实验五&实验六 环境准备预实验

> 实验要求:
>
> ```
> 1. 掌握Scala、Spark on Yarn、 Kafka的部署过程
> 1. 理解组件环境之间的依赖关联
> 1. 能够对环境进行简单的测试
> ```

零、启动服务器集群

1. 在华为云上依次启动`server0`, `server1`, `server2`
2. 检查服务器正常启动后进入下面步骤

一、

Scala部署

> Spark和Kafka依赖于Scala语言环境, Scala语言是一种基于JVM并对Java语法进行简化的面向大数据的语言, Spark和Kafka一般会内部集成Scala环境, 但为了统一环境, 我们在集群内部配置独立的Scala环境

1. **在server0服务器上下载Scala包**

   ```
   cd /opt/software
   wget https://downloads.lightbend.com/scala/2.11.12/scala-2.11.12.tgz
   ```

2. **解压压缩包**

   ```
   tar -xvf scala-2.11.12.tgz -C /opt/module/
   ```

3. **配置环境变量**

   修改`/etc/profile.d/hadoopenv.sh`文件, 增加如下内容

   ```
   # SCALA_HOME
   export SCALA_HOME=/opt/module/scala-2.11.12
   export PATH=$PATH:$SCALA_HOME/bin
   ```

   修改完成后使环境变量生效

   ```
   source /etc/profile
   ```

检查环境是否正确

```
scala -version
> Scala code runner version 2.11.12 -- Copyright 2002-2017, LAMP/EPFL
```

4. **同步文件**

   将scala安装目录同步到其他两台服务器, 注意替换`server1`、`server2`为自己的主机名

```
rsync -av /opt/module/scala-2.11.12 server1:/opt/module
rsync -av /opt/module/scala-2.11.12 server2:/opt/module
```

同步环境变量文件

```
sudo rsync -av /etc/profile.d/hadoopenv.sh server1:/etc/profile.d
sudo rsync -av /etc/profile.d/hadoopenv.sh server2:/etc/profile.d
```

在 `server1` 和 `server2` 上测试scala是否正确安装

```
scala -version
> Scala code runner version 2.11.12 -- Copyright 2002-2017, LAMP/EPFL
```

二、Spark on Yarn部署

1. 在`server0`上下载spark压缩包, 这里下载的是spark针对hadoop2.7编译后的版本

```
cd /opt/software
wget https://archive.apache.org/dist/spark/spark-2.4.8/spark-2.4.8-bin-hadoop2.7.tgz
```

2. 在`server0`上解压缩

```
tar -xvf spark-2.4.8-bin-hadoop2.7.tgz -C /opt/module
cd /opt/module
mv spark-2.4.8-bin-hadoop2.7 spark-2.4.8
```

3. 在`server0`上修改配置文件

   进入`/opt/module/spark-2.4.8/conf/`目录, 复制 `spark-env.sh.template` 为 `spark-env.sh`

```
cd /opt/module/spark-2.4.8/conf/
mv spark-env.sh.template spark-env.sh
```

修改`spark-env.sh`, 增加一下内容

```
export JAVA_HOME=/opt/module/jdk1.8.0_341
YARN_CONF_DIR=/opt/module/hadoop-2.7.7/etc/hadoop
```

确保`/opt/module/hadoop-2.7.7/etc/hadoop/yarn-site.xml`中有如下配置

```
<property>
     <name>yarn.nodemanager.pmem-check-enabled</name>
     <value>false</value>
</property>
<property>
  <name>yarn.nodemanager.vmem-check-enabled</name>
  <value>false</value>
</property>
```

4. 在`server0`上配置环境变量

   创建`/etc/profile.d/sparkenv.sh`

   ```
   sudo vim /etc/profile.d/sparkenv.sh
   ```

   在环境变量文件中增加以下内容

   ```
   # SPARK_HOME
   export SPARK_HOME=/opt/module/spark-2.4.8
   export PATH=$PATH:$SPARK_HOME/bin::$SPARK_HOME/sbin
   ```

   保存退出后执行`source /etc/profile`使环境变量生效

5. 启动HDFS和yarn并进行测试

   在`server0`上启动HDFS

   ```
   start-dfs.sh
   ```

   在`server1`上启动yarn

   ```
   start-yarn.sh
   ```

   在`server0`上提交测试任务, 如果报错显示集群处于安全模式, 就等待几分钟后再尝试

```
cd /opt/module/spark-2.4.8
spark-submit \
--class org.apache.spark.examples.SparkPi \
--master yarn \
--deploy-mode client \
./examples/jars/spark-examples_2.11-2.4.8.jar \
10
```

