快速使用手册

## 目的

指在帮助用户快速搭建工具框架,利用一个小demo迅速熟悉工具功能.

## 安装VMware15

## 安装CentOS-7.5

## 虚拟机样机准备

准备一台原生虚拟机,最低配置: 4G+50G,修改静态ip,主机名,关闭防火墙,配置用户权限,

### 设置静态ip

查看ip ==> 设置静态ip ==> 重启网络 ==> 查看ip ==> window配置ping测试

[root@hdp102 ~]# ifconfig

[root@hdp102 ~]# vim /etc/sysconfig/network-scripts/ifcfg-ens33

YPE="Ethernet"

PROXY\_METHOD="none"

BROWSER\_ONLY="no"

BOOTPROTO="static"

DEFROUTE="yes"

IPV4\_FAILURE\_FATAL="no"

IPV6INIT="yes"

IPV6\_AUTOCONF="yes"

IPV6\_DEFROUTE="yes"

IPV6\_FAILURE\_FATAL="no"

IPV6\_ADDR\_GEN\_MODE="stable-privacy"

NAME="ens33"

UUID="63c8d575-6aa5-446a-b193-89ed05dbfe1c"

DEVICE="ens33"

ONBOOT="yes"

#IP地址

IPADDR=192.168.10.20

#网关

GATEWAY=192.168.10.2

#域名解析器

DNS1=192.168.10.2

[root@hdp102 ~]# service network restart

Window 执行测试: ping 192.168.10.20

### 配置主机名

查看主机名 ==> 修改主机名称 ==> 重启生效 ==> 查看验证 ==> window下ping主机名验证

[root@hdp102 ~]# hostname

[root@hdp102 ~]# vi /etc/hostname

### 修改hosts映射文件

添加Linux和window映射信息

[root@hdp100 ~]# vim /etc/hosts

Window路径: C:\Windows\System32\drivers\etc路径

### 关闭linux防火墙

关闭防火墙服务和关闭自启

[root@hdp100 ~]# systemctl stop firewalld

[root@hdp100 ~]# systemctl disable firewalld

### 配置hugh普通用户root权限

[root@hdp100 ~]# vim /etc/sudoers

修改/etc/sudoers文件，找到下面一行（102行），在%wheel下面添加一行：

## Allow root to run any commands anywhere

root ALL=(ALL) ALL

%wheel ALL=(ALL) ALL

hugh ALL=(ALL) NOPASSWD: ALL

### 重启生效

## HDP集群环境快速搭建

* 从样机克隆三台虚拟机,配置ip,修改主机名; 检查映射,用户,防火墙
* 配置ssh免密,安装jdk
* 集群脚本

### 虚拟机集群准备

克隆3台虚拟机,配置ip, 和 主机名,重启

[root@hdp100 ~]# vim /etc/sysconfig/network-scripts/ifcfg-ens33

[root@hdp100 ~]# hostnamectl --static set-hostname hdp102

[root@hdp100 ~]# vim /etc/hosts

[root@hdp100 ~]# vim /etc/sudoers

[root@hdp100 ~]# systemctl status firewalld

[root@hdp100 ~]# ifconfig

验证: 虚拟机之间, 互相ping测试

### 配置ssh免密登录

如果家目录不存在.ssh目录,家目录执行 ssh localhost

配置hdp102三台服务器免密登录。进入.ssh目录

1）生成公钥和私钥：

[hugh@hdp102 .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）

2）将公钥拷贝到要免密登录的目标机器上

[hugh@hdp102 .ssh]$ ssh-copy-id hdp102

[hugh@hdp102 .ssh]$ ssh-copy-id hdp103

[hugh@hdp102 .ssh]$ ssh-copy-id hdp104

**输入yes,密码**

3）配置hdp102三台服务器免密登录

### 安装JDK1.8

#### 卸载原有得

[hugh@hdp104 ~]$ rpm -qa | grep openjdk

java-1.8.0-openjdk-headless-1.8.0.161-2.b14.el7.x86\_64

java-1.7.0-openjdk-1.7.0.171-2.6.13.2.el7.x86\_64

java-1.8.0-openjdk-1.8.0.161-2.b14.el7.x86\_64

java-1.7.0-openjdk-headless-1.7.0.171-2.6.13.2.el7.x86\_64

[hugh@hdp104 ~]$ sudo rpm -e --nodeps java-1.8.0-openjdk-headless-1.8.0.161-2.b14.el7.x86\_64

