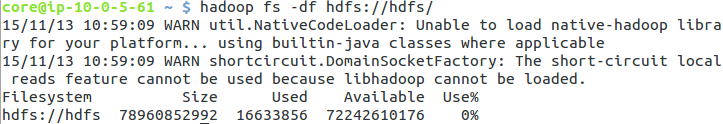
This test was performed once manually to assert that no data is lost on HDFS when scaling up/down number of slave nodes.

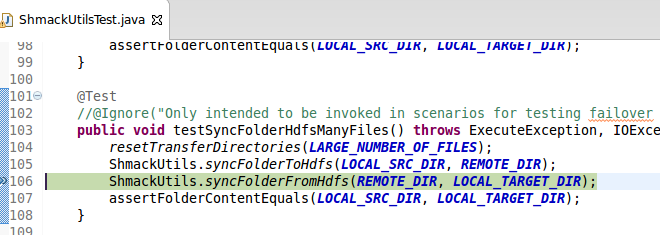
# Steps performed

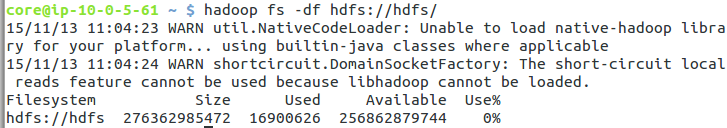
## Start with 10 Slaves



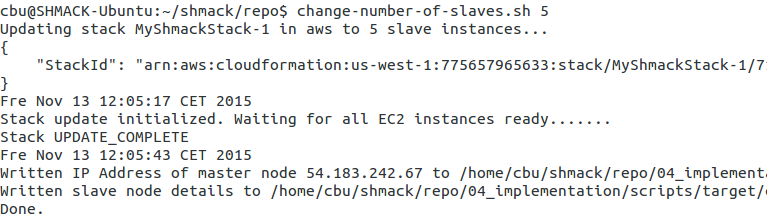
## Write 1000 files into hdfs

Run testcase until this line: ShmackUtils.syncFolderToHdfs

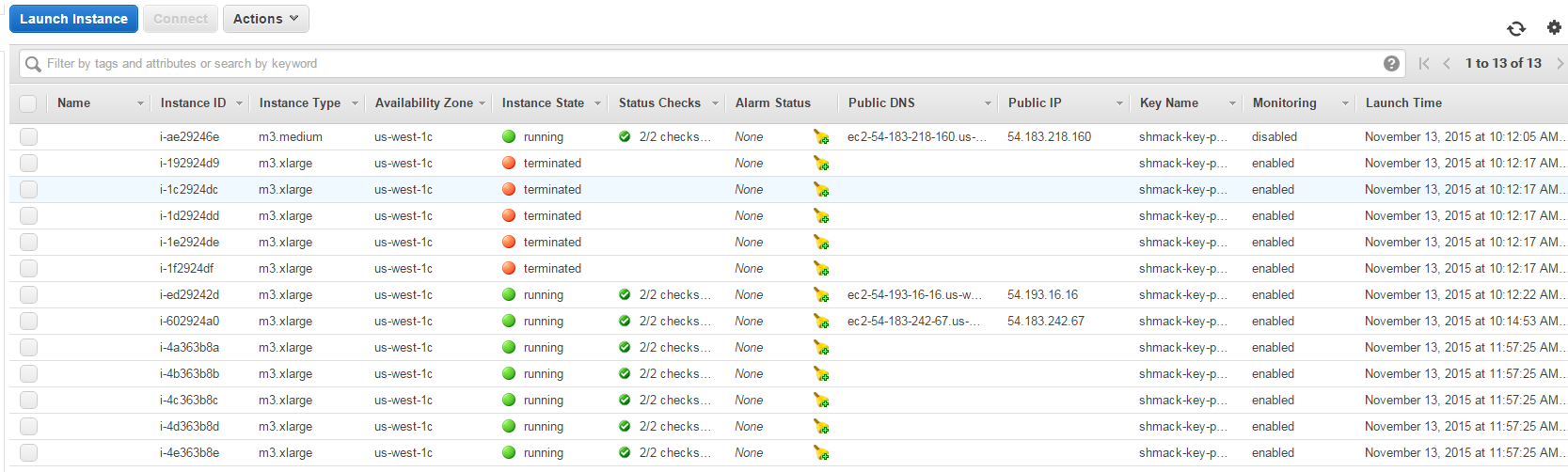




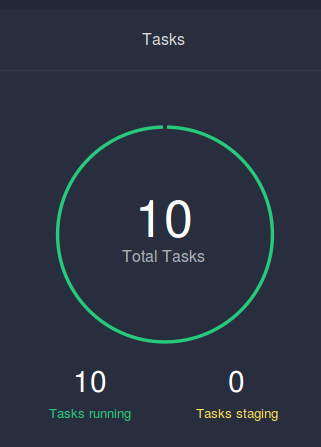
## Scale cluster down to 5 slave nodes



Interesting not the new nodes are removed but old ones… This could be a problem!  
(see “Launch time”)

