# 大数据部署文档

1. 机器准备，准备三台节点机器，测试系统为centos ，线上为ubuntu，执行命令基本无差别，并重新命名为：master slave1 slave2 命令为：hostnamectl set-hostname master
   1. 对于centos注意要先关闭selinux
   2. 防火墙本地可以先关闭，线上可能将udp tcp 局域网内开放
2. 给三台节点配置ip，分配hosts，目前本地配置为：
   1. 192.168.1.193 master
   2. 192.168.1.194 slave1
   3. 192.168.1.195 slave2
3. 给三台机器配置无密码登陆（默认使用root登陆）
   1. 修改三台机器的 /etc/ssh/sshd\_config 的以下两个配置注释打开
      1. AuthorizedKeysFile .ssh/authorized\_keys
      2. PubkeyAuthentication yes
   2. 三台机器都先执行 ssh-keygen -t rsa 会在登陆用户生成 .ssh目录
   3. 在master机器建立 authorized\_keys 文件 命令为：touch ~/.ssh/authorized\_keys
   4. 将slave1 slave2的~/.ssh/id\_rsa.pub 文件scp 到 master 节点，命令为：
      1. Salve1执行：scp ~/.ssh/id\_rsa.pub master:~/.ssh/slave1.id\_rsa.pub
      2. Salve2执行：scp ~/.ssh/id\_rsa.pub master:~/.ssh/slave2.id\_rsa.pub
      3. 以上两个命令会让输入master的密码
   5. 将id\_rsa.pub写入 authorized\_keys 在master执行以下命令
      1. cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys
      2. cat ~/.ssh/slave1.id\_rsa.pub >> ~/.ssh/authorized\_keys
      3. cat ~/.ssh/slave2.id\_rsa.pub >> ~/.ssh/authorized\_keys
   6. 将authorized\_keys scp 到slave1 slave2 命令为：
      1. scp ~/.ssh/authorized\_keys slave1:~/.ssh/authorized\_keys
      2. scp ~/.ssh/authorized\_keys slave2:~/.ssh/authorized\_keys
   7. 三台机器目录授权
      1. chmod 700 ~/.ssh
      2. chmod 600 ~/.ssh/authorized\_keys
   8. 验证无密登陆 ssh master ssh slave1 ssh slave2
4. 上传Hadoop和jdk安装包到安装目录,如/home
5. 执行hadoopinstall.sh脚本，等待安装完毕。该脚本包含jdk、hadoop、yarn的自动安装与配置，安装好后会启动hadoop与yarn集群
   1. 注意hadoop-3.2.1/etc/hadoop/yarn-site.xml的配置，
      1. 其中yarn.nodemanager.resource.memory-mb 代表节点的资源大小
      2. 其中yarn.scheduler.maximum-allocation-mb 代表任务的最大资源
      3. 其中yarn.nodemanager.resource.cpu-vcores 代表节点的cpu数量
      4. 其中yarn.scheduler.maximum-allocation-vcores 代表任务的最大cpu数量
   2. 等待脚本执行完毕，浏览器输入master ip 访问 8088 9870端口查看yarn与hadoop是否起来
   3. 启动集群命令（重启后需要）

start-dfs.sh

start-yarn.sh

mapred --daemon start historyserver

yarn --daemon start timelineserver

* 1. 关闭集群

stop-yarn.sh

stop-dfs.sh

mapred --daemon stop historyserver

yarn --daemon stop timelineserver

1. zk集群
   1. 先处理master节点，将zookeeper-3.4.9.tar.gz 安装包上传到/home目录
   2. 进入/home，解压到当前目录tar -zxvf zookeeper-3.4.9.tar.gz
   3. 配置zk
      1. cd zookeeper-3.4.9/conf (进入配置目录)
      2. cp zoo\_sample.cfg zoo.cfg (复制示例配置文件)
      3. mkdir -p /home/softdata/zkdata (创建数据目录)
      4. vi zoo.cfg (修改配置文件)
         1. 修改其中的 dataDir (此参数存放zk数据)

