完全分布式

安装部署mapred、yarn、web访问hadoop

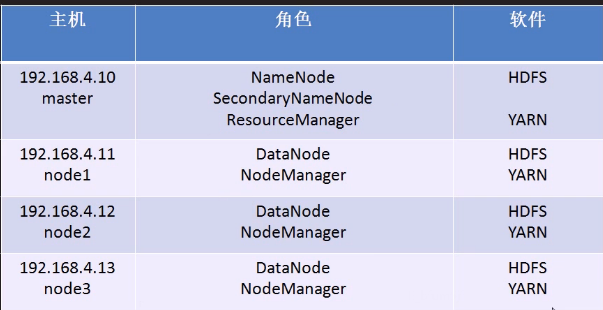
hadoop三大核心组件

分布式文件系统：HDFS已经部署完毕

分布式计算框架：mapreduce

集群资源管理：yarn

系统规划



新安装resourcemanager和nodemanager

分布式计算框架maprred-site.xml

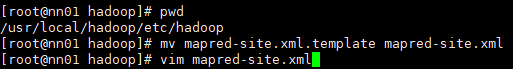
改名

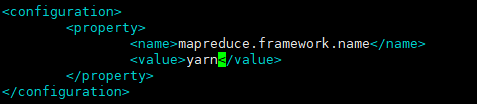
FROM:mapred-site.xml.template

TO:mapred-site.xml

资源管理类

mapreduce.framework.name



被谁管理

yarn资源管理安装与配置yarm-site.xml

resourcemanager管理地址yarn.resourcemanager.hostname

nodemanager使用哪个计算框架yarn.nodemanager.aux-services

mapreduice\_shuffle是我们使用的计算框架名称

yarm-site.xml

<property>

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

<value>nn01</value>

</property>

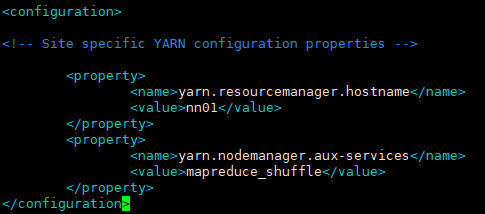
<property>

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

<value>mapreduce\_shuffle</value>

</property>



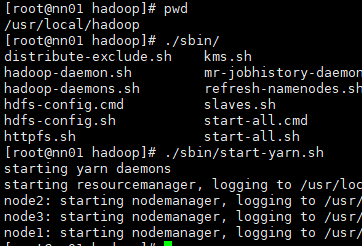


配置完成后把配置同步到所有主机



启动yarn服务

./sbin/start-yarn.sh

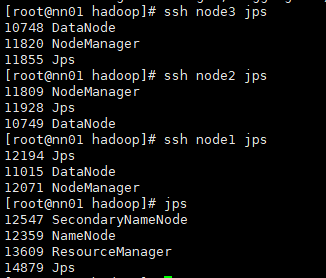


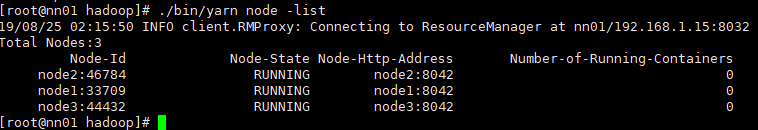


在所有主机上执行jps,查看是否启动成功

resourcemanager

nodemanager





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>

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

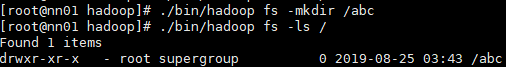
./bin/hadoop fs -ls /

对应shell命令的ls /



./bin/hadoop fs -mkdir /abc

对应shell命令mkdir /abc



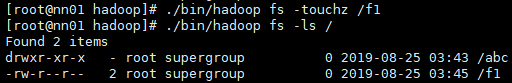
./bin/hadoop fs -rmdir /abc

对应shell命令rmdir /abc



./bin/hadoop fs -touchz /urfile

对应shell命令touch /urfile



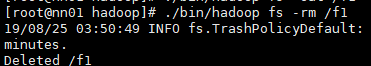
./bin/hadoop fs -cat /urfile

对应shell命令cat /urfile



./bin/hadoop fs -rm /urfile

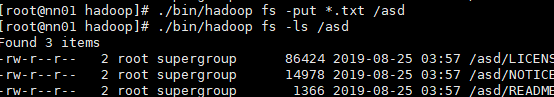
对应shell命令rm /urfile





上传文件

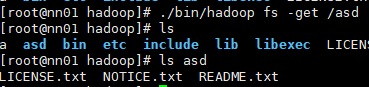
./bin/hadoop fs -put localfile /remotefile



