

Hadoop与Spark集群配置总体流程及踩坑记录

作者：lalalapotter

伪分布式配了好多次，但还是首次在多台服务器上配置hadoop和spark，总的来说还是逼不得已，课程PJ的资源太少，只能自己摸索配置了。加油！

环境设置

python2.7

pyspark 2.2.0

JDK1.8

hadoop 2.8.3

Scala 2.12

spark 2.2.0

服务器地址:

10.88.3.55

10.88.3.81

10.88.3.82

注意：配置文件中原本没有的，可以利用template文件复制之后重命名，再做修改

Hadoop集群配置

首先检查三个服务器之间是否可以ping的通，已检查。

安装JDK

使用了服务器自带的openJDK，此处不知道会不会产生问题？（验证后发现自带openJDK这个比较轻量级的包还是不能满足我们的需要，因此重新下载安装JDK1.8）

只需下载安装包（binary），解压之后放到/opt文件夹下，并重命名为（jdk）

ps：这里也可以利用软连接

ps：本人使用连接<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>，下载了这个安装包：jdk-8u171-linux-x64.tar.gz

修改/etc/profile，添加

```
export JAVA_HOME=/opt/jdk
export JRE_HOME=$JAVA_HOME/jre
export CLASSPATH=.:$JAVA_HOME/lib
export PATH=$PATH:$JAVA_HOME/bin
```

不要忘了source /etc/profile

java -version来测试

安装hadoop

直接下载hadoop2.8.3 binary包后，解压，放到/opt中，重命名为hadoop

下载地址：<http://hadoop.apache.org/releases.html>

修改/etc/profile，添加

```
export HADOOP_HOME=/opt/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

不要忘了source /etc/profile

hadoop version来测试

配置hadoop

- hadoop-env.sh

```
export JAVA_HOME=${JAVA_HOME}
#改成：
export JAVA_HOME=/opt/jdk
```

- core-site.xml

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://mip:9000</value>
  </property>
</configuration>
```

其中mip为主机ip：10.88.3.55

- hdfs-site.xml

```
<configuration>
  <property>
    <name>dfs.nameservices</name>
    <value>hadoop-cluster</value>
  </property>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
```

```

        <value>file:///data/hadoop/hdfs/nn</value>
    </property>
    <property>
        <name>dfs.namenode.checkpoint.dir</name>
        <value>file:///data/hadoop/hdfs/snn</value>
    </property>
    <property>
        <name>dfs.namenode.checkpoint.edits.dir</name>
        <value>file:///data/hadoop/hdfs/snn</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>file:///data/hadoop/hdfs/dn</value>
    </property>
</configuration>

```

其中dfs.datanode.data.dir这一个在实际问题中在主节点用了（fs.datanode.data.dir），虽然讲道理dfs也是可以的，但是貌似在实际中有一点问题

- mapred-site.xml

```

<configuration>
    <property>
        <!--指定Mapreduce运行在yarn上-->
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
</configuration>

```

此处，如果遇上中文乱码，服务器可能存在编码问题，可以删去，否则可以修改/etc/environment文件，加入

```

LC_CTYPE=zh_CN.UTF-8
LC_ALL=zh_CN.UTF-8
LANG=zh_CN.UTF-8

```

再重启服务器或者重启语言的配置就行了

- yarn-site.xml

```

<configuration>
    <!-- 指定ResourceManager的地址-->
    <property>
        <name>yarn.resourcemanager.hostname</name>
        <value>mip</value>
    </property>
    <!-- 指定reducer获取数据的方式-->
    <property>
        <name>yarn.nodemanager.aux-services</name>

```

```
        <value>mapreduce_shuffle</value>
    </property>
    <property>
        <name>yarn.nodemanager.local-dirs</name>
        <value>file:///data/hadoop/yarn/nm</value>
    </property>
</configuration>
```

此处mip也是主节点ip: 10.88.3.55 (当然也可以加上用户名)

- 创建上面配置的目录

```
sudo mkdir -p /data/hadoop/hdfs/nn
sudo mkdir -p /data/hadoop/hdfs/dn
sudo mkdir -p /data/hadoop/hdfs/snn
sudo mkdir -p /data/hadoop/yarn/nm
```

在所有结点都同样配置之后, 可以启动hadoop集群:

HDFS集群的启动/关闭

- 启动/关闭名字节点: **hadoop-daemon.sh start/stop namenode**
- 启动/关闭第二名字节点: **hadoop-daemon.sh start/stop secondarynamenode**
- 启动/关闭数据节点: **hadoop-daemon.sh start/stop datanode**

yarn集群的启动/关闭

- 启动/关闭资源管理器: **yarn-daemon.sh start/stop resourcemanager**
- 启动/关闭节点管理器: **yarn-daemon.sh start/stop nodemanager**

