浏览器查看：

Hadoop：http://localhost:50070/

Hbase：[http://hdp-node-01:16010/](http://192.168.1.127:60010/" \t "https://blog.csdn.net/peace1213/article/details/_blank)

Zookeeper：

Spark：http://hdp-node-01:8080/

环境变量：

export JAVA\_HOME=/usr/local/apps/jdk1.8

export HADOOP\_HOME=/usr/local/apps/hadoop-2.6.5

export ZK\_HOME=/usr/local/apps/zookeeper

export HBASE\_HOME=/usr/local/apps/hbase

export KAFKA\_HOME=/usr/local/apps/kafka

export FLUME\_HOME=/usr/local/apps/flume

export PATH=$PATH:$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin:$ZK\_HOME/bin:$HBASE\_HOME/bin:$KAFKA\_HOME/bin:$FLUME\_HOME/bin:$PATH

**Zookeeper配置过程**

(hdp-node-01 hdp-node-02 hdp-node-03 hdp-node-04)

1.官网下载zookeeper镜像zookeeper-3.4.12.tar.gz

2.发送到linux，解压到/usr/local/apps

# tar -zxvf zookeeper-3.4.12.tar.gz -C /usr/local/apps

# cd /usr/local/apps

# mv zookeeper-3.4.12.tar.gz zookeeper

4.在安装路径下的./conf 配置文件

# mv zoo\_sample.cfg zoo.cfg

# vi  zoo.cfg

【tickTime=2000

clientPort=2181

initLimit=5

syncLimit=2

dataDir=/usr/local/apps/zookeeper/data

dataLogDir=/usr/local/apps/zookeeper/datalog

server.1=192.168.27.101:2888:3888

server.2=192.168.27.102:2888:3888

server.3=192.168.27.103:2888:3888

server.4=192.168.27.104:2888:3888

】

1. 回到/zookeeper,创建目录/data 和 /datalog

# cd ..

# mkdir data

# mkdir datalog

1. 分别在四台服务器上配置

(hdp-node-01)

# echo 1 >data/myid

(hdp-node-02)

# echo 2 >data/myid

(hdp-node-03)

# echo 3 >data/myid

(hdp-node-04)

# echo 4 >data/myid

1. 每台server启动进入/zookeeper/bin

# /bin/zkServer.sh start

1. 查看状态

# /bin/zkServer.sh status

**Hbase配置过程**

(hdp-node-01 )

1.官网下载hbase镜像hbase-2.0.1-bin.tar.gz

2.发送到linux，解压到/usr/local/apps

# tar -zxvf hbase-2.0.1-bin.tar.gz -C /usr/local/apps

# cd /usr/local/apps

# mv hbase-2.0.1-bin.tar.gz hbase

2.5 删除问题jar包

低版本不兼容：rm -rf /usr/local/apps/hadoop-2.6.5/share/hadoop/yarn/lib/jline-0.9.94.jar

然后还存在SLF4j jar包冲突的warning，但是没有关系。

rm -rf /usr/local/apps/hadoop-2.6.5/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar

如果问题不处理，hbase shell会报错。hadoop2.6.5，不是版本问题！

1. 配置hbase-env.sh

vim conf/hbase-env.sh

export JAVA\_HOME=/usr/local/apps/jdk-1.8

export HBASE\_CLASSPATH=/usr/local/hadoop/conf

export HBASE\_MANAGES\_ZK=false #不使用自带zookeeper

java路径

HBASE\_CLASSPATH指向存放有Hadoop配置文件的目录

HBASE\_MANAGES\_ZK=false指示HBase使用已有的Zookeeper而不是自带的

4.配置hbase-site.xml

<configuration>

<property>

<name>hbase.rootdir</name>

<value>hdfs://hdp-node-01:9000/hbase</value>

<description>The directory shared by region servers.</description>

</property>

<property>

<name>hbase.hregion.max.filesize</name>

<value>1073741824</value>

<description>

Maximum HStoreFile size. If any one of a column families' HStoreFiles has

grown to exceed this value, the hosting HRegion is split in two.

Default: 256M.

