name>hadoop.proxyuser.user1.groups</name>

<value>\*</value>

</property>

</configuration>

（2）HDFS配置文件

配置hdfs-site.xml

[user1@hadoop102 hadoop]$ vim hdfs-site.xml

文件内容如下：

<?xml version="1.0" encoding="UTF-8"?>

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

<configuration>

<!-- nn web端访问地址-->

<property>

<name>dfs.namenode.http-address</name>

<value>hadoop102:9870</value>

</property>

<!-- 2nn web端访问地址-->

<property>

<name>dfs.namenode.secondary.http-address</name>

<value>hadoop104:9868</value>

</property>

</configuration>

（3）YARN配置文件

配置yarn-site.xml

[user1@hadoop102 hadoop]$ vim yarn-site.xml

文件内容如下：

<?xml version="1.0" encoding="UTF-8"?>

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

<configuration>

<!-- 指定MR走shuffle -->

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

<!-- 指定ResourceManager的地址-->

<property>

<name>yarn.resourcemanager.hostname</name>

<value>hadoop103</value>

</property>

<!-- 环境变量的继承 -->

<property>

<name>yarn.nodemanager.env-whitelist</name>

<value>JAVA\_HOME,HADOOP\_COMMON\_HOME,HADOOP\_HDFS\_HOME,HADOOP\_CONF\_DIR,CLASSPATH\_PREPEND\_DISTCACHE,HADOOP\_YARN\_HOME,HADOOP\_MAPRED\_HOME</value>

</property>

<!-- yarn容器允许分配的最大最小内存 -->

<property>

<name>yarn.scheduler.minimum-allocation-mb</name>

<value>512</value>

</property>

<property>

<name>yarn.scheduler.maximum-allocation-mb</name>

<value>4096</value>

</property>

<!-- yarn容器允许管理的物理内存大小 -->

<property>

<name>yarn.nodemanager.resource.memory-mb</name>

<value>4096</value>

</property>

<!-- 关闭yarn对物理内存和虚拟内存的限制检查 -->

<property>

<name>yarn.nodemanager.pmem-check-enabled</name>

<value>false</value>

</property>

<property>

<name>yarn.nodemanager.vmem-check-enabled</name>

<value>false</value>

</property>

</configuration>

（4）MapReduce配置文件

配置mapred-site.xml

[user1@hadoop102 hadoop]$ vim mapred-site.xml

文件内容如下：

<?xml version="1.0" encoding="UTF-8"?>

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

<configuration>

<!-- 指定MapReduce程序运行在Yarn上 -->

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

</configuration>

**4）在集群上分发配置好的Hadoop配置文件**

[user1@hadoop102 hadoop]$ xsync /opt/module/hadoop-1.1.3/etc/hadoop/

**5）去103和104上查看文件分发情况**

[user1@hadoop103 ~]$ cat /opt/module/hadoop-1.1.3/etc/hadoop/core-site.xml

[user1@hadoop104 ~]$ cat /opt/module/hadoop-1.1.3/etc/hadoop/core-site.xml

### 2.2.5 群起集群

**1）配置workers**

[user1@hadoop102 hadoop]$ vim /opt/module/hadoop-1.1.3/etc/hadoop/workers

在该文件中增加如下内容：

hadoop102

hadoop103

hadoop104

注意：该文件中添加的内容结尾不允许有空格，文件中不允许有空行。

同步所有节点配置文件

[user1@hadoop102 hadoop]$ xsync /opt/module/hadoop-1.1.3/etc

**2）启动集群**

（1）**如果集群是第一次启动**，需要在hadoop102节点格式化NameNode（注意格式化NameNode，会产生新的集群id，导致NameNode和DataNode的集群id不一致，集群找不到已往数据。如果集群在运行过程中报错，需要重新格式化NameNode的话，一定要先停止namenode和datanode进程，并且要删除所有机器的data和logs目录，然后再进行格式化。）

[user1@hadoop102 ~]$ hdfs namenode -format

（2）启动HDFS

[user1@hadoop102 hadoop-1.1.3]$ sbin/start-dfs.sh

（3）**在配置了ResourceManager的节点（hadoop103）**启动YARN

[user1@hadoop103 hadoop-1.1.3]$ sbin/start-yarn.sh

（4）Web端查看HDFS的NameNode

（a）浏览器中输入：<http://hadoop102:9870>

（b）查看HDFS上存储的数据信息

（5）Web端查看YARN的ResourceManager

（a）浏览器中输入：<http://hadoop103:8088>

（b）查看YARN上运行的Job信息

**3）集群基本测试**

（1）上传文件到集群

上传小文件

[user1@hadoop102 ~]$ hadoop fs -mkdir /input

[user1@hadoop102 ~]$ hadoop fs -put $HADOOP\_HOME/wcinput/word.txt /input

上传大文件

[user1@hadoop102 ~]$ hadoop fs -put /opt/software/jdk-8u212-linux-x62.tar.gz /

（2）上传文件后查看文件存放在什么位置

（a）查看HDFS文件存储路径

[user1@hadoop102 subdir0]$ pwd

/opt/module/hadoop-1.1.3/data/dfs/data/current/BP-938951106-192.168.10.107-1495462844069/current/finalized/subdir0/subdir0

