**Preparations**

To set up our Cloudera cluster, firstly we need two machine to make a cluster. We’ll be using virtual machines provided by Amazon service [AWS](https://aws.amazon.com/). Specifically EC2 instances. For our practical lesson, we will use next machines:

Large instance :

25 GB hard drive

>= 8 GB RAM

CentOS, RHEL, SLES

Small instance :

25 GB hard drive

>= 2 GB RAM

CentOS, RHEL, SLES

We will need more powerful machine(large) as a master, that will host Cloudera Manager - cluster management tool provided by [Cloudera](http://www.cloudera.com/).

After our machines are created, we should open needed ports that Cloudera needs and provide proper access from local machine to our EC2 instances and access between them both without password.

**Opening ports**

From AWS console we should open few ports that are required for Cloudera Manager such as

22 - Default SSH port

7180 - Cloudera Manager UI

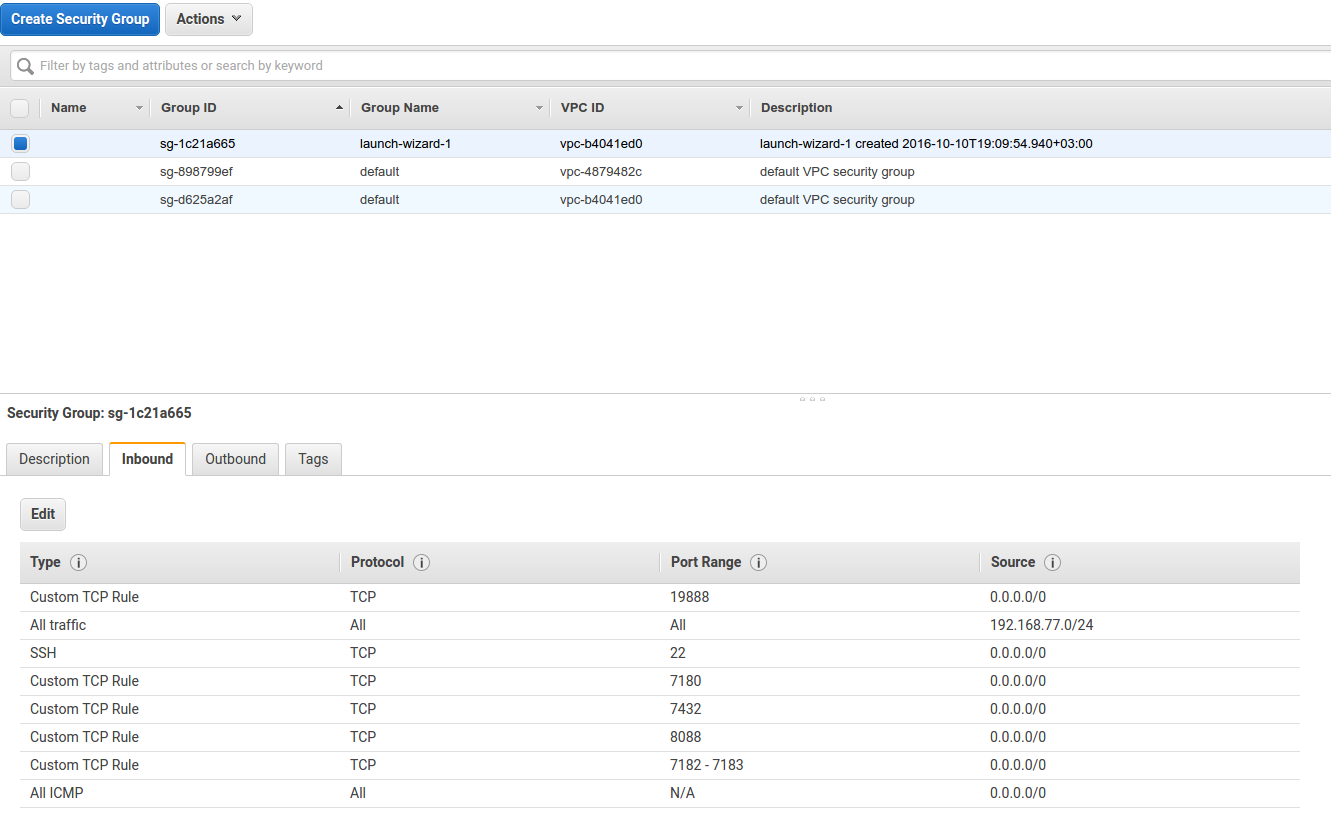
7182 - Agent heartbeat

7432 - Embedded SQL database for internal CLoudera Manager use

19888 - Yarn History Server UI

8088 - Yarn Resource Manager UI

icmp -1 - ping echo



**Giving Access**

AWS provides standard way to connect to instances using only key without password.

ssh -i lits.pem ec2-user@external-ip

To allow access without password between your instances, you should copy your key(in this case lits.pem) to each of your nodes, then login as a root and move into ssh folder.

Sudo -s

Cd ~/.ssh/

// copy the lits.pem key into this folder and rename it to id\_rsa

Cp /home/ec2-user/lits.pem ~/.ssh

Mv lits.pem id\_rsa

//give the key right access rights and proper user

Chmod 600 id\_rsa

Chown root:root id\_rsa

After that try to connect to other node with root access and check whether you’ll be asked to enter password or not.

//from slave to master

Ssh root@192.168.0.1

//from master to slave

Ssh root@192.168.0.2

**Disable SELinux**

Next small thing that should be done on large node, we should disable SELinux on both instances. To do that, we should edit SELinux config file and change SELINUX property to disabled

Sudo vi /etc/sysconfig/selinux

SELINUX=disabled

Save changes and now look if your changes were applied

Less /etc/sysconfig/selinux

**Edit hostnames**

AWS machines are created with already defined hostnames. We should change that for easiness and Cloudera conventions, so that we won’t have any troubles with host discovery.