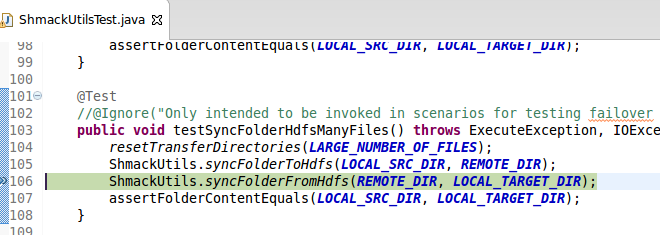


## Wait until all DCOS task are ready



## Try to read the 1000 files from hdfs

Finish Testcase (continue from Breakpoint)



org.apache.commons.exec.ExecuteException: Failed to execute [/bin/bash, run-on-dcos-master.sh, hadoop, fs, -copyToLocal, hdfs://hdfs/tmp/ssh-transfer-test/\*, /tmp/hdfs-xchange/from-hdfs/tmp/ssh-transfer-test/] - Output: com.zuehlke.shmack.sparkjobs.infrastructure.ExecuteResult@33cb5951[

standardOutput=

standardError=15/11/13 11:10:38 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

15/11/13 11:10:38 WARN hdfs.DFSUtil: Namenode for hdfs remains unresolved for ID nn1. Check your hdfs-site.xml file to ensure namenodes are configured properly.

15/11/13 11:10:38 WARN hdfs.DFSUtil: Namenode for hdfs remains unresolved for ID nn2. Check your hdfs-site.xml file to ensure namenodes are configured properly.

-copyToLocal: java.net.UnknownHostException: namenode1.hdfs.mesos

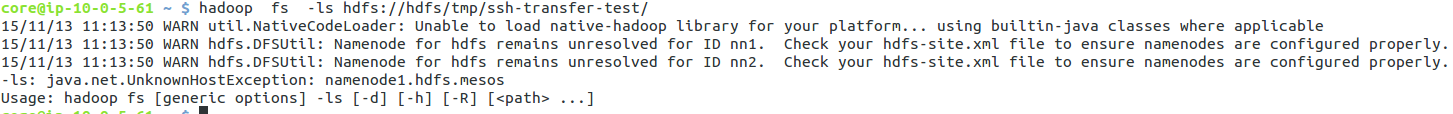
Usage: hadoop fs [generic options] -copyToLocal [-p] [-ignoreCrc] [-crc] <src> ... <localdst>

exitValue=255

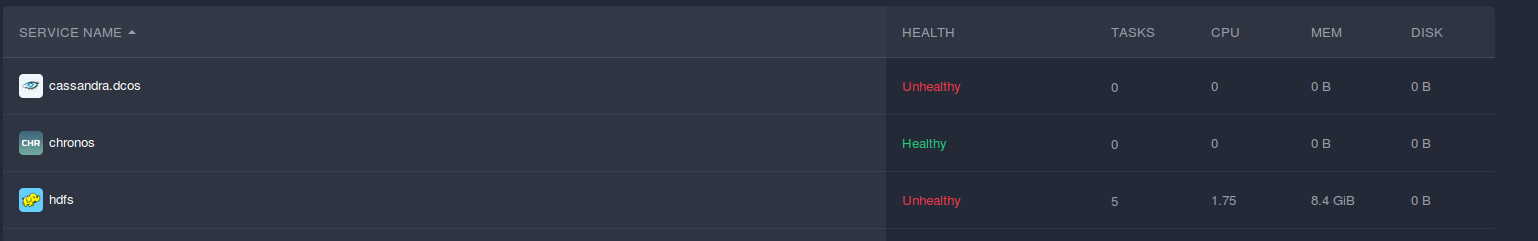
] (Exit value: 255)

