**[Flink（二）CentOS7.5搭建Flink1.6.1分布式集群](https://www.cnblogs.com/frankdeng/p/9400627.html)**

**一. Flink的下载**

安装包下载地址：[http://flink.apache.org/downloads.html](http://flink.apache.org/downloads.html" \t "_blank)  ，选择对应Hadoop的Flink版本下载

[admin@bigdata11 software]$ wget http://mirrors.tuna.tsinghua.edu.cn/apache/flink/flink-1.6.1/flink-1.6.1-bin-hadoop27-scala\_2.11.tgz

[admin@bigdata11 software]$ ll

-rw-rw-r-- 1 admin admin 301867081 Sep 15 15:47 flink-1.6.1-bin-hadoop27-scala\_2.11.tgz

**Flink 有三种部署模式，分别是 Local、Standalone Cluster 和 Yarn Cluster。**

**二. Local模式**

对于 Local 模式来说，JobManager 和 TaskManager 会公用一个 JVM 来完成 Workload。如果要验证一个简单的应用，Local 模式是最方便的。实际应用中大多使用 Standalone 或者 Yarn Cluster，而local模式只是将安装包解压启动（./bin/start-local.sh）即可，在这里不在演示。

**三. Standalone 模式**

**快速入门教程地址：**[**https://ci.apache.org/projects/flink/flink-docs-release-1.6/quickstart/setup\_quickstart.html**](https://ci.apache.org/projects/flink/flink-docs-release-1.6/quickstart/setup_quickstart.html)

**1.  软件要求**

* Java 1.8.x或更高版本，
* ssh（必须运行sshd才能使用管理远程组件的Flink脚本）

集群部署规划

|  |  |  |  |
| --- | --- | --- | --- |
| 节点名称 | master | worker | zookeeper |
| bigdata11 | master |  | zookeeper |
| bigdata12 | master | worker | zookeeper |
| bigdata13 |  | worker | zookeeper |

**2. 解压**

[admin@bigdata11 software]$ tar zxvf flink-1.6.1-bin-hadoop27-scala\_2.11.tgz -C /opt/module/

[admin@bigdata11 software]$ cd /opt/module/

[admin@bigdata11 module]$ ll

drwxr-xr-x 8 admin admin 125 Sep 15 04:47 flink-1.6.1

**3. 修改配置文件**

[admin@bigdata11 conf]$ ls

flink-conf.yaml log4j-console.properties log4j-yarn-session.properties logback.xml masters sql-client-defaults.yaml

log4j-cli.properties log4j.properties logback-console.xml logback-yarn.xml slaves zoo.cfg

修改flink/conf/masters，slaves，flink-conf.yaml

[admin@bigdata11 conf]$ sudo vi masters

bigdata11:8081

[admin@bigdata11 conf]$ sudo vi slaves

bigdata12

bigdata13

[admin@bigdata11 conf]$ sudo vi flink-conf.yaml

taskmanager.numberOfTaskSlots：2

jobmanager.rpc.address: bigdata11

可选配置：

* 每个JobManager（jobmanager.heap.mb）的可用内存量，
* 每个TaskManager（taskmanager.heap.mb）的可用内存量，
* 每台机器的可用CPU数量（taskmanager.numberOfTaskSlots），
* 集群中的CPU总数（parallelism.default）和
* 临时目录（taskmanager.tmp.dirs）

**4. 拷贝安装包到各节点**

[admin@bigdata11 module]$ scp -r flink-1.6.1/ admin@bigdata12:`pwd`

[admin@bigdata11 module]$ scp -r flink-1.6.1/ admin@bigdata13:`pwd`

**5. 配置环境变量**

配置所有节点Flink的环境变量

[admin@bigdata11 flink-1.6.1]$ sudo vi /etc/profile

export FLINK\_HOME=/opt/module/flink-1.6.1

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

[admin@bigdata11 flink-1.6.1]$ source /etc/profile

**6. 启动flink**

[admin@bigdata11 flink-1.6.1]$ ./bin/start-cluster.sh

Starting cluster.