On Large instance we should provide name with pattern like hostname.domain (e.g. master.cloudera). To do that we should change entry in /etc/hostname file.

Sudo vi /etc/hostname

Same should be done on small instance, only with different hostname(e.g. slave.cloudera) as hostnames should be unique per cluster.

Now we need to add both addresses to /etc/hosts to identify self and each others. But before we need to know actual IP address of a machine. To do that you should execute ifconfig.

Sudo ifconfig

Now editing:

Sudo vi /etc/hosts

// on master.cloudera

192.168.0.1 master.cloudera

192.168.0.2 slave.cloudera

//same on slave

192.168.0.1 master.cloudera

192.168.0.2 slave.cloudera

Additionally add both addresses to your local machine.

To check whether changes were applied try next steps:  
 //from local machine ping master

Ping 192.168.0.1

Ping master.cloudera

//from local machine ping slave

Ping 192.168.0.2

Ping slave.cloudera

**Install Cloudera Manager on large instance**.

Now we can start actual installation of Cloudera Manager. To do this we should download latest installer from official [download page](http://www.cloudera.com/downloads/manager/5-8-2.html) on site using wget.

Sudo yum install wget

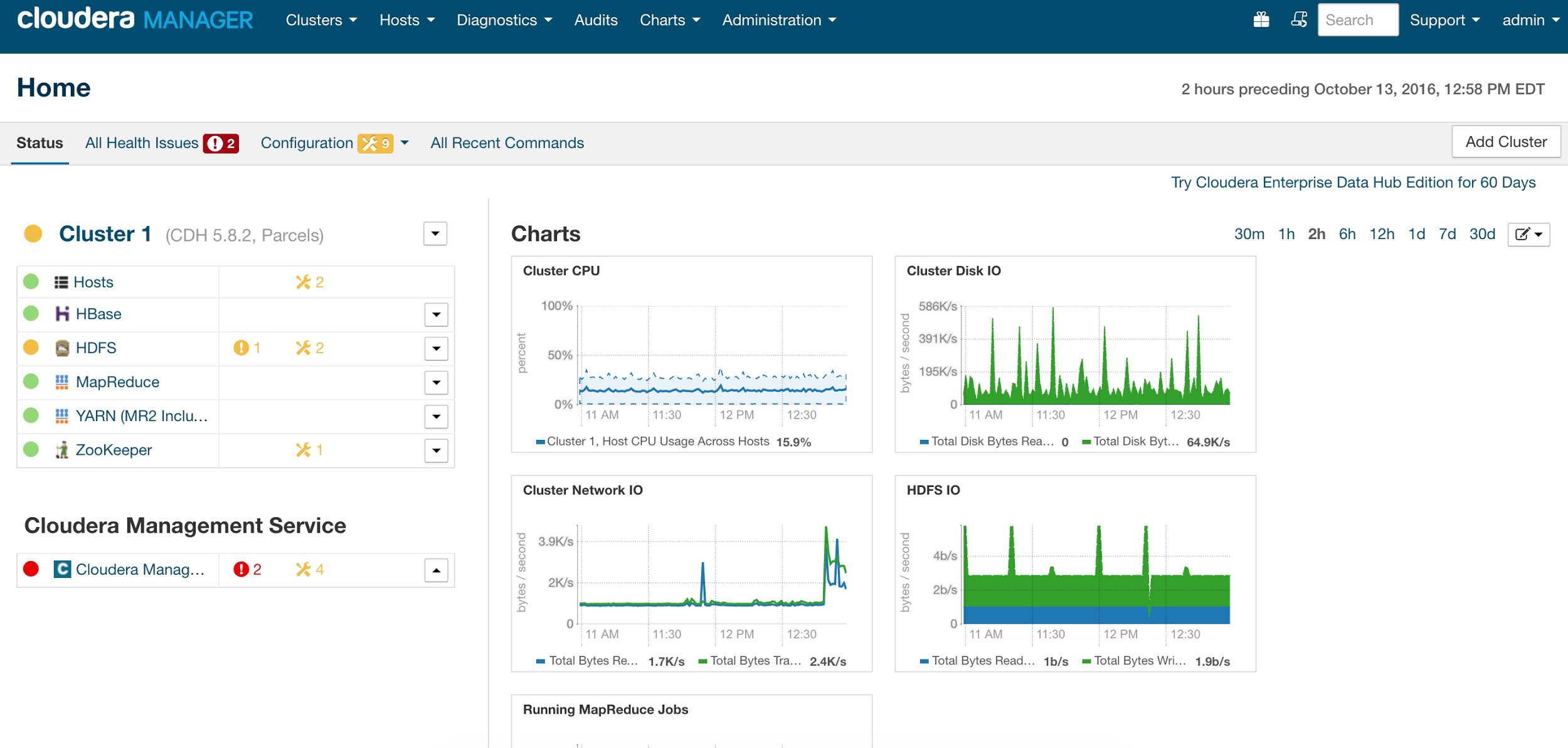
wget <http://archive.cloudera.com/cm5/installer/latest/cloudera-manager-installer.bin>

Chmod +x cloudera-manager-installer.bin

Sudo ./cloudera-manager-installer.bin

After execution just follow the installation steps, when selecting host just point out the cmaster.cloudera and check whether Cloudera Manager installer could fetch domain. Besides that, it is advised not to select all services. So when there will be a choice what services to install, choose **Custom** and in the next window choose only HDFS, YARN, MapReduce and HBASE. That’s the only services that are required for our practical lesson.

As result you should see Cloudera Manager dashboard with services status and charts.