## Try on console on dcos master 🡪 ls failing

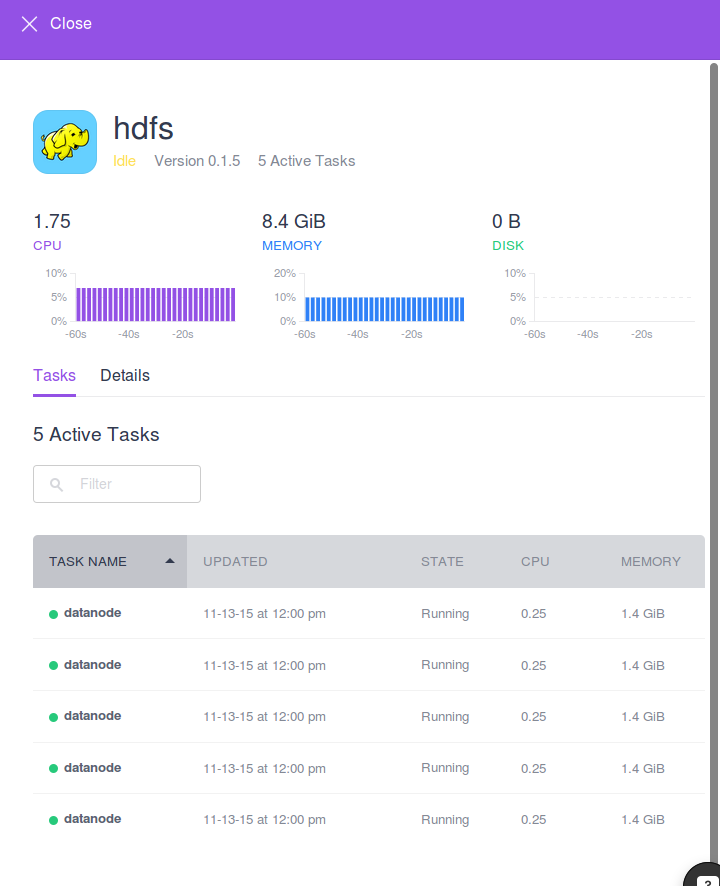
hadoop fs -ls hdfs://hdfs/tmp/ssh-transfer-test/



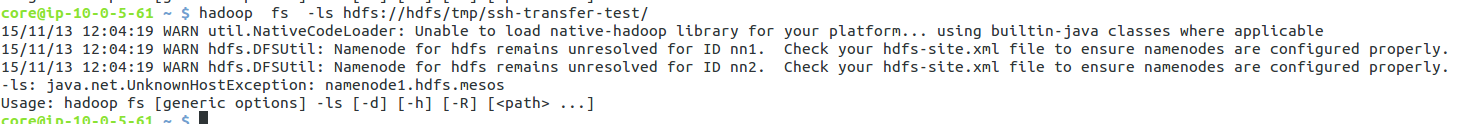
## DCOS Console is also unhappy



No namenode here:



## Wait approx. 1 hr 🡪 Still no namenode 🡪 All data lost



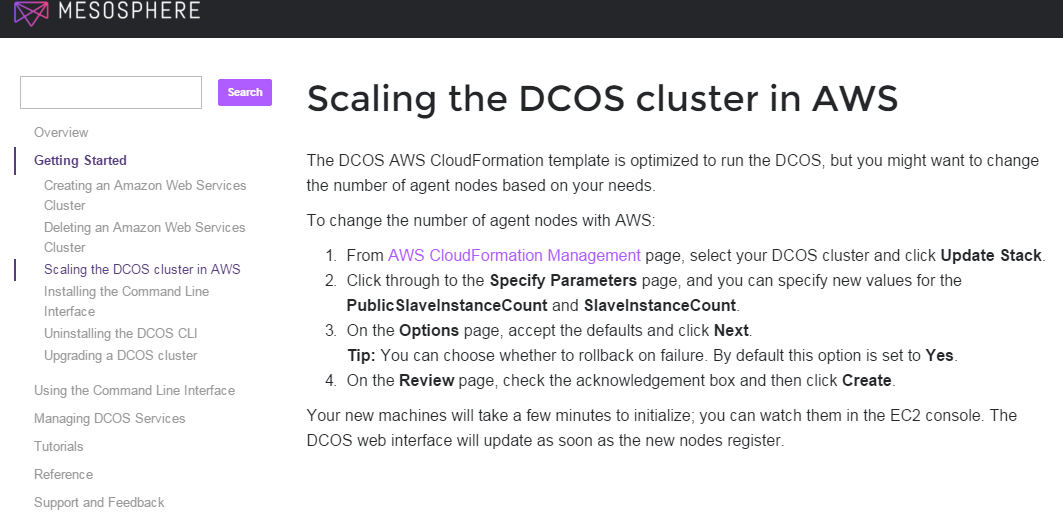
# Conclusion

It is not easily possible to scale down a DCOS-Cluster without using all Data in HDFS.

