CS 696 Intro to Big Data: Tools and Methods
Fall Semester, 2016
Doc 26 Spark Install
Dec 6, 2016

Installing & Running Spark

http://spark.apache.org/

Choose “Pre-build for Hadoop 2.7 and later”

Download spark-2.0.2-bin-hadoop2.7.tgz

Unpack the file

Spark download comes with Hadoop HDFS
So do not need Hadoop installed

Install directory

bin

    REPL's for running interactive spark

examples

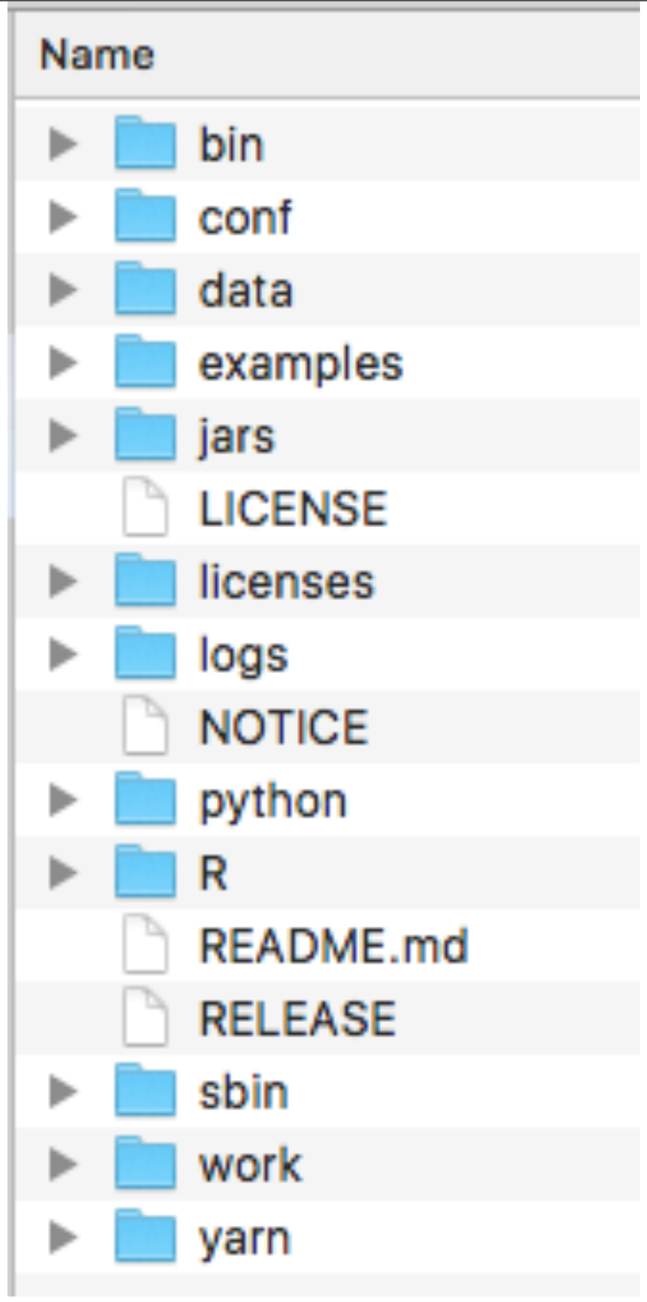
    Lots of Java examples

jars

    All the jars

sbin

    Shell scripts to start and stop


Name
bin
conf
data
examples
jars
LICENSE
licenses
logs
NOTICE
python
R
README.md
RELEASE
sbin
work
yarn

Some Useful Set up

Set SPARK_HOME

Add $SPARK_HOME/bin & $SPARK_HOME/sbin in your path

Spark REPL for Scala :)

spark-shell

```
scala> 1 + 2
res0: Int = 3

scala> val data = sc.parallelize(Array(1, 2, 3, 4))
data: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize at
<console>:24

scala> data.saveAsTextFile("foo.txt")

scala>
```

No repl for Java :(

Sample Program

```
import org.apache.spark.api.java.JavaSparkContext;
import org.apache.spark.api.java.JavaRDD;
import java.util.Arrays;

public final class SampleProgram {
    public static void main(String[] args) throws Exception {

        System.out.println("Start");
        JavaSparkContext sc = new JavaSparkContext();
        JavaRDD<Integer> rdd = sc.parallelize(Arrays.asList(1, 2, 3));
        rdd.saveAsTextFile("outputDir");
        System.out.println("Done");
        sc.stop();
    }
}
```

Compile & Create Jar file

To compile program I put all the jar files in $SPARK_HOME/jars in the classpath

To run the jar file using spark use the command

```
spark-submit --class “SampleProgram” SampleProgram.jar
```

If the jar file contains a manifest file indicating the main class you can drop
    --class “SampleProgram”