dataDir=/home/softdata/zkdata

* + - 1. 增加节点通信端口

server.1=master:2888:3888

server.2=slave1:2888:3888

server.3=slave2:2888:3888

* + 1. 在dataDir创建myid文件
       1. echo '1' > /home/softdata/zkdata/myid
    2. 分发到slave1和slave2节点
       1. scp -r /home/zookeeper-3.4.9 slave1:/home
       2. scp -r /home/zookeeper-3.4.9 slave2:/home
       3. 登陆slave1 创建数据目录(6.3.3操作)，创建myid(6.5.1)，slave1值为2
       4. 登陆slave2 创建数据目录(6.3.3操作)，创建myid(6.5.1)，slave2值为3
    3. 修改环境变量(三台机器都需要)
       1. vi /etc/profile增加

#set zookeeper environment

export ZK\_HOME=/home/zookeeper-3.4.9

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

* + - 1. 执行source /etc/profile 使环境变量生效
  1. 启动zk (三台机器都需要)
     1. zkServer.sh start
     2. 查看状态
     3. zkServer.sh status
     4. 如果显示Mode: leader 或者Model:follower 代表集群启动成功

1. Hbase集群
   1. 先处理master节点，上传hbase-2.2.6-bin.tar.gz文件到/home目录
   2. 进入/home，解压到当前目录 tar -zxvf hbase-2.2.6-bin.tar.gz
   3. 配置hbase
      1. cd hbase-2.2.6/conf
      2. vi hbase-env.sh
      3. 增加配置参数

export JAVA\_HOME=/home/jdk

export HBASE\_MANAGES\_ZK=false

* + 1. echo '<?xml version="1.0"?>

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

<!--

/\*

\* Licensed to the Apache Software Foundation (ASF) under one

\* or more contributor license agreements. See the NOTICE file

\* distributed with this work for additional information

\* regarding copyright ownership. The ASF licenses this file

\* to you under the Apache License, Version 2.0 (the

\* "License"); you may not use this file except in compliance

\* with the License. You may obtain a copy of the License at

\*

\* http://www.apache.org/licenses/LICENSE-2.0

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

-->

<configuration>

<!--

The following properties are set for running HBase as a single process on a

developer workstation. With this configuration, HBase is running in

"stand-alone" mode and without a distributed file system. In this mode, and

without further configuration, HBase and ZooKeeper data are stored on the

local filesystem, in a path under the value configured for `hbase.tmp.dir`.

This value is overridden from its default value of `/tmp` because many

systems clean `/tmp` on a regular basis. Instead, it points to a path within

this HBase installation directory.

Running against the `LocalFileSystem`, as opposed to a distributed

filesystem, runs the risk of data integrity issues and data loss. Normally

HBase will refuse to run in such an environment. Setting

`hbase.unsafe.stream.capability.enforce` to `false` overrides this behavior,

permitting operation. This configuration is for the developer workstation

only and \_\_should not be used in production!\_\_

See also https://hbase.apache.org/book.html#standalone\_dist

<property>

<name>hbase.cluster.distributed</name>

<value>false</value>

</property>

<property>

<name>hbase.tmp.dir</name>

<value>./tmp</value>

</property>

<property>

<name>hbase.unsafe.stream.capability.enforce</name>

<value>false</value>

</property> -->

<!-- 这个目录是region服务器共享的目录，用来持久存储HBase的数据，默认值为：${hbase.tmp.dir}/hbase，如果不修改这个配置，数据将会在集群重启时丢失。 -->