**Cross the fingers and Cloudera Manager should be installed when installer will finish the job.**

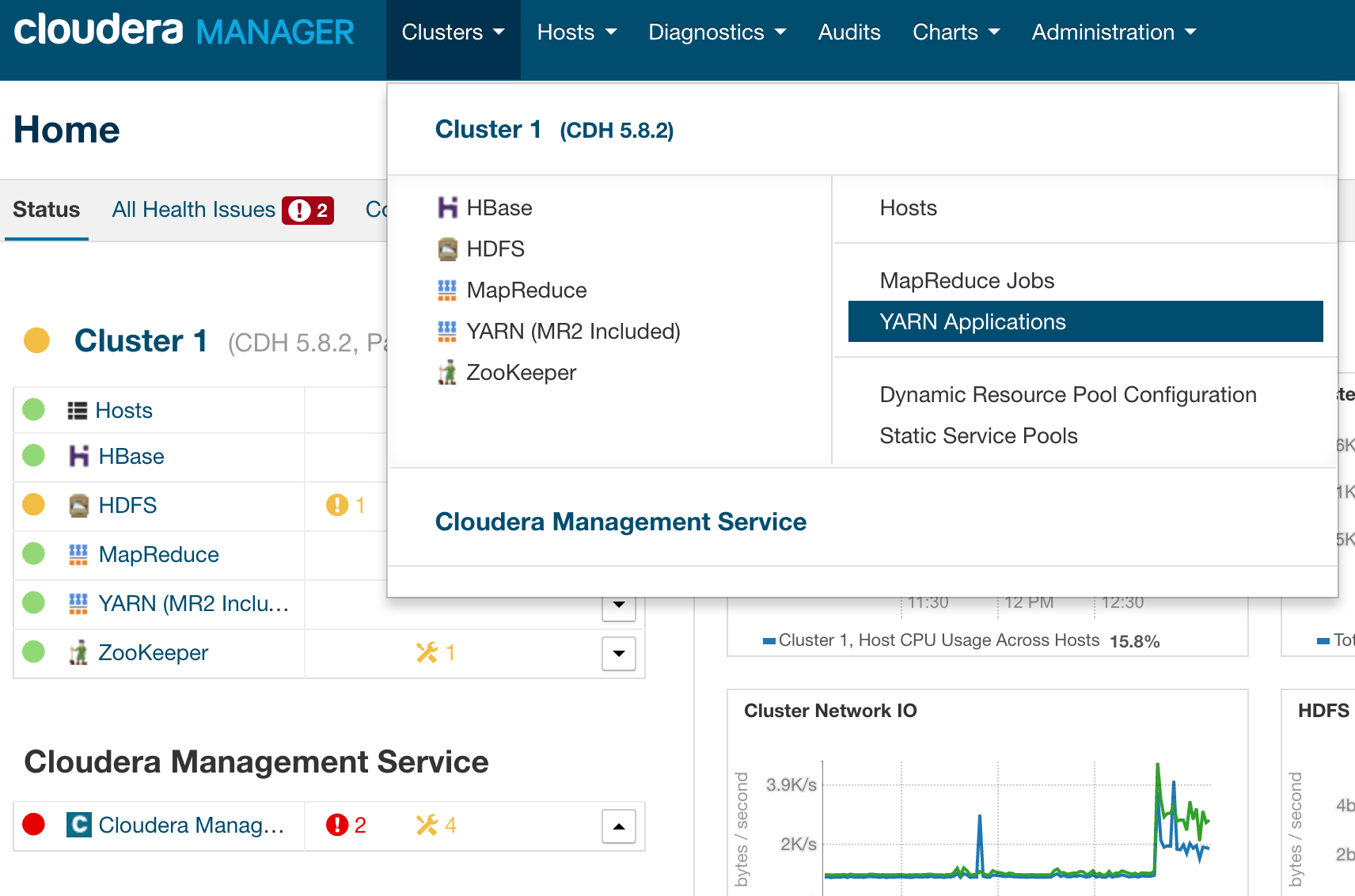
**Running test application**

Up to you.

**Check job components**

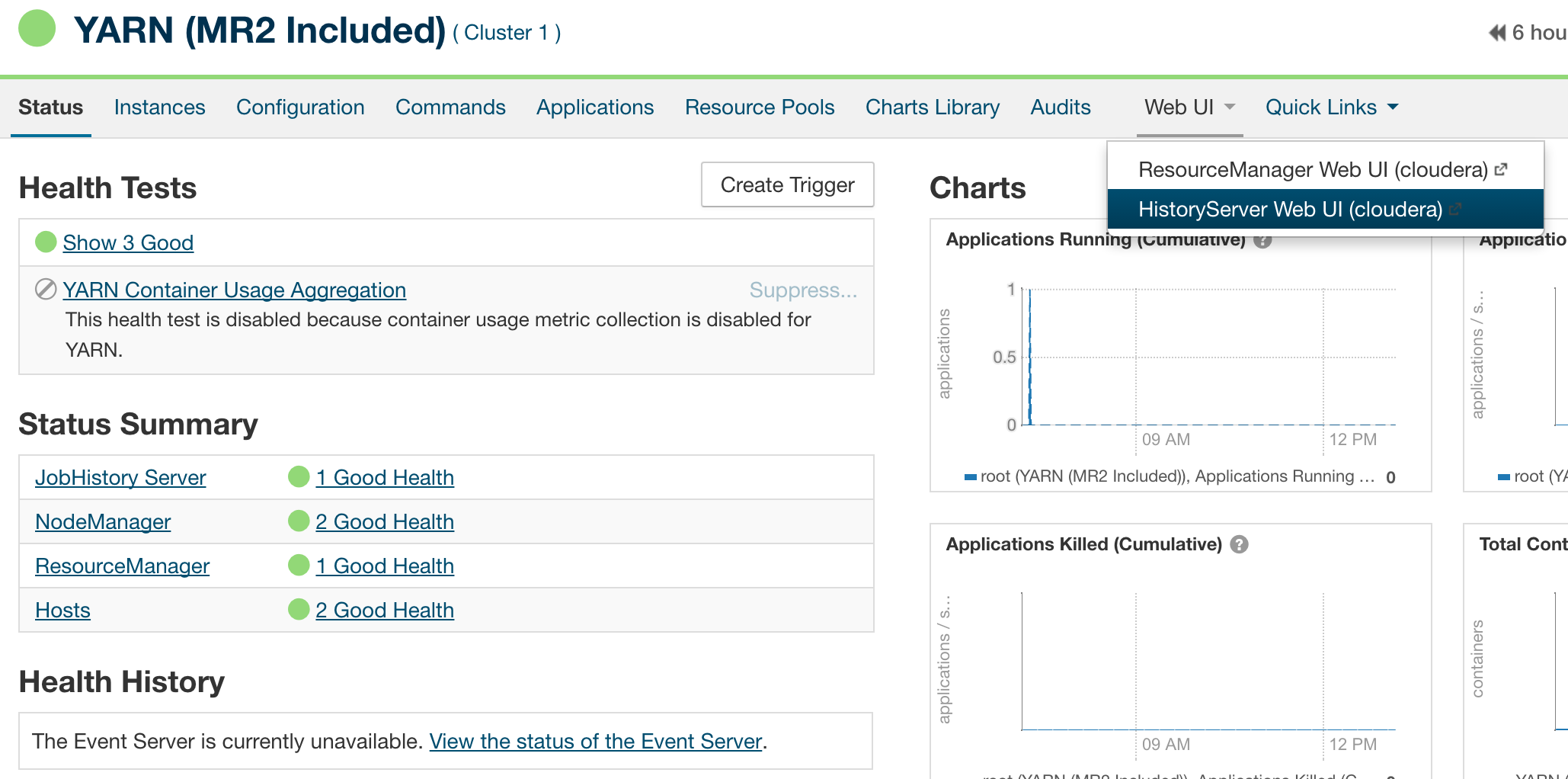
The importance in this step is to see that our application was ran using single cluster node. We should look at execution time and location of MapReduce tasks.