**This issue blocks use in productive environments.**

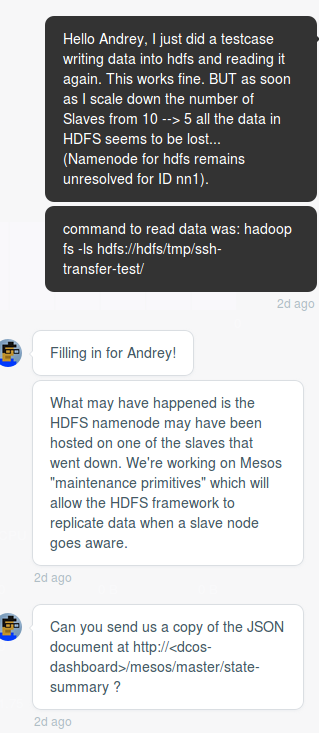
This is the complete documentation for scaling clusters:

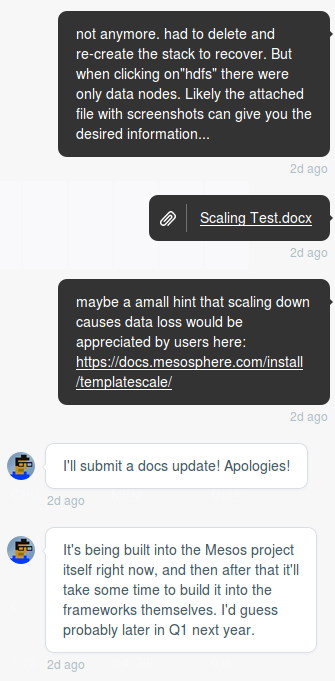
<https://docs.mesosphere.com/install/templatescale/>

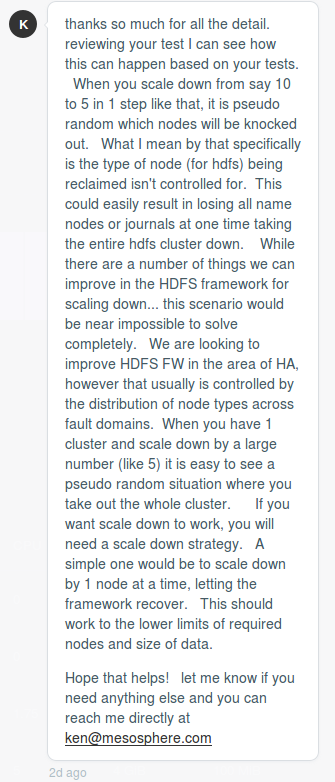


* It does not forbid to scale down and there are no hints that scaling down causes trouble.

# Communication with mesosphere







Email to Ken Sipe (2015-11-16):

Hello Ken,

thank so much for you detailed and quick answer!

A scaling-down strategy sounds very promising to me.

I already implemented scripts to invoke the aws-cli / dcos for creating , deleting and scaling the cluster.

These scripts already periodically check the status of the stack, which I assume to be a good basis for extension.

What I now would like to do is extends the script to scale down using a 1-by-1-scaling-down-strategy as you suggested like this

1. Scale down by 1 node (EC2-Instance)
2. Check if target number of slaves is reached  
   --> Exit if reached
3. Wait for system to become stable again  
   --> how can I determine "stable" ?
4. go to (1)

The only piece of information missing is related to (3).

How do I determine whether the system is stable to be able to continue deletion?

It would be ideal for me to have a dcos-command or some URL which returns some JSON telling me that I may continue.