[hugh@hdp104 ~]$ sudo rpm -e --nodeps java-1.7.0-openjdk-1.7.0.171-2.6.13.2.el7.x86\_64

[hugh@hdp104 ~]$ sudo rpm -e --nodeps java-1.8.0-openjdk-1.8.0.161-2.b14.el7.x86\_64

[hugh@hdp104 ~]$ sudo rpm -e --nodeps java-1.7.0-openjdk-headless-1.7.0.171-2.6.13.2.el7.x86\_64

[hugh@hdp104 ~]$ rpm -qa | grep openjdk

[hugh@hdp104 ~]$

#### 安装

[hugh@hdp102 software]$ tar -zxvf jdk-8u212-linux-x64.tar.gz -C /opt/module/

#### **配置环境变量:**

在profile.d 目录下创建文件存放自己变量

[root@hdp102 profile.d]# vim /etc/profile.d/my\_env.sh

#JAVA\_HOME

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

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

#### 配置生效和验证:

[hugh@hdp102 jdk1.8.0\_212]$ source /etc/profie

[hugh@hdp102 software]$ java -version

### 脚本工具

#### xsync 脚本

1）在/root目录下创建bin目录，并在bin目录下创建文件xsync，文件内容如下：

[hugh@hdp102 ~]$ mkdir bin

[hugh@hdp102 ~]$ cd bin/

[hugh@hdp102 ~]$ vi xsync

在该文件中编写如下代码

#!/bin/bash

#1 获取输入参数个数，如果没有参数，直接退出

pcount=$#

if((pcount==0)); then

echo no args;

exit;

fi

#2 获取文件名称

p1=$1

fname=`basename $p1`

echo fname=$fname

#3 获取上级目录到绝对路径

pdir=`cd -P $(dirname $p1); pwd`

echo pdir=$pdir

#4 获取当前用户名称

user=`whoami`

#5 循环

for((host=103; host<105; host++)); do

echo ------------------- hdp3-$host --------------

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

done

2）修改脚本 xsync 具有执行权限

[root@hadoop102 bin]$ chmod 777 xsync

#### xcall脚本

#! /bin/bash

for i in hdp102 hdp103 hdp104

do

echo --------- $i ----------

ssh $i "$\*"

done

#### mydate脚本

#!/bin/bash

log\_date=$1

for i in hdp102 hdp103 hdp104

do

ssh -t $i "sudo date -s $log\_date"

done

### Hadoop集群环境搭建验证

hadoop集群安装环境搭建成功(jdk，ssh，tool)

## HDP3.0 集群部署

### 集群部署规划

|  |  |  |  |
| --- | --- | --- | --- |
|  | hdp102 | hdp103 | hdp104 |
| HDFS | NameNode  DataNode | DataNode | SecondaryNameNode  DataNode |
| YARN | NodeManager | ResourceManager  NodeManager | NodeManager |