Starting standalonesession daemon on host bigdata11.

Starting taskexecutor daemon on host bigdata12.

Starting taskexecutor daemon on host bigdata13.

jps查看进程

**7.  WebUI查看**

[http://bigdata11:8081](http://node21:8081/)

**8. Flink 的 HA**

首先，我们需要知道 Flink 有两种部署的模式，分别是 Standalone 以及 Yarn Cluster 模式。对于 Standalone 来说，Flink 必须依赖于 Zookeeper 来实现 JobManager 的 HA（Zookeeper 已经成为了大部分开源框架 HA 必不可少的模块）。在 Zookeeper 的帮助下，一个 Standalone 的 Flink 集群会同时有多个活着的 JobManager，其中只有一个处于工作状态，其他处于 Standby 状态。当工作中的 JobManager 失去连接后（如宕机或 Crash），Zookeeper 会从 Standby 中选举新的 JobManager 来接管 Flink 集群。

对于 Yarn Cluaster 模式来说，Flink 就要依靠 Yarn 本身来对 JobManager 做 HA 了。其实这里完全是 Yarn 的机制。对于 Yarn Cluster 模式来说，JobManager 和 TaskManager 都是被 Yarn 启动在 Yarn 的 Container 中。此时的 JobManager，其实应该称之为 Flink Application Master。也就说它的故障恢复，就完全依靠着 Yarn 中的 ResourceManager（和 MapReduce 的 AppMaster 一样）。由于完全依赖了 Yarn，因此不同版本的 Yarn 可能会有细微的差异。这里不再做深究。

**1） 修改配置文件**

修改flink-conf.yaml，HA模式下，jobmanager不需要指定，在master file中配置，由zookeeper选出leader与standby。

#jobmanager.rpc.address: bigdata11

high-availability:zookeeper

#指定高可用模式（必须）

high-availability.zookeeper.quorum:bigdata11:2181,bigdata12:2181,bigdata13:2181

#ZooKeeper仲裁是ZooKeeper服务器的复制组，它提供分布式协调服务（必须）

high-availability.storageDir:hdfs:///flink/ha/

#JobManager元数据保存在文件系统storageDir中，只有指向此状态的指针存储在ZooKeeper中（必须）

high-availability.zookeeper.path.root:/flink

#根ZooKeeper节点，在该节点下放置所有集群节点（推荐）

high-availability.cluster-id:/flinkCluster

＃自定义集群（推荐）

state.backend: filesystem

state.checkpoints.dir: hdfs:///flink/checkpoints

state.savepoints.dir: hdfs:///flink/checkpoints

修改conf/zoo.cfg

server.1=bigdata11:2888:3888

server.2=bigdata12:2888:3888

server.3=bigdata13:2888:3888

修改conf/masters

bigdata11:8081

bigdata12:8081

修改slaves

bigdata12

bigdata13

同步配置文件conf到各节点

**2） 启动HA**

先启动zookeeper集群各节点（测试环境中也可以用Flink自带的start-zookeeper-quorum.sh），启动dfs ,再启动flink

[admin@bigdata11 flink-1.6.1]$ start-cluster.sh

WebUI查看，这是会自动产生一个主Master，如下

**3） 验证HA**

手动杀死bigdata12上的master，此时，bigdata11上的备用master转为主mater。

**4）手动将JobManager / TaskManager实例添加到群集**

您可以使用bin/jobmanager.sh和bin/taskmanager.sh脚本将JobManager和TaskManager实例添加到正在运行的集群中。

添加JobManager

bin/jobmanager.sh ((start|start-foreground) [host] [webui-port])|stop|stop-all

添加TaskManager

bin/taskmanager.sh start|start-foreground|stop|stop-all

[admin@bigdata12 flink-1.6.1]$ jobmanager.sh start bigdata12

新添加的为从master。

**9. 运行测试任务**

[admin@bigdata11 flink-1.6.1]$ flink run -m bigdata11:8081 ./examples/batch/WordCount.jar --input /opt/wcinput/wc.txt --output /opt/wcoutput/

