



Programming Assignment 5

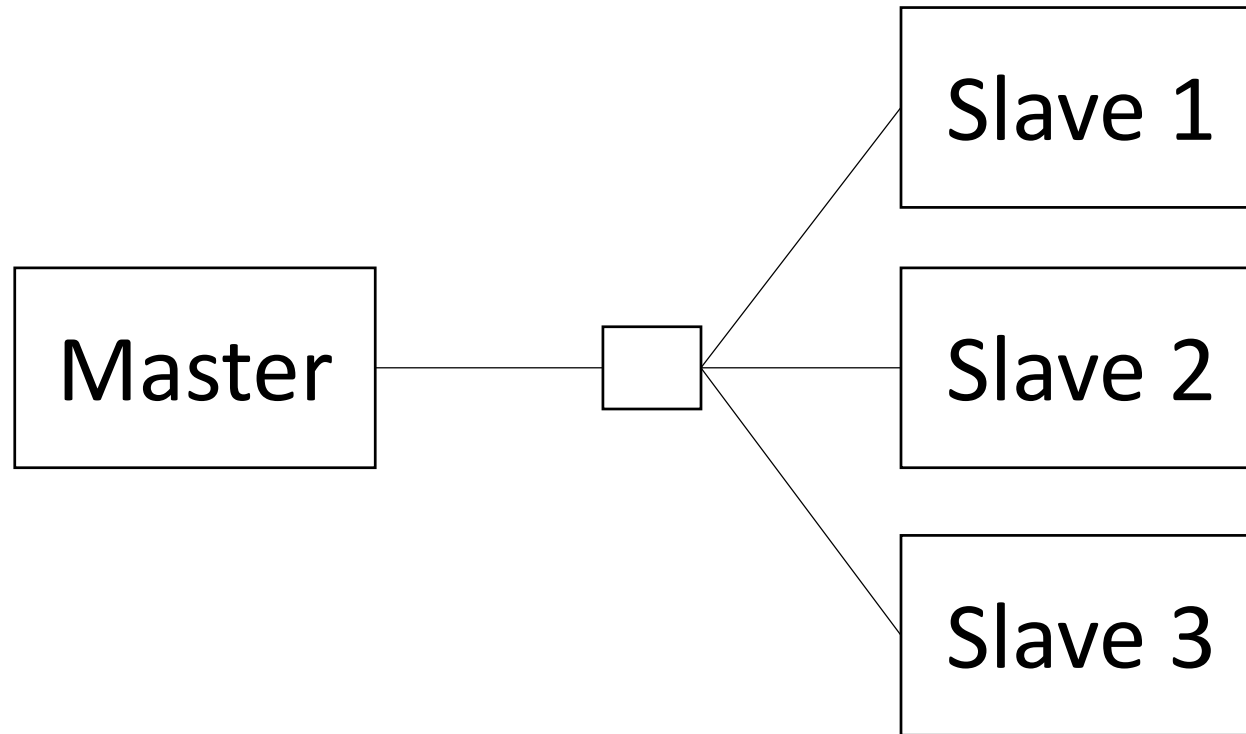
CS 4740 Cloud Computing

Department of Computer Science,
University of Virginia, USA

Goal

- Gain hands-on experience on how to configure a multi-node Hadoop cluster on AWS EC2 instances

Hadoop Architecture



Steps

- Tutorial videos
- Step 1: Request four t2.micro instances on EC2 – free tier
 - Remember to shut down all of them after the assignment
 - Pay attention to the Ubuntu OS version you request. It affects what JAVA version you need to install

Video Link: <https://www.youtube.com/watch?v=cr5RmnyWbYw>

- Step 2: Login to the EC2 instances on Windows PC
 - Using Putty and WinSCP
 - Terminal on Macbook - search how to connect to EC2 instance on Macbook

Video Link: <https://www.youtube.com/watch?v=IRQqR0Fm1oE>

Step 1: Request four t2.micro instances on EC2 – free tier

Configuring the instances simultaneously

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances ⓘ [Launch into Auto Scaling Group ⓘ](#)

You may want to consider launching these instances into an Auto Scaling Group to help you maintain application availability and for easy scaling in the future. Learn how Auto Scaling can help your application stay healthy and cost effective.

