**Steps to execute Hadoop Benchmarks on Google Cloud and Amazon EC2**

1. Create Container Cluster on google cloud platform
2. The following fields are required:

**Name:** The name of this container cluster. It must be unique within the project and the zone.

**Zone**: Select the zone in which you want to create the container cluster

**Cluster size:** The number of instances to include in this container cluster. We had created as single instance for single node cluster so therefore the cluster size was one. We created one master node and two slave nodes for multi-node cluster

**Machine Type:** The Google Compute Engine [machine type](https://cloud.google.com/compute/docs/machine-types) to use for instances in this container cluster. We used machine type n1-standard-4 (Standard 4 CPU machine type with 4 virtual CPUs and 15 GB of memory) for master node in both single node cluster and multi-node cluster. We used machine type n1-standard-2 (Standard 2 CPU machine type with 2 virtual CPUs and 7.5 GB of memory) for slave node.

**Command for creating the cluster:**

**gcloud container clusters create NAME --zone ZONE \ --num-nodes=1 --machine-type=n1-standard-4 --local-ssd-count=1**

**Hadoop Benchmark Execution Commands**

**Single Node and Multi Node Cluster.**

1. After creating the cluster ssh into the cluster
2. As Docker is pre-installed in google cluster, We will install and run Hadoop in the docker
3. We pulled the Hadoop 2.7.1 docker image by the following command

**sudo docker sequenceiq/hadoop-docker:2.7.1pull**

1. To check whether the hadoop docket image got downloaded correctly.

**docker images**

1. Now we will run Hadoop 2.7.1 in docker

**docker run -it –p 50070:50070 sequenceiq/hadoop-docker:2.7.1 /etc/bootstrap.sh -bash**

1. Now we will go Hadoop directory by the command given below and execute the various benchmark

**TestDFSIO Benchmark**

**Write**

time bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-client-jobclient-2.7.1.jar TestDFSIO -write -nrFiles 8 -fileSize 1000

**Read**

time bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-client-jobclient-2.7.1.jar TestDFSIO -read -nrFiles 8 -fileSize 1000

**Clean**

bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-client-jobclient-2.7.1.jar TestDFSIO -clean

**NameNode(NN) Benchmark**

**time bin/hadoop jar** share/hadoop/mapreduce/hadoop-mapreduce-client-jobclient-2.7.1.jar **nnbench -operation open\_read \  
   -maps 12 -reduces 6 -blockSize 1 -bytesToWrite 1000 -numberOfFiles 1000 \-replicationFactorPerFile 16 -readFileAfterOpen true \  
   -baseDir /benchmarks/NNBench-`hostname -s`**

**MapReduce Benchmark**

time bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-client-jobclient-2.7.1.jar mrbench -numRuns 25

**TeraSort Benchmark**

**TeraGen**

time bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar teragen 200000000 /**benchmark**/terasort-input

**Tera Sort**

time bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar terasort /benchmark/terasort-input /benchmark/terasort-output

**Tera Validat**e

time bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar teravalidate /benchmark/terasort-output /benchmark/terasort-validate