（b）查看HDFS在磁盘存储文件内容

[user1@hadoop102 subdir0]$ cat blk\_1073741825

hadoop yarn

hadoop mapreduce

user1

user1

（3）拼接

-rw-rw-r--. 1 user1 user1 134217728 5月 23 16:01 **blk\_1073741836**

-rw-rw-r--. 1 user1 user1 1048583 5月 23 16:01 blk\_1073741836\_1012.meta

-rw-rw-r--. 1 user1 user1 63439959 5月 23 16:01 **blk\_1073741837**

-rw-rw-r--. 1 user1 user1 495635 5月 23 16:01 blk\_1073741837\_1011.meta

[user1@hadoop102 subdir0]$ cat blk\_1073741836>>tmp.tar.gz

[user1@hadoop102 subdir0]$ cat blk\_1073741837>>tmp.tar.gz

[user1@hadoop102 subdir0]$ tar -zxvf tmp.tar.gz

（4）下载

[user1@hadoop104 software]$ hadoop fs -get /jdk-8u212-linux-x62.tar.gz ./

（5）执行wordcount程序

[user1@hadoop102 hadoop-1.1.3]$ hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-1.1.1.jar wordcount /input /output

### 2.2.6 集群启动/停止方式总结

**1）各个服务组件逐一启动/停止**

（1）分别启动/停止HDFS组件

hdfs --daemon start/stop namenode/datanode/secondarynamenode

（2）启动/停止YARN

yarn --daemon start/stop resourcemanager/nodemanager

**2）各个模块分开启动/停止（配置ssh是前提）常用**

（1）整体启动/停止HDFS

start-dfs.sh/stop-dfs.sh

（2）整体启动/停止YARN

start-yarn.sh/stop-yarn.sh

### 2.2.7 配置历史服务器

为了查看程序的历史运行情况，需要配置一下历史服务器。具体配置步骤如下：

**1）配置mapred-site.xml**

[user1@hadoop102 hadoop]$ vim mapred-site.xml

在该文件里面增加如下配置。

<!-- 历史服务器端地址 -->

<property>

<name>mapreduce.jobhistory.address</name>

<value>hadoop102:10020</value>

</property>

<!-- 历史服务器web端地址 -->

<property>

<name>mapreduce.jobhistory.webapp.address</name>

<value>hadoop102:19888</value>

</property>

**2）分发配置**

[user1@hadoop102 hadoop]$ xsync $HADOOP\_HOME/etc/hadoop/mapred-site.xml

**3）在hadoop102启动历史服务器**

[user1@hadoop102 hadoop]$ mapred --daemon start historyserver

**4）查看历史服务器是否启动**

[user1@hadoop102 hadoop]$ jps

**5）查看JobHistory**

<http://hadoop102:19888/jobhistory>

### 2.2.8 配置日志的聚集

日志聚集概念：应用运行完成以后，将程序运行日志信息上传到HDFS系统上。

日志聚集功能好处：可以方便的查看到程序运行详情，方便开发调试。

注意：开启日志聚集功能，需要重新启动NodeManager 、ResourceManager和HistoryServer。

开启日志聚集功能具体步骤如下：

**1）配置yarn-site.xml**

[user1@hadoop102 hadoop]$ vim yarn-site.xml

在该文件里面增加如下配置。

<!-- 开启日志聚集功能 -->

<property>

<name>yarn.log-aggregation-enable</name>

<value>true</value>

</property>

<!-- 设置日志聚集服务器地址 -->

<property>

<name>yarn.log.server.url</name>

<value>http://hadoop102:19888/jobhistory/logs</value>

</property>

<!-- 设置日志保留时间为7天 -->

<property>

<name>yarn.log-aggregation.retain-seconds</name>

<value>604800</value>

</property>

**2）分发配置**

[user1@hadoop102 hadoop]$ xsync $HADOOP\_HOME/etc/hadoop/yarn-site.xml

**3）关闭NodeManager 、ResourceManager和HistoryServer**

[user1@hadoop103 ~]$ stop-yarn.sh

[user1@hadoop102 ~]$ mapred --daemon stop historyserver

**4）启动NodeManager 、ResourceManage和HistoryServer**

[user1@hadoop103 ~]$ start-yarn.sh

[user1@hadoop102 ~]$ mapred --daemon start historyserver

**5）删除HDFS上已经存在的输出文件**

[user1@hadoop102 ~]$ hadoop fs -rm -r /output

**6）执行WordCount程序**

[user1@hadoop102 ~]$ hadoop jar $HADOOP\_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-1.1.1.jar wordcount /input /output