下载文件

./bin/hadoop fs -get /remotefile





提交验证

hadoop验证

./bin/hadoop fs -mkdir /input

导入要分析的文件

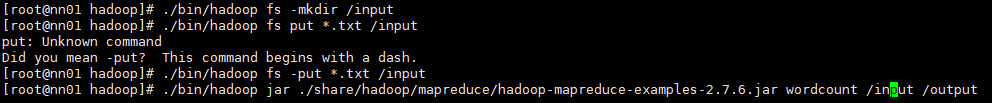
./bin/hadoop fs -put \*.txt /input

提交分析作业（要加/才行，否则是单机分析）

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

查看结果

./bin/hadoop fs -cat /output/\*





HDFS进阶应用

NFS网关用途

1. 用户可以通过操作系统兼容的本地NFSv3客户端来阅读HDFS文件系统（可以mount）
2. 用户可以从HDFS文件系统下载文档到本地文件系统
3. 用户可以通过挂载点直接流化数据。支持文件附加，但是不支持随机写

NFS网关支持NFSv3和允许HDFS作为客户端文件系统的一部分挂载

特性与注意事项

不支持随机写

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

调试与排错

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

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

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

核心配置core-site.xml

hadoop.proxyuser.{代理用户}.groups

hadoop.proxyuser.{代理用户}.hosts

这里的{代理用户}是机器上真实运行nfs3的用户

在非安全模式，运行nfs网关的用户为代理用户

groups为挂载点用户所使用的组

hosts位挂载点主机地址

NFS网关

配置步骤：

1. 启动一个新的系统，禁用selinux、禁用firewalld
2. 卸载rpcbind、nfs-utils
3. 配置hosts，添加所有namenode和datanode的主机名与ip映射关系
4. 安装java openjdk的环境
5. 同步namemode的/usr/local/hadoop到本机
6. 配置hdfs-site.xml

nfs.exports.allowed.hosts默认情况下，export可以被任何客户端挂载。为了更好的访问控制，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的正则表达式或者是ipv4地址。访问权限使用rw或ro来指定导出目录的读/写或者机器只读访问。如果访问策略没被提供，默认为只读的。每个条目使用分号来分割。默认“ro”

<property>

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

<value>\* rw</value>

</property>

nfs.dump.dir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的数值（如1mb）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

<property>

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

<value>/tmp/nfstmp</value>

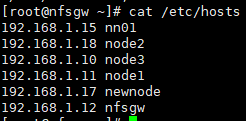
</property>

配置完成后要创建目录并把该文件夹的属组改成代理用户

1. 启动服务与挂载

步骤：

1. 配置hosts



1. 添加用户（nn01，nfsgw）



1. 配置core-site.xml（停止集群）

增加、同步到所有主机、启动集群

core-site.xml

<property>

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

<value>\*</value>

</property>

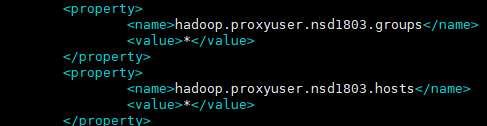
<property>

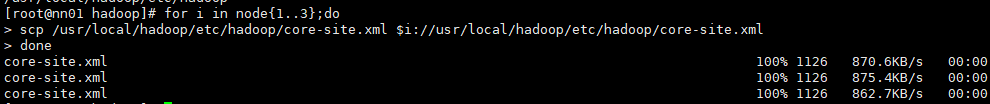
<name>hadoop.proxyuser.nsd1803.hosts</name>

<value>\*</value>

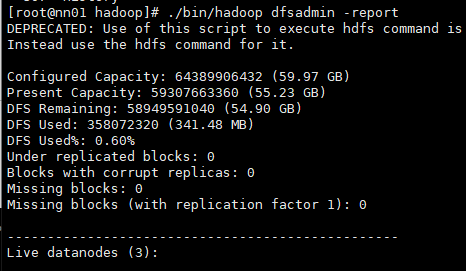
</property>



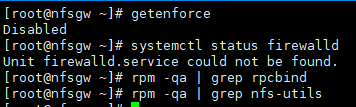








1. 在nfsgw上检查是否禁用selinux、禁用firewalld，卸载rpcbind、nfs-utils，安装openjdk，同步namenode的hadoop到该主机

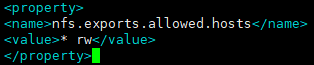


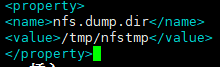




1. 配置hdfs-site.xml







1. 创建临时转储目录，并把该文件夹的属组改成代理用户





1. 创建数据根目录

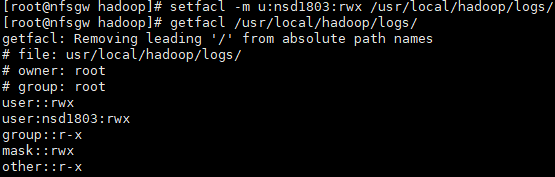


1. 启动与挂载

启动

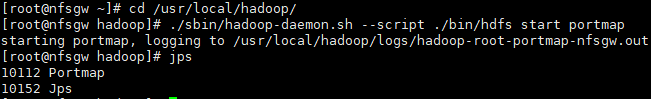
设置/usr/local/hadoop/logs权限，为代理用户赋予读写执行权限