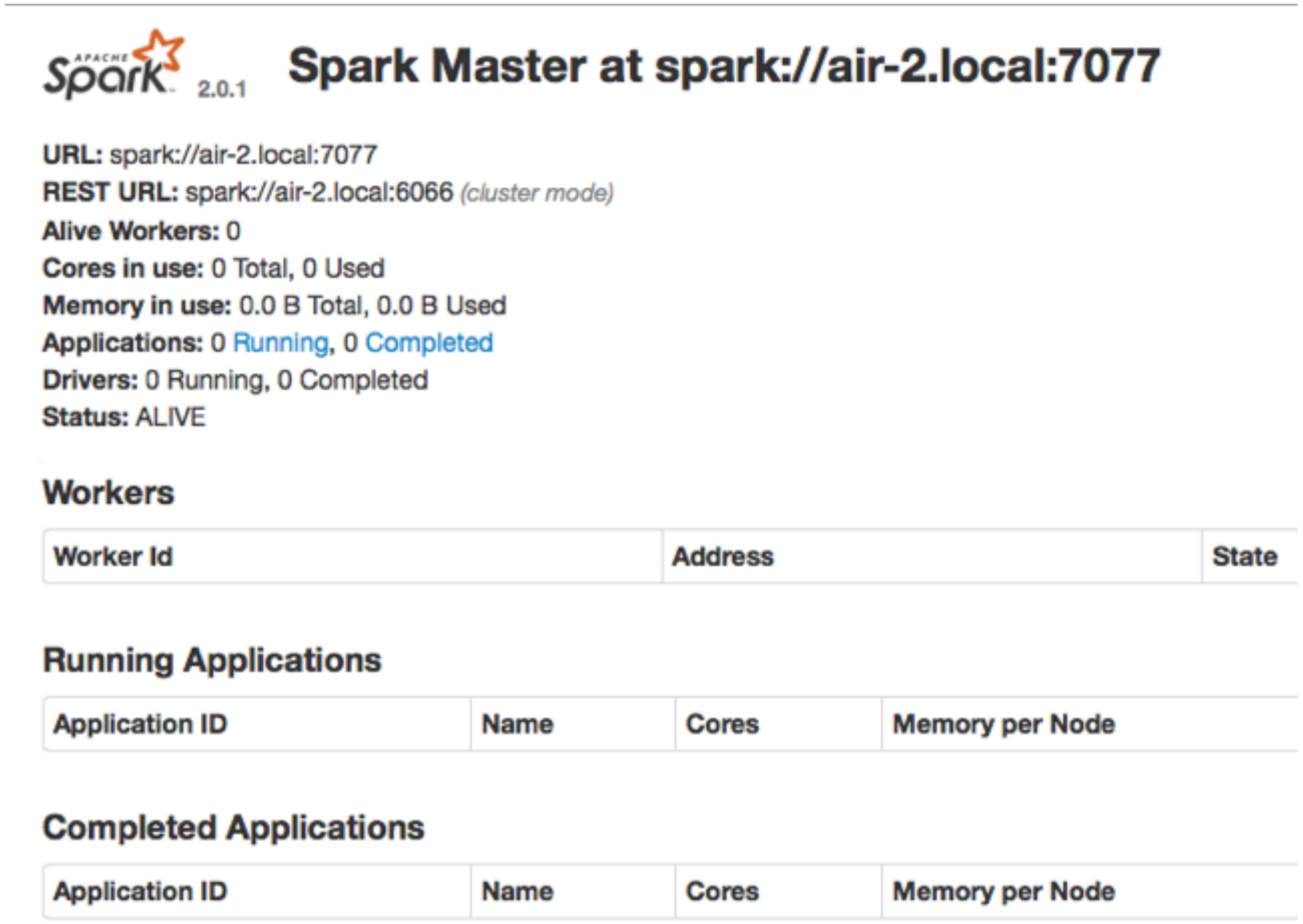
spark-submit
    Located in $SPARK_HOME/bin
    Starts & stops Spark

Starting Spark on Cluster or Single machine

Starting master

start-master.sh (located in $SPARK_HOME/sbin)

View the Web UI at localhost:8080



Spark 2.0.1 **Spark Master at spark://air-2.local:7077**

URL: spark://air-2.local:7077
REST URL: spark://air-2.local:6066 *(cluster mode)*
Alive Workers: 0
Cores in use: 0 Total, 0 Used
Memory in use: 0.0 B Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers

Worker Id	Address	State

Running Applications

Application ID	Name	Cores	Memory per Node

Completed Applications

Application ID	Name	Cores	Memory per Node

air-2.local is name of my machine on my home network

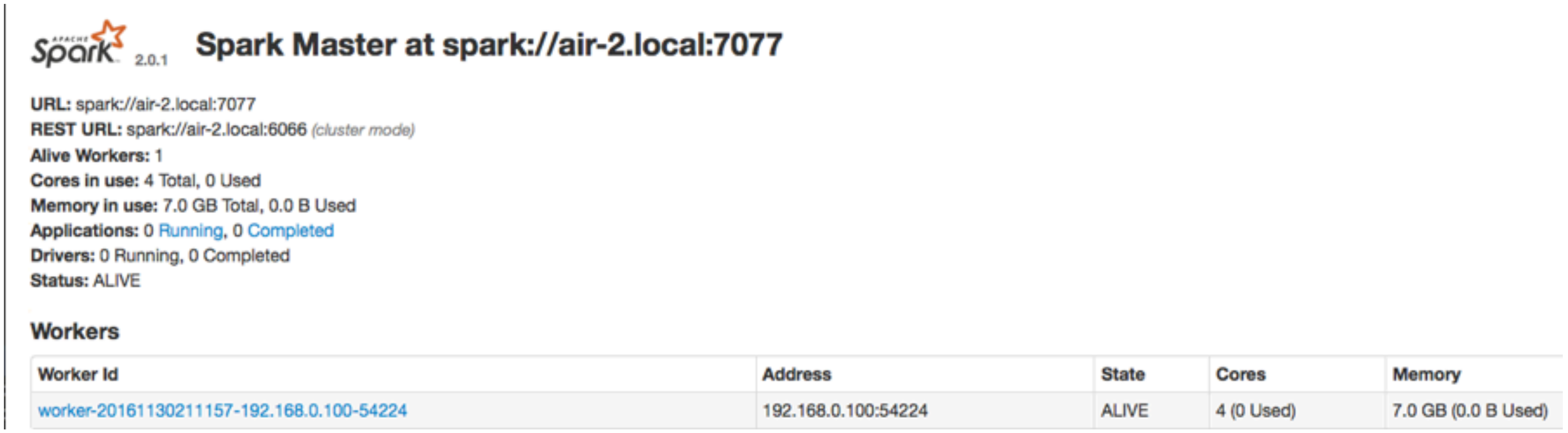
Starting a slave

start-slave.sh spark://air-2.local:7077

start-slave.sh is in $SPARK_HOME/sbin

air-2.local is my machine name, replace it with your machine name

Master Web UI shows Worker & gives access to Worker Web UI



Spark Master at spark://air-2.local:7077

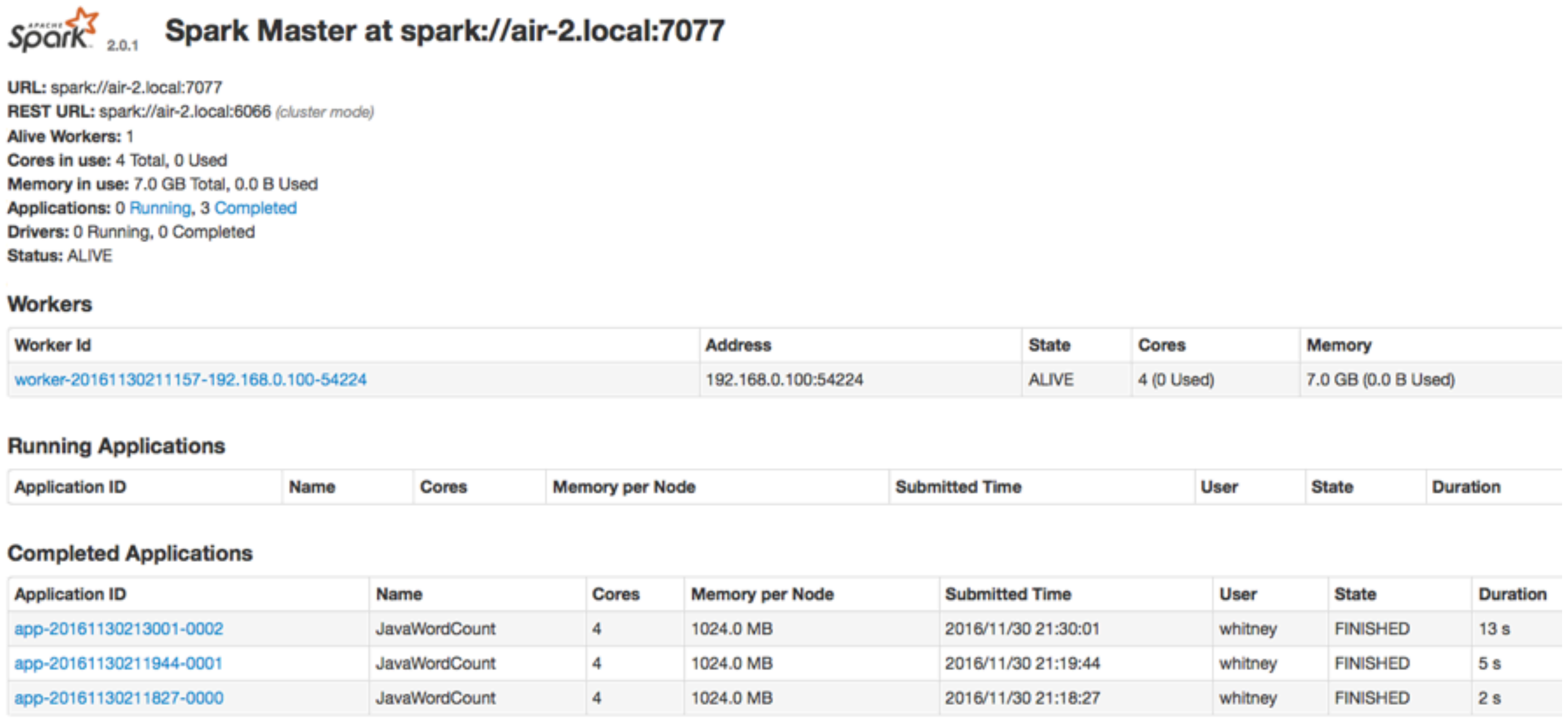
URL: spark://air-2.local:7077
REST URL: spark://air-2.local:6066 *(cluster mode)*
Alive Workers: 1
Cores in use: 4 Total, 0 Used
Memory in use: 7.0 GB Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20161130211157-192.168.0.100-54224	192.168.0.100:54224	ALIVE	4 (0 Used)	7.0 GB (0.0 B Used)

