HDFS

壹、大数据及HDFS简述

一、大数据简述

在互联技术飞速发展过程中，越来越多的人融入互联网。也就意味着各个平台的用户所产生的数据也越来越多，可以说是爆炸式的增长，以前传统的数据处理的技术已经无法胜任了。比如淘宝，每天的活跃用户量是很大的一个数目。马云之前说过某个省份的女性bar的size最小问题，不管是玩笑还什么，细想而知，基于淘宝用户的购物记录确实可以分析出来。

对企业的用户数据进行分析，可以知道公司产品的运营情况，比方说一个APP的用户每天登陆了几乎都没有什么实质性的操作，那就说明这个玩意儿已经快凉了，程序员赶快可以跑路了。

每个人登录哪些电商网站的首页都是不一样，这后面就是根据用户的近期浏览或者关注的，根据这些来生成推送每个人关注的商品。

对于这些海量的数据的处理分析所诞生的技术，也就是大数据。

对于这些数据两个核心点，一个如何存储，另一个就是怎么使用。

相关的技术：

存储框架：

HDFS——分布式文件存储系统（HADOOP生态中的存储框架）

HBASE——分布式数据库系统

KAFKA——分布式消息缓存系统(实时流式数据处理场景中应用广泛)

运算框架：（要解决的核心问题就是帮用户将处理逻辑在很多机器上并行）

MAPREDUCE—— 分布式计算（HADOOP中的运算框架）

SPARK —— 离线批处理/实时流式计算

STORM —— 实时流式计算

其他框架：

HIVE —— 数据仓库工具：可以接收sql，翻译成mapreduce或者spark程序运行

FLUME——数据采集

SQOOP——数据迁移

ELASTIC SEARCH —— 分布式的搜索引擎

.......

.......

.......

1. HDFS简述

hadoop中有3个核心组件：

分布式文件系统：HDFS —— 实现将文件分布式存储在很多的服务器上

分布式运算编程框架：MAPREDUCE —— 实现在很多机器上分布式并行运算

分布式资源调度平台：YARN —— 帮用户调度大量的mapreduce程序，并合理分配运算资源

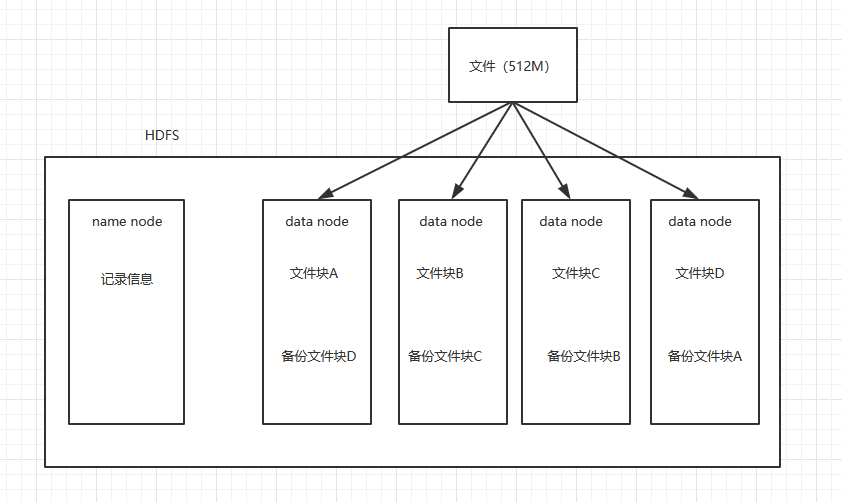
HDFS---分布式文件系统，相当于就是一个目录树，一层一层的，这个是虚拟的出来一个结构，由HDFS管理，并不能实际看见，只能通过客户端去访问的时候可以看见这些结构。

1.一个数据在HDFS上面存储的时候会根据大小来进行分块，被分块之后，存放在多个机器上面（data node），比如一个512M的文件，如果按照128M来分，就会被分成4块，然后存储到4个节点上。

2.一般来说为了保证数据的高可用，我们会把同一个数据块备份到不同的节点上面，某个节点挂了，还可以在其他节点上面找到数据。意思就说数据块A既会在A机器上存储，也会在机器B上面存储一份，甚至更多的备份。