<property>

<name>hbase.rootdir</name>

<!-- hadoop引入JQM负载均衡时，这里配置dfs.nameservices指定的集群逻辑名称 -->

<value>hdfs://master:8020/hbase</value>

</property>

<!-- 指定hbase集群为分布式集群 -->

<property>

<name>hbase.cluster.distributed</name>

<value>true</value>

</property>

<!-- 指定zookeeper集群，有多个用英文逗号分隔 -->

<property>

<name>hbase.zookeeper.quorum</name>

<value>master:2181,slave1:2181,slave2:2181</value>

</property>

<!-- Zookeeper元数据快照的存储目录（需要和Zookeeper的zoo.cfg 配置文件中的属性一致） -->

<property>

<name>hbase.zookeeper.property.dataDir</name>

<value>/data/bigdata/zookeeper</value>

</property>

<!-- 指定HBase Master web页面访问端口，默认端口号16010 -->

<property>

<name>hbase.master.info.port</name>

<value>16010</value>

</property>

<!-- 指定HBase RegionServer web页面访问端口，默认端口号16030 -->

<property>

<name>hbase.regionserver.info.port</name>

<value>16030</value>

</property>

<!-- 解决启动HMaster无法初始化WAL的问题 -->

<property>

<name>hbase.unsafe.stream.capability.enforce</name>

<value>false</value>

</property>

<property>

<name>hbase.table.sanity.checks</name>

<value>false</value>

</property>

<property>

<name>hbase.hregion.memstore.flush.size</name>

<value>5368709120</value>

<description>

Memstore will be flushed to disk if size of the memstore

exceeds this number of bytes. Value is checked by a thread that runs

every hbase.server.thread.wakefrequency.

</description>

</property>

<property>

<name>hbase.hregion.memstore.block.multiplier</name>

<value>8</value>

<description>

Block updates if memstore has hbase.hregion.block.memstore

time hbase.hregion.flush.size bytes. Useful preventing

runaway memstore during spikes in update traffic. Without an

upper-bound, memstore fills such that when it flushes the

resultant flush files take a long time to compact or split, or

worse, we OOME.

</description>

</property>

</configuration>' > hbase-site.xml

* + 1. 修改节点
       1. vi regionservers 内容替换为

master

slave1

slave2

* + 1. 将hadoop的配置复制到hbase目录下
       1. cp $HADOOP\_HOME/etc/hadoop/core-site.xml /home/hbase-2.2.6/conf
       2. cp $HADOOP\_HOME/etc/hadoop/hdfs-site.xml /home/hbase-2.2.6/conf
    2. 配置环境变量 vi /etc/profile(三台机器都要)

#HBase

export HBASE\_HOME=/home/hbase-2.2.6/

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

* + 1. 使环境变量生效 source /etc/profile
  1. 分发到slave1 slave2

scp -r /home/hbase-2.2.6 slave1:/home

scp -r /home/hbase-2.2.6 slave2:/home

* 1. 在master节点 输入start-hbase.sh 启动hbase 然后到各节点输入jps看是否有进程 hmaster hregionserver
  2. 输入master节点ip加上端口16010，访问hbase的web页面

1. Kafka集群
   1. 先处理master节点，上传kafka\_2.12-2.7.0.tgz文件到/home目录
   2. 进入/home，解压到当前目录 tar -zxvf kafka\_2.12-2.7.0.tgz
   3. 配置kafka
      1. cd kafka\_2.12-2.7.0/config
      2. 创建日志目录 mkdir -p /home/softdata/kafka/kafka-logs
      3. vi server.properties
         1. 配置 broker 的ID

broker.id=0 //第一个kafka配置为 0，第二个配置为1，以此类推

* + - 1. 配置监听相关

host.name=master

host.ip=192.168.1.193

port=9097

advertised.host.name=master

listeners=PLAINTEXT://0.0.0.0:9097

* + - 1. 配置请求大小

socket.request.max.bytes=2048576000

message.max.bytes=6000000

* + - 1. 配置分区

num.partitions=10

* + - 1. 配置可删除topic

delete.topic.enable=true

* + - 1. 配置日志目录

log.dirs=/home/softdata/kafka/kafka-logs

* + - 1. 配置zk

zookeeper.connect=master:2181,slave1:2181,slave2:2181

* + - 1. 配置zk连接时间

zookeeper.connection.timeout.ms=18000

* + - 1. 配置自动创建topic

auto.create.topics.enable=true

* + - 1. 配置延时空消息组开启reblance

group.initial.rebalance.delay.ms=0

* + - 1. 配置topic副本数量

default.replication.factor=2

* + 1. 分发到每个节点，注意分发后，需要到各自中改掉host.ip,host.name,broker的ID,advertised.host.name
       1. scp -r /home/kafka\_2.12-2.7.0 slave1:/home
       2. scp -r /home/kafka\_2.12-2.7.0 slave2:/home
       3. 登陆slave1 mkdir -p /home/softdata/kafka/kafka-logs
       4. 登陆slave2 mkdir -p /home/softdata/kafka/kafka-logs
    2. 增加环境变量（三台需要）
       1. vi /etc/profile