Purchasing option ⓘ ☐ Request Spot instances

Network ⓘ vpc-119b066a (default) ↕ ⌂ Create new VPC

Subnet ⓘ No preference (default subnet in any Availability Zone) ↕ ⌂ Create new subnet

Auto-assign Public IP ⓘ Use subnet setting (Enable) ↕

Placement group ⓘ ☐ Add instance to placement group

Capacity Reservation ⓘ Open ↕ ⌂ Create new Capacity Reservation

IAM role ⓘ None ↕ ⌂ Create new IAM role

Shutdown behavior ⓘ Stop ↕

Stop - Hibernate behavior ⓘ ☐ Enable hibernation as an additional stop behavior ⓘ

Enable termination protection ⓘ ☐ Protect against accidental termination ⓘ

Monitoring ⓘ ☐ Enable CloudWatch detailed monitoring ⓘ
Additional charges apply.

Tenancy ⓘ Shared - Run a shared hardware instance ↕
Additional charges will apply for dedicated tenancy.

Elastic Inference ⓘ ☐ Add an Elastic Inference accelerator ⓘ
Additional charges apply.

Cancel Previous **Review and Launch** Next: Add Storage

Step 1: Request four t2.micro instances on EC2 – free tier

Final Outcome

Launch Instance ▾ Connect Actions ▾													
Filter by tags and attributes or search by keyword													
1 to 4 of 4													
<input type="checkbox"/>	Name ▾	Instance ID ▲	Instance Type ▾	Availability Zone ▾	Instance State ▾	Status Checks ▾	Alarm Status	Public DNS (IPv4) ▾	IPv4 Public IP ▾	IPv6 IPs ▾	Key Name ▾	Monitoring ▾	Lau
<input type="checkbox"/>	NameNode	i-02151e1913f7f06c1	t2.micro	us-east-1c	● running	✓ 2/2 checks ...	None	ec2-3-84-252-182.com...	3.84.252.182	-	my-hadoop-key	disabled	Apri
<input type="checkbox"/>	Datanode1	i-0453c57aa65a3597a	t2.micro	us-east-1c	● running	✓ 2/2 checks ...	None	ec2-54-175-221-213.co...	54.175.221.213	-	my-hadoop-key	disabled	Apri
<input type="checkbox"/>	Datanode2	i-0c959861fe69de98a	t2.micro	us-east-1c	● running	✓ 2/2 checks ...	None	ec2-3-91-149-195.com...	3.91.149.195	-	my-hadoop-key	disabled	Apri
<input type="checkbox"/>	Datanode3	i-0d1fc99c91a96938f	t2.micro	us-east-1c	● running	✓ 2/2 checks ...	None	ec2-3-89-219-13.comp...	3.89.219.13	-	my-hadoop-key	disabled	Apri

Step 3: setting up Passwordless SSH

- Setup Passwordless SSH
 - Nodes communicate frequently with each other
 - Too slow to authenticate every time
 - Command: `ssh-keygen -f ~/.ssh/sshkey_rsa -t rsa -P ""`
 - Replace the name `sshkey_rsa` with `id_rsa` and also replace the name for the following steps that use `sshkey_rsa`

Video Link: https://www.youtube.com/watch?v=u7flf_R-gaM&t=3s

Step 3: setting up Passwordless SSH

Commands

```
# set permission level on PEM on all nodes
sudo chmod 600 ~/.ssh/my-hadoop-key.pem

#copy config and pem file to all nodes
scp ~/.ssh/my-hadoop-key.pem ~/.ssh/config Datanode1:~/.ssh
scp ~/.ssh/my-hadoop-key.pem ~/.ssh/config Datanode2:~/.ssh
scp ~/.ssh/my-hadoop-key.pem ~/.ssh/config Datanode3:~/.ssh

#generate key files on Namenode
ssh-keygen -f ~/.ssh/id_rsa -t rsa -P ""

#copy the generated key from sshkey_rsa.pub to authorized keys of namenodes
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys

#copy the file to all datanodes
cat ~/.ssh/id_rsa.pub | ssh Datanode1 'cat >> ~/.ssh/authorized_keys'
cat ~/.ssh/id_rsa.pub | ssh Datanode2 'cat >> ~/.ssh/authorized_keys'
cat ~/.ssh/id_rsa.pub | ssh Datanode3 'cat >> ~/.ssh/authorized_keys'
```


Step 4: Install Hadoop on the four nodes

- Install Hadoop on the four nodes
 - Install JAVA
 - Configure environment variables, e.g.,
 - JAVA_HOME
 - HADOOP_HOME