[admin@bigdata11 flink-1.6.1]$ flink run -m bigdata11:8081 ./examples/batch/WordCount.jar --input hdfs:///user/admin/input/wc.txt --output hdfs:///user/admin/output2

**四. Yarn Cluster模式**

**1. 引入**

在一个企业中，为了最大化的利用集群资源，一般都会在一个集群中同时运行多种类型的 Workload。因此 Flink 也支持在 Yarn 上面运行。首先，让我们通过下图了解下 Yarn 和 Flink 的关系。

在图中可以看出，Flink 与 Yarn 的关系与 MapReduce 和 Yarn 的关系是一样的。Flink 通过 Yarn 的接口实现了自己的 App Master。当在 Yarn 中部署了 Flink，Yarn 就会用自己的 Container 来启动 Flink 的 JobManager（也就是 App Master）和 TaskManager。

启动新的Flink YARN会话时，客户端首先检查所请求的资源（容器和内存）是否可用。之后，它将包含Flink和配置的jar上传到HDFS（步骤1）。

客户端的下一步是请求（步骤2）YARN容器以启动*ApplicationMaster*（步骤3）。由于客户端将配置和jar文件注册为容器的资源，因此在该特定机器上运行的YARN的NodeManager将负责准备容器（例如，下载文件）。完成后，将启动*ApplicationMaster*（AM）。

该*JobManager*和AM在同一容器中运行。一旦它们成功启动，AM就知道JobManager（它自己的主机）的地址。它正在为TaskManagers生成一个新的Flink配置文件（以便它们可以连接到JobManager）。该文件也上传到HDFS。此外，*AM*容器还提供Flink的Web界面。YARN代码分配的所有端口都是*临时端口*。这允许用户并行执行多个Flink YARN会话。

之后，AM开始为Flink的TaskManagers分配容器，这将从HDFS下载jar文件和修改后的配置。完成这些步骤后，即可建立Flink并准备接受作业。

**2. 修改环境变量**

export  HADOOP\_CONF\_DIR= /opt/module/hadoop-2.7.6/etc/hadoop

**3. 部署启动**

[admin@bigdata11 flink-1.6.1]$ yarn-session.sh -d -s 2 -tm 800 -n 2

-n : TaskManager的数量，相当于executor的数量

-s : 每个JobManager的core的数量，executor-cores。建议将slot的数量设置每台机器的处理器数量

-tm : 每个TaskManager的内存大小，executor-memory

-jm : JobManager的内存大小，driver-memory

上面的命令的意思是，同时向Yarn申请3个container，其中 2 个 Container 启动 TaskManager（-n 2），每个 TaskManager 拥有两个 Task Slot（-s 2），并且向每个 TaskManager 的 Container 申请 800M 的内存，以及一个ApplicationMaster（Job Manager）。

Flink部署到Yarn Cluster后，会显示Job Manager的连接细节信息。

Flink on Yarn会覆盖下面几个参数，如果不希望改变配置文件中的参数，可以动态的通过-D选项指定，如 -Dfs.overwrite-files=true -Dtaskmanager.network.numberOfBuffers=16368