3.分块存储之后怎么直到数据存在哪些机器上呢，这个时候就需要一个管理者来记录这些数据信息（name node）

也就是说一个HDFS系统是由name node服务器和多个data node服务组成



贰、HDFS集群搭建

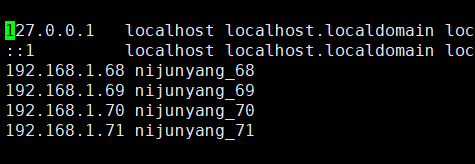
一、准备工作

1.准备几台机器，我这里使用VMware准备了四台机器，一个name node，三个data node。

VMware安装虚拟机：<https://www.cnblogs.com/nijunyang/p/12001312.html>

1. Hadoop生态几乎都是用的java开发的，因此四台机器还需要安装JDK。
2. 集群内主机域名映射，将四台机器的IP和主机名映射分别写到hosts文件中(切记主机名不要带非法字符，图片中的下划线”\_”请忽略)

vim /etc/hosts



配好一个之后可以直接将这个复制到其他机器上面去：

scp /etc/hosts nijunyang69:/etc/

scp /etc/hosts nijunyang70:/etc/

scp /etc/hosts nijunyang71:/etc/

二、hdfs集群安装

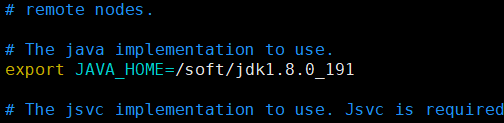
1.下载hadoop安装包到linux服务器上面，并进行解压，我这里使用的的2.8.5，

tar -zxvf hadoop-2.8.5.tar.gz

1. 指定java环境变量：

hadoop-2.8.5/etc/hadoop/hadoop-env.sh 文件中指定java环境变量：

export JAVA\_HOME=/soft/jdk1.8.0\_191



3.配置核心参数：

1)指定hadoop的默认文件系统为：hdfs

2)指定hdfs的namenode节点为哪台机器

修改/etc/hadoop/core-site.xml 指定hadoop默认文件系统为hdfs,并且指定name node

<configuration>

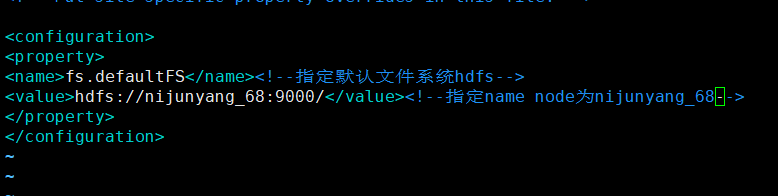
<property>

<name>fs.defaultFS</name><!--指定默认文件系统hdfs-->

<value>hdfs://nijunyang68:9000/</value><!--指定name node为nijunyang\_68-->

</property>

</configuration>



1. 指定namenode软件存储元数据的本地目录
2. 指定datanode软件存放文件块的本地目录

修改/etc/hadoop/hdfs-site.xml

<configuration>

<property>

<name>dfs.namenode.name.dir</name>

<value>/hadoop/hdpdata/name/</value>

</property>

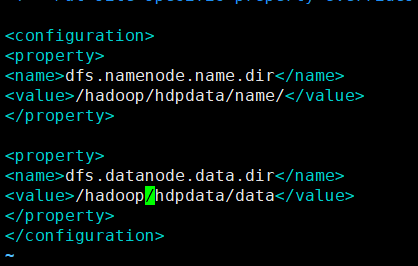
<property>

<name>dfs.datanode.data.dir</name>

<value>/hadoop/hdpdata/data</value>

</property>

</configuration>

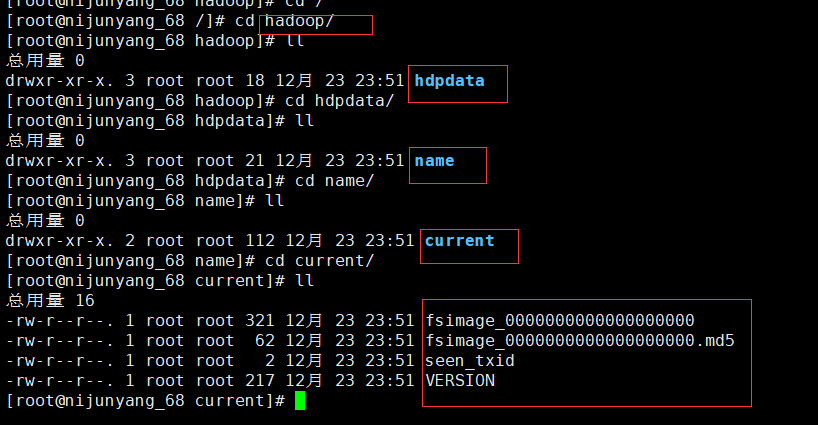


Namenode 和datanode分开，因为一个节点既可以起namenode也可以起datanode

每台机器都执行同样的操作，配置好上述配置，可以使用scp -r /soft/hadoop-2.8.5 nijunyang69:/soft 这个命令将第一台机器配置好的全部打包拷贝到另外机器上面去。

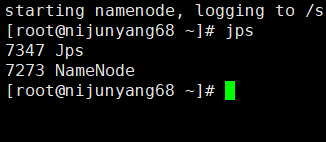
1. 配置hadoop环境变量
2. 初始化namenode：hadoop namenode -format