Video Link: <https://www.youtube.com/watch?v=Z0cOE2SRo5c>

Step 4: Install Hadoop on the four nodes

Commands-1

- #Install Hadoop - Following commands are for all four machines

```
sudo apt-get update
```

- #Installing Java on Ubuntu 16.04

```
sudo apt-get install openjdk-8-jdk-headless
```

- #check java version

```
java -version
```

- #Download Hadoop from Apache to Downloads folder in home

```
wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.3/hadoop-2.7.3.tar.gz -P ~/Downloads/Hadoop
```

- #Uncompress the Hadoop tar file into the /usr/local folder

```
sudo tar zxvf ~/Downloads/Hadoop/hadoop-2.7.3.tar.gz -C /usr/local
```

- #Move all Hadoop related file from /usr/local to /usr/local/hadoop

```
sudo mv /usr/local/hadoop-* /usr/local/hadoop
```

Step 4: Install Hadoop on the four nodes

Commands-2

- #Set environment variables on all the nodes in /home/ubuntu/.profile

```
export JAVA_HOME=/usr
```

```
export PATH=$PATH:$JAVA_HOME/bin
```

```
export HADOOP_HOME=/usr/local/hadoop
```

```
export PATH=$PATH:$HADOOP_HOME/bin
```

```
export HADOOP_CONF_DIR=/usr/local/hadoop/etc/hadoop
```

- #load variables

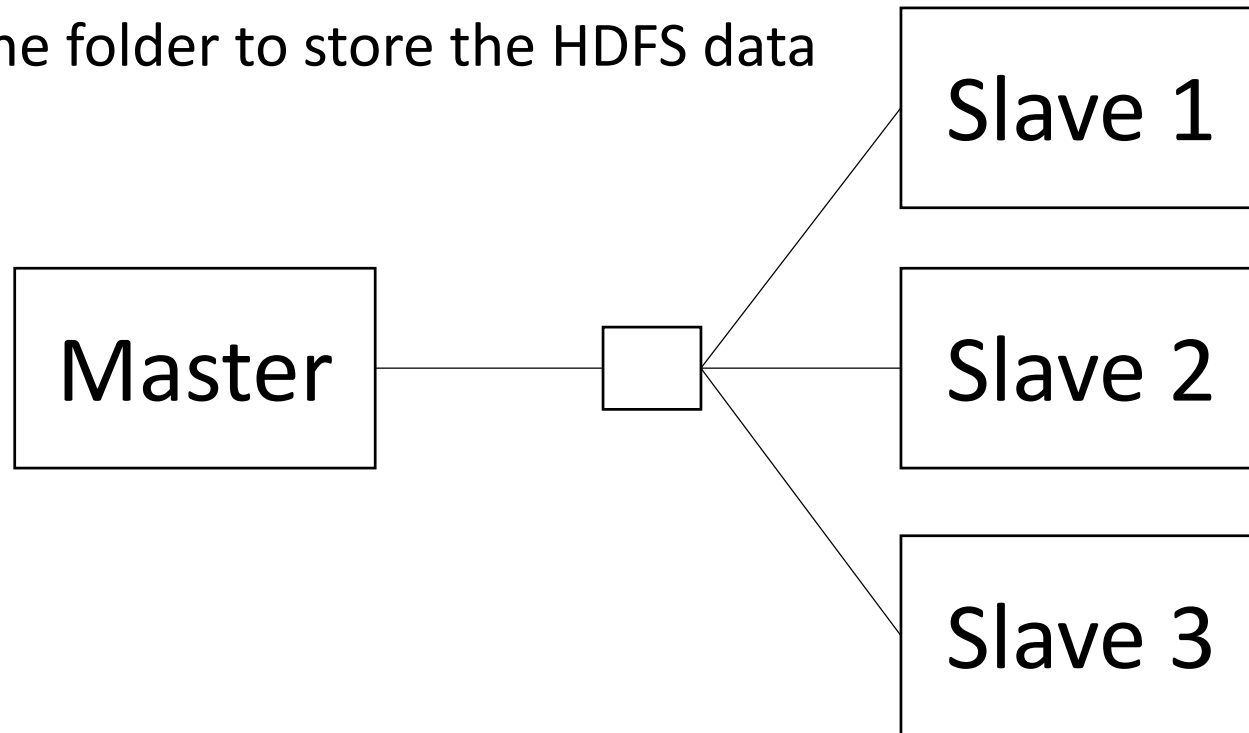
```
. ~/.profile
```

- #change ownership of hadoop folder

```
sudo chown -R ubuntu /usr/local/hadoop
```