</description>

</property>

<property>

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

<value>1073741824</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.cluster.distributed</name>

<value>true</value>

<description>The mode the cluster will be in. Possible values are

false: standalone and pseudo-distributed setups with managed Zookeeper

true: fully-distributed with unmanaged Zookeeper Quorum (see hbase-env.sh)

</description>

</property>

<property>

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

<value>2181</value>

<description>Property from ZooKeeper's config zoo.cfg.

The port at which the clients will connect.

</description>

</property>

<property>

<name>zookeeper.session.timeout</name>

<value>120000</value>

</property>

<property>

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

<value>6000</value>

</property>

<property>

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

<value>hdp-node-01,hdp-node-02,hdp-node-03,hdp-node-04</value>

<description>Comma separated list of servers in the ZooKeeper Quorum.

For example, "host1.mydomain.com,host2.mydomain.com,host3.mydomain.com".

By default this is set to localhost for local and pseudo-distributed modes of operation. For a fully-distributed setup, this should be set to a full list of ZooKeeper quorum servers. If HBASE\_MANAGES\_ZK is set in hbase-env.sh this is the list of servers which we will start/stop ZooKeeper on.

</description>

</property>

<property>

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

<value>/usr/local/apps/hbase/tmp</value>

</property>

</configuration>

hbase.zookeeper.quorum 里面不要用hostname，如果我是在其他机器上执行hbase相关代码，那还需要更改那台机器的host文件。

1，hbase.rootdir：hbase所使用的文件系统为HDFS，根目录为hdfs://192.168.1.127:9000/hbase，该目录应该由HBase自动创建，只需要指定到正确的HDFS NameNode上即可。

2，hbase.hregion.max.filesize：设置HStoreFile的大小，当 大于这个数时，就会split 成两个文件

3，hbase.hregion.memstore.flush.size：设置memstore的大小，当大于这个值时，写入磁盘

4，hbase.cluster.distributed：指定hbase为分布式模式

5，hbase.zookeeper.property.clientPort：指定zk的连接端口

6，zookeeper.session.timeout：RegionServer与Zookeeper间的连接超时时间。当超时时间到后，ReigonServer会被Zookeeper从RS集群清单中移除，HMaster收到移除通知后，会对这台server负责的regions重新balance，让其他存活的RegionServer接管.

7，hbase.zookeeper.property.tickTime：

8，hbase.zookeeper.quorum：默认值是 localhost，列出zookeepr的master,slave1,slave2

9，hbase.tmp.dir：指定HBase将元数据存放路径

5.在/hbase创建/tmp目录

# mkdir tmp

6.配置regionservers 相当于hadoop的slave

hdp-node-02

hdp-node-03

hdp-node-04

7.发送到子服务器

scp –r /usr/local/apps/hbase hdp-node-02:/usr/local/apps/

scp –r /usr/local/apps/hbase hdp-node-03:/usr/local/apps/

scp –r /usr/local/apps/hbase hdp-node-04:/usr/local/apps/

8.

启动HBase集群：

bin/start-hbase.sh

单独启动一个HMaster进程：

bin/hbase-daemon.sh start master

单独停止一个HMaster进程：

bin/hbase-daemon.sh stop master

单独启动一个HRegionServer进程：

bin/hbase-daemon.sh start regionserver

单独停止一个HRegionServer进程：

bin/hbase-daemon.sh stop regionserver

如果遇到org.apache.zookeeper.KeeperException$NoNodeException: KeeperErrorCode = NoNode for /hbase/master问题：

create 'behavior\_user\_app\_201807','timeLen'

create 'behavior\_user\_hour\_time\_201807','timeLen'

create 'behavior\_user\_day\_time\_201807','timeLen'

create 'behavior\_user\_hour\_app\_time\_201807','timeLen'

jps zookeeper对应QuorumPeerMain

# 启动顺序

(hdp-node-01)

先启动hadoop集群start-all.sh

在master节点会启动SecondaryNameNode，NameNode，ResourceManager

在slave节点会启动NodeManager，DataNode

(hdp-node-01 hdp-node-02 hdp-node-03 hdp-node-04)