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



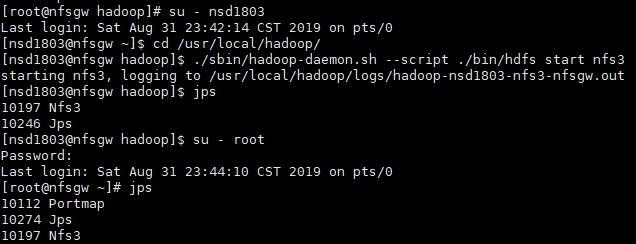
使用root启动portmap服务

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



使用代理用户启动nfs3

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



挂载nfs

目前NFS v3仅使用TCP作为传输协议。不支持NLM（锁），因此需要安装”nolock”。强烈建议使用安装选项sync，因为它可以最小化或避免重新排序写入，这将导致更可预测的吞吐量。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

如果必须使用软安装，用户应该给它一个相对较长的超时（至少不小于主机上的默认超时）

挂载NFS参数

vers=3

proto=tcp

nolock

noacl

noatime

sync

mount -t nfs

-o vers=3,proto=tcp,nolock,noacl,noatime,sync ip.xx.xx.xx:/

/localdir



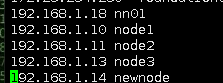


1. 挂载完成后进行读写测试

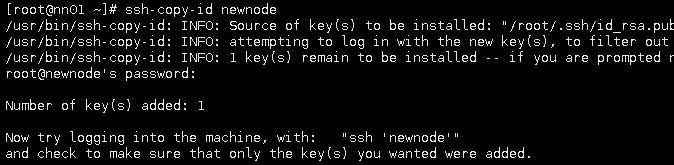
hadoop节点管理

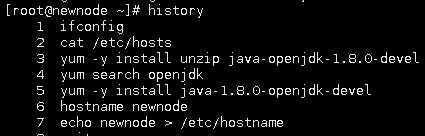
HDFS增加节点

1. 配置所有hadoop环境，包括主机名、ssh免密码登录、禁用selinux、iptables、安装java环境

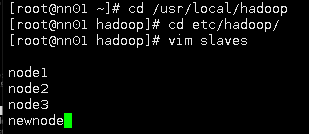








1. 修改namenode的slaves文件增加该节点



1. 把所有的配置文件复制到所有主机配置文件目录下
2. 在新节点启动datanode

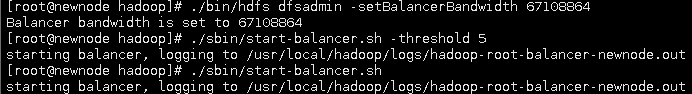
./sbin/hadoop-daemon.sh start datanode



1. 设置同步带宽，并同步数据

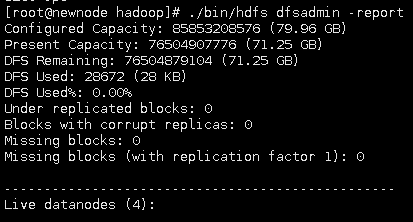
./bin/hdfs dfsadmin -setBalancerBandwidth 67108864

./sbin/start-balancer.sh -threshold 5



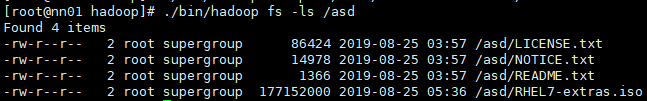
1. 查看集群状态

./bin/hdfs dfsadmin -report



HDFS删除节点





配置namenode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>

</property>



增加exclude配置文件，写入要删除的节点主机名



更新数据

./bin/hdfs dfsadmin -refreshNodes



删除节点状态

查看状态./bin/hdfs dfsadmin -report

noraml正常状态

decommissioned in program数据正在迁移

decommissioned数据迁移完成





注意：只有当状态变成decommissioned才能down机下线



HDFS修复节点

修复节点比较简单

单独配置一台新的datanode，主机名必须跟原来的一样

启动服务

./sbin/hadoop-daemon.sh start datanode

数据恢复是自动的

上线以后会自动恢复数据，如果数据量非常巨大，可能需要一段时间

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

sbin/yarn-daemon.sh start nodemanager

删除节点

sbin/yarn-daemon.sh stop nodemanager

查看节点（resourcemanager）

./bin/yarn node -list

yarn的系统环境配哦哦之与hdfs的基础环境配置相同，这里就不重复列出了

由于yarn不包含数据，所以在增加删除修复节点的时候比较简单，hdfs要注意数据安全

HDFS还有很多其他的应用方式，比如native-hdfs，可以用来测试

用到的依赖

cmake，fuse-devel

protobuf

protobuf-c

native-hdfs-fuse