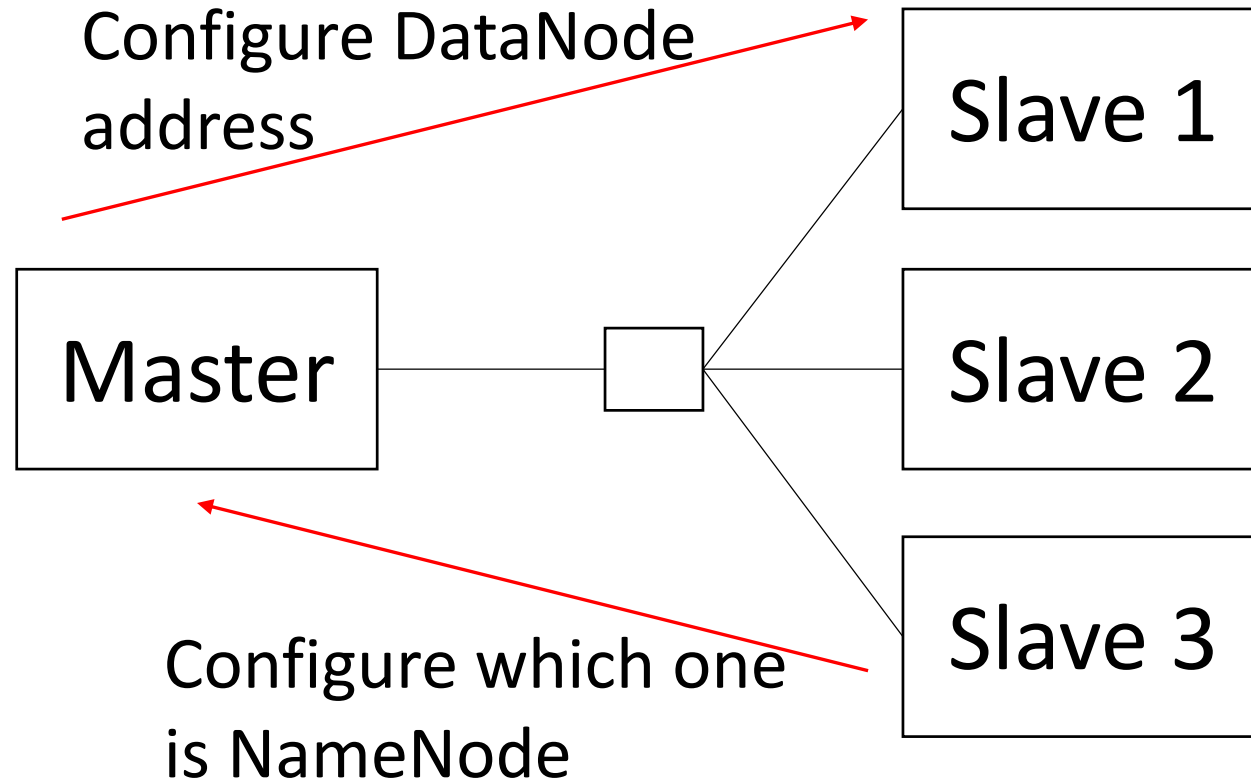
Step 5: Configure NameNode and DataNode

- Configure NameNode and DataNode
 - Hadoop distributed file system - HDFS
 - Configure the folder to store the HDFS data



Step 5: Configure NameNode and DataNode

- Configure NameNode and DataNode



Step 5: Configure NameNode and DataNode Sub-Steps

- 3 types of configuration
 - Sub-step1: Applicable to both Namenode and Datanodes
 - Sub-step2: Applicable to only Namenode
 - Sub-step3: Applicable to only Datanodes

Video Link: https://www.youtube.com/watch?v=PF1dUAKs_bg
<https://www.youtube.com/watch?v=k-DGPlwfitM>

Step 5: Configure NameNode and DataNode

sub-step1: Applicable to both type of nodes

- Change in core-site.xml (Excluded in the following command)
- Change in hadoop-env.sh
- Change in mapred-site.xml
- Change in yarn-site.xml

One way to do it easily: Only change in Namenode and scp the files datanodes.