Running Program

```
spark-submit --class "JavaWordCount" --master spark://air-2.local:7077 /
                                             SparkWordCount.jar twain.txt output
```



Spark 2.0.1 **Spark Master at spark://air-2.local:7077**

URL: spark://air-2.local:7077
REST URL: spark://air-2.local:6066 *(cluster mode)*
Alive Workers: 1
Cores in use: 4 Total, 0 Used
Memory in use: 7.0 GB Total, 0.0 B Used
Applications: 0 Running, 3 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20161130211157-192.168.0.100-54224	192.168.0.100:54224	ALIVE	4 (0 Used)	7.0 GB (0.0 B Used)

Running Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration

Completed Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20161130213001-0002	JavaWordCount	4	1024.0 MB	2016/11/30 21:30:01	whitney	FINISHED	13 s
app-20161130211944-0001	JavaWordCount	4	1024.0 MB	2016/11/30 21:19:44	whitney	FINISHED	5 s
app-20161130211827-0000	JavaWordCount	4	1024.0 MB	2016/11/30 21:18:27	whitney	FINISHED	2 s

Twain Word count

```
(154090,the)
(121298,and)
(78844,of)
(72260,a)
(70270,to)
(46934,in)
(43329,i)
(39348,it)
(38431,was)
(37942,that)
```

