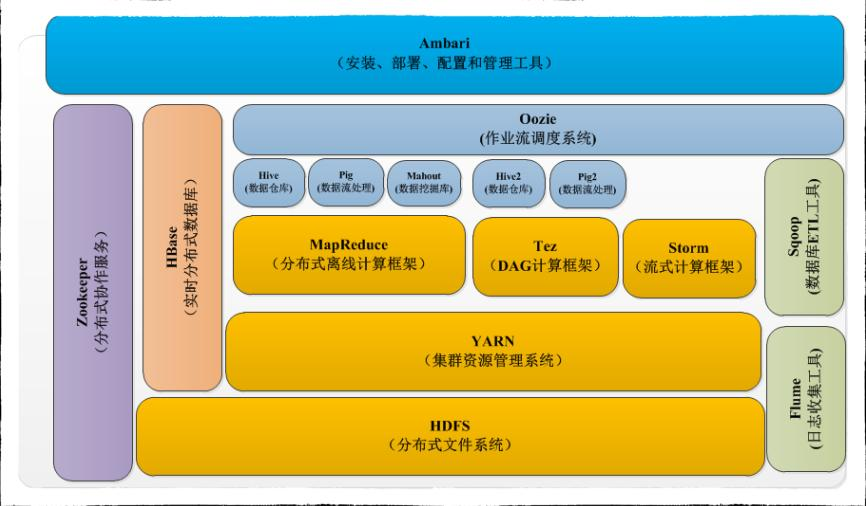
hadoop 2.7.6

MapReduce 分布式计算框架

YARN 集群资源管理系统

HDFS 分布式文件系统



• HDFS(Hadoop分布式文件系统)

• Mapreduce(分布式计算框架) （开发）

• Zookeeper(分布式协作服务)

• Hbase(分布式列存数据库) （开发）

• Hive(基于Hadoop的数据仓库) （开发）

• Sqoop(数据同步工具)

• Pig(基于Hadoop的数据流系统) （开发）

• Mahout(数据挖掘算法库) （开发）

• Flume(日志收集工具)