**Word Count**

Step 1. Create java file for WordCount program

Step 2. bash-4.1# export JAVA\_HOME=/usr/java/default

bash-4.1#export PATH=${JAVA\_HOME}/bin:${PATH}

export HADOOP\_CLASSPATH=${JAVA\_HOME}/lib/tools.jar

Step 3. Compile java file and create a jar file

bash-4.1# bin/hadoop com.sun.tools.javac.Main WordCount.java

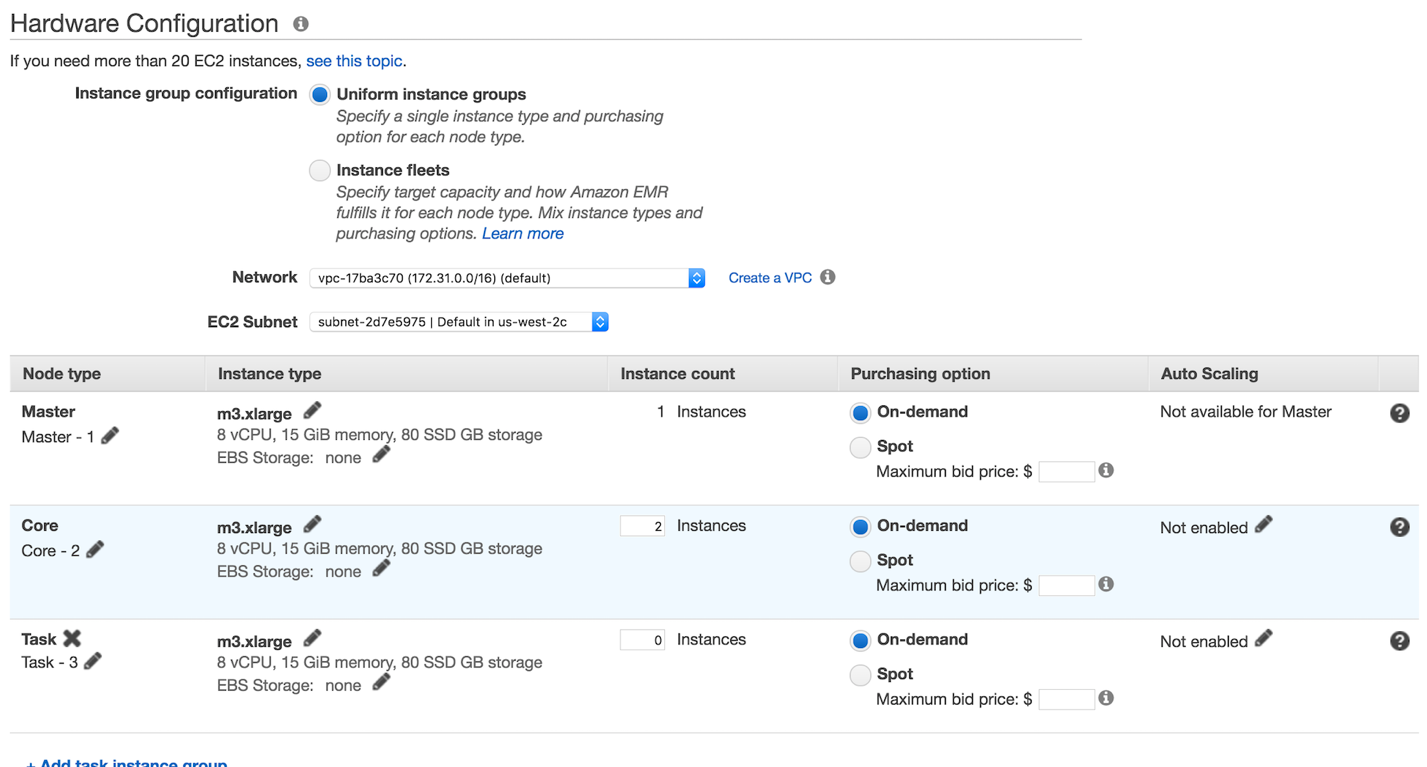
bash-4.1# jar cf wc.jar WordCount\*.class

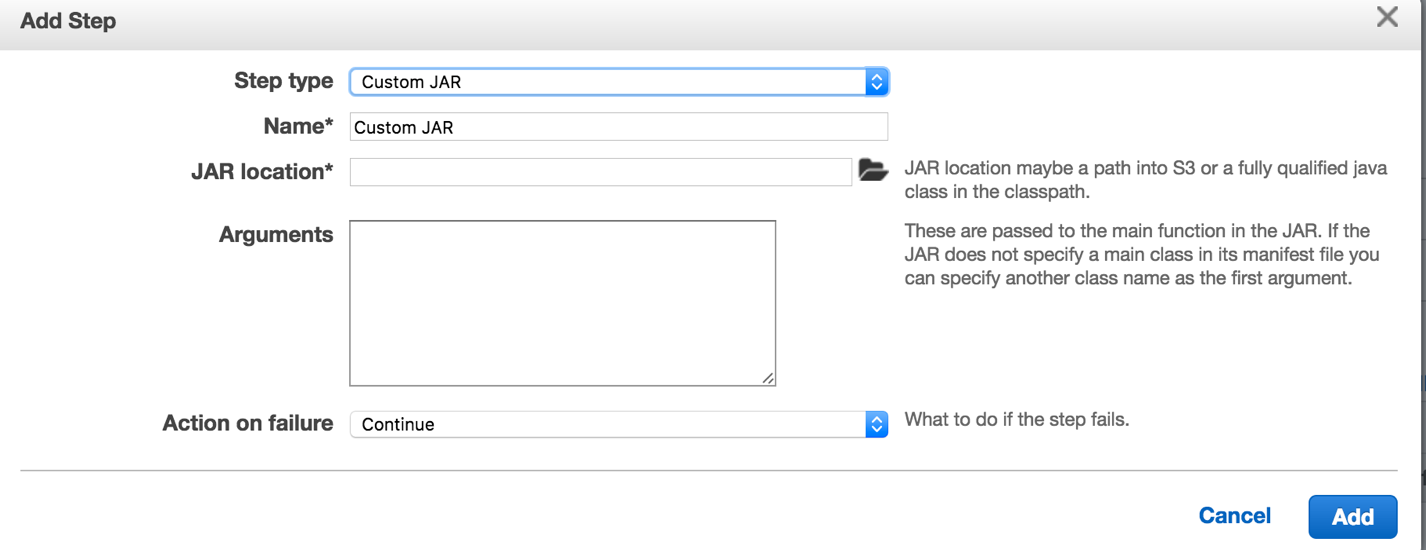
Step 4. Take the input files using wget command and move them to input files

Step 5. Run the WordCount.java program

**bash-4.1# time bin/hadoop jar wc.jar WordCount cloud/inputs cloud/output**

In Amazon EMR after you create a cluster of your configuration you can add steps to run various jobs.