Hadoop HDFS warning

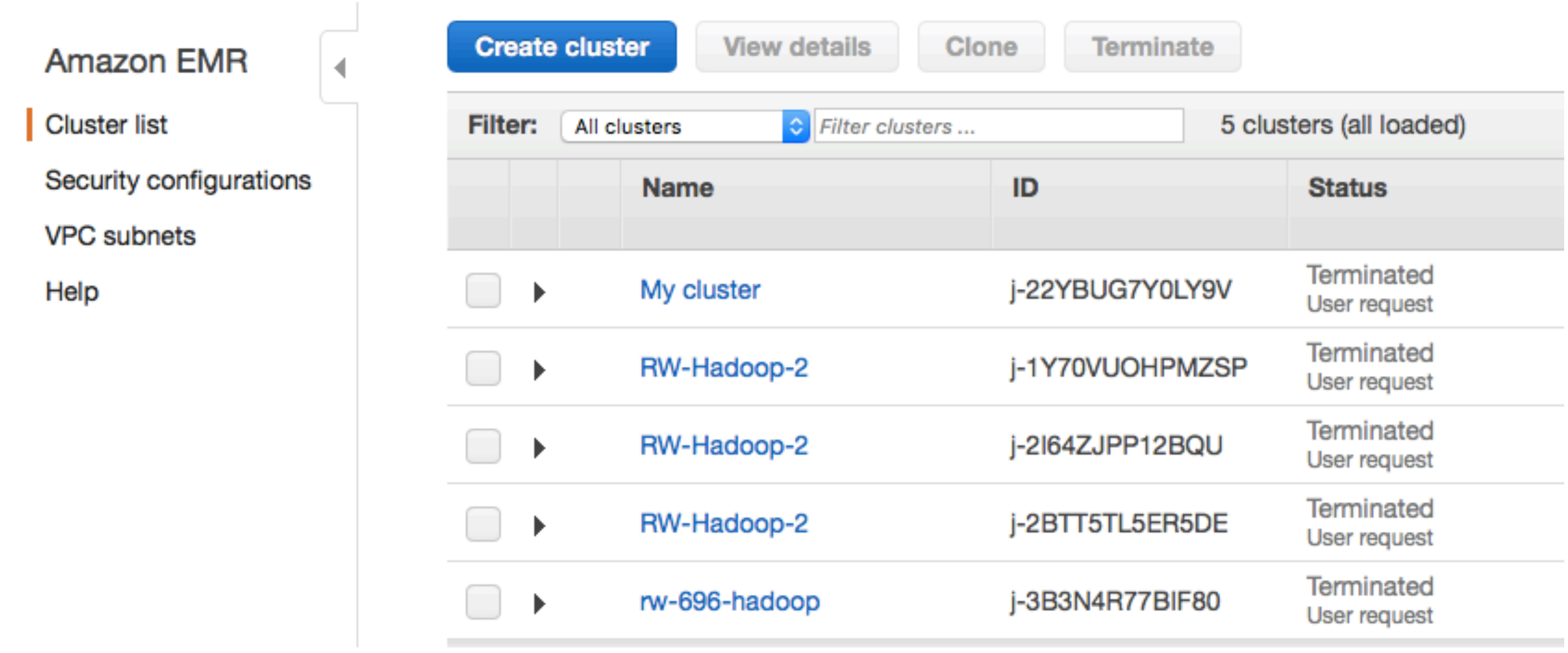
If you have Hadoop installed &

Set the config file to use HDFS &

Set HADOOP_CONF_DIR environmental variable

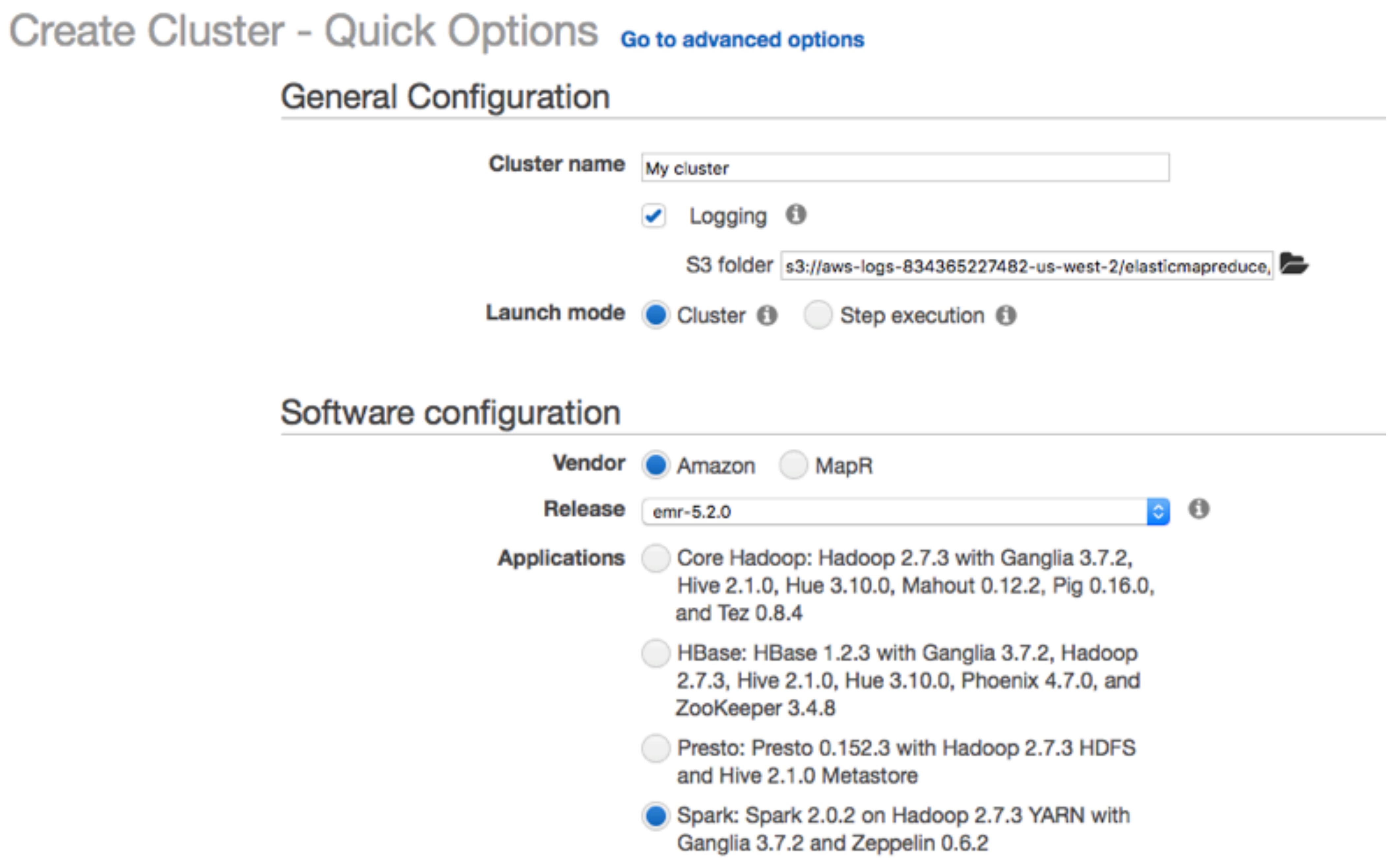
THEN Spark will use HDFS as its file system