To see the overview and status of the execution of your job. Go to tab at the top named **Clusters** and choose **Yarn Applications**.

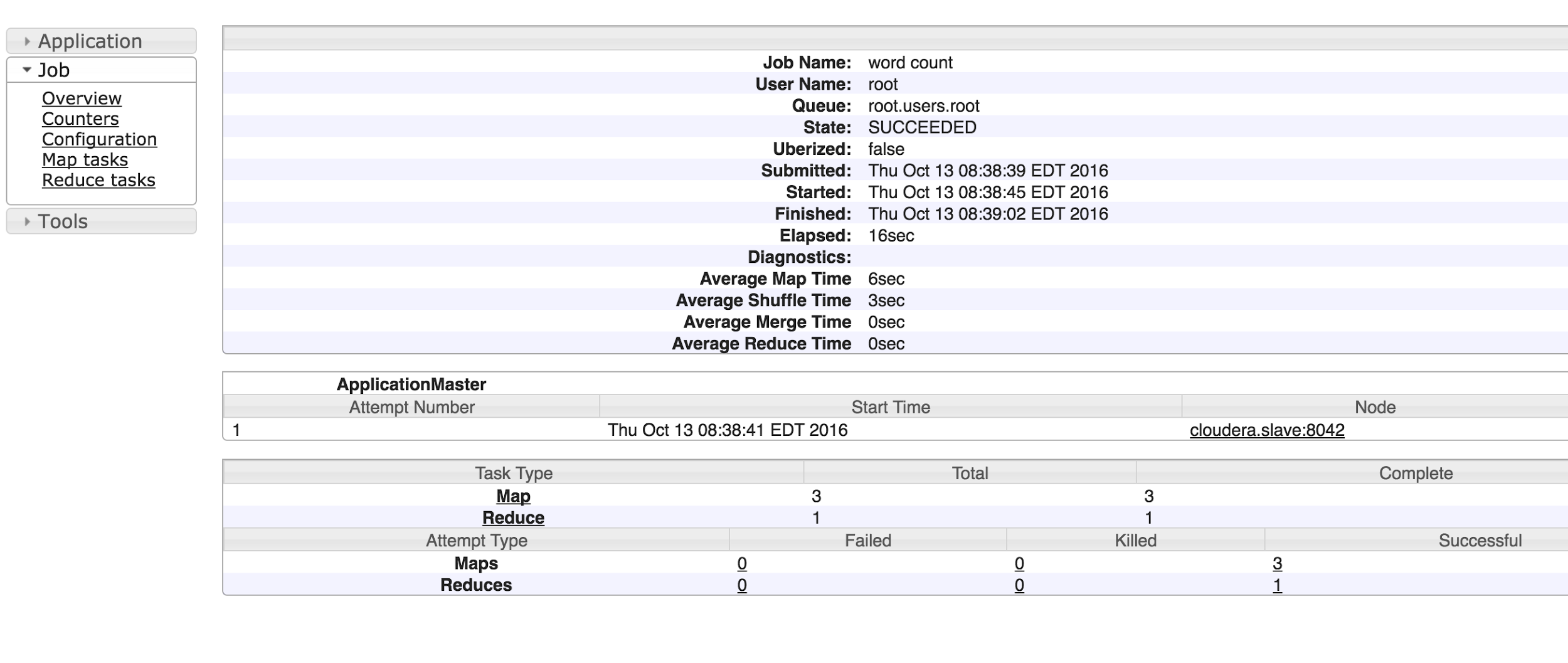


You will see latest MapReduce jobs runs and the state of each. Look at the **Duration** field to know how many time it took for job to run.