HDFS 角色及概念

– Client

– Namenode

– Secondarynode

– Datanode

• NameNode 不能做热备份，数据可能不是实时，真正有变更时需要等Secondarynode 定期推送获取最新信息

– Master节点,管理HDFS的名称空间和数据块映射信息,配置副本策略,处理所有客户端请求。

• Secondary NameNode

– 定期合并 fsimage（数据块的映射信息） 和fsedits（数据变更日志）,推送给NameNode

– 紧急情况下,可辅助恢复NameNode

• DataNode

– 数据存储节点,存储实际的数据

– 汇报存储信息给NameNode。

• Client

– 切分文件

– 访问HDFS

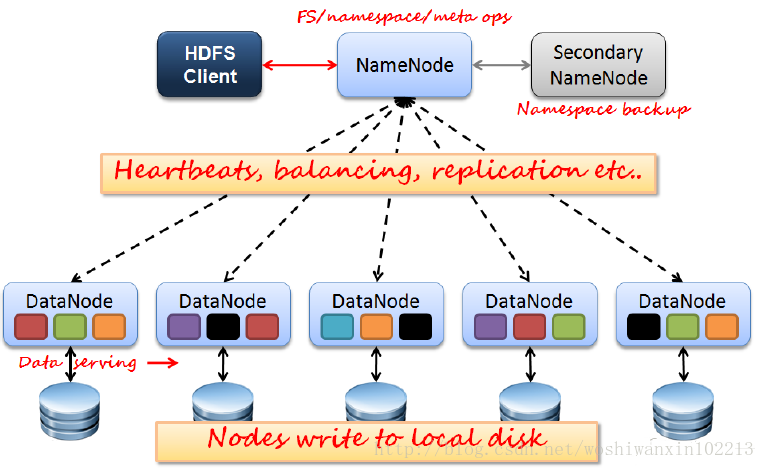
– 不NameNode交互,获取文件位置信息

– 不DataNode交互,读取和写入数据。

• Block

– 每块缺省64MB / 128MB大小 数据切块

– 每块可以多个副本（备份）存在不同机器上



1.HDFS Client 将数据进行切分，然后访问NameNode--->

2.NameNode（管理者角色）处理所有客户端请求，获取存储文件的位置,记录fsimage（数据块的映射信息）--->

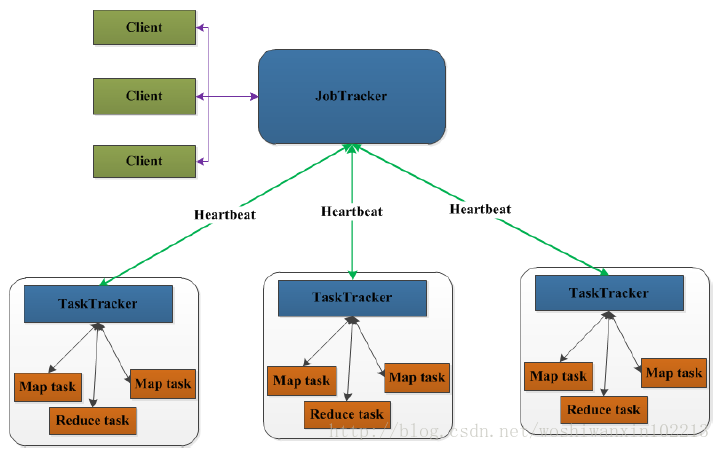
3.将数据存储到真正的DataNode中，存储完数据要回送消息给NameNode

Secondarynode 定期合并 fsimage（数据块的映射信息） 和fsedits（数据变更日志）,推送给NameNode

HDFS的空间约等于所有DataNode主机的空间之和

每个DataNode的空间最好一样大

MapReduce



• 角色和概念

– JobTracker

– TaskTracker

– Map Task （开发）

– Reducer Task （开发）

• JobTracker

– Master节点,只有一个

– 管理所有作业

– 作业/任务的监控、错误处理等

– 将任务分解成一系列任务,并分派给TaskTracker

• TaskTracker

– Slave节点,一般是多台

– 运行Map Task和Reduce Task

– 并与JobTracker交互,汇报任务状态。

• Map Task:解析每条数据记录,传递给用户编写的map(),幵执行,将输出结果写入本地磁盘(如果为map-only作业,直接写入HDFS)。

• Reducer Task:从Map Task的执行结果中,远程读取输入数据,对数据进行排序,将数据按照分组传递给用户编写的reduce凼数执行。

JobTracker（只有一个Master，管理节点）负责接收Client请求，分配给底层的TaskTracker（Slave）进行运算，TaskTracker计算完将计算结果返回给JobTracker，再返回给Client客户端