Spark on AWS



Name	ID	Status
My cluster	j-22YBUG7Y0LY9V	Terminated User request
RW-Hadoop-2	j-1Y70VUOHPMZSP	Terminated User request
RW-Hadoop-2	j-2I64ZJPP12BQU	Terminated User request
RW-Hadoop-2	j-2BTT5TL5ER5DE	Terminated User request
rw-696-hadoop	j-3B3N4R77BIF80	Terminated User request

You can either use Spark option on Quick Options
or use Advanced Options



Create Cluster - Quick Options Go to advanced options

General Configuration

Cluster name: My cluster

[✓] Logging

S3 folder: s3://aws-logs-834365227482-us-west-2/elasticmapreduce/

Launch mode: [✓ Cluster] Step execution

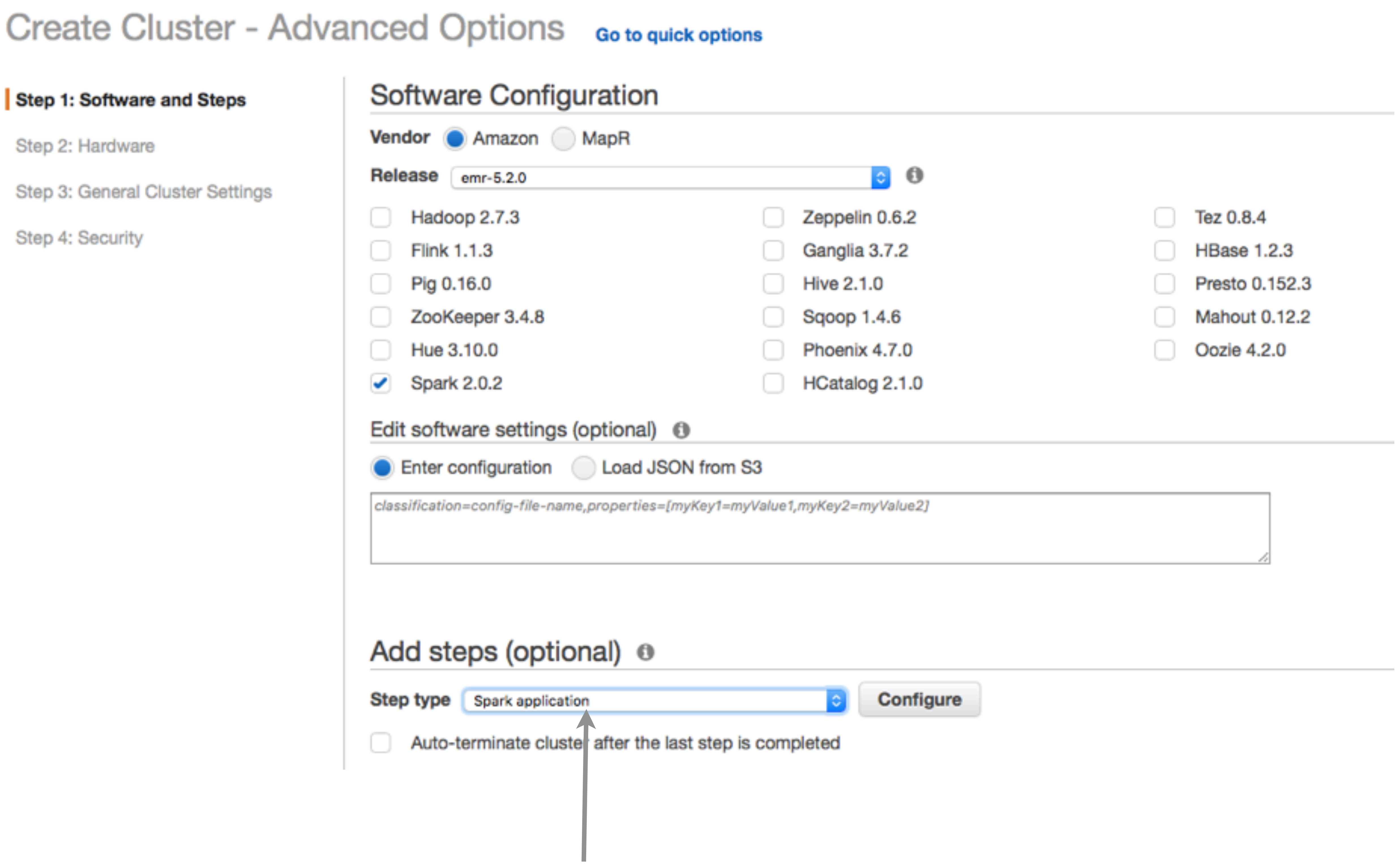
Software configuration

Vendor: [✓ Amazon] MapR

Release: emr-5.2.0

Applications:
- Core Hadoop: Hadoop 2.7.3 with Ganglia 3.7.2, Hive 2.1.0, Hue 3.10.0, Mahout 0.12.2, Pig 0.16.0, and Tez 0.8.4
- HBase: HBase 1.2.3 with Ganglia 3.7.2, Hadoop 2.7.3, Hive 2.1.0, Hue 3.10.0, Phoenix 4.7.0, and ZooKeeper 3.4.8
- Presto: Presto 0.152.3 with Hadoop 2.7.3 HDFS and Hive 2.1.0 Metastore
- [✓ Spark: Spark 2.0.2 on Hadoop 2.7.3 YARN with Ganglia 3.7.2 and Zeppelin 0.6.2]