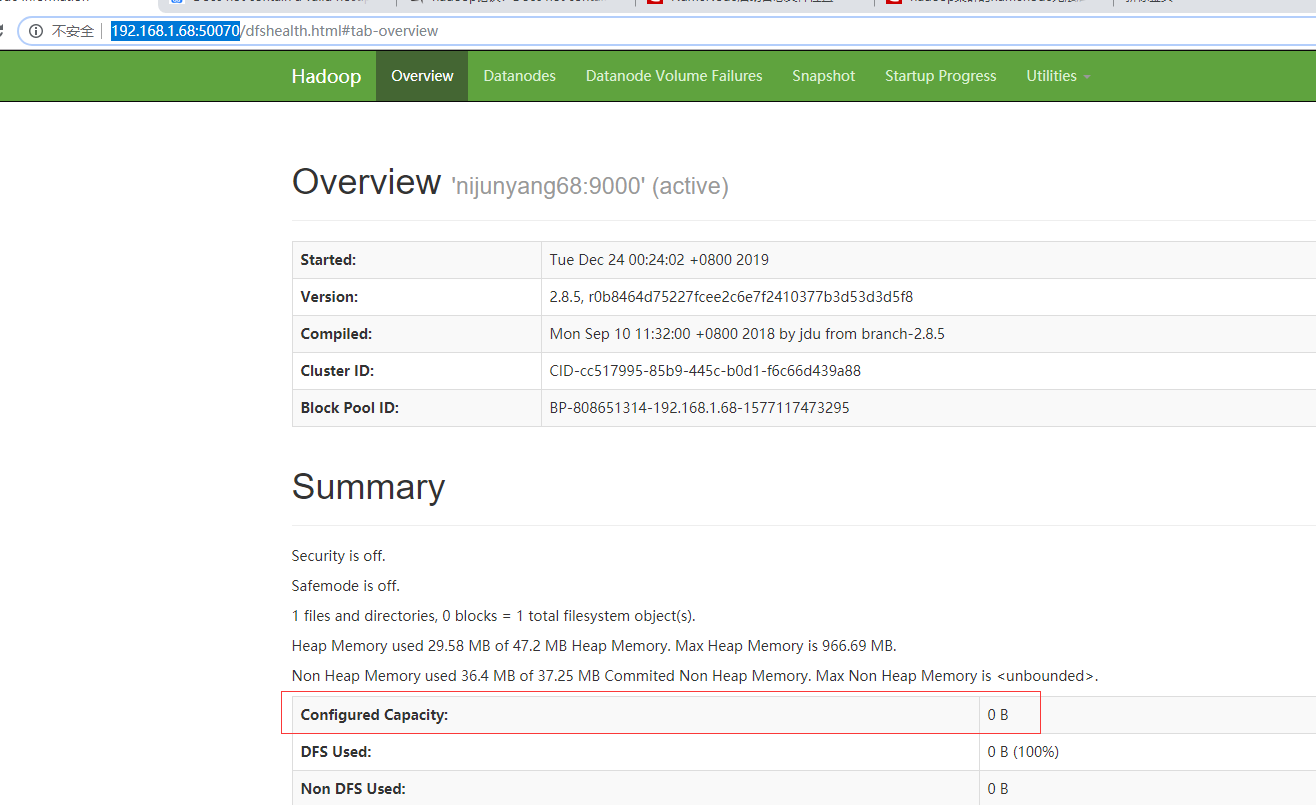
这时我们设置的namenode数据目录下面就会初始化出来对应的文件夹



1. 启动namenode：在设置的namenode上面：hadoop-daemon.sh start namenode

Jps查看可以看到一个namenode的java进程，同时通过默认的50070端口可以进行web访问

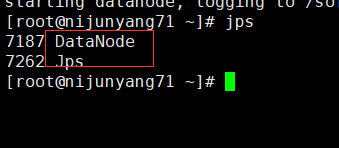


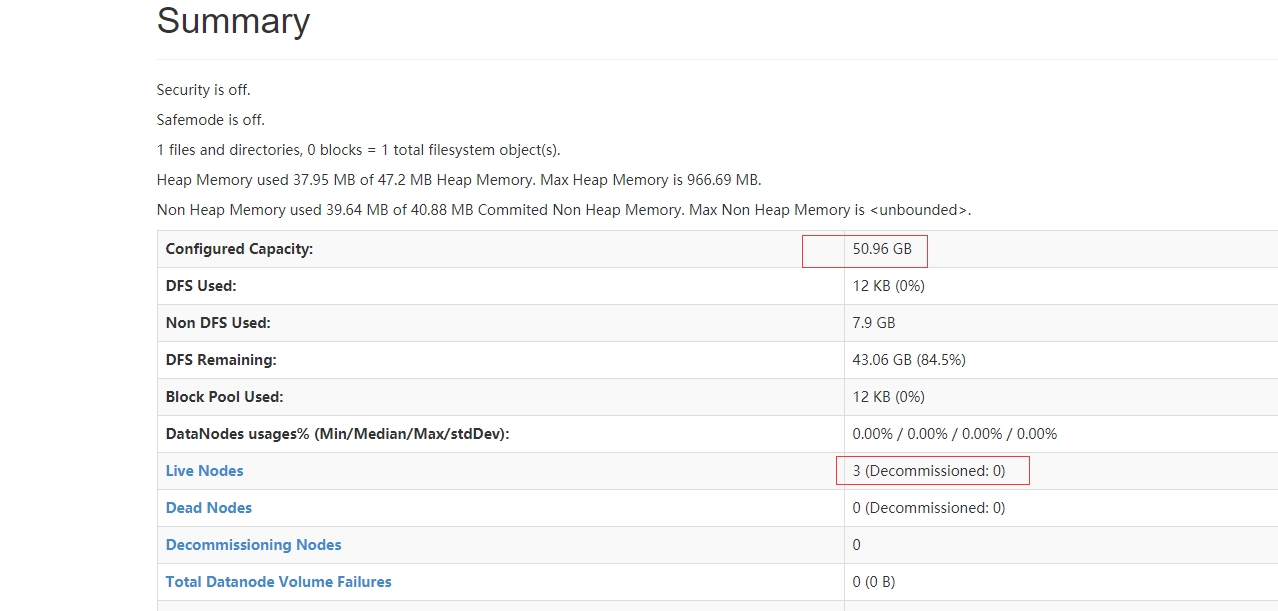


可以看到现在HDFS的容量还是0，因为我还没有启动datanode

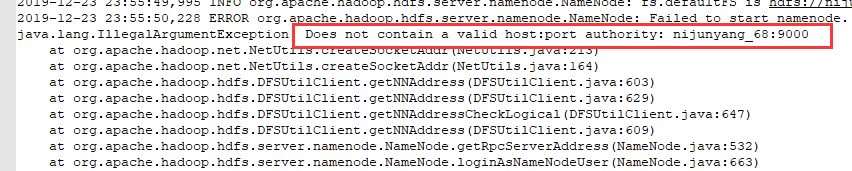
1. 依次启动datanode：hadoop-daemon.sh start datanode

同样可以看到一个datanode的java进程启动了，再看web页面这个时候的hdfs容量大小差不多就是三个datanode之和了。





至此整个hdfs集群基本就搭建完毕了，中间的一个小插曲，主机名一定不要带”.” “/” “\_”等特殊符号，否则启动无服务的时候可能报错 无法启动



1. 脚本一键启动