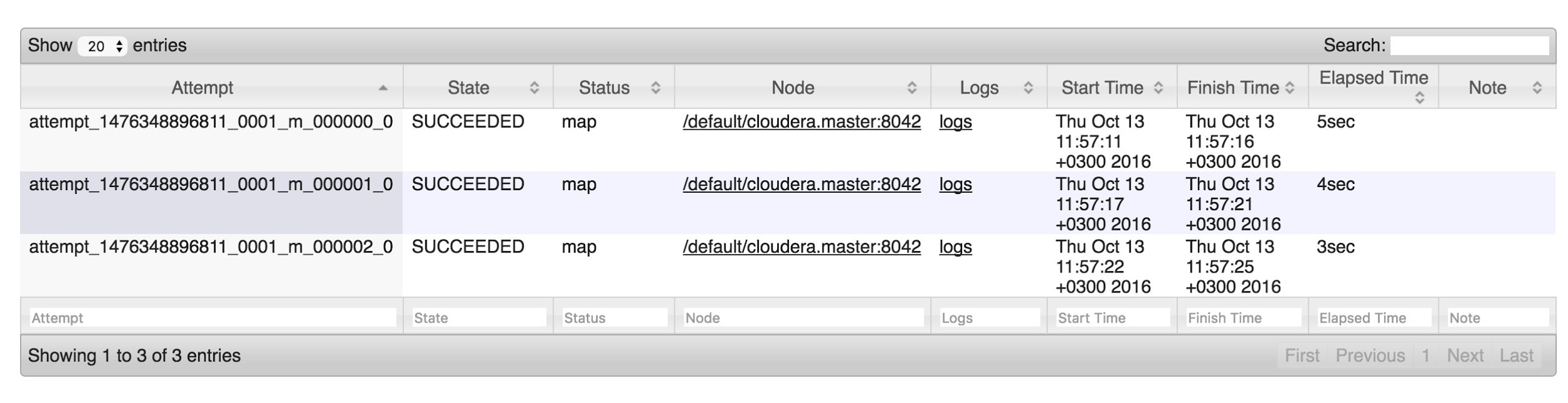
Now let’s look at MapReduce tasks and check that all of them ran on the same machine. To do that, go to YARN service, and from dropdown **Web UI** choose **HistoryServer Web UI** option.



There we search for our job that has an id of a pattern job\_12345678\_0001 and click on it. From there we click on the **Successful** mapper count.



It will show on what machines MapReduce tasks were executed.

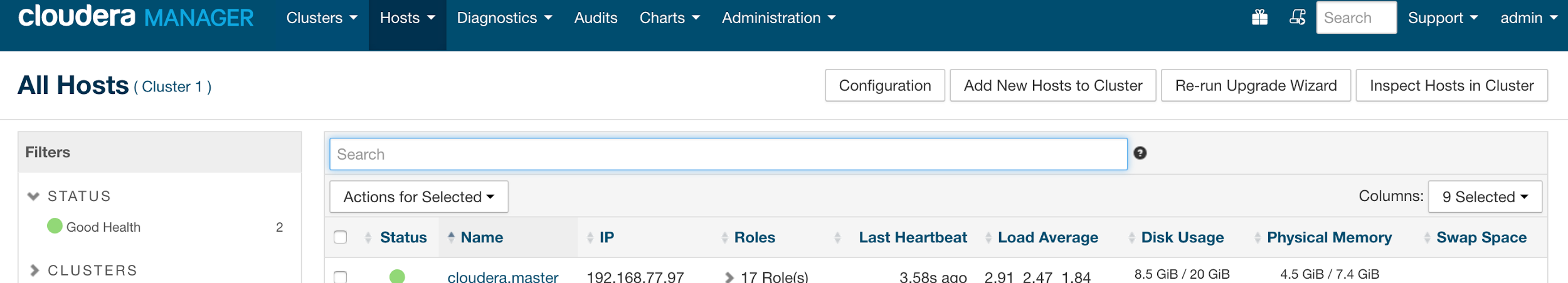


As you can see, all the MapReduce tasks were executed on master instance.

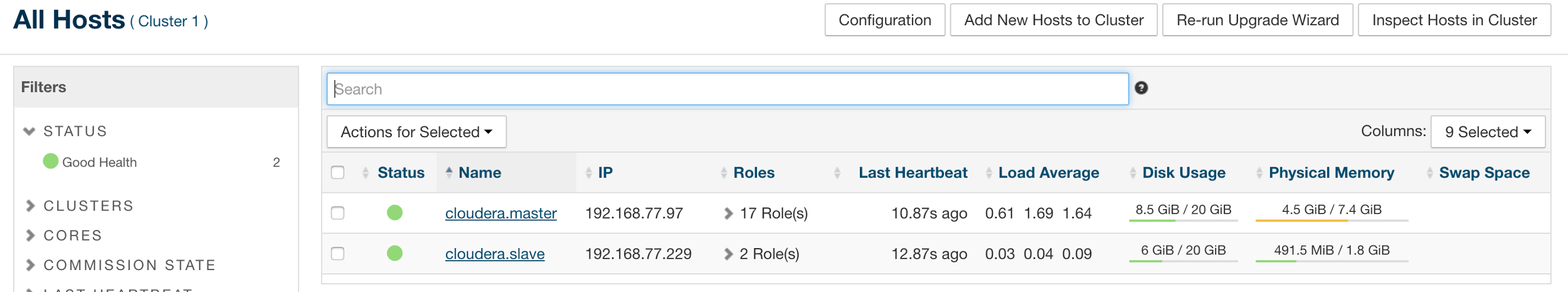
**Now let us make a real cluster!**

**Adding new host to cluster.**

From dashboard, go to **Hosts** service. On the top right corner hit **Add New Hosts to Cluster**. After that you will see similar to Cloudera Manager installation guide that will install all Cloudera Agent libraries into selected host so that host could be visible from CLoudera Manager UI.