Yarn

• Yarn 角色

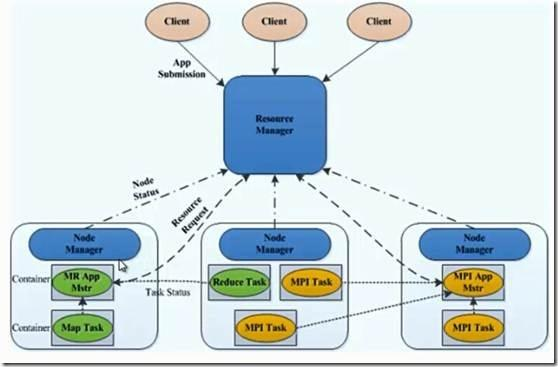
– Resourcemanager

– Nodemanager

– ApplicationMaster

– Container

– Client



• ResourceManager

– 处理客户端请求

– 启动 / 监控 ApplicationMaster

– 监控 NodeManager

– 资源分配不调度

• NodeManager 节点管理代理

– 单个节点上的资源管理

– 处理来自 ResourceManager 的命令

– 处理来自 ApplicationMaster 的命令

• Container 计算CPU 、内存等资源使用

– 对任务运行行环境的抽象,封装了 CPU 、内存等

– 多维资源以及环境变量、启动命令等任务运行相关的信息资源分配不调度

• ApplicationMaster ResourceManager启动的一个程序

– 数据切分

– 为应用程序申请资源,并分配给内部任务

– 任务监控不容错

• Client

– 用户不 YARN 交互的客户端程序

– 提交应用程序、监控应用程序状态,杀死应用程序等

• YARN 的核心思想

• 将 JobTracker 和 TaskTacker 进行分离,它由下面几大构成组件:

– ResourceManager 一个全局的资源管理器

– NodeManager 每个节点(RM)代理

– ApplicationMaster 表示每个应用

– 每一个 ApplicationMaster 有多个 Container 在NodeManager 上运行

Hadoop 安装配置

Hadoop 单机模式安装配置

1./etc/hostname node1

2./etc/hosts 192.168.1.11 node1

3.装包

yum -y install java-1.8.0-openjdk-devel

(安装 Openjdk 和 openjdk-devel) jps

hadoop-2.7.6.tar.gz

tar -xf hadoop-2.7.6.tar.gz

mv hadoop-2.7.6 /usr/local/hadoop

设置JAVA\_HOME环境变量

rpm -ql java-1.8.0-openjdk

vim /usr/local/hadoop/etc/hadoop/hadoop-env.sh

JAVA\_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.131-11.b12.el7.x86\_64/jre"

HADOOP\_CONF\_DIR="/usr/local/hadoop/etc/hadoop"

– 测试

cd /usr/local/hadoop/bin

./hadoop version

./bin/hadoop

统计热词

jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar --jar路径 wordcount input output

回车有说明

./bin/hadoop -----hadoop解释器

jar

./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar jar包路径

wordcount 方法

input 要统计的目录（可以多个目录）

output 统计完的结果放入的目录（输出目录要不能存在）

Hadoop 伪分布式 （可以做测试环境）

• 伪分布式

– 伪分布式的安装和完全分布式类似,但区别是所有角色安装在一台机器上,使用本地磁盘,一般生产环境都会使用完全分布式,伪分布式一般用来学习和测试方面的功能

– 伪分布式的配置和完全分布式配置类似

完全分布式

HDFS 分布式文件系统

• 系统规划

主机 角色 软件

192.168.1.10 namenode NameNode HDFS

SecondaryNameNode

ResourceManager YARN

192.168.1.11 Node1 DataNode HDFS

NodeManager YARN

192.168.1.12 Node2 DataNode HDFS

NodeManager YARN

192.168.1.13 node3 DataNode HDFS

NodeManager YARN

hostname 最好不要有特殊符号

namenode 存储映射信息 磁盘

datanode 存储真正数据 磁盘

resourcemanager 分配计算 cpu 内存 读取数据切分

nodemanager 计算 cpu 内存 坏了，整个集群数据全部丢失

1. 所有虚拟机都要配置/etc/hosts，四台之间要互相通信
2. 所有虚拟机装java-1.8.0

3.• 配置 SSH 信任关系(在namenode配置)

– 注意:不能出现要求输入 yes 的情况,namenode连每台机器都要能登录成功,包括本机!!!

vim /etc/ssh/ssh\_config

Host \*

GSSAPIAuthentication yes

StrictHostKeyChecking no

– ssh-keygen -b 2048 -t rsa -N '' -f key（文件名）

– ssh-copy-id -i ./key.pub [root@ip.xx.xx.xx](mailto:root@ip.xx.xx.xx)

4.• HDFS 完全分布式系统配置

– 环境配置文件 hadoop-env.sh

JAVA\_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.131-11.b12.el7.x86\_64/jre"