Any hint would be aapreciated ;-)

Thanks and best regards,

Chris

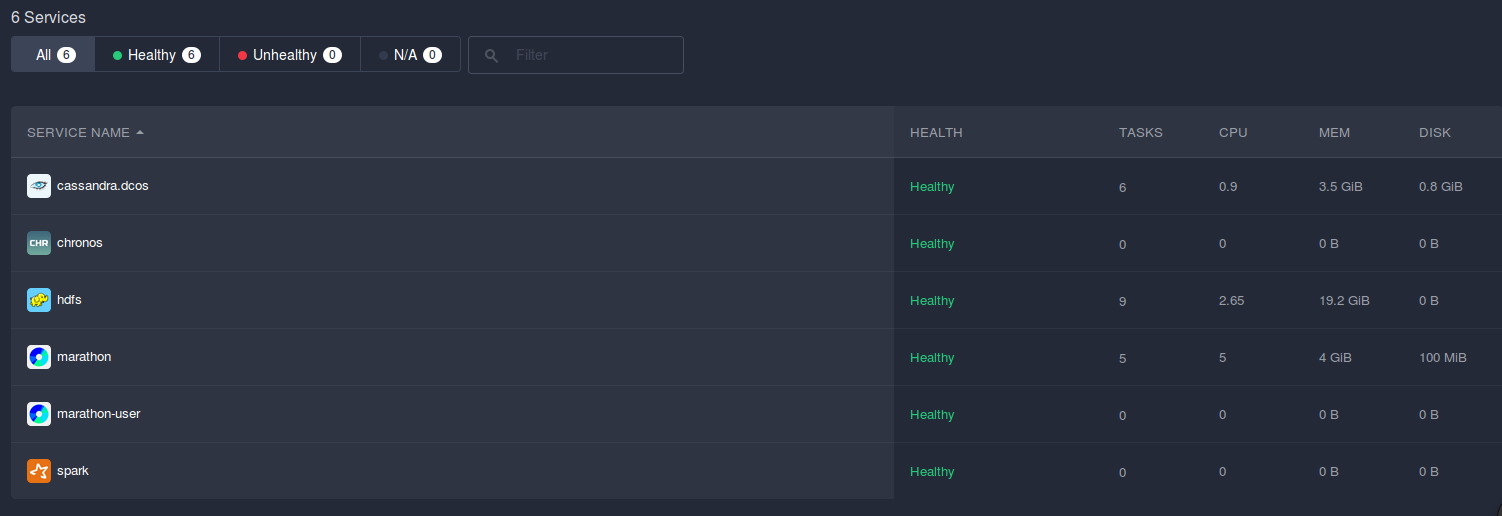
Ken wrote on 2015-11-13:

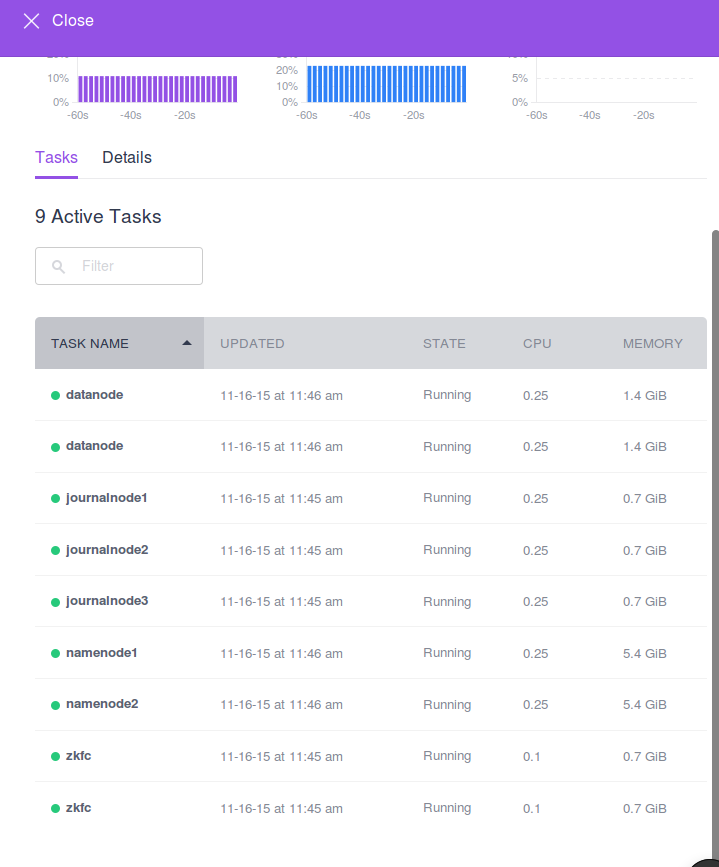
thanks so much for all the detail. reviewing your test I can see how this can happen based on your tests. When you scale down from say 10 to 5 in 1 step like that, it is pseudo random which nodes will be knocked out. What I mean by that specifically is the type of node (for hdfs) being reclaimed isn't controlled for. This could easily result in losing all name nodes or journals at one time taking the entire hdfs cluster down. While there are a number of things we can improve in the HDFS framework for scaling down... this scenario would be near impossible to solve completely. We are looking to improve HDFS FW in the area of HA, however that usually is controlled by the distribution of node types across fault domains. When you have 1 cluster and scale down by a large number (like 5) it is easy to see a pseudo random situation where you take out the whole cluster. If you want scale down to work, you will need a scale down strategy. A simple one would be to scale down by 1 node at a time, letting the framework recover. This should work to the lower limits of required nodes and size of data.