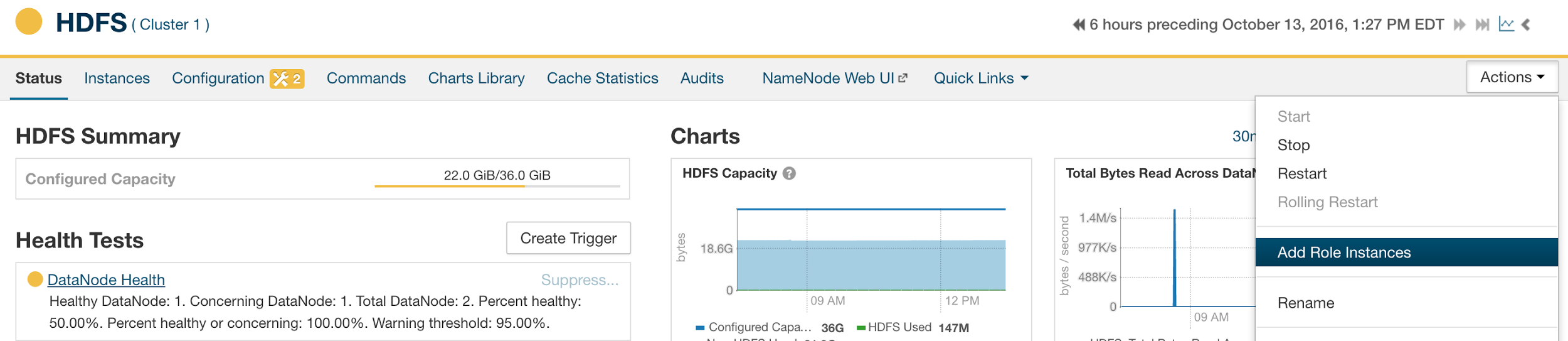


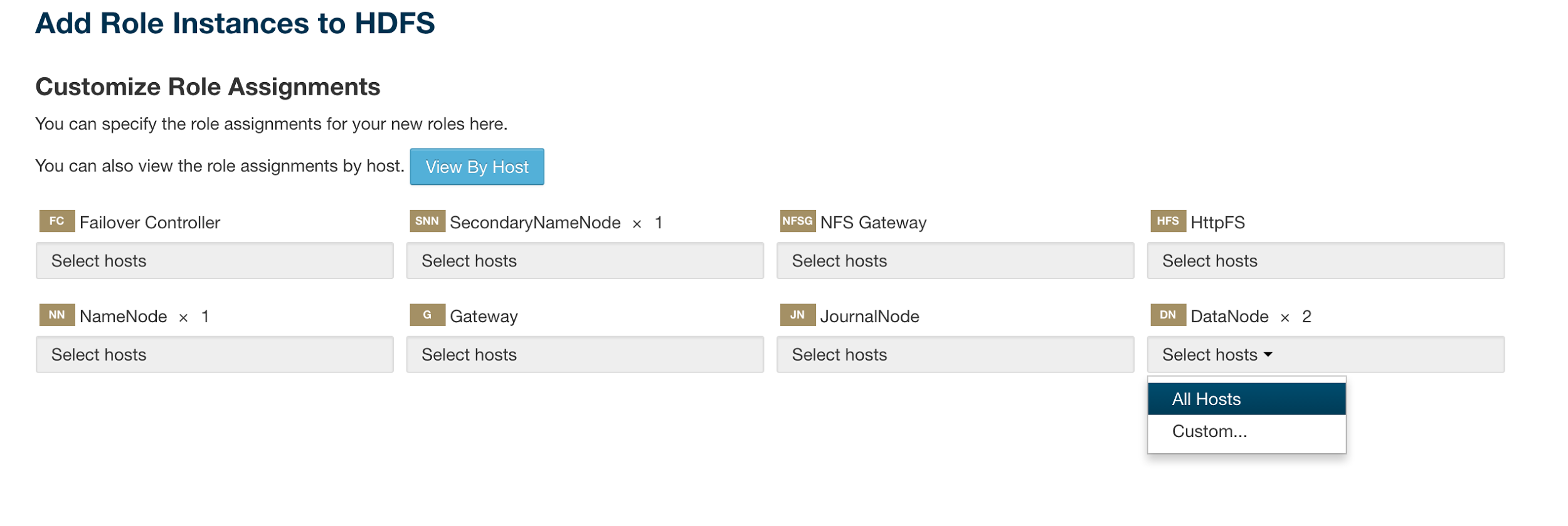
As a result, Cloudera Manager will install all needed libs onto the second machine and in **Hosts** service you should see two hosts available.