HADOOP\_CONF\_DIR="/usr/local/hadoop/etc/hadoop"

– Xml 文件配置格式

<property>

<name>关键字</name>

<value>变量值</value>

<description> 描述 </description>

</property>

官方文档

<http://hadoop.apache.org/>

name value（默认值，可以修改） description

– 核心配置文件 core-site.xml

– fs.defaultFS 文件系统配置参数

– hadoop.tmp.dir 数据目录配置参数,数据根目录，相当于mysql根目录

每一台datanode都有该目录存数据，namenode存储的是数据映射关系

/var/hadoop 给所有主机都创建该文件夹

vim core-site.xml

<configuration>

<property>

<name>fs.defaultFS</name>

<value>hdfs://namenode:9000</value>

//集群时写hdfs://namenode主机名:9000

</property>

<property>

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

<value>/var/hadoop</value>

</property>

</configuration>

– HDFS配置文件 hdfs-site.xml

vim hdfs-site.xml

<configuration>

<property>

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

<value>namenode:50070</value> //给datanode用

</property>

<property>

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

<value>namenode:50090</value> //和namenode在同一台机

</property>

<property>

<name>dfs.replication</name>//切块数据复制几份

<value>2</value>原数据＋复制＝2份，默认是3份

</property>

</configuration>

在datanode中声明namenode是哪台主机

|  |  |
| --- | --- |
| dfs.namenode.name.dir | file://${hadoop.tmp.dir}/dfs/name |

namenode数据存储2种，默认都放在${hadoop.tmp.dir}/dfs/name下，如果不想存在该目录，修改该配置

|  |  |
| --- | --- |
| dfs.datanode.data.dir | file://${hadoop.tmp.dir}/dfs/data |

datanode的数据默认存在本机的${hadoop.tmp.dir}/dfs/data，如果不想存在该目录，修改该配置，还可以用数据恢复和备份

– 节点配置文件slaves 配置所有的datanodes

vim /usr/local/hadoop/etc/hadoop/slaves

node1

node2

node3

5.– 同步配置

– hadoop 所有节点的配置参数完全一样,我们在一台配置好以后,要把配置文件分发到其它所有主机上去

hadoop配置文件特点：

在一台中安装完直接把文件夹拷贝到其他机上，因为配置完全一模一样

6.在 namenode 上执行格式化操作

./bin/hdfs namenode -format

7.在 namenode上启动集群

./sbin/start-dfs.sh

会自动创建一个logs文件夹

./sbin/stop-dfs.sh

jps 每台机输入jps查看自己的角色

在 namenode上运行以下命令：查看集群是否组建成功

./bin/hdfs dfsadmin -report

Live datanodes (3) -----------------------代表成功，3台datanode,哪台没有上那台机排错

排错：

cd logs

.out屏幕输出

.log 日志输出

* [core-default.xml](http://hadoop.apache.org/docs/r2.7.6/hadoop-project-dist/hadoop-common/core-default.xml) 全局核心配置文件
* [hdfs-default.xml](http://hadoop.apache.org/docs/r2.7.6/hadoop-project-dist/hadoop-hdfs/hdfs-default.xml)
* [mapred-default.xml](http://hadoop.apache.org/docs/r2.7.6/hadoop-mapreduce-client/hadoop-mapreduce-client-core/mapred-default.xml)
* [yarn-default.xml](http://hadoop.apache.org/docs/r2.7.6/hadoop-yarn/hadoop-yarn-common/yarn-default.xml)