启动 / 关闭MR作业日志服务器

- **mr-jobhistory-daemon.sh start/stop historyserver**

- 名字节点、资源管理器: 这是在主节点中启动或关闭的。

数据节点、节点管理器: 这是在从节点中启动或关闭的。

MR作业日志管理器: 这是在主节点中启动或关闭的。

- web端打开: mip:50070 (此处为10.88.3.55)
- 按照命令启动之后, 发现在web端, 没有datanode, 查看datanode的logs, 发现: 遇到datanode启动但不能初始化的问题, 报错信息: Datanode denied communication with namenode because hostname cannot be resolved

在主节点hdfs-site.xml添加

```
<property>
  <name>dfs.namenode.datanode.registration.ip-hostname-check</name>
  <value>false</value>
</property>
```

- 上面的各个操作要是遇到了权限不足的问题，都可以通过下面的语句来加权限

```
sudo chmod -R 777 filename
```

- 配置主从节点之间的免密登录

- 在所有的主从节点中执行

如果以前配置过免密登录的话，建议删除重新建立过，因为我们需要配置的是多台服务器：

```
rm -r ~/.ssh
```

在~/文件下，执行

```
ssh-keygen
```

为了在主节点中生成公钥和私钥，在从节点生成.ssh目录，这个中间步骤一直回车就好

- 在主节点中执行

```
scp ~/.ssh/id_rsa.pub 从节点的用户名@从节点ip:~
```

注意：第一次远程连接的话，首先输入yes，然后是从节点密码

- 在所有的从节点中执行

我们把主节点的公钥已经拿到了所有的从节点中，接下来就是：

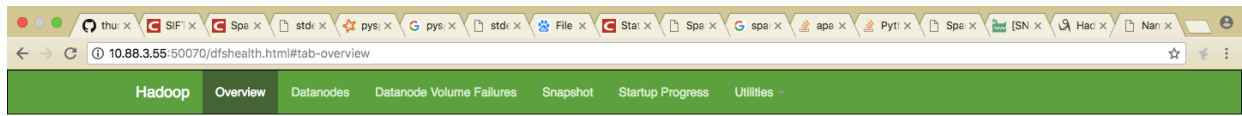
```
cat id_rsa.pub >> .ssh/authorized_keys
```

- 实现主节点控制从节点

- 在主节点中，在/opt/hadoop/etc/hadoop/slaves中添加相应的从节点用户名@ip（这样，用户名和主节点不一样也可以了）
- 在主节点中

```
cat .ssh/id_rsa.pub >> .ssh/authorized_keys
```

至此，hadoop集群已安装完毕，web端效果图：



Overview '10.88.3.55:9000' (active)

Started:	Mon Jun 11 11:31:52 +0800 2018
Version:	2.8.3, rb3fe56402d908019d99af1f1f4fc65cb1d1436a2
Compiled:	Tue Dec 05 11:43:00 +0800 2017 by jdu from branch-2.8.3
Cluster ID:	CID-10c58730-4d54-443f-alfa-2d466e0abeeb
Block Pool ID:	BP-1448653284-127.0.1.1-1528633548574

Summary

Security is off.

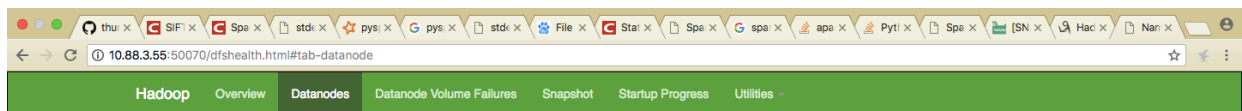
Safemode is off.

8 files and directories, 4 blocks = 12 total filesystem object(s).

Heap Memory used 137.67 MB of 322 MB Heap Memory. Max Heap Memory is 889 MB.

Non Heap Memory used 51.85 MB of 53.06 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

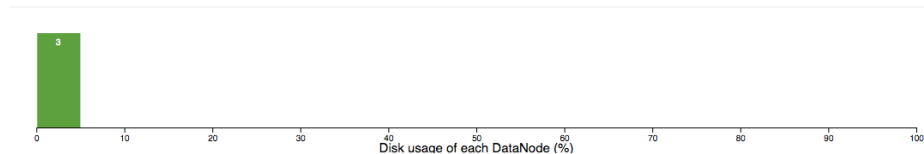
Configured Capacity:	416.09 GB
DFS Used:	202.21 MB (0.05%)
Non DFS Used:	49.01 GB
DFS Remaining:	345.6 GB (83.06%)
Block Pool Used:	202.21 MB (0.05%)