Hope that helps! let me know if you need anything else and you can reach me directly at ken@mesosphere.com

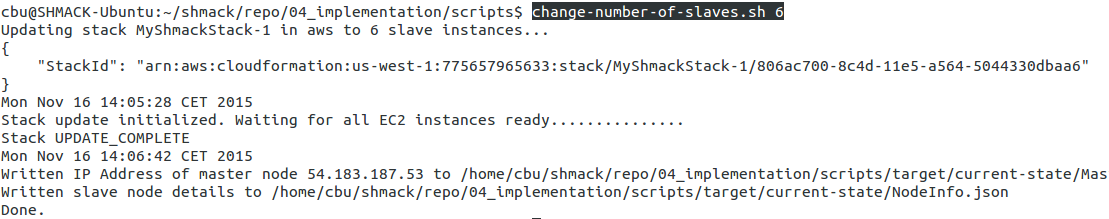
# Scale 1-by-1

## 5 workers (default stack setup)

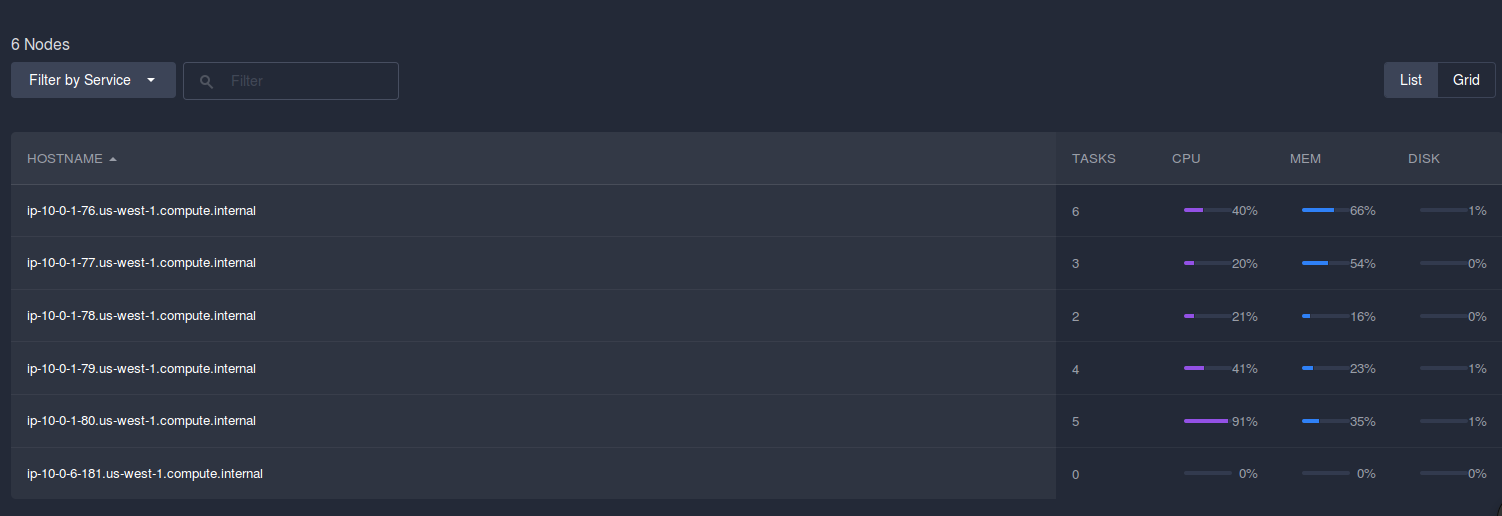




## Scale up to 6 workers



Added node does not receive tasks:



But HDFS nodes increase (+ 1 datanode):