HDFS可以单独用来做分布式文件系统存储使用，类似ceph

分布式计算框架mapreduce部署

改名：mv mapred-site.xml.template mapred-site.xml

vim mapred-site.xml

<configuration>

<property>

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

<value>yarn</value>

</property>

</configuration>

local自己管理自己 yarn有集群管理者，要遵循集群管理者分配和调度

YARN

管理分布式计算框架mapreduce

vim yarn-site.xml

<configuration>

<property>

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

<value>namenode</value>

//value为resourcemanager所在的主机名，告诉所有的nodemanager，resourcemanager是谁

</property>

<property>

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

<value>mapreduce\_shuffle</value>

//声明你所管理的分布式框架是谁，value为计算框架的名称（开发给的计算框架名称，没有就用官网给的例子mapreduce\_shuffle测试）

</property>

</configuration>

启动服务：sbin/start-yarn.sh

也可以start-all.sh会启动YARN和hdfs

先验证角色 jps

测试集群是否成功：

bin/yarn node -list

• 使用 web 访问hadoop

– namenode web页面(nn01)

– http://192.168.1.10:50070/

– secondory namenode web 页面(nn01)

– http://192.168.1.10:50090/

– datanode web 页面(node1,node2,node3)

– <http://192.168.1.11:50075/>

• 使用 web 访问hadoop

– resourcemanager web页面(nn01)

– http://192.168.1.10:8088/

– nodemanager web 页面(node1,node2,node3)