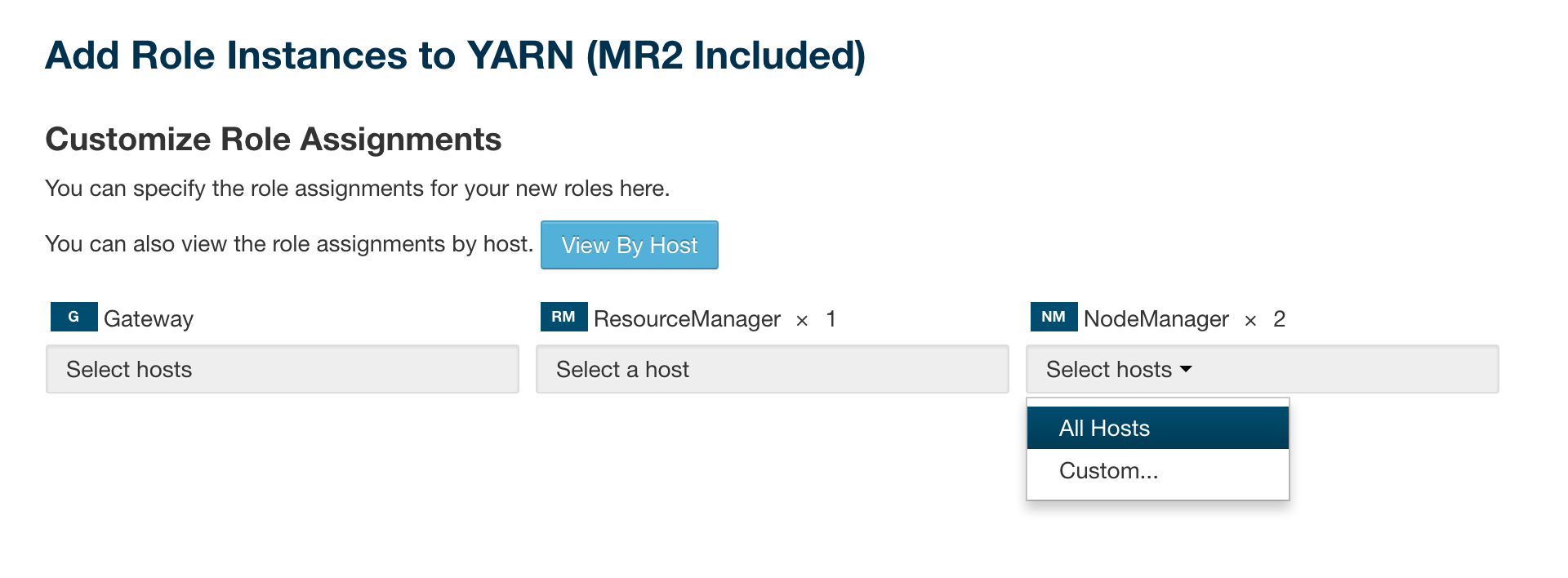
**Adding roles for new node**

To make our job actually distributed, we need to add certain roles to our new host. TO do this Go to HDFS Service tab on the right side of the screen there will be **Actions** button. And hit **Add Role Instances**.Choose Data Node and select in drop down menu **All Hosts**.





We added node that will replicate data from master node. But to actually execute your MapReduce job on slave instance, you need one more thing. Add NodeManager role to slave instance. To do that, go to **YARN** service and hit same **Actions** menu button. Hit same role addition button and chose All hosts on **NodeManager** menu dropdown.



After applying these changes, your job should be running on both machines. We need to check that out.

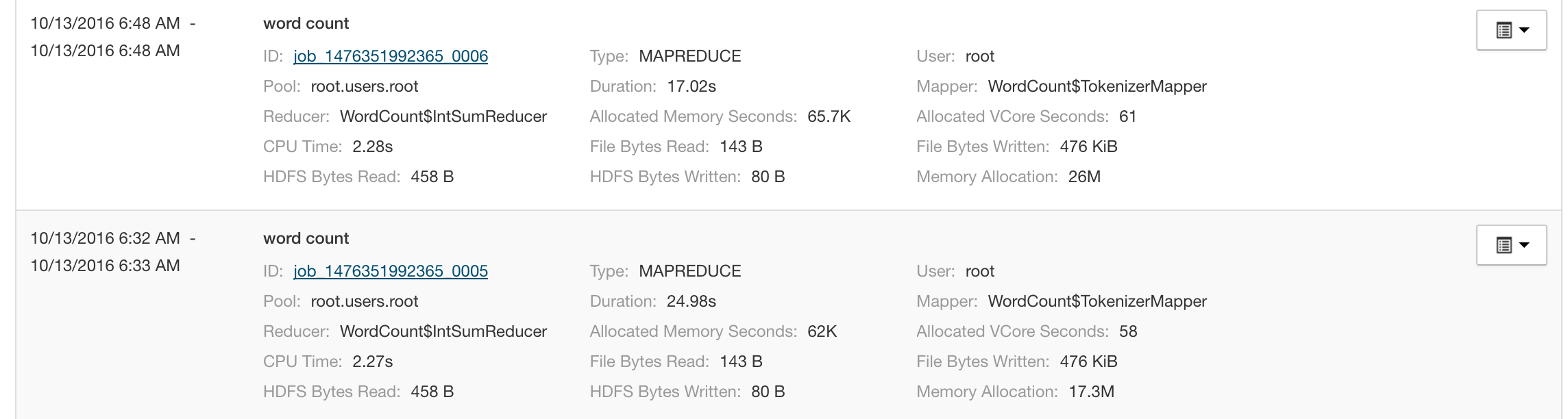
**Running test application**

Again.

The second run should be faster as now we’ve doubled the amount of machine were our computation take place. Lets check.

**Checking again**

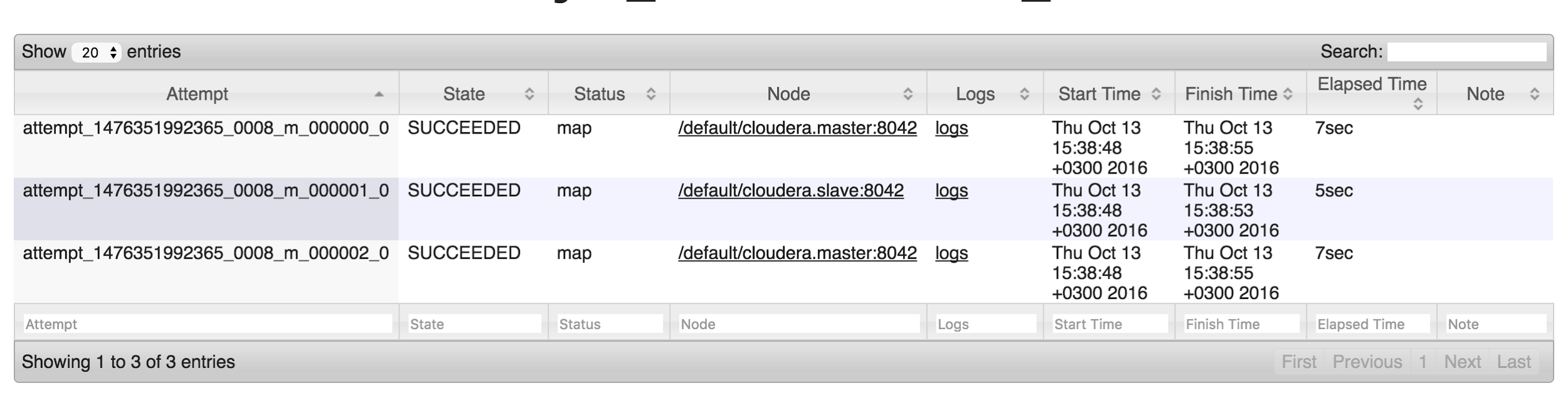
Now we can go again to **Clusters** tab at the top of the window and hit **YARN Applications**. Now we should see new run of our job and the duration of the job should be lower.