配置SSH免密登录，在任意一台机器配置SSH免密链接其他机器

1)生成秘钥：ssh-keygen

2)设置免密连接：

ssh-copy-id nijunyang68

ssh-copy-id nijunyang69

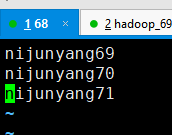
ssh-copy-id nijunyang70

ssh-copy-id nijunyang71

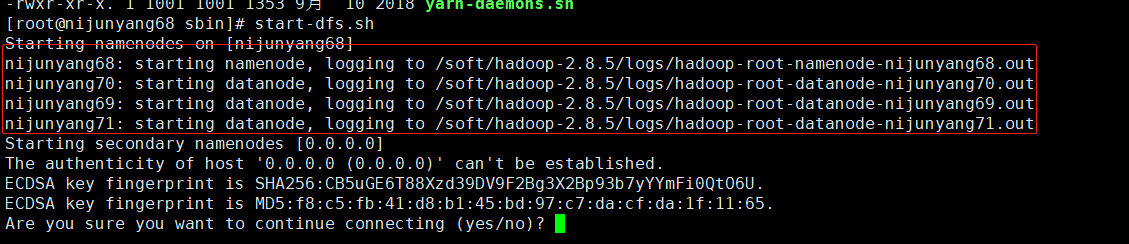
设置好之后就可以当前机器直接通过SSH连接其他机器，不需要输入密码

1. 修改文件hadoop-2.8.5/etc/hadoop/etc/hadoop/slaves，加入需要启动的namenode

默认有个本机。如果不需要再本机启动namenode就把本机删掉



1. 现在就可以sbin目录下的集群启动脚本/停止脚本：start-dfs.sh/stop-dfs.sh



虽然集群起来了，但是还有个Starting secondary namenodes 启动在本机上，这个时候我最好去将secondary namenodes配置到另外的机器上面去，修改之前的/etc/hadoop/hdfs-site.xml，加入secondary namenodes的配置：