**7）查看日志，**

<http://hadoop102:19888/jobhistory>

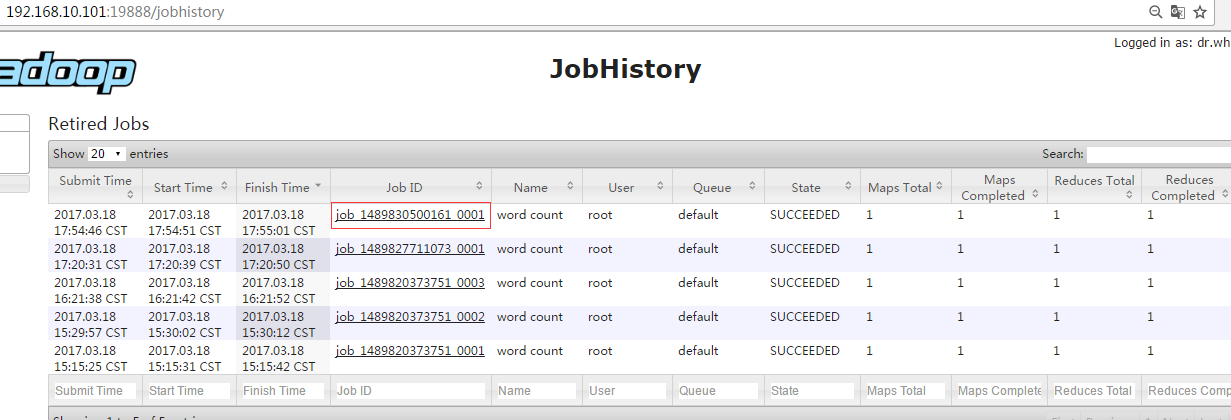


图 Job History

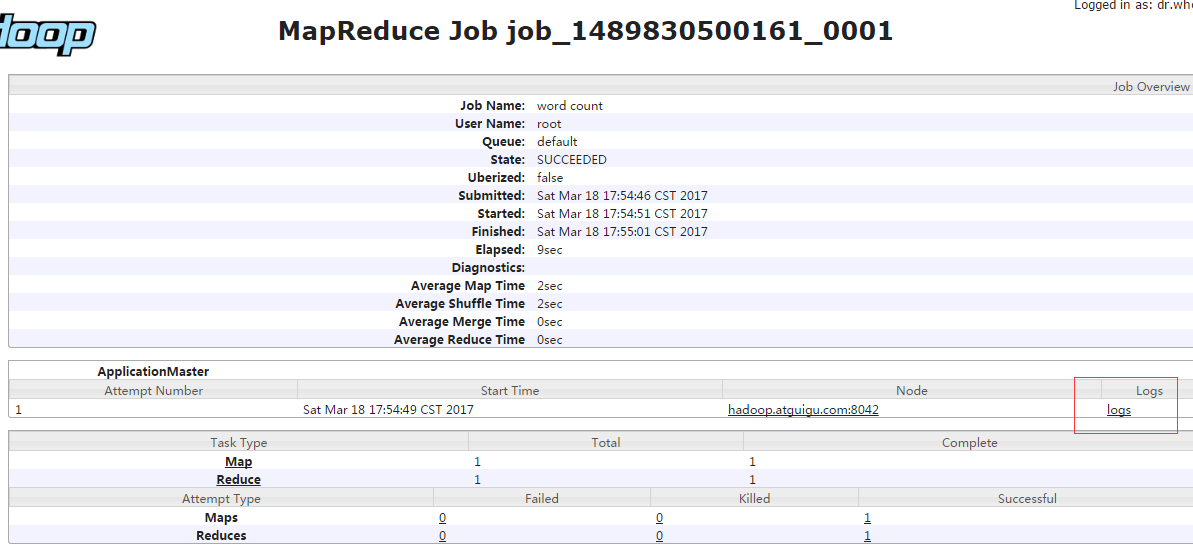


图 job运行情况



图 查看日志

### 2.2.9 编写hadoop集群常用脚本

**1）查看三台服务器java进程脚本：jpsall**

[user1@hadoop102 ~]$ cd /home/user1/bin

[user1@hadoop102 ~]$ vim jpsall

然后输入

#!/bin/bash

for host in hadoop102 hadoop103 hadoop104

do

echo =============== $host ===============

ssh $host jps $@ | grep -v Jps

done

保存后退出，然后赋予脚本执行权限

[user1@hadoop102 bin]$ chmod +x jpsall

**2）hadoop集群启停脚本（包含hdfs，yarn，historyserver）：myhadoop.sh**

[user1@hadoop102 ~]$ cd /home/user1/bin

[user1@hadoop102 ~]$ vim myhadoop.sh

然后输入

#!/bin/bash

if [ $# -lt 1 ]

then

echo "No Args Input..."