E.g., `scp $HADOOP_CONF_DIR/mapred-site.xml $HADOOP_CONF_DIR/yarn-site.xml $HADOOP_CONF_DIR/core-site.xml $HADOOP_CONF_DIR/hadoop-env.sh`

Datanode1:`$HADOOP_CONF_DIR`

Datanode1: As set in config (in slide 8: `scp ~/.ssh/my-hadoop-key.pem ~/.ssh/config`
Datanode1:`~/.ssh`)

Replace Datanode1 with each of other Datanodes and run the above command

Step 5: Configure NameNode and DataNode

Applicable to only namenode

- Change in /etc/hosts
- Change in hdfs-site.xml
- Creating Hadoop data directory
- Creating two files named masters and slaves

Step 5: Configure NameNode and DataNode

Applicable to datanodes

- Change in hdfs-site.xml
- Creating Hadoop data directory

Step 6: Start the cluster

- Start the cluster
 - Monitor cluster health by going to `ec2-{instance-address}.amazonaws.com:50070`
 - Showing all three datanodes running properly
- Video Link: https://www.youtube.com/watch?v=Z_5pJMCoeM8

Deliverables

- **What to submit:** You need to submit screenshots for some configuration files, screenshots for testing some simple Hadoop command, and screenshot showing Teragen, Terasort is running to show that you successfully configure the cluster and the benchmark runs successfully.
- You **MUST** submit one screenshot of your cluster health by going to `ec2-{instance-address}.amazonaws.com:50070` when your cluster is running. See the example below.

Deliverables

1. screenshots for some configuration files

- hdfs-site.xml
- mapred-site.xml
- yarn-site.xml
- etc.

Deliverable

2. screenshots for testing some simple Hadoop command

- Some Basic command on hdfs

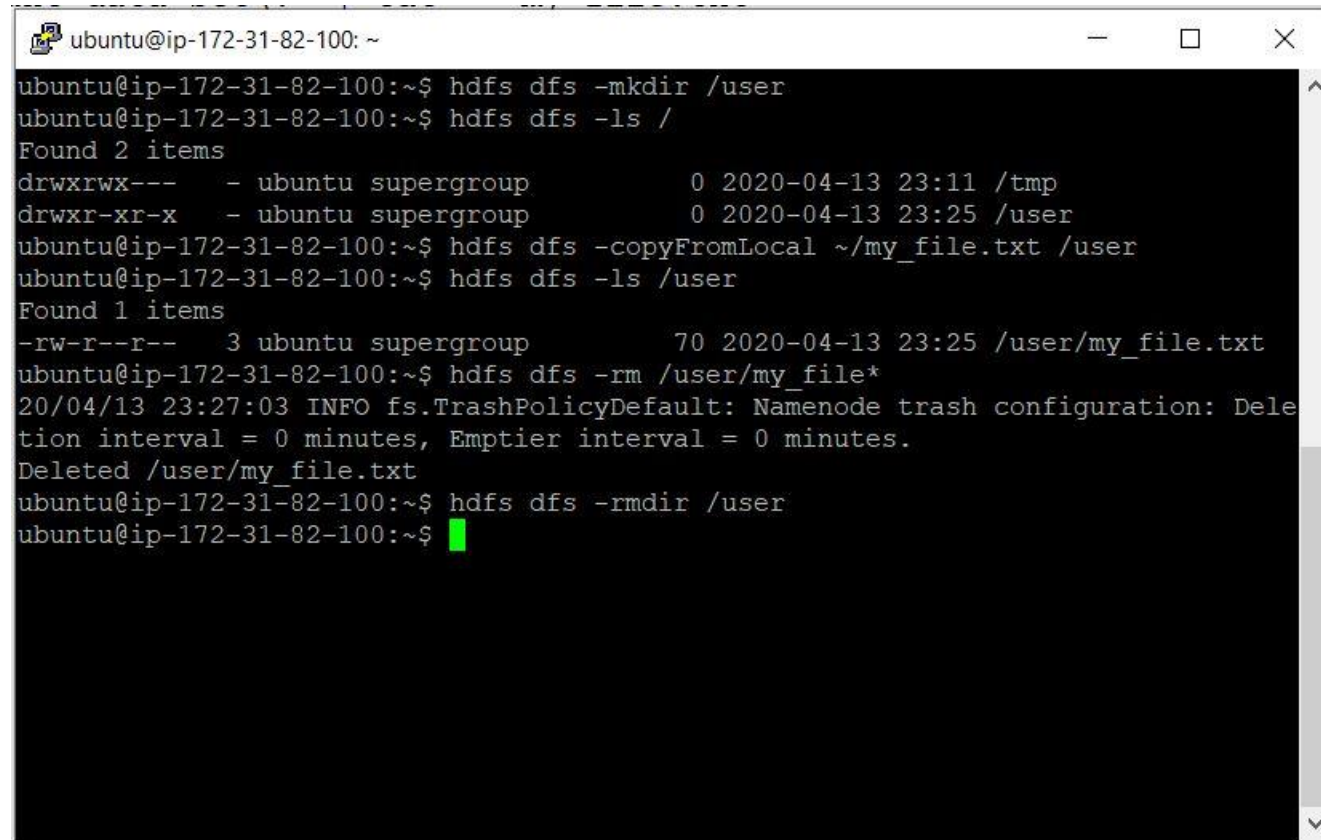
Such as- creating directory, copy from local to hdfs, deleting directory, showing java process (using JPS), etc.