再启动zookeeper集群：分别登录master和slave，执行：zkServer.sh start

两个节点都有QuorumPeerMain进程

启动zookeeper客户端：zkCli.sh -server master:2181,slave:2181

客户端有进程：ZooKeeperMain

(hdp-node-01)

启动HBase：start-hbase.sh

master节点有HMaster

slave节点有HRegionServer

启动HBase的shell命令：hbase shell

关闭的过程相反

**Kafka配置过程**

(hdp-node-01 )

1.官网下载kafka镜像kafka\_2.11-1.1.0.tgz

2.发送到linux，解压到/usr/local/apps

# tar -zxvf kafka\_2.11-1.1.0.tgz -C /usr/local/apps

# cd /usr/local/apps

# mv kafka\_2.11-1.1.0.tgz kafka

1. 进入/config目录，配置server.properties

broker.id=1 (每台服务器的broker.id要区别，和zookeeper的myid类似)

host.name=192.168.27.101 (当前 Broker 服务器的 IP 地址或者机器名)

port=9092 (Broker 的监听端口，用于监听 Producer 或者 Consumer 的连接)

advertised.host.name=192.168.27.101

advertised.port=9092

advertised.listeners=PLAINTEXT://192.168.27.101:9092

zookeeper.connect=192.168.27.101:2181,192.168.27.102:2181,192.168.27.103:2181,192.168.27.104:2181

zookeeper.connection.timeout.ms=6000

log.dirs=/usr/local/apps/kafka/kafka-logs (日志保存目录)

(有的就修改，没有的添加)

1. 在主目录新建目录/kafka-logs

# mkdir kafka-logs

5.发送到子服务器

scp –r /usr/local/apps/kafka hdp-node-02:/usr/local/apps/

scp –r /usr/local/apps/kafka hdp-node-03:/usr/local/apps/

scp –r /usr/local/apps/kafka hdp-node-04:/usr/local/apps/

6.修改相关配置

/config/server.properties

7.启动kafka集群

(hdp-node-01 hdp-node-02 hdp-node-03 hdp-node-04)

kafka-server-start.sh -daemon /usr/local/apps/kafka/config/server.properties

(1)创建分区和topics

kafka-topics.sh --create --zookeeper 192.168.27.101:2181,192.168.27.102:2181,192.168.27.103:2181,192.168.27.104 --replication-factor 2 --partitions 2 --topic ruready

(2)查看topics状态

kafka-topics.sh --describe --zookeeper 192.168.27.101:2181,192.168.27.102:2181,192.168.27.103:2181,192.168.27.104 --topic ruready

8.测试

(1)启动消息生产者和消息消费者

(hdp-node-01)

bin/kafka-console-producer.sh --broker-list 192.168.27.101:9092,192.168.27.102:9092,192.168.27.103:9092,192.168.27.104:9092 --topic ruready

(hdp-node-02 hdp-node-03 hdp-node-04)

bin/kafka-console-consumer.sh --zookeeper 192.168.27.101:2181,192.168.27.102:2181,192.168.27.103:2181,192.168.27.104:2181 --topic ruready --from-beginning

(2)hdp-node-01输入信息，hdp-node-02或hdp-node-03或hdp-node-04会显示hdp-node-01输的的信息

**Flume配置过程**

(hdp-node-01 )

1.官网下载flume镜像apache-flume-1.7.0-bin

2.发送到linux，解压到/usr/local/apps

# tar -zxvf apache-flume-1.7.0-bin -C /usr/local/apps

# cd /usr/local/apps

# mv apache-flume-1.7.0-bin flume

1. 配置flume-env 添加JAVA的环境变量

# cd /conf

# cp -r flume-env.sh.template flume-env.sh

# vi flume-env.sh

export JAVA\_HOME=/usr/local/apps/jdk1.8

1. 发送到子机器

scp –r /usr/local/apps/flume hdp-node-02:/usr/local/apps/

scp –r /usr/local/apps/flume hdp-node-03:/usr/local/apps/

scp –r /usr/local/apps/flume hdp-node-04:/usr/local/apps/