jobmanager.rpc.address：因为JobManager会经常分配到不同的机器上

taskmanager.tmp.dirs：使用Yarn提供的tmp目录

parallelism.default：如果有指定slot个数的情况下

yarn-session.sh会挂起进程，所以可以通过在终端使用CTRL+C或输入stop停止yarn-session。

如果不希望Flink Yarn client长期运行，Flink提供了一种detached YARN session，启动时候加上参数-d或—detached

在上面的命令成功后，我们就可以在 Yarn Application 页面看到 Flink 的纪录。如下图。

如果在虚拟机中测试，可能会遇到错误。这里需要注意内存的大小，Flink 向 Yarn 会申请多个 Container，但是 Yarn 的配置可能限制了 Container 所能申请的内存大小，甚至 Yarn 本身所管理的内存就很小。这样很可能无法正常启动 TaskManager，尤其当指定多个 TaskManager 的时候。因此，在启动 Flink 之后，需要去 Flink 的页面中检查下 Flink 的状态。这里可以从 RM 的页面中，直接跳转（点击 Tracking UI）。这时候 Flink 的页面如图

yarn-session.sh启动命令参数如下：

[admin@bigdata11 flink-1.6.1]$ yarn-session.sh --help

Usage:

Required

-n,--container <arg> Number of YARN container to allocate (=Number of Task Managers)

Optional

-D <property=value> use value for given property

-d,--detached If present, runs the job in detached mode

-h,--help Help for the Yarn session CLI.

-id,--applicationId <arg> Attach to running YARN session

-j,--jar <arg> Path to Flink jar file

-jm,--jobManagerMemory <arg> Memory for JobManager Container with optional unit (default: MB)

-m,--jobmanager <arg> Address of the JobManager (master) to which to connect. Use this flag to connect to a different JobManager than the one specified i

n the configuration. -n,--container <arg> Number of YARN container to allocate (=Number of Task Managers)

-nl,--nodeLabel <arg> Specify YARN node label for the YARN application

-nm,--name <arg> Set a custom name for the application on YARN

-q,--query Display available YARN resources (memory, cores)

-qu,--queue <arg> Specify YARN queue.

-s,--slots <arg> Number of slots per TaskManager

-st,--streaming Start Flink in streaming mode

-t,--ship <arg> Ship files in the specified directory (t for transfer)

-tm,--taskManagerMemory <arg> Memory per TaskManager Container with optional unit (default: MB)

-yd,--yarndetached If present, runs the job in detached mode (deprecated; use non-YARN specific option instead)

-z,--zookeeperNamespace <arg> Namespace to create the Zookeeper sub-paths for high availability mode

**4. 提交任务**

之后，我们可以通过这种方式提交我们的任务

[admin@bigdata11 flink-1.6.1]$ ./bin/flink run -m yarn-cluster -yn 2 ./examples/batch/WordCount.jar --input /opt/wcinput/wc.txt --output /opt/wcoutput/

以上命令在参数前加上y前缀，-yn表示TaskManager个数。

在这个模式下，同样可以使用-m yarn-cluster提交一个"运行后即焚"的detached yarn（-yd）作业到yarn cluster。

**5. 停止yarn cluster**

yarn application -kill application\_1539058959130\_0001

**6. Yarn模式的HA**

**应用最大尝试次数（yarn-site.xml），您必须配置为尝试应用的最大数量的设置yarn-site.xml，当前YARN版本的默认值为2（表示允许单个JobManager失败）。**

<property>

<name>yarn.resourcemanager.am.max-attempts</name>

<value>4</value>

<description>The maximum number of application master execution attempts</description>

</property>

**申请尝试（flink-conf.yaml），您还必须配置最大尝试次数conf/flink-conf.yaml： yarn.application-attempts：10**

**示例：高度可用的YARN会话**

1. 配置HA模式和zookeeper法定人数在conf/flink-conf.yaml：
2. high-availability: zookeeper
3. high-availability.zookeeper.quorum: bigdata11:2181,bigdata12:2181,bigdata13:2181
4. high-availability.storageDir: hdfs:///flink/recovery
5. high-availability.zookeeper.path.root: /flink

yarn.application-attempts: 10

1. 配置ZooKeeper的服务器中conf/zoo.cfg（目前它只是可以运行每台机器的单一的ZooKeeper服务器）：
2. server.1=bigdata11:2888:3888
3. server.2=bigdata12:2888:3888

server.3=bigdata13:2888:3888

1. 启动ZooKeeper仲裁：

$ bin / start-zookeeper-quorum.sh

1. 启动HA群集：

$ bin / yarn-session.sh -n 2

**五.错误异常**

**1.身份认证失败**

[root@bigdata11 flink-1.6.1]# flink run examples/streaming/SocketWindowWordCount.jar --port 9000