<property>

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

<value>nijunyang69:50090</value>

</property>

----------------------------------------------------------------

叁、HDFS客户端操作及java操作

Hdfs的备份数量和切块大小都是可以配置的，默认是备份3，切块大小默认128M

文件的切块大小和存储的副本数量，都是由客户端决定！

所谓的由客户端决定，是通过配置参数来定的

hdfs的客户端会读以下两个参数，来决定切块大小、副本数量：

切块大小的参数： dfs.blocksize

副本数量的参数： dfs.replication

因此我们只需要在客户端的机器上面hdfs-site.xml中进行配置：

*<property>*

*<name>dfs.blocksize</name>*

*<value>64m</value>*

*</property>*

*<property>*

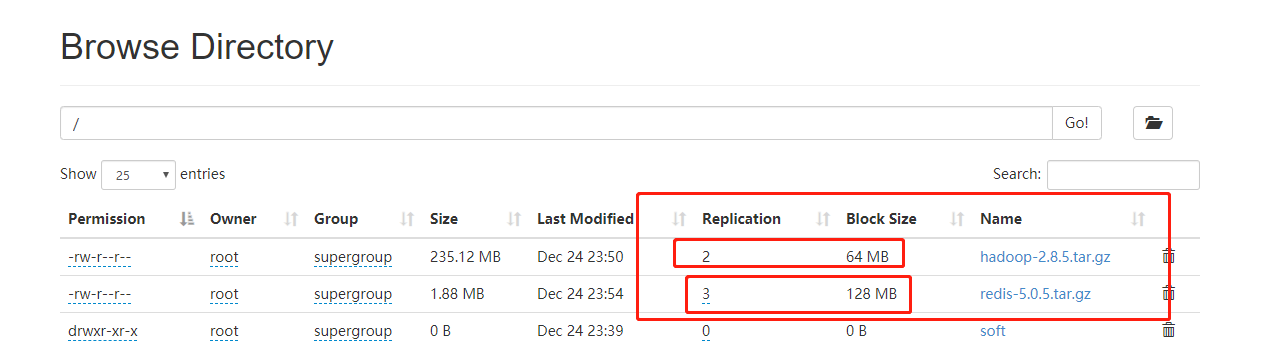
*<name>dfs.replication</name>*

*<value>2</value>*

*</property>*

我们在两个客户端进行上传，一个客户端修改为上述配置，查看上传文件信息

可以看见一个文件是3和128m，另外一个是2和64m



1. 客户端命令行操作
2. 上传文件到hdfs中

hadoop fs -put /本地文件 /aaa

1. 下载文件到客户端本地磁盘

hadoop fs -get /hdfs中的路径 /本地磁盘目录

1. 在hdfs中创建文件夹

hadoop fs -mkdir -p /aaa/xxx

1. 移动hdfs中的文件（更名）

hadoop fs -mv /hdfs的路径1 /hdfs的另一个路径2

复制hdfs中的文件到hdfs的另一个目录

hadoop fs -cp /hdfs路径\_1 /hdfs路径\_2

1. 删除hdfs中的文件或文件夹

hadoop fs -rm -r /aaa

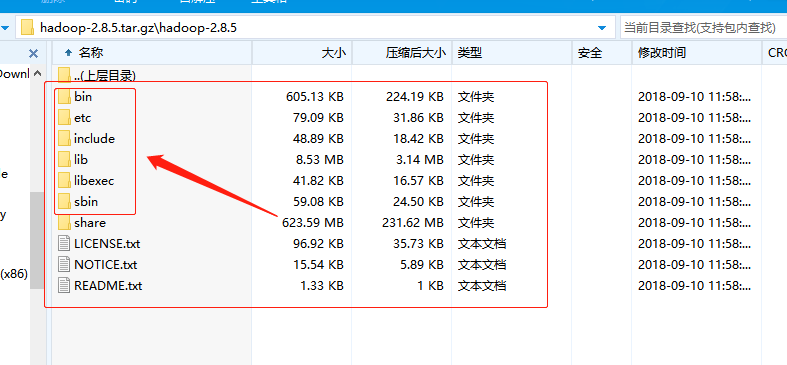
