

CS 553 Cloud Computing

Programming Assignment 2

Project Report

Dinesh Chowday Jasti (A20335710)

Jayanth Vangari (A20337867)

Sai Sravan Rachiraju (A20332891)

The following are the most important aspects in this assignment:

- Creating a Virtual Cluster using AMAZON WEB SERVICES(AWS)
- Implementing Word count in JAVA
- Implementing Word count in HADOOP
- Implementing Word count in SWIFT
- Implementing Word count using MPI
- Implementing Sorting across 4 approaches (JAVA, HADOOP, SWIFT, MPI)
- Measuring the performance of the above

Creating a Virtual Cluster using AMAZON WEB SERVICES(AWS):

First we log in into the Amazon AWS service website (<http://aws.amazon.com/>) and we select EC2 from the network and services section. We select launch instance here. Then we configure our service to meet our requirements. We choose the AMI we need. Then we choose the instance type which can be micro, small, medium, large Xlarge, 2Xlarge, 4Xlarge or 8Xlarge. Then we configure the instance i.e we can choose the instance to be either spot instance or on Demand instance. We then add storage to our selected instance and configure the security group. We add TCP, UDP and ICMP protocols from the security group. We then create the instance. We access the instance by connecting to the instance by suing the public IP address of it and later on we can configure the settings of the AMI that we are using.

Implementing Word count in JAVA:

We implement a java program to count the occurrence of each word in a given input file of size 1MB and 1GB. We print the output into a file in the increasing order of the frequency of the word.

Implementing Word count in HADOOP:

After downloading and installing HADOOP and extracting its components and adding the library paths to the bash we connect to the AWS instance using ssh -i key (IP address). The word count application is implemented using HADOOP for 1MB on a single node and 10GB dataset on 16 nodes.

Implementing Word count in SWIFT:

The same word count application is implemented using SWIFT programming language for a 10GB dataset on 16 nodes.

WORD COUNT IN MPI:

The same word count application is implemented using MPI (Message Passing Interface) for a 10GB dataset on 16 nodes.

PERFORMANCE CALCULATION:

From the given data sets, all the three version of word count i.e. JAVA, HADOOP and SWIFT their performances are obtained on 1 node and 16 nodes. These are compared and performance graphs are drawn for them. The performance result of shared memory Word count is compared with the performance of swift word count and Hadoop word count which are implemented on 16 nodes. For each of the above separate graphs are drawn to show their performance.

SORTING:

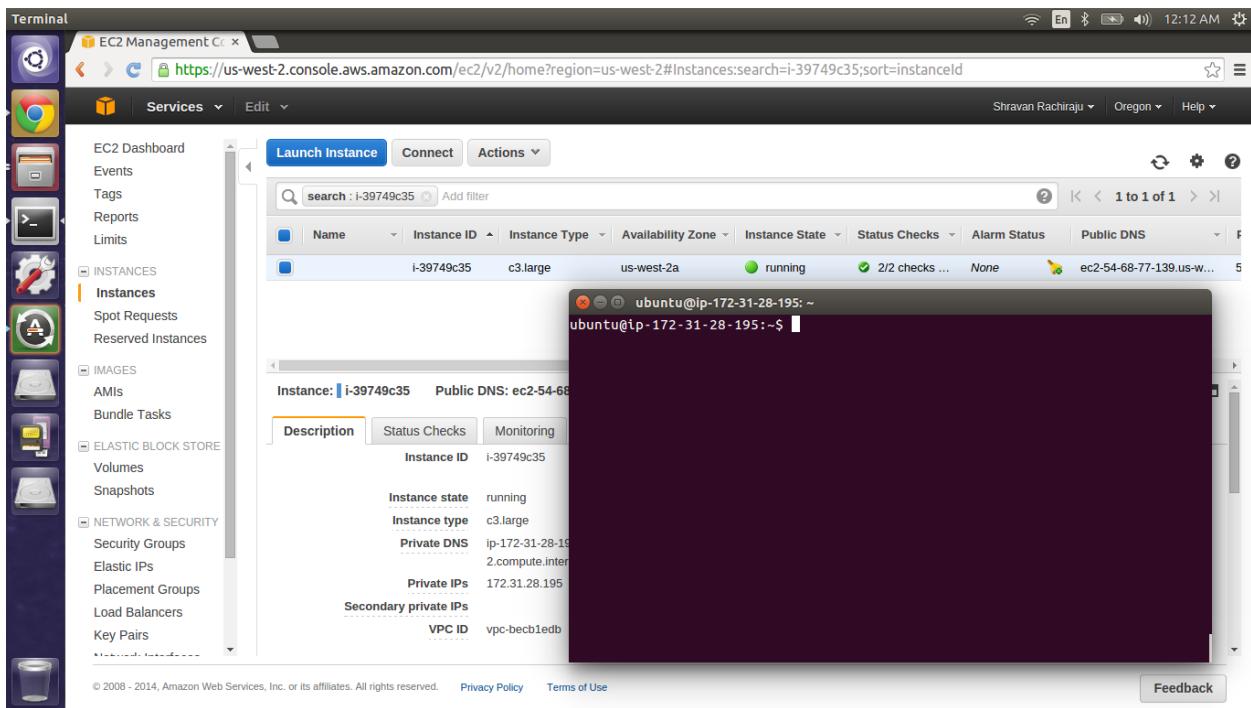
Implementation and evaluation the performance of sorting for all the 4 versions of word count i.e. for JAVA, HADOOP, SWIFT, MPI on a 10GB dataset is done. This sorting performance includes reading, sorting and writing data to disks.