Advanced Options



Create Cluster - Advanced Options   Go to quick options

Step 1: Software and Steps
Step 2: Hardware
Step 3: General Cluster Settings
Step 4: Security

Software Configuration

Vendor [✓ Amazon] MapR

Release emr-5.2.0

☐ Hadoop 2.7.3	☐ Zeppelin 0.6.2	☐ Tez 0.8.4
☐ Flink 1.1.3	☐ Ganglia 3.7.2	☐ HBase 1.2.3
☐ Pig 0.16.0	☐ Hive 2.1.0	☐ Presto 0.152.3
☐ ZooKeeper 3.4.8	☐ Sqoop 1.4.6	☐ Mahout 0.12.2
☐ Hue 3.10.0	☐ Phoenix 4.7.0	☐ Oozie 4.2.0
✓ Spark 2.0.2	☐ HCatalog 2.1.0	

Edit software settings (optional)

[✓ Enter configuration] Load JSON from S3

classification=config-file-name,properties=[myKey1=myValue1,myKey2=myValue2]

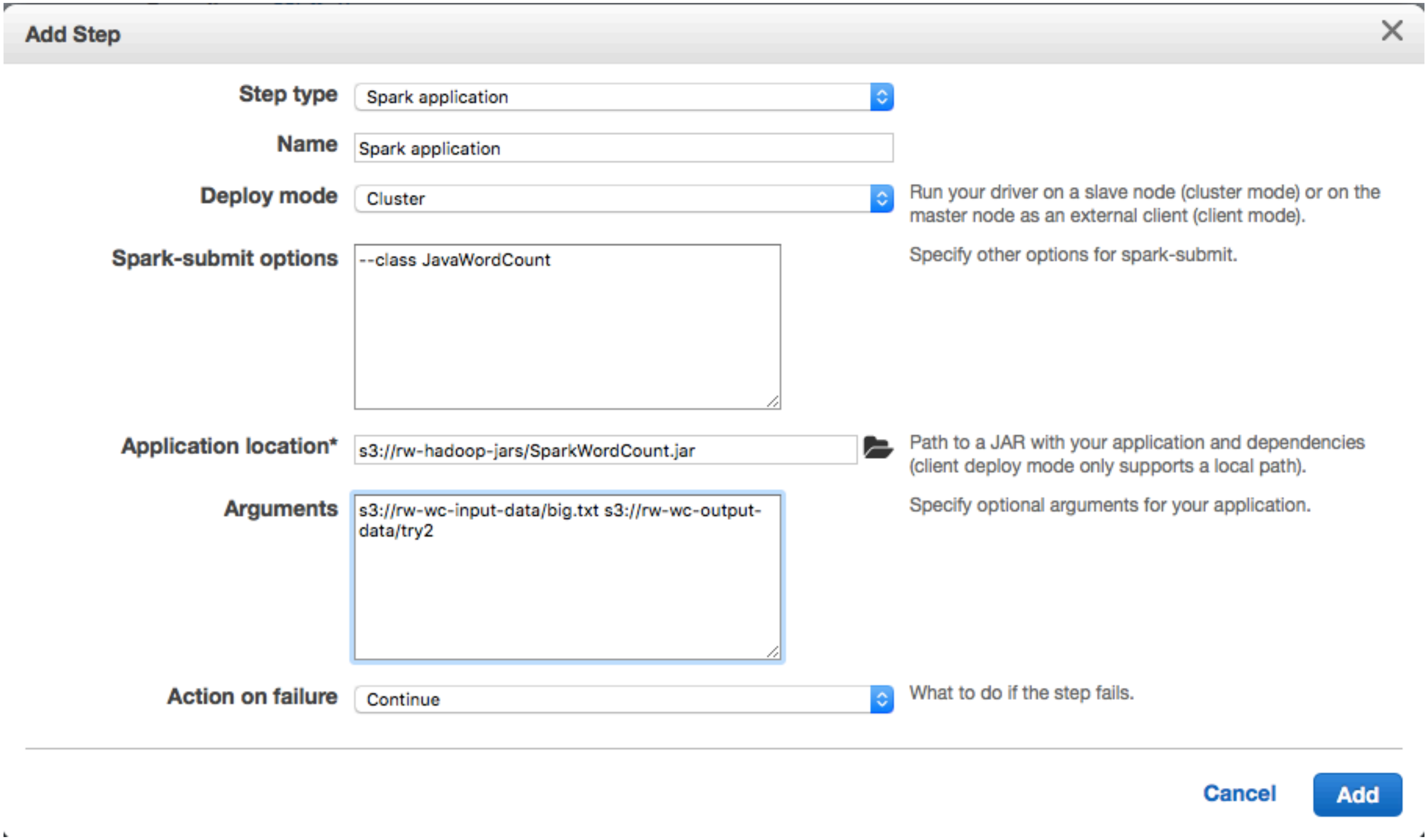
Add steps (optional)