export KAFKA\_HOME=/home/kafka\_2.12-2.7.0

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

* + - 1. source /etc/profile
  1. 启动kafka（三台需要）
     1. kafka-server-start.sh $KAFKA\_HOME/config/server.properties &
     2. 观察日志无异常，按ctrl+d退出登陆
  2. 输入jps查看是否有kafka
  3. 测试kafka
     1. kafka-topics.sh --create --zookeeper master:2181 --replication-factor 1 --partitions 1 --topic ranyi
     2. kafka-topics.sh --list -zookeeper master:2181

1. flink运行环境
   1. 在master节点，上传flink-1.11.2-bin-scala\_2.12.tgz文件到/home目录
   2. 进入/home，解压到当前目录 tar -zxvf flink-1.11.2-bin-scala\_2.12.tgz
   3. 配置环境变量 vi /etc/profile

export FLINK\_HOME=/home/flink-1.11.2

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

* 1. 使配置环境变量生效 source /etc/profile
  2. 后续程序在master上提供到yarn集群即可

1. es集群
   1. 在master节点，上传elasticsearch-6.5.4.tar.gz文件到/home目录
   2. 进入/home，解压到当前目录 tar -zxvf elasticsearch-6.5.4.tar.gz
   3. 创建es文件目录 mkdir -p /home/softdata/es/data（三台）
   4. 创建es 日志目录 mkdir -p /home/softdata/es/logs（三台）
   5. 配置es
      1. cd elasticsearch-6.5.4/config
      2. vi elasticsearch.yml

cluster.name: escluster

node.name: es1

node.master: true

node.data: true

path.data: /home/softdata/es/data

path.logs: /home/softdata/es/logs

bootstrap.memory\_lock: true

bootstrap.system\_call\_filter: false

http.port: 9200

network.host: 0.0.0.0

discovery.zen.minimum\_master\_nodes: 2

discovery.zen.ping\_timeout: 60s

discovery.zen.ping.unicast.hosts: ["192.168.1.195:9300","192.168.1.194:9300","192.168.1.193:9300"]

#index.number\_of\_shards: 2

#index.number\_of\_replicas: 1

cluster.max\_shards\_per\_node: 5000

* + 1. 分发到slave1 和slave2

scp -r /home/elasticsearch-6.5.4/ slave1:/home

scp -r /home/elasticsearch-6.5.4/ slave2:/home

* + 1. slave1 修改node.name为es2
    2. slave2 修改node.name为es3

以下操作，三台均需要

* 1. 新增es用户 useradd elasticsearch
  2. 设置权限信息
     1. chown -R elasticsearch: /home/softdata/es
     2. chown -R elasticsearch: /home/elasticsearch-6.5.4
  3. 操作系统调优（三台）
     1. vi /etc/sysctl.conf

fs.file-max=655360

vm.max\_map\_count=655360

解释：

（1）vm.max\_map\_count=655360

系统最大打开文件描述符数

（2）vm.max\_map\_count=655360

限制一个进程拥有虚拟内存区域的大小

sysctl -p生效

* + 1. vi /etc/security/limits.conf

\* soft nofile 65536

\* hard nofile 65536

\* soft nproc 65536

\* hard nproc 65536

\* soft memlock unlimited

\* hard memlock unlimited

解释:

(nofile)最大开打开文件描述符

(nproc)最大用户进程数

(memlock)最大锁定内存地址空间

* 1. 启动es（注意内存配置）
     1. su elasticsearch
     2. /home/elasticsearch-6.5.4/bin/elasticsearch -d