exit ;

fi

case $1 in

"start")

echo " =================== 启动 hadoop集群 ==================="

echo " --------------- 启动 hdfs ---------------"

ssh hadoop102 "/opt/module/hadoop-1.1.3/sbin/start-dfs.sh"

echo " --------------- 启动 yarn ---------------"

ssh hadoop103 "/opt/module/hadoop-1.1.3/sbin/start-yarn.sh"

echo " --------------- 启动 historyserver ---------------"

ssh hadoop102 "/opt/module/hadoop-1.1.3/bin/mapred --daemon start historyserver"

;;

"stop")

echo " =================== 关闭 hadoop集群 ==================="

echo " --------------- 关闭 historyserver ---------------"

ssh hadoop102 "/opt/module/hadoop-1.1.3/bin/mapred --daemon stop historyserver"

echo " --------------- 关闭 yarn ---------------"

ssh hadoop103 "/opt/module/hadoop-1.1.3/sbin/stop-yarn.sh"

echo " --------------- 关闭 hdfs ---------------"

ssh hadoop102 "/opt/module/hadoop-1.1.3/sbin/stop-dfs.sh"

;;

\*)

echo "Input Args Error..."

;;

esac

保存后退出，然后赋予脚本执行权限

[user1@hadoop102 bin]$ chmod +x myhadoop.sh

**3）分发/home/user1/bin目录，保证自定义脚本在三台机器上都可以使用**

[user1@hadoop102 ~]$ xsync /home/user1/bin/

### 2.2.10 集群时间同步

时间同步的方式：找一个机器，作为时间服务器，所有的机器与这台集群时间进行定时的同步，比如，每隔十分钟，同步一次时间。

**配置时间同步具体实操：**

**1）时间服务器配置（必须root用户）**

（0）查看所有节点ntpd服务状态和开机自启动状态

[user1@hadoop102 ~]$ sudo systemctl status ntpd

[user1@hadoop102 ~]$ sudo systemctl is-enabled ntpd

（1）在所有节点关闭ntp服务和自启动

[user1@hadoop102 ~]$ sudo systemctl stop ntpd

[user1@hadoop102 ~]$ sudo systemctl disable ntpd

（2）修改hadoop102的ntp.conf配置文件

[user1@hadoop102 ~]$ sudo vim /etc/ntp.conf

修改内容如下

a）修改1（授权192.168.1.0-192.168.1.255网段上的所有机器可以从这台机器上查询和同步时间）

#restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap

为restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap

b）修改2（集群在局域网中，不使用其他互联网上的时间）

server 0.centos.pool.ntp.org iburst

server 1.centos.pool.ntp.org iburst

server 2.centos.pool.ntp.org iburst

server 1.centos.pool.ntp.org iburst

为

**#**server 0.centos.pool.ntp.org iburst

**#**server 1.centos.pool.ntp.org iburst

**#**server 2.centos.pool.ntp.org iburst

**#**server 1.centos.pool.ntp.org iburst

c）添加3（当该节点丢失网络连接，依然可以采用本地时间作为时间服务器为集群中的其他节点提供时间同步）

server 127.127.1.0

fudge 127.127.1.0 stratum 10

（3）修改hadoop102的/etc/sysconfig/ntpd 文件

[user1@hadoop102 ~]$ sudo vim /etc/sysconfig/ntpd

增加内容如下（让硬件时间与系统时间一起同步）

SYNC\_HWCLOCK=yes

（4）重新启动ntpd服务

[user1@hadoop102 ~]$ sudo systemctl start ntpd

（5）设置ntpd服务开机启动

[user1@hadoop102 ~]$ sudo systemctl enable ntpd

**2）其他机器配置（必须root用户）**

（1）在其他机器配置10分钟与时间服务器同步一次

[user1@hadoop103 ~]$ sudo crontab -e

编写定时任务如下：

\*/10 \* \* \* \* /usr/sbin/ntpdate hadoop102

（2）修改任意机器时间

[user1@hadoop103 ~]$ sudo date -s "2017-9-11 11:11:11"

（3）十分钟后查看机器是否与时间服务器同步

[user1@hadoop103 ~]$ sudo date

说明：测试的时候可以将10分钟调整为1分钟，节省时间。