Datanode Information

✓ In service ⬇ Down ✂ Decommissioned ⬇ Decommissioned & dead

Datanode usage histogram



In operation

Show25entries

Search:

Node	Http Address	Last contact	Capacity	Blocks	Block pool used	Version
✓10-88-3-55:50010 (10.88.3.55:50010)	http://10-88-3-55:50075	1s	198.01 GB	4	202.15 MB (0.1%)	2.8.3
✓yang01:50010 (10.88.3.81:50010)	http://yang01:50075	1s	109.04 GB	0	32 KB (0%)	2.8.3
✓yang02:50010 (10.88.3.82:50010)	http://yang02:50075	2s	109.04 GB	0	28 KB (0%)	2.8.3

Showing 1 to 3 of 3 entries

Previous

1

Next

安装scala

下载scala安装包，并解压到/opt文件夹下，重命名为scala2

修改/etc/profile，添加

```
export SCALA_HOME=/opt/scala2
export PATH=$PATH:$SCALA_HOME/bin
```

scala -version 来测试

安装spark

下载spark安装包，并解压到/opt文件夹下，重命名为spark2.2

ps: 下载地址为<https://www.apache.org/dyn/closer.lua/spark/spark-2.2.0/spark-2.2.0-bin-hadoop2.7.tgz>

修改/etc/profile，添加

```
export SPARK_HOME=/opt/spark2.2
export PATH=$SPARK_HOME/bin:$PATH
```

pyspark来测试

测试/opt/spark/bin目录中的spark-shell脚本是否可以运行，来测试scala版本与spark是否配套

配置spark

- /opt/spark/conf/spark-env.sh文件配置，添加

```
export JAVA_HOME=/opt/jdk
export SCALA_HOME=/opt/scala2
export HADOOP_HOME=/opt/hadoop

export STANDALONE_SPARK_MASTER_HOST=10.88.3.55
export SPARK_MASTER_IP=$STANDALONE_SPARK_MASTER_HOST

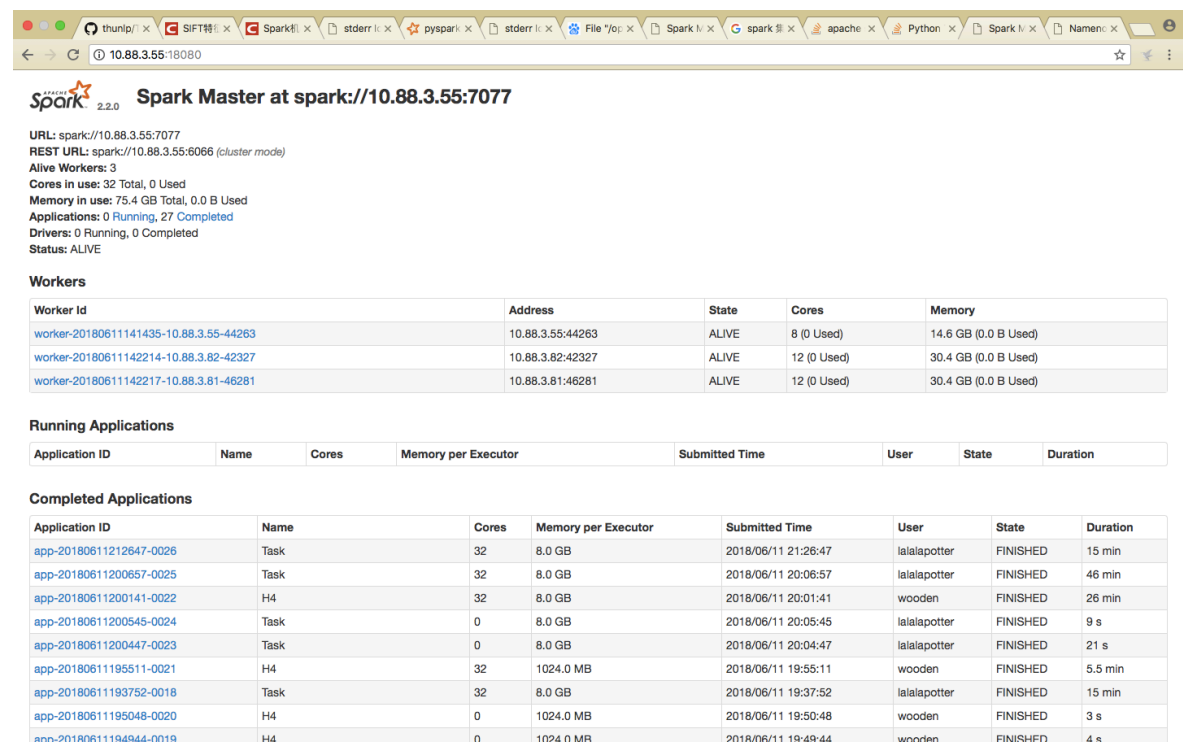
export SPARK_LAUNCH_WITH_SCALA=0
export SPARK_LIBRARY_PATH=${SPARK_HOME}/lib
export SCALA_LIBRARY_PATH=${SPARK_HOME}/lib
export SPARK_MASTER_WEBUI_PORT=18080
#此处设置端口为18080，由于默认的8080端口很有可能被其他程序使用，所以先让出来

if [ -n "$HADOOP_HOME" ]; then
    export
    SPARK_LIBRARY_PATH=$SPARK_LIBRARY_PATH:${HADOOP_HOME}/lib/native
fi
```