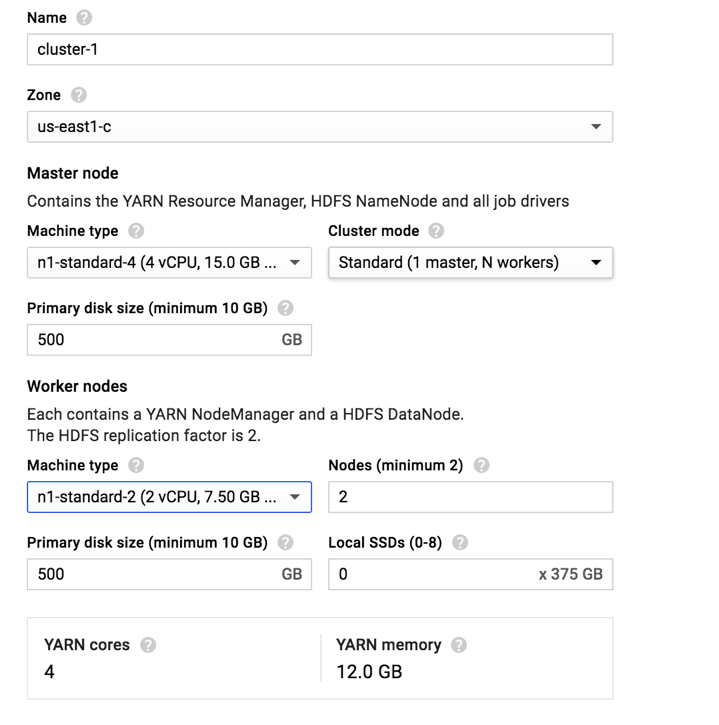


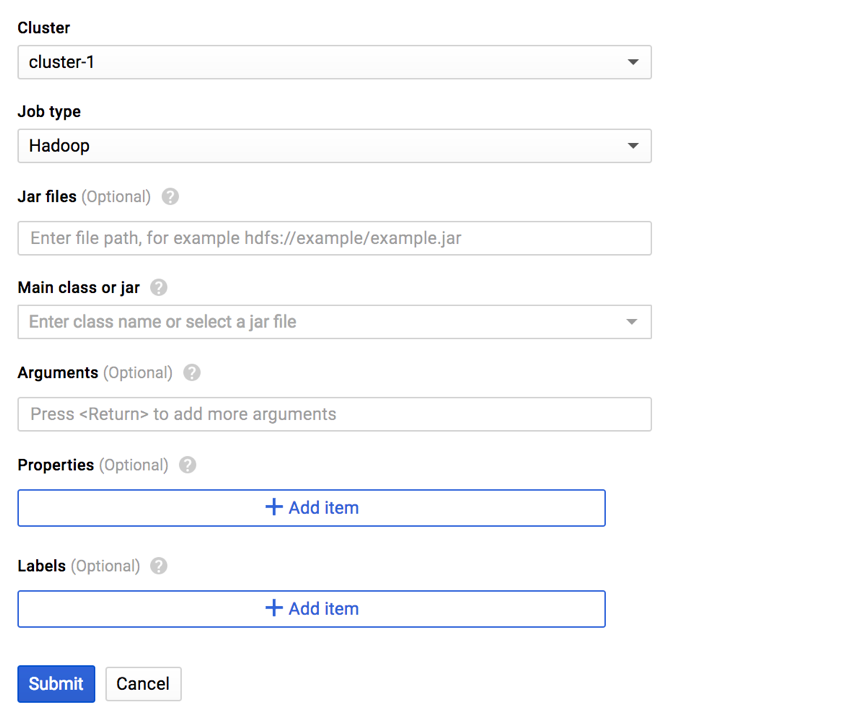


you store all your data in S3 and you can mention the path of your bucket to access.

* select Custom Jar
* Give name to the step
* Give your jar path, which is located in S3 bucket
* you can give appropriate arguments next

In Google DataProc after you create a cluster of your configuration there you can add various jobs.





As we saw in AWS EMR similarly here you store your data in google bucket and give the appropriate path in jar file and give the Main class and arguments required to run.

**GitHub Link:**

[**https://github.com/enamshah09/DeltaComparisonOfContainers.git**](https://github.com/enamshah09/DeltaComparisonOfContainers.git)

.