Video Link: <https://www.youtube.com/watch?v=X8Tn3FETqXw>

Deliverable

2. screenshots for testing some simple Hadoop command

deleting directory

A terminal window titled 'ubuntu@ip-172-31-82-100: ~' with standard window controls. It displays a series of Hadoop commands and their outputs. The commands include creating a directory, listing contents, copying a file, listing again, removing a file, and finally removing the directory. The terminal output shows file permissions, ownership, size, and timestamps for the files and directory involved.

```
ubuntu@ip-172-31-82-100:~$ hdfs dfs -mkdir /user
ubuntu@ip-172-31-82-100:~$ hdfs dfs -ls /
Found 2 items
drwxrwx---  - ubuntu supergroup          0 2020-04-13 23:11 /tmp
drwxr-xr-x  - ubuntu supergroup          0 2020-04-13 23:25 /user
ubuntu@ip-172-31-82-100:~$ hdfs dfs -copyFromLocal ~/my_file.txt /user
ubuntu@ip-172-31-82-100:~$ hdfs dfs -ls /user
Found 1 items
-rw-r--r--   3 ubuntu supergroup          70 2020-04-13 23:25 /user/my_file.txt
ubuntu@ip-172-31-82-100:~$ hdfs dfs -rm /user/my_file*
20/04/13 23:27:03 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 0 minutes, Emptier interval = 0 minutes.
Deleted /user/my_file.txt
ubuntu@ip-172-31-82-100:~$ hdfs dfs -rmdir /user
ubuntu@ip-172-31-82-100:~$
```

Deliverable

2. screenshots for testing some simple Hadoop command

Running JPS command for namenode

```
ubuntu@ip-172-31-82-100: ~  
2658 Jps  
2549 SecondaryNameNode  
2332 NameNode  
ubuntu@ip-172-31-82-100:~$ $HADOOP_HOME/sbin/start-yarn.sh  
starting yarn daemons  
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-ubuntu-resource  
manager-ip-172-31-82-100.out  
Datanode1: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-ubuntu-n  
odemanager-ip-172-31-83-151.out  
Datanode2: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-ubuntu-n  
odemanager-ip-172-31-84-177.out  
Datanode3: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-ubuntu-n  
odemanager-ip-172-31-82-215.out  
ubuntu@ip-172-31-82-100:~$ $HADOOP_HOME/sbin/mr-jobhistory-daemon.sh start histo  
ryserver  
starting historyserver, logging to /usr/local/hadoop/logs/mapred-ubuntu-historys  
erver-ip-172-31-82-100.out  
ubuntu@ip-172-31-82-100:~$ jps  
2996 JobHistoryServer  
2549 SecondaryNameNode  
3050 Jps  
2716 ResourceManager  
2332 NameNode  
ubuntu@ip-172-31-82-100:~$
```

Running JPS command for datanode

```
ubuntu@ip-172-31-82-215: ~  
Using username "ubuntu".  
Authenticating with public key "imported-openssh-key"  
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1101-aws x86_64)  
  
 * Documentation:  https://help.ubuntu.com  
 * Management:    https://landscape.canonical.com  
 * Support:       https://ubuntu.com/advantage  
  
42 packages can be updated.  
28 updates are security updates.  
  
New release '18.04.4 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Mon Apr 13 18:39:06 2020 from 73.148.105.25  
ubuntu@ip-172-31-82-215:~$ jps  
2531 Jps  
2341 NodeManager  
2202 DataNode  
ubuntu@ip-172-31-82-215:~$
```