- 在同一目录下，slaves文件中，添加三个worker节点的用户名@ip
- 每个节点都这样配置
- 利用10.88.3.55:18080查看sparkUI
- 启动spark要在hadoop启动的前提下
- 遇到spark启动之后，在web端没有显示worker的情况，在主从节点的/opt/spark/sbin目录下利用

```
./start-master.sh -h 10.88.3.55
./start-slave.sh spark://10.88.3.55:7077
```

分别启动master和worker，这样，spark集群就启动了，在web端也有了显示：



Spark Master at spark://10.88.3.55:7077

URL: spark://10.88.3.55:7077
REST URL: spark://10.88.3.55:6066 (cluster mode)
Alive Workers: 3
Cores in use: 32 Total, 0 Used
Memory in use: 75.4 GB Total, 0.0 B Used
Applications: 0 Running, 27 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20180611141435-10.88.3.55-44263	10.88.3.55:44263	ALIVE	8 (0 Used)	14.6 GB (0.0 B Used)
worker-20180611142214-10.88.3.82-42327	10.88.3.82:42327	ALIVE	12 (0 Used)	30.4 GB (0.0 B Used)
worker-20180611142217-10.88.3.81-46281	10.88.3.81:46281	ALIVE	12 (0 Used)	30.4 GB (0.0 B Used)

Running Applications

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

Completed Applications

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
app-20180611212647-0026	Task	32	8.0 GB	2018/06/11 21:26:47	lalalapotter	FINISHED	15 min
app-20180611200657-0025	Task	32	8.0 GB	2018/06/11 20:06:57	lalalapotter	FINISHED	46 min
app-20180611200141-0022	H4	32	8.0 GB	2018/06/11 20:01:41	wooden	FINISHED	26 min
app-20180611200545-0024	Task	0	8.0 GB	2018/06/11 20:05:45	lalalapotter	FINISHED	9 s
app-20180611200447-0023	Task	0	8.0 GB	2018/06/11 20:04:47	lalalapotter	FINISHED	21 s
app-20180611195511-0021	H4	32	1024.0 MB	2018/06/11 19:55:11	wooden	FINISHED	5.5 min
app-20180611193752-0018	Task	32	8.0 GB	2018/06/11 19:37:52	lalalapotter	FINISHED	15 min
app-20180611195048-0020	H4	0	1024.0 MB	2018/06/11 19:50:48	wooden	FINISHED	3 s
app-20180611194944-0019	H4	0	1024.0 MB	2018/06/11 19:49:44	wooden	FINISHED	4 s

- 运行测试程序是遇到以下错误代码：

```
18/06/11 15:52:49 ERROR Executor: Exception in task 5.3 in stage 0.0
(TID 8)
org.apache.spark.api.python.PythonException: Traceback (most recent
call last):
  File "/opt/spark2.2/python/lib/pyspark.zip/pyspark/worker.py", line
166, in main
    func, profiler, deserializer, serializer = read_command(pickleSer,
infile)
  File "/opt/spark2.2/python/lib/pyspark.zip/pyspark/worker.py", line
55, in read_command
    command = serializer._read_with_length(file)
  File "/opt/spark2.2/python/lib/pyspark.zip/pyspark/serializers.py",
line 169, in _read_with_length
    return self.loads(obj)
  File "/opt/spark2.2/python/lib/pyspark.zip/pyspark/serializers.py",
line 454, in loads
    return pickle.loads(obj)
  File "/opt/spark2.2/python/lib/pyspark.zip/pyspark/cloudpickle.py",
line 784, in _make_skel_func
    closure = _reconstruct_closure(closures) if closures else None
  File "/opt/spark2.2/python/lib/pyspark.zip/pyspark/cloudpickle.py",
line 776, in _reconstruct_closure
    return tuple([_make_cell(v) for v in values])
TypeError: 'int' object is not iterable
```

检查后发现是pyspark的版本不对齐，对齐后问题解决

总结

至此，hadoop集群和spark集群的能够正常运行了，果然人还是要逼着自己才能干点事。