Step type Spark application   Configure

☐ Auto-terminate cluster after the last step is completed

Spark Application Setup

You have to give --class ClassName in Spark-submit options

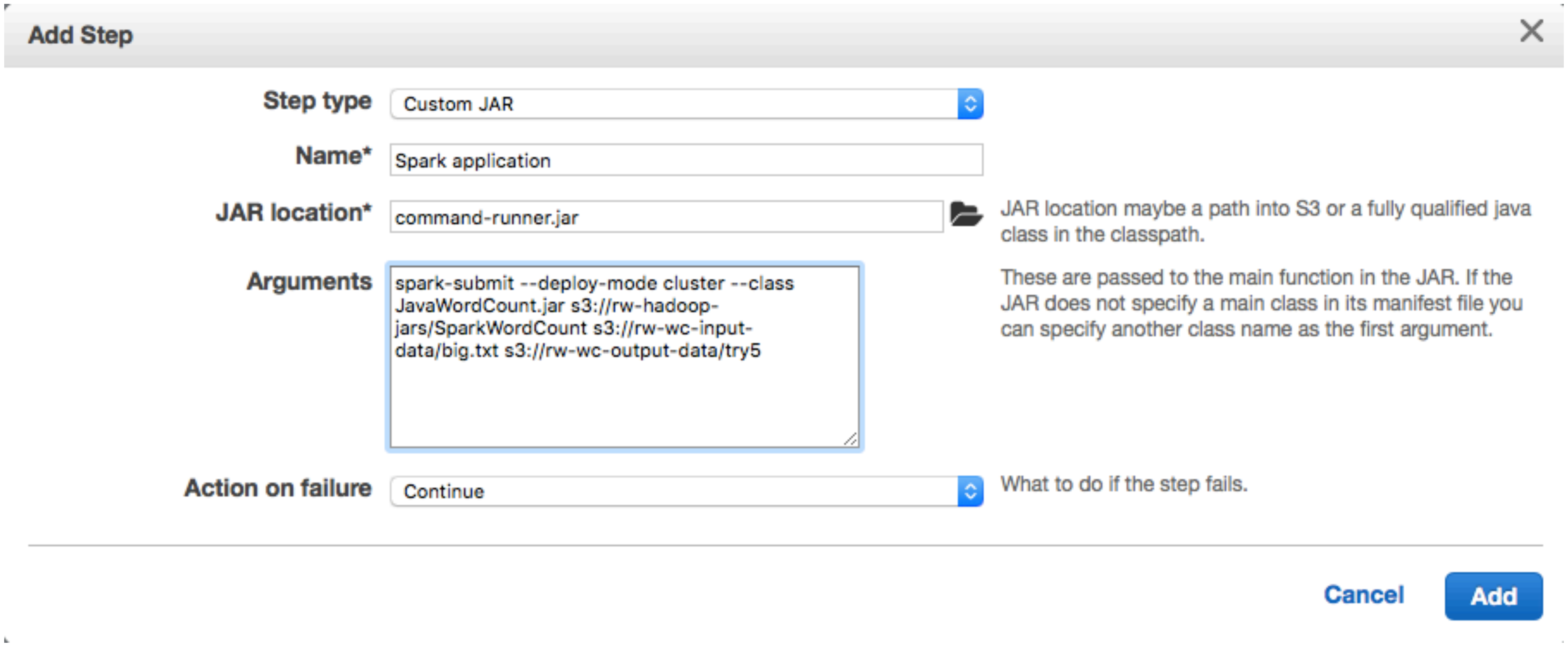


Add Step

Step type	Spark application	
Name	Spark application	
Deploy mode	Cluster	Run your driver on a slave node (cluster mode) or on the master node as an external client (client mode).
Spark-submit options	--class JavaWordCount	Specify other options for spark-submit.
Application location*	s3://rw-hadoop-jars/SparkWordCount.jar	Path to a JAR with your application and dependencies (client deploy mode only supports a local path).
Arguments	s3://rw-wc-input-data/big.txt s3://rw-wc-output-data/try2	Specify optional arguments for your application.
Action on failure	Continue	What to do if the step fails.

Using the custom jar option

Useful when cloning steps



Add Step

Step type: Custom JAR

Name*: Spark application

JAR location*: command-runner.jar

JAR location maybe a path into S3 or a fully qualified java class in the classpath.

Arguments:

```
spark-submit --deploy-mode cluster --class
JavaWordCount.jar s3://rw-hadoop-
jars/SparkWordCount s3://rw-wc-input-
data/big.txt s3://rw-wc-output-data/try5
```

These are passed to the main function in the JAR. If the JAR does not specify a main class in its manifest file you can specify another class name as the first argument.

Action on failure: Continue

What to do if the step fails.