1. 查看hdfs中的文本文件内容

hadoop fs -cat /demo.txt

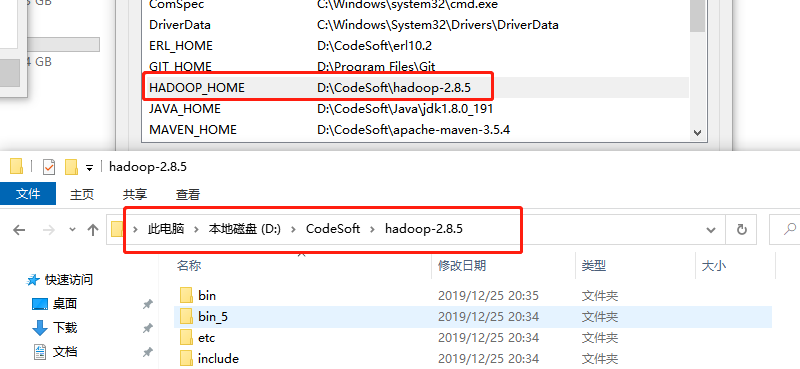
hadoop fs -tail -f /demo.txt

更多指令https://www.cnblogs.com/houkai/p/3848089.html

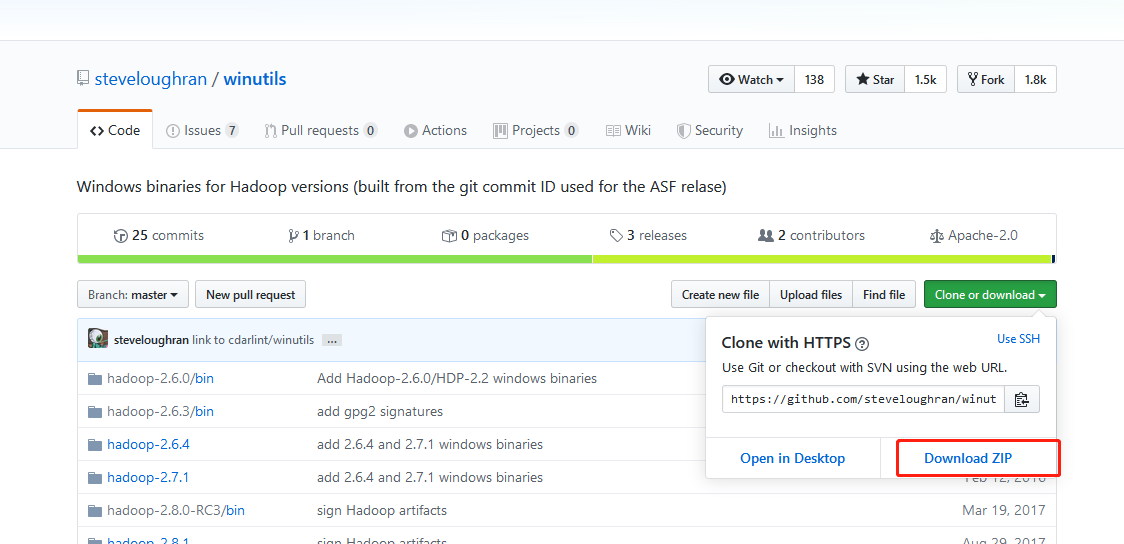
1. java操作客户端
2. 首先需要搭建本地开发环境，因为本地启动应用的时候会从hadoop里面回去调用c的函数操作本地文件系统，因此我们需要在本地配置hadoop的环境信息。将hadoop压缩包解压出来，留下脚本所在的目录就可以了，其他的一些目录可以丢掉



配置hadoop环境变量，将bin目录的里面的文件替换问windows的脚本文件。



可以去这儿下载：[https://github.com/steveloughran/winutils](https://github.com/steveloughran/winutils" \t "https://www.cnblogs.com/chevin/p/_blank)



这是别人已经编译好的windows脚本，换到自己的bin目录里面去就行了。当然也可以自己拿到hadoop去编译。

1. 准备完毕就可以导包撸代码了

肆、HDFS工作机制

一、namenode管理元数据

元数据：hdfs的目录结构以及文件文件的块信息（块副本数量，存放位置等）。

Namenode把元数据存在内存中，以方便改动，同时也会在某个时间点上面将其写到磁盘上（fsimage镜像文件）。同时还会把引起元数据变化的操作记录在edits日志文件中。重新启动或者是服务挂了的时候，也可以从磁盘文件和日志文件中还原数据。

久而久之edits文件会越来越大，如果重启可能会花费很长时间。这个时候就有了Sencondary NameNode，Secondary NameNode定期合并fsimage和edits日志，把edits日志文件大小控制在一个限度下。因为内存需求和NameNode差不多，所以一般都是另外配置一台机器。Sencondary NameNode整合完毕会将新的fsimages传回给namenode。这个过程叫做checkpoint。那么什么执行这个操作呢。默认的时一个小时，也可以通过参数配置