### 集群部署和环境配置

[hugh@hdp102 module]$ tar -zxvf hadoop-3.1.3.tar.gz -C /opt/module/

[hugh@hdp102 module]$ vim /etc/profile.d/my\_env.sh

export HADOOP\_HOME=/opt/module/hadoop-3.1.3

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

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

[hugh@hdp102 module]$ xsync hadoop-3.1.3/

[root@hdp102 profile.d]# xsync my\_env.sh

[hugh@hdp102 module]$ xcall source /etc/profile

[hugh@hdp102 module]$ xcall hadoop version

### 集群核心配置

[hugh@hdp102 hadoop]$ vim core-site.xml

<?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://hdp102:9820</value>

</property>

<!-- 指定hadoop数据的存储目录 -->

<property>

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

<value>/opt/module/hadoop-3.1.3/data</value>

</property>

<!-- 配置HDFS网页登录使用的静态用户为hugh -->

<property>

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

<value>hugh</value>

</property>

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

<property>

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

<value>\*</value>

</property>

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

<property>

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

<value>\*</value>

</property>

<!-- 配置该hugh(superUser)允许通过代理的用户-->

<property>

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

<value>\*</value>

</property>

</configuration>

[hugh@hdp3-102 hadoop]$ vim hdfs-site.xml

<?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>hdp102:9870</value>

</property>

<!-- 2nn web端访问地址-->

<property>

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

<value>hdp104:9868</value>

</property>

<!-- 测试环境指定HDFS副本的数量1 -->

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

</configuration>

[hugh@hdp3-102 hadoop]$ vim yarn-site.xml

<?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>hdp103</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>

[hugh@hdp3-102 hadoop]$ vim mapred-site.xml

<?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>

[hugh@hdp102 hadoop]$ vim workers

hdp102

hdp103

hdp104

### HDP集群初始化和测试

分发配置文件和初始化

[hugh@hdp102 etc]$ xsync hadoop/

格式化NameNode（hdp3-102）

[hugh@hdp102 hadoop-3.1.3]$ bin/hdfs namenode -format

在任意节点启动HDFS（hdp3-102）

[hugh@hdp102 hadoop-3.1.3]$ sbin/start-dfs.sh

在配置了ResourceManager的节点（hdp3-103）启动YARN

[hugh@hdp103 hadoop-3.1.3]$ sbin/start-yarn.sh

WC测试

创建文件

[hugh@hdp102 ~]$ vim wc.csv

上传文件到hdfs

[hugh@hdp102 ~]$ hdfs dfs -put mydata/wc.txt /user/hugh/input

查看hdfs文件

[hugh@hdp102 ~]$ hdfs dfs -cat /user/hugh/input/wc.txt

测试包测试wordcount

[hugh@hdp102 hadoop-3.1.3]$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.1.3.jar wordcount /user/hugh/input/ /user/hugh/output

查看计算结果

[hugh@hdp102 hadoop-3.1.3]$ hdfs dfs -cat /user/hugh/output/\*

下载文件到本地

[hugh@hdp102 mydata]$ hdfs dfs -get /user/hugh/output/part-r-00000 ./

### 集群其他配置

#### 配置历史服务器

[hugh@hdp102 hadoop]$ vim mapred-site.xml

添加

<!-- 历史服务器端地址 -->

<property>

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

<value>hdp102:10020</value>

</property>

<!-- 历史服务器web端地址 -->

<property>

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

<value>hdp102:19888</value>

</property>

#### 配置日志的聚集

[hugh@hdp102 hadoop]$ vim yarn-site.xml

<!-- 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://hdp102:19888/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>

配置HDP3.0支持LZO压缩

将编译好后的hadoop-lzo-0.4.20.jar 放入hadoop-3.1.3/share/hadoop/common/

[hugh@hdp102software]$cp hadoop-lzo-0.4.20.jar /opt/module/hadoop-3.1.3/share/hadoop/common/

[hugh@hdp102software]$ls /opt/module/hadoop-3.1.3/share/hadoop/common/ | grep hadoop

hadoop-common-3.1.3.jar

hadoop-common-3.1.3-tests.jar

hadoop-kms-3.1.3.jar

hadoop-lzo-0.4.20.jar

hadoop-nfs-3.1.3.jar

同步 jar包

[hugh@hdp102 common]$ xsync hadoop-lzo-0.4.20.jar

[hugh@hdp102 common]$ xcall ls /opt/module/hadoop-3.1.3/share/hadoop/common/hadoop-lzo-0.4.20.jar