As you can see in the example , it took single-node cluster almost 25 seconds to cope with simple job. And it took only 17 seconds to run same job with two-node cluster. That’s some improvement!

To check whether our job was truly distributed, let’s go to **YARN** service and again look at **HistoryServer Web UI**. Choose latest job and again click on **Successful** mappers count as we did before.

We should see that some tasks were executed on master node and some of task were executed on slave node which makes it distributed.



**Preparing test application**

<https://drive.google.com/open?id=0B0KJHPn8wXJ9SWNQR2Y2ZTJWa1E>

After we downloaded test jar, We should upload it to master instance into home folder for example.

Scp -i lits.pem your-jar-with-dependencies.jar ec2-user@192.168.0.1:~/

Now we need to prepare test data that we will store in files for our application. We will simply create few directories in HDFS and create few files with content that our MapReduce job will process.

//log in as hdfs user

$ sudo su hdfs

//make home/root directory in HDFS   
$ hadoop fs -mkdir /home /home/root

//give root user full access to created directory  
$ hadoop fs -chown root /home/root

//log out of hdfs user  
$ exit

//log in as a root user  
$ sudo -s

//make directory for our job’s input files  
$ hadoop fs -mkdir /home/root/wordcount /home/root/wordcount/input

//now we create few files for our job and move these files into HDFS

$ echo "Hadoop is an elephant" > file0  
$ echo "Hadoop is as yellow as can be" > file1  
$ echo "Oh what a yellow fellow is Hadoop" > file2  
$ hadoop fs -put file\* /home/root/wordcount/input

//now let’s run the test application on our created cluster

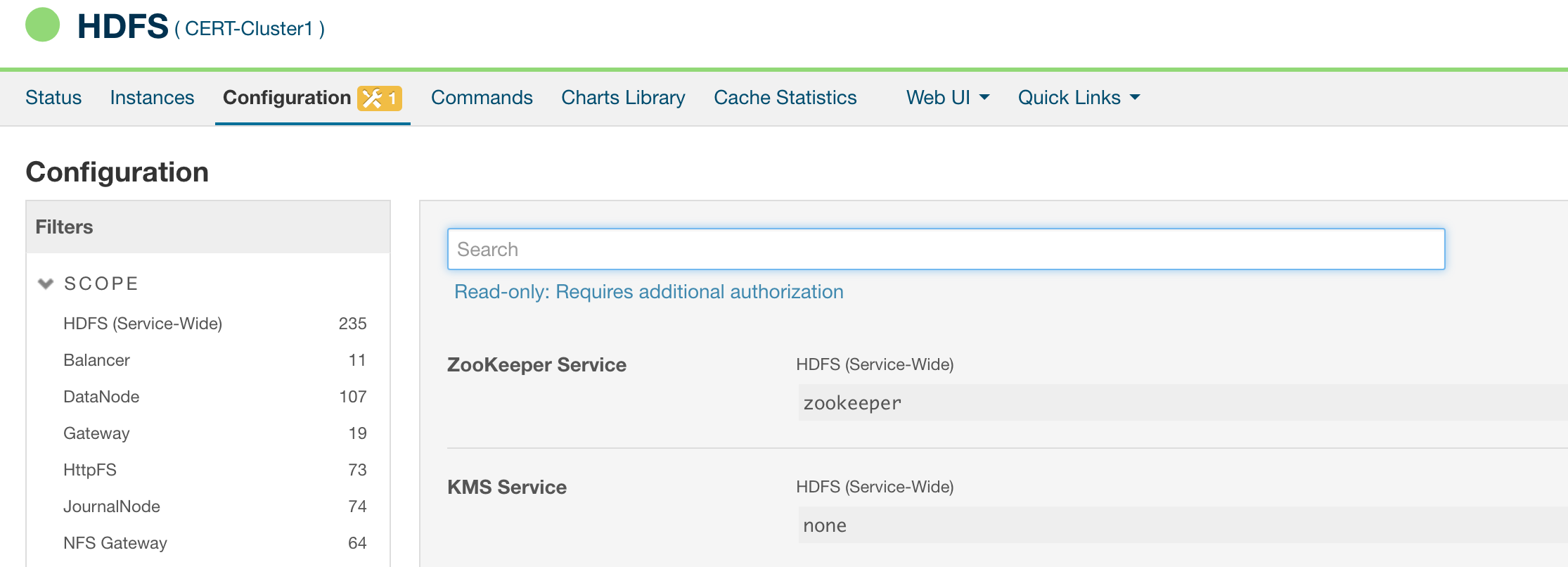
hadoop jar your-jar-with-dependencies.jar WordCount /home/root/wordcount/input /home/root/wordcount/output

To see the results, you can look into the output folder to check whether words were properly counted

hadoop fs -cat /home/root/wordcount/output/\*

**There’s a chance that job won’t be kicked off from the start, so please check possible tweaks for your configuration**

Mostly all problems are related to wrong configuration values that are set by default. So to change configuration, go to service which configuration should be changed and hit **Configuration** menu item



Problems with permissions

HDFS permissions dfs.permissions - false

Memory allocation

yarn.nodemanager.resource.memory-mb 2500

Yarn.scheduler.minimum-allocation-mb 750

**Useful links**

<https://blog.cloudera.com/blog/2013/03/how-to-create-a-cdh-cluster-on-amazon-ec2-via-cloudera-manager/>

<http://www.cloudera.com/documentation/archive/manager/4-x/4-7-2/Cloudera-Manager-Managing-Clusters/Managing-Clusters-with-Cloudera-Manager.html>

<https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html#Example:_WordCount_v1.0>

<https://www.cloudera.com/documentation/other/tutorial/CDH5/topics/ht_usage.html>

<http://www.cloudera.com/documentation/archive/manager/4-x/4-8-6/Cloudera-Manager-Installation-Guide/cmig_uninstall_CM.html>