fs.checkpoint.period 设置checkpoint间隔时间

fs.checkpoint.size 规定edits文件的最大值，一旦超过这个值则强制checkpoint，不管是否到达最大时间间隔。

dfs.namenode.checkpoint.dir Sencondary NameNode保存数据的位置

这个过程和redis的混合持久化有点点类似，RDB快照和AOF重放

二、读写文件

<https://www.w3cschool.cn/hadoop/2myl1p37.html>

<https://www.w3cschool.cn/hadoop/8rkl1p3a.html>

-----------------------------------------------------

Yarn

简述：

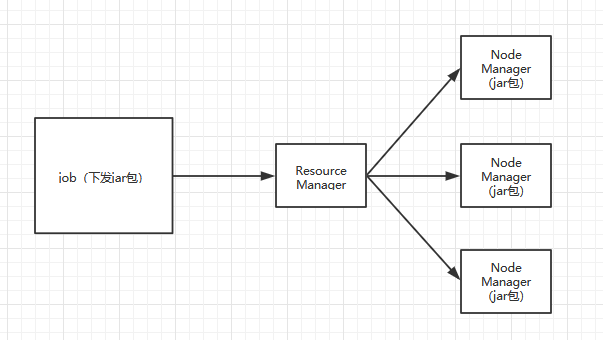
Yarn是haddop体系中的分布式资源调度平台。

Yarn中有两大核心角色Resource Manager和Node Manager

Resource Manager负责接收用户提交的分布式计算程序/任务，并为其划分资源，管理监控各个Node Manager。

Node Manager 接收resoResource Manager分配过来的任务，并计算。

通俗一点说就是计算程序会被打成一个jar包，然后分配到每个node manager上面去，这样每个node manager 执行的代码都是一样，只是可能数据源不一样。



Yarn集群安装：

node manager在物理上应该跟data node部署在一起，方便数据的读取

Yarn的软件在hadoop里面的都是有的，就和hdfs一样，我们只需要去配置一下，然后启动就可以了

每台机器都对etc/hadoop/yarn-site.xml进行配置

<property><!--配置redource manager-->

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

<value>nijunyang68</value>

</property>

<property><!--配置mapreduce shuffle-->

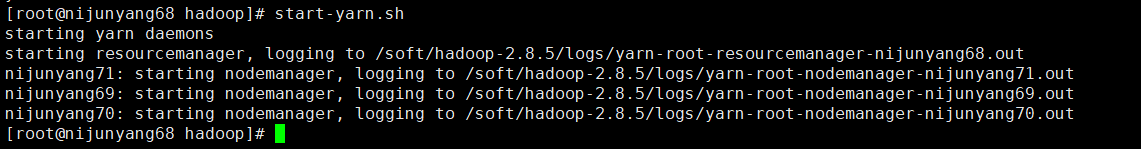
<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

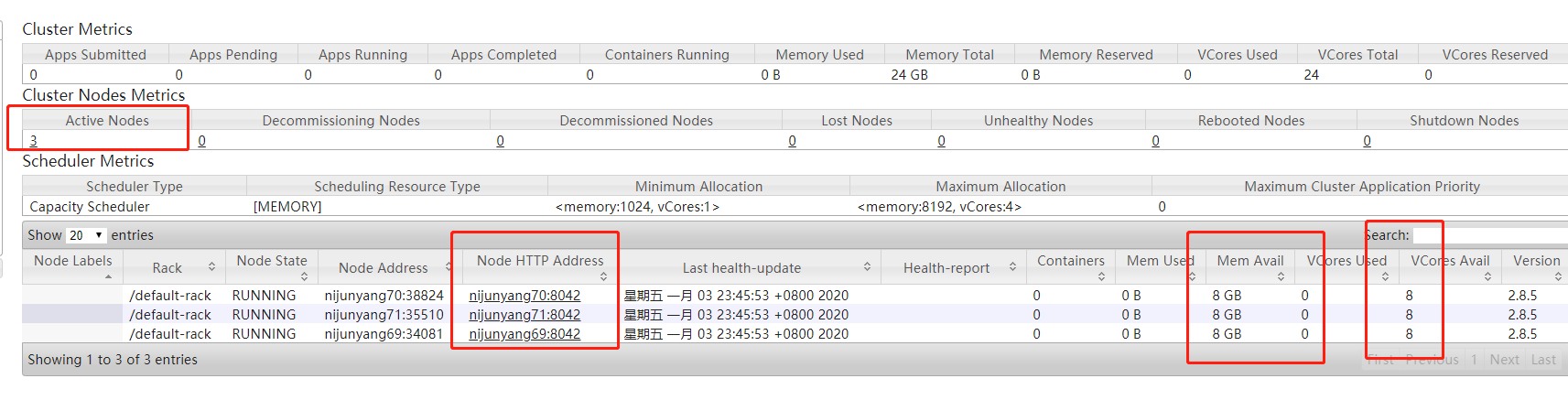
</property>

因为之前配置hdfs集群的时候已经在slaves中将集群IP的都配置进去了，所以现在只需要一键执行脚本就可以了：start-yarn.sh

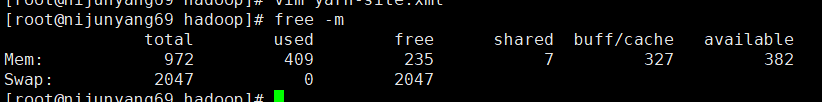
注意在哪台机器启动redource manager就在那儿执行这个脚本，上面的配置只是告诉集群的中机器谁是redource manager，所以执行这个脚本需要在配置中的那个机器上面去执行。从日志中也可以看见，resource manager是在本机启动的，node manager是在其他机器上面启动的