--------- hdp102 ----------

/opt/module/hadoop-3.1.3/share/hadoop/common/hadoop-lzo-0.4.20.jar

--------- hdp103 ----------

/opt/module/hadoop-3.1.3/share/hadoop/common/hadoop-lzo-0.4.20.jar

--------- hdp104 ----------

/opt/module/hadoop-3.1.3/share/hadoop/common/hadoop-lzo-0.4.20.jar

core-site.xml 配置文件支持LZO

[hugh@hdp102 hadoop]$ vim core-site.xml

<property>

<name>io.compression.codecs</name>

<value>

org.apache.hadoop.io.compress.GzipCodec,

org.apache.hadoop.io.compress.DefaultCodec,

org.apache.hadoop.io.compress.BZip2Codec,

org.apache.hadoop.io.compress.SnappyCodec,

com.hadoop.compression.lzo.LzoCodec,

com.hadoop.compression.lzo.LzopCodec

</value>

</property>

<property>

<name>io.compression.codec.lzo.class</name>

<value>com.hadoop.compression.lzo.LzoCodec</value>

</property>

分发: core-site.xml

重启集群验证支持LZO压缩

1.hdfs创建一个新目录

[hugh@hdp102 ~]$ hadoop fs -mkdir -p /user/hugh/input2

2.上传一个文件input2目录

[hugh@hdp102 mydata]$ hadoop fs -put wc.txt /user/hugh/input2

3.执行命令将文件压缩输出到output2目录

hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.1.3.jar wordcount \

-Dmapreduce.output.fileoutputformat.compress=true \

-Dmapreduce.output.fileoutputformat.compress.codec=com.hadoop.compression.lzo.LzopCodec \

/user/hugh/input2 /user/hugh/output2

4.查看hdfs输出目录.

测试LZO支持切片

LZO必须对文件创建索引,才能支持切片功能

1.将测试lzO数据上传

[hugh@hdp102 software]$ hadoop fs -mkdir -p /user/hugh/input3

[hugh@hdp102 software]$ hadoop fs -put bigtable.lzo /user/hugh/input3

2.执行测试,使用LZO压缩文件

hadoop jar /opt/module/hadoop-3.1.3/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.1.3.jar wordcount \

-Dmapreduce.job.inputformat.class=com.hadoop.mapreduce.LzoTextInputFormat \

/user/hugh/input3 /user/hugh/output3

4.检查发现切片失效

2020-09-26 00:40:43,614 INFO mapreduce.JobSubmitter: number of splits:1

5.对测试数据创建索引

hadoop jar /opt/module/hadoop-3.1.3/share/hadoop/common/hadoop-lzo-0.4.20.jar \

com.hadoop.compression.lzo.DistributedLzoIndexer /user/hugh/input3/bigtable.lzo

6.查看hdfs目录下/user/hugh/input3,生产一个索引文件bigtable.lzo.index

7.再执行命令测试,查看切片个数

hadoop jar /opt/module/hadoop-3.1.3/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.1.3.jar wordcount \

-Dmapreduce.job.inputformat.class=com.hadoop.mapreduce.LzoTextInputFormat \

/user/hugh/input3 /user/hugh/output4

8.生产2个切片,LZO支持切片验证成功

2020-09-26 00:49:47,415 INFO mapreduce.JobSubmitter: number of splits:2

## 日志模拟数据

将application.properties、gmall2020-mock-log-2020-04-01.jar、path2.json上传到hadoop102的/opt/applog目录下

[hugh@hdp102 applog]$ sudo mkdir applog

[hugh@hdp102 applog]$ sudo chown -R hugh: applog/

[atguigu@hadoop102 applog]$ ls

application.properties gmall2020-mock-log-2020-04-01.jar path2.json

配置文件application.properteis文件,修改业务日期

执行命令生产数据

[hugh@hdp102 applog]$ java -jar gmall2020-mock-log-2020-04-01.jar

查看数据 在/opt/module/applog/log目录下查看生成日志

分发日志模拟数据applog到所有节点

[hugh@hdp102 module]$ xsync applog/

数据生产脚本 mylog.sh

#!/bin/bash

for i in hdp102 hdp103; do

echo "========== $i loading ... =========="

ssh $i "cd /opt/module/applog/; java -jar gmall2020-mock-log-2020-04-01.jar >/dev/null 2>&1 &"

done

[hugh@hdp102 bin]$ chmod 775 mylog.sh

执行测试

[hugh@hdp102 module]$ mylog.sh

查看验证

[hugh@hdp102 log]$ more /opt/module/applog/log/app.2020-09-25.log