Deliverable

3: screenshot showing Teragen, Terasort is running

- Run Hadoop TeraGen and TeraSort benchmarks
- `hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-*examples*.jar teragen 10000000 /terasort-input`
- `--generate` unsorted data and store the data in `/terasort-input`
- `hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-*examples*.jar terasort /terasort-input /terasort-output`
- `--sort` the generated unsorted data and store the data in `/terasort-output`

Deliverable

3: screenshot showing Teragen, Terasort is running

```
ubuntu@ip-172-31-82-100: ~  
Total time spent by all maps in occupied slots (ms)=10146  
Total time spent by all reduces in occupied slots (ms)=0  
Total time spent by all map tasks (ms)=10146  
Total vcore-milliseconds taken by all map tasks=10146  
Total megabyte-milliseconds taken by all map tasks=10389504  
Map-Reduce Framework  
  Map input records=10000  
  Map output records=10000  
  Input split bytes=164  
  Spilled Records=0  
  Failed Shuffles=0  
  Merged Map outputs=0  
  GC time elapsed (ms)=120  
  CPU time spent (ms)=980  
  Physical memory (bytes) snapshot=200331264  
  Virtual memory (bytes) snapshot=3870457856  
  Total committed heap usage (bytes)=32636928  
org.apache.hadoop.examples.terasort.TeraGen$Counters  
  CHECKSUM=21555350172850  
File Input Format Counters  
  Bytes Read=0  
File Output Format Counters  
  Bytes Written=1000000  
ubuntu@ip-172-31-82-100:~$
```

```
ubuntu@ip-172-31-82-100: ~  
Computing partitions took 261ms  
Spent 452ms computing partitions.  
20/04/14 11:59:02 INFO client.RMProxy: Connecting to ResourceManager at ec2-3-84-  
-252-182.compute-1.amazonaws.com/172.31.82.100:8032  
20/04/14 11:59:03 WARN hdfs.DFSClient: Caught exception  
java.lang.InterruptedException  
  at java.lang.Object.wait(Native Method)  
  at java.lang.Thread.join(Thread.java:1252)  
  at java.lang.Thread.join(Thread.java:1326)  
  at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.closeResponder(DF  
SOutputStream.java:609)  
  at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.endBlock(DFSOutput  
Stream.java:370)  
  at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.run(DFSOutputStre  
am.java:546)  
20/04/14 11:59:03 INFO mapreduce.JobSubmitter: number of splits:2  
20/04/14 11:59:03 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_15  
86865198921_0007  
20/04/14 11:59:03 INFO impl.YarnClientImpl: Submitted application application_15  
86865198921_0007  
20/04/14 11:59:04 INFO mapreduce.Job: The url to track the job: http://ec2-3-84-  
252-182.compute-1.amazonaws.com:8088/proxy/application_1586865198921_0007/  
20/04/14 11:59:04 INFO mapreduce.Job: Running job: job_1586865198921_0007  
[
```

Deliverable

- You **MUST** submit one screenshot of your cluster health by going to `ec2-{instance-address}.amazonaws.com:50070` when your cluster is running. See the example below.



Overview 'ec2-3-84-252-182.compute-1.amazonaws.com:9000' (active)

Started:	Mon Apr 13 23:09:51 UTC 2020
Version:	2.7.3, rbaa91f7c6bc9cb92be5982de4719c1c8af91ccff
Compiled:	2016-08-18T01:41Z by root from branch-2.7.3
Cluster ID:	CID-93f74612-6004-421a-90b1-3f77845abbc5
Block Pool ID:	BP-657146638-172.31.82.100-1586819369760

Summary

Security is off.

Safemode is off.

7 files and directories, 0 blocks = 7 total filesystem object(s).

Heap Memory used 34.64 MB of 45.97 MB Heap Memory. Max Heap Memory is 966.69 MB.

Non Heap Memory used 38.74 MB of 39.44 MB Committed Non Heap Memory. Max Non Heap Memory is -1 B.

Configured Capacity:	23.08 GB
DFS Used:	72 KB (0%)
Non DFS Used:	5.9 GB
DFS Remaining:	17.17 GB (74.43%)
Block Pool Used:	72 KB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Use Nodes	9 (Decommissioned: 0)



Questions?