默认8088端口可以查看yarn控制台



说一点，上面显示内存大小是不对的，因为我们没有配置，都是使用的默认，并不是我机器的实际值，实际上我的虚拟机总共才1G的内存



配置详情：

<https://hadoop.apache.org/docs/stable/hadoop-yarn/hadoop-yarn-common/yarn-default.xml>

<property>

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

<value>200</value>

</property>

内存有个最小分配限制1024，如果小于1024，集群是无法启动的。

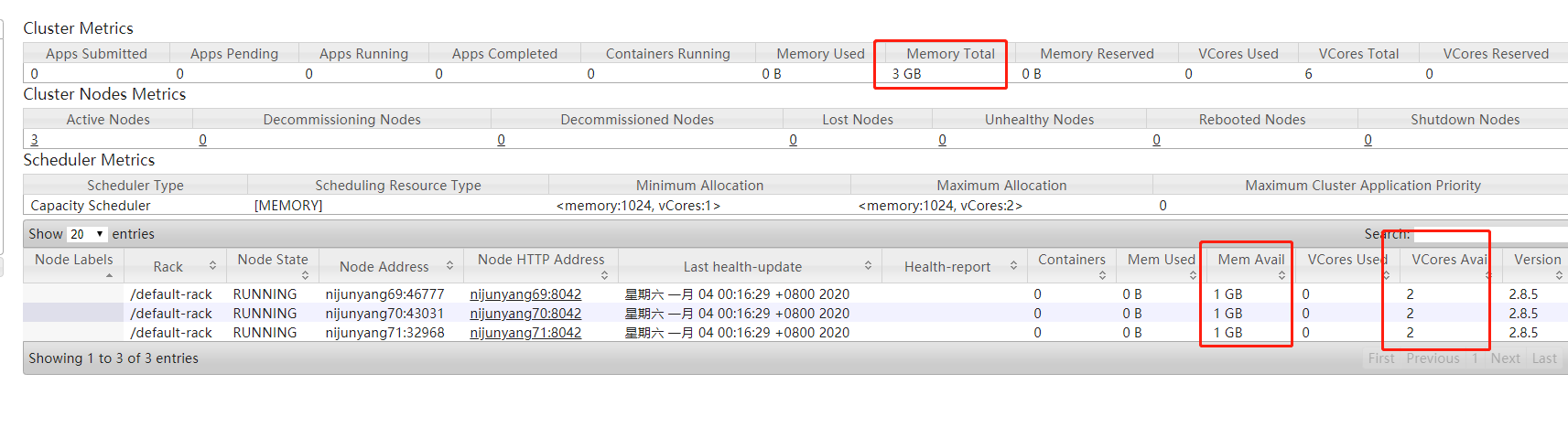
核数也并不是实际CPU的核数，我的虚拟机也才1核，这儿的意思是假如我内存200m，现在有一个任务需要100m内存，那么我这个机器就可以起两个任务，所以可以把核数配置成2，如果配置成那么久只能起一个任务。意思就是我CPU虽然是一核，但是我一个人100M，我200内存可以起两个任务，那么我CPU的运算能力就平均分给这两个任务。

<property>

<name>yarn.nodemanager.resource.cpu-vcores</name>

<value>2</value>

</property>



Map Reduce

壹、

Mapreduce是hadoop体系中的分布式计算框架。

操作主要分两个阶段

通过一个映射过程（Map任务），把一组键值对映射成一组新的键值对，然后通过归约操作（Reduce任务），对新键对中相同键的数据进行操作。

map: (K1, V1) → list(K2, V2)

combine: list(K2, V2) → （K2, list(V2)）

reduce: (K2, list(V2)) → list(K2, V3)

比如说统计单词出现次数。文本内容如下：

Think twice before acting

Think twice before acting

第一行数据（行号，内容）map之后：{（think，1），（twice ，1），（before ，1），（acting，1）}

第二行数据map之后：{（think，1），（twice ，1），（before ，1），（acting，1）}

之后进行合并：{（think，{1,1}），（twice ，{1,1}），（before ，{1,1}），（acting，{1,1}）}

归约操作：{（think，2），（twice ，2），（before ，2），（acting，2）}

Map之后还会有一个一个分发机制，key相同数据分发到同一个reduce任务，这样才能保证最后数据完整性。

上述过程在mapreduce框架中都已经封装完毕，我们自己只需要实现我们自己的map和reduce操作即可。

1.重写org.apache.hadoop.mapreduce.Mapper的map方法

2.重写org.apache.hadoop.mapreduce.Reducer的reduce方法