执行成功结果

```
22/12/03 16:45:56 INFO scheduler.TaskSetManager: Finished task 7.0 in stage 0.0 (TID 7) in 40 ms on hadoop1-202211
22/12/03 16:45:56 INFO scheduler.TaskSetManager: Finished task 9.0 in stage 0.0 (TID 9) in 47 ms on hadoop1-202211
22/12/03 16:45:56 INFO scheduler.TaskSetManager: Finished task 8.0 in stage 0.0 (TID 8) in 95 ms on hadoop3-202211
22/12/03 16:45:56 INFO cluster.YarnScheduler: Removed TaskSet 0.0, whose tasks have all completed, from pool
22/12/03 16:45:56 INFO scheduler.DAGScheduler: ResultStage 0 (reduce at SparkPi.scala:38) finished in 1.449 s
22/12/03 16:45:56 INFO scheduler.DAGScheduler: Job 0 finished: reduce at SparkPi.scala:38, took 1.537635 s
Pi is roughly 3.141067141067141
22/12/03 16:45:56 INFO server.AbstractConnector: Stopped Spark@1e34c607{HTTP/1.1, (http/1.1)}{0.0.0.0:4040}
22/12/03 16:45:56 INFO ui.SparkUI: Stopped Spark web UI at http://hadoop1-2022110946:4040
22/12/03 16:45:56 INFO cluster.YarnClientSchedulerBackend: Interrupting monitor thread
22/12/03 16:45:56 INFO cluster.YarnClientSchedulerBackend: Shutting down all executors
22/12/03 16:45:56 INFO cluster.YarnSchedulerBackend$YarnDriverEndpoint: Asking each executor to shut down
22/12/03 16:45:56 INFO cluster.SchedulerExtensionServices: Stopping SchedulerExtensionServices
(serviceOption=None,
 services=List(),
 started=false)
```

三、Kafka部署

1. 在`server0`上下载kafka安装包

```
cd /opt/software
wget https://archive.apache.org/dist/kafka/2.4.1/kafka_2.11-2.4.1.tgz
```

2. 在`server0`上解压安装包

```
tar -xvf kafka_2.11-2.4.1.tgz -C /opt/module
cd /opt/module
mv kafka_2.11-2.4.1 kafka-2.4.1
```

3. 在`server0`上修改配置kafka
    1. 在kafka安装目录下创建`logs`目录

    ```
    mkdir /opt/module/kafka-2.4.1/logs
    ```

    2. 修改配置文件`/opt/module/kafka-2.4.1/config/server.properties`，先删除所有内容后，再粘贴以下配置，注意修改`server0/1/2`为自己的主机名

    ```
    #broker的全局唯一编号，不能重复
    ```

```
broker.id=0
#删除topic功能使能,当前版本此配置默认为true，已从配置文件移除
delete.topic.enable=true
#处理网络请求的线程数量
num.network.threads=1
#用来处理磁盘IO的线程数量
num.io.threads=1
#发送套接字的缓冲区大小
socket.send.buffer.bytes=102400
#接收套接字的缓冲区大小
socket.receive.buffer.bytes=102400
#请求套接字的缓冲区大小
socket.request.max.bytes=104857600
#kafka运行日志存放的路径
log.dirs=/opt/module/kafka-2.4.1/logs
#topic在当前broker上的分区个数
num.partitions=1
#用来恢复和清理data下数据的线程数量
num.recovery.threads.per.data.dir=1
#segment文件保留的最长时间，超时将被删除
log.retention.hours=168
#配置连接Zookeeper集群地址
zookeeper.connect=server0:2181,server1:2181,server2:2181
```

4. 同步文件夹到`server1`和`server2`

```
rsync -av /opt/module/kafka-2.4.1 server1:/opt/module
rsync -av /opt/module/kafka-2.4.1 server2:/opt/module
```

5. 修改配置文件中的ID

在`server1`和`server2`中分别修改`/opt/module/kafka-2.4.1/config/server.properties`中的`block.id`

```
# server1
block.id=1
# server2
block.id=2
```

6. 启动zookeeper集群

在三台节点上执行zk启动命令

```
zkServer.sh start
```

7. 启动kafka集群

在三台节点上分别执行启动命令

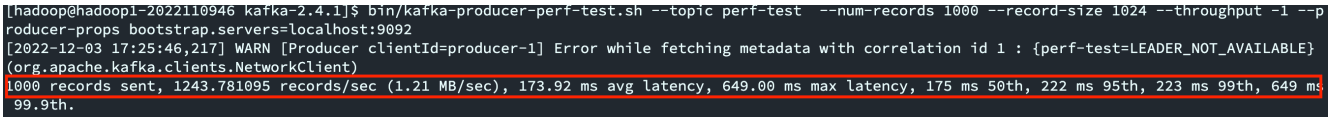
```
cd /opt/module/kafka-2.4.1
bin/kafka-server-start.sh -daemon config/server.properties
```

7. 测试生产者

   在`server0`上执行

   ```
   bin/kafka-producer-perf-test.sh --topic perf-test  --num-records 1000 --record-size 1024 --throughput -1 --producer-props bootstrap.servers=localhost:9092
   ```

   成功结果, 出现测试数据即可

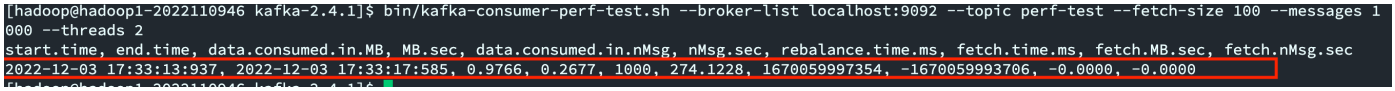


8. 测试消费者

   在`server0`上执行

```
bin/kafka-consumer-perf-test.sh --broker-list localhost:9092 --topic perf-test --fetch-size 100 --messages 1000 --threads 2
```

成功结果, 出现测试数据即可



9. 关闭kafka集群

   在三台节点的`/opt/module/kafka-2.4.1`目录下执行

```
bin/kafka-server-stop.sh stop
```

四、 关闭集群

1. 关闭kafka集群, 见3.9
2. 关闭yarn和hdfs

   `server0`执行`stop-dfs.sh`, `server1`执行`stop-yarn.sh`

3. 关闭zookeeper

   三台节点上执行`zkServer.sh stop`

4. 将三台节点关机