– <http://192.168.1.11:8042/>

HDFS 基本使用

HDFS 基本命令

./bin/hadoop fs 回车 所支持的shell命令

– ./bin/hadoop fs –ls /

– 对应 shell 命令 ls /

– ./bin/hadoop fs -mkdir /abc

– 对应 shell 命令 mkdir /abc

– ./bin/hadoop fs –rmdir /abc

– 对应 shell 命令 rmdir /abc

• HDFS 基本命令

– ./bin/hadoop fs –touchz /urfile

– 对应 shell 命令 touch /urfile

– ./bin/hadoop fs –cat /urfile

– 对应 shell 命令 cat /urfile

– ./bin/hadoop fs –rm /urfile

– 对应 shell 命令 rm /urfile

集群文件删除一定要写绝对路径

上传（put）下载（get） 本地文件只能上传到集群中

– 上传文件

– ./bin/hadoop fs –put localfile /remotefile

– 下载文件

– ./bin/hadoop fs –get /remotefile

节点管理

hdfs节点管理

• HDFS 增加节点

– 1、启动一个新的系统,禁用 selinux、禁用 firewalld

– 2、设置 ssh 免密码登录

– 3、在所有节点增加新新节点的主机信息 /etc/hosts

– 4、安装 java openjdk 运行环境

– 5、拷贝namnode的 /usr/local/hadoop 到本机

– 6、修改namenode的slaves文件增加该节点

– 7、在该节点启动Datanode

./sbin/hadoop-daemon.sh start datanode （单独启动角色用）

– 8、设置同步带宽,并同步数据 //将新增加的节点均匀分摊其他节点的数据

./bin/hdfs dfsadmin -setBalancerBandwidth 67108864 //设置带宽，网速的一半

./sbin/start-balancer.sh

– 9、查看集群状态

./bin/hdfs dfsadmin -report

• HDFS 修复节点

– 修复节点比较简单,不增加节点基本一致

– 需要注意新节点的 ip 不 主机名 要不损坏节点一致

– 启动服务

./sbin/hadoop-daemon.sh start datanode

– 数据恢复是自动的

– 我们上线以后会自动恢复数据,如果数据量非常巨大,可能需要一定的时间，千万不能手动物理拷贝

• HDFS 删除节点 （危险系数大）

1. 无数据才能删除
2. 查看状态为Decommissioned才能删除
3. 否则会导致整个集群不能用

– 配置NameNode主机上的 hdfs-site.xml

– 增加 dfs.hosts.exclude 配置

<property>

<name>dfs.hosts.exclude</name>

<value>/usr/local/hadoop/etc/hadoop/exclude</value> 创建该文件

</property

– 增加 exclude 配置文件,写入要删除的节点 主机名

vim /usr/local/hadoop/etc/hadoop/exclude

newnode -------------要删除的主机名

– 开始导出数据，将自己的数据均匀分摊到其他主机上

./bin/hdfs dfsadmin -refreshNodes （Decommissioned in Program）

• HDFS 删除节点状态

– 查看状态 ./bin/hdfs dfsadmin -report

– Normal 正常状态

– Decommissioned in Program 数据正在迁移，不能中断

– Decommissioned 数据迁移完成

– 注意:只有当状态变成 Decommissioned 才能 down机下线

连接到要删除的主机上

./sbin/hadoop-daemon.sh stop datanode

grep -p “newnode” -A 14 xxxx.txt5

查询newnode后面的14行

Yarn节点管理

• Yarn 的相关操作

– 由于在 2.x hadoop 引入了 yarn 框架,对于计算节点的操作已经变得非常简单

– 增加节点

在新加节点主机中运行：

sbin/yarn-daemon.sh start nodemanager

– 删除节点

在要删除的节点主机中运行： （在空闲没有计算任务的时候停止服务）

sbin/yarn-daemon.sh stop nodemanager

– 查看节点列表 (Resourcemanager)

在namenode主机上运行：

./bin/yarn node -list

NFS网关

把hdfs虚拟出来的大的硬盘把它变成一个NFS服务器，其他机器可以用这个mount到其他主机上，容量可以无限横向扩容，而且有数据备份

HDFS可以用replication做数据冗余备份，client和NFS在一台机器上，该台机器如果挂了就不能访问后端数据，因为本身不存数据所以无关紧要，做2台高可用就可以了，客户端这台机负责接收前台数据和返回客户端数据