Starting execution of program

------------------------------------------------------------

The program finished with the following exception:

org.apache.flink.client.program.ProgramInvocationException: Job failed. (JobID: b7a99ac5db242290413dbebe32ba52b0)

at org.apache.flink.client.program.rest.RestClusterClient.submitJob(RestClusterClient.java:267)

at org.apache.flink.client.program.ClusterClient.run(ClusterClient.java:486)

at org.apache.flink.streaming.api.environment.StreamContextEnvironment.execute(StreamContextEnvironment.java:66)

at org.apache.flink.streaming.examples.socket.SocketWindowWordCount.main(SocketWindowWordCount.java:92)

at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)

at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)

at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)

at java.lang.reflect.Method.invoke(Method.java:498)

at org.apache.flink.client.program.PackagedProgram.callMainMethod(PackagedProgram.java:529)

at org.apache.flink.client.program.PackagedProgram.invokeInteractiveModeForExecution(PackagedProgram.java:421)

at org.apache.flink.client.program.ClusterClient.run(ClusterClient.java:426)

at org.apache.flink.client.cli.CliFrontend.executeProgram(CliFrontend.java:804)

at org.apache.flink.client.cli.CliFrontend.runProgram(CliFrontend.java:280)

at org.apache.flink.client.cli.CliFrontend.run(CliFrontend.java:215)

at org.apache.flink.client.cli.CliFrontend.parseParameters(CliFrontend.java:1044)

at org.apache.flink.client.cli.CliFrontend.lambda$main$11(CliFrontend.java:1120)

at java.security.AccessController.doPrivileged(Native Method)

at javax.security.auth.Subject.doAs(Subject.java:422)

at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1754)

at org.apache.flink.runtime.security.HadoopSecurityContext.runSecured(HadoopSecurityContext.java:41)

at org.apache.flink.client.cli.CliFrontend.main(CliFrontend.java:1120)

Caused by: java.net.ConnectException: Connection refused (Connection refused)

at java.net.PlainSocketImpl.socketConnect(Native Method)

at java.net.AbstractPlainSocketImpl.doConnect(AbstractPlainSocketImpl.java:350)

at java.net.AbstractPlainSocketImpl.connectToAddress(AbstractPlainSocketImpl.java:206)

at java.net.AbstractPlainSocketImpl.connect(AbstractPlainSocketImpl.java:188)

at java.net.SocksSocketImpl.connect(SocksSocketImpl.java:392)

at java.net.Socket.connect(Socket.java:589)

at org.apache.flink.streaming.api.functions.source.SocketTextStreamFunction.run(SocketTextStreamFunction.java:96)

at org.apache.flink.streaming.api.operators.StreamSource.run(StreamSource.java:87)

at org.apache.flink.streaming.api.operators.StreamSource.run(StreamSource.java:56)

at org.apache.flink.streaming.runtime.tasks.SourceStreamTask.run(SourceStreamTask.java:99)

at org.apache.flink.streaming.runtime.tasks.StreamTask.invoke(StreamTask.java:300)

at org.apache.flink.runtime.taskmanager.Task.run(Task.java:711)

at java.lang.Thread.run(Thread.java:748)

通过查看日志，发现有如下报错

2018-10-20 02:32:19,668 ERROR org.apache.flink.shaded.curator.org.apache.curator.ConnectionState - Authentication failed

解决法案：添加定时任务认证kerberos