• 特性与注意事项

– HDFS超级用户是不NameNode进程本身具有相同标识的用户,超级用户可以执行任何操作,因为权限检查永远不会为超级用户失败。

– 在非安全模式,运行网关的用户是代理用户

– 在安全模式时,Kerberos keytab中的用户是代理用户

• 调试与日志排错（出错就开调试模式）

– 在配置 NFS 网关过程中经常会碰到各种各样的错误,如果出现了错误,打开调试日志是一个不错的选择

• log4j.property

– log4j.logger.org.apache.hadoop.hdfs.nfs=DEBUG

– log4j.logger.org.apache.hadoop.oncrpc=DEBUG

• 配置代理用户

– 在 namenode 和 nfsgw 上添加代理用户

– 代理用户的 uid gid 用户名 必须完全相同

– 如果因特殊原因客户端的用户和NFS网关的用户 uid不能保持一致需要我们配置 nfs.map 的静态映射关系

– nfs.map

uid 10 100 # Map the remote UID 10 the local UID 100

gid 11 101 # Map the remote GID 11 to the local GID 101

node5 做NFS网关配置

1.配置/etc/hosts（同步到所有主机）

namenode

node1

node2

node3

nfsgw

2.添加用户（namenode、nfsgw两台主机上执行） uid和gid完全相同

groupadd -g 200 nsd1803

useradd -u 200 -g 200 nsd1803

3、在namenode上配置core-site.xml 同步配置到node1-node3,不包括nfsgw

1）./sbin/stop-all.sh 先停止集群

2）vim core-site.xml

<property>

<name>hadoop.proxyuser.nsd1803.groups</name>

<value>\*</value> //允许所有

</property>

<property>

<name>hadoop.proxyuser.nsd1803.hosts</name>

<value>\*</value>

</property>

启动集群

./sbin/start-dfs.sh

1. NFS 网关（以下全是在nfsgw主机上操作）

– 1）、启动一个新的系统,禁用 selinux、禁用 firewalld

– 2）、卸载 rpcbind 、nfs-utils

– 3）、配置 /etc/hosts,添加所有 namenode 和datanode 的主机名不 ip 对应关系

– 4）、安装 java openjdk 的运行环境

– 5）、同步 namenode 的 /usr/local/hadoop 到本机

– 6）、配置 hdfs-site.xml （namenode上操作）

vim hdfs-site.xml

<property>

<name>nfs.exports.allowed.hosts</name>

<value>\* rw</value> //允许所有主机可读写

</property>

– 默认情况下,export可以被任何客户端挂载。为了更好的控制访问,可以设置属性。值字符串为机器名和访问策略,通过空格来分割。机器名的格式可以是单一的主机,Java的正则表达式或者IPv4地址。访问权限使用rw或ro来指定导出目录的读/写或机器只读访问。如果访问策略没被提供,默认为只读的。每个条使用“;”来分割。

<property>

<name>nfs.dump.dir</name>

<value>/var/nfstmp</value>

//临时存储的目录，当上传的整个文件上传完整了才会进行操作

</property>

配置完该属性后要创建 /var/nfstmp 文件夹，mkdir /var/nfstmp

//由nfs的代理用户进行转储，所以该目录对代理用户要有可读写权限

chown 200:200 /var/nfstmp

用户需要更新文件转储目录参数。NFS客户端经常重新安排写操作,顺序的写操作会以随机到达NFS网关。这个目录常用于临时存储无序的写操作。对于每个文件,无序的写操作会在他们积累在内存中超过一定阈值(如。1 mb)被转储。需要确保有足够的空间的目录。例如,如果应用上传10个100M,那么这个转储目录推荐有1GB左右的空间,以便每个文件都发生最坏的情况。只有NFS网关需要在设置该属性后重启。

7）创建数据根目录 /var/hadoop

mkdir /var/hadoop

8）– 设置 /usr/local/hadoop/logs 权限,为代理用户赋予读写执行的权限

setfacl -m u:nsd1803:rwx /usr/local/hadoop/logs

9）、启动服务,以下顺序不能错

– 使用 root 用户启动 portmap 服务

centos6 : rpcbind centos7:portmap

./sbin/hadoop-daemon.sh --script ./bin/hdfs start portmap

– 使用代理用户启动 nfs3

su -l nsd1803

./sbin/hadoop-daemon.sh --script ./bin/hdfs start nfs3

客户端测试：

客户端安装nfs-utils

mount -t nfs -o vers=3,proto=tcp,nolock,noacl,noatime,sync 192.168.1.15:/ /mnt

vers=3 目前NFS只支持V3版本

proto=tcp 仅使用TCP作为传输协议

nolock不支持锁NLM

noatime 禁用时间更新 数据切块完只修改时间啥也不干就到下一步了，浪费性能（优化）

atime mtime ctime

stat /etc/passwd

Access time最近访问：2018-08-02 17:04:43.299039430 +0800

Modify time最近更改：2018-08-02 17:04:43.098037406 +0800

内容或属性发生变化，文件Md5值变化了一定会更改

Change time最近改动：2018-08-02 17:04:43.123037658 +0800

属性发生变化

sync 不支持随机写，只支持顺序写，会导致不可预测的吞吐量，未指定同步选项可能会导致上传大文件时出现不可靠的行为

查看注册服务

rpcinfo -p 192.168.1.15

查看共享目录

showmount -e 192.168.1.15

**1、Namenode**

**它是Hadoop 中的主服务器，管理文件系统名称空间和对集群中存储的文件的访问。**

**2、Datanode**

**它负责管理连接到节点的存储（一个集群中可以有多个节点）。每个存储数据的节点运行一个 datanode 守护进程。**

**3、secondaryNameNode**

**它不是 namenode 的冗余守护进程，而是提供周期检查点和清理任务。 出于对可扩展性和容错性等考虑，我们一般将SecondaryNameNode运行在一台非NameNode的机器上。**

**4、ResourceManager**

**负责调度 DataNode上的工作。每个 DataNode有一个NodeManager，它们执行实际工作。**

**5、NodeManager**

**负责执行ResourceManager分发的任务**