Creating an instance in Amazon(AWS):

- Log in into the Amazon AWS service website (<http://aws.amazon.com>).
- Select EC2 from the network and services section.
- Select launch instance. Then configure the service and chose the best AMI to meet the requirements.
- Then choose the instance type which can be micro, small, medium, large Xlarge, 2Xlarge, 4Xlarge or 8Xlarge. For our assignment it is c3.large
- Then we configure the instance i.e we can choose the instance to be either spot instance or on Demand instance.
- Add storage to our selected instance.
- Configure the security group. We add TCP, UDP and ICMP protocols from the security group. We then create the instance. We access the instance by connecting to the instance by suing the public IP address of it and later on we can configure the settings of the AMI that we are using.

The screenshot shows the AWS EC2 Management Console in Google Chrome. The URL is https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:search=i-39749c35;sort=instanceId. The interface includes a sidebar with icons for EC2 Dashboard, Events, Tags, Reports, Limits, Instances (selected), Spot Requests, Reserved Instances, Images (AMIs, Bundle Tasks), Elastic Block Store (Volumes, Snapshots), and Network & Security (Security Groups, Elastic IPs, Placement Groups, Load Balancers, Key Pairs). The main area shows a table of instances with one row selected: Instance ID: i-39749c35, Instance Type: c3.large, Availability Zone: us-west-2a, Instance State: running, Status Checks: Initializing, Public DNS: ec2-54-68-77-139.us-west-2.compute.amazonaws.com. Below the table, a detailed view for the selected instance shows fields like Description, Instance ID, Instance state, Instance type, Private DNS, Private IPs, Secondary private IPs, VPC ID, Subnet ID, Public DNS, Public IP, Elastic IP, Availability zone, Security groups, Scheduled events, AMI ID, and Platform. The instance is described as 'running' with a Public DNS of ec2-54-68-77-139.us-west-2.compute.amazonaws.com and a Public IP of 54.68.77.139.

The above screen shot is of a c3.large instance.



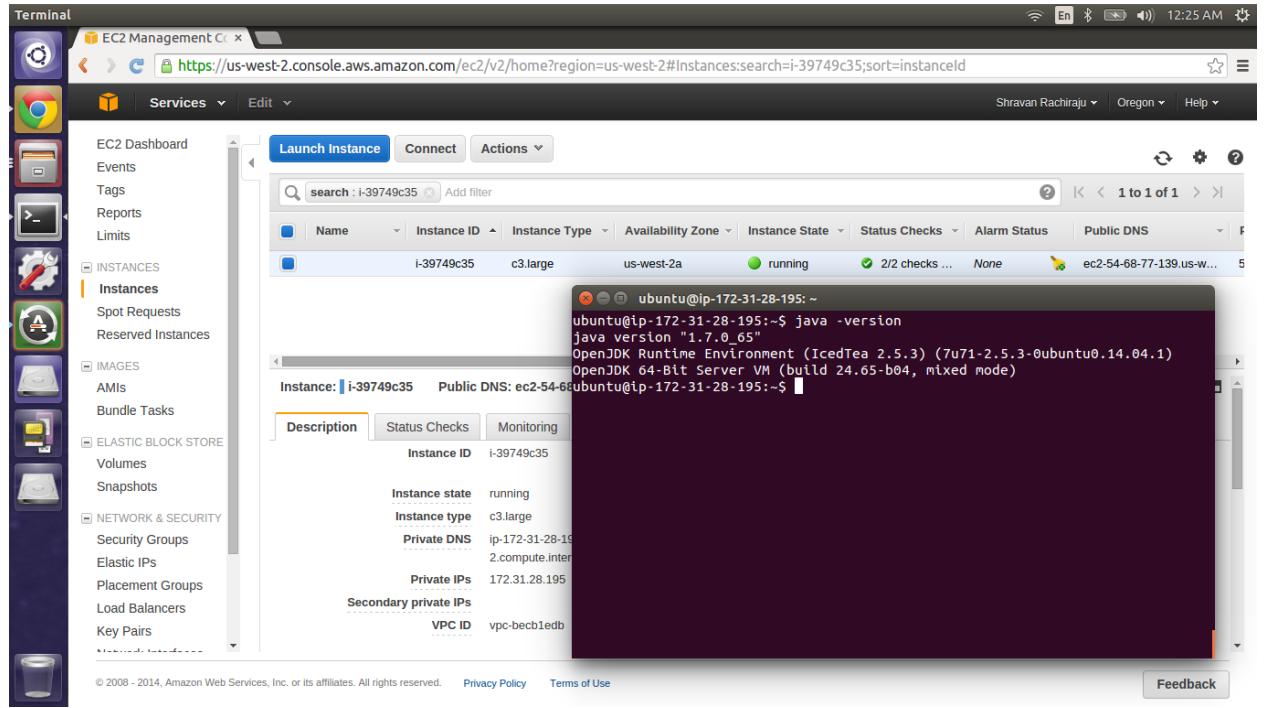
The above screen shot is of a connected instance via the Ubuntu terminal using the below commands:

```
chmod 400 keygiven( This is to change the permission to be read by owner. )
```

```
ssh -i key IP address
```

In our case we use ssh -i newkey.pem Ubuntu@54.68.77.139

- After getting connected to the AMI we now have to install java on the virtual node.
- Install java by using sudo apt-get install openjdk-7-jre-headless.
- Install javac by using sudo apt-get install openjdk-7-jdk.
- Now update all the installed applications by using sudo apt-get update
- Now check the java version by using java -version.



java –version is shown in the above screen shot

WORD COUNT IN JAVA:

Configuration:

| | |
|---------------|--------------------|
| VERSION | : JAVA 1.7 |
| INSTANCE TYPE | : c3. Large UBUNTU |
| RAM | : 3.75 GB |
| NO. OF CORES | : 2 virtual cores |
| STORAGE | : 32 GB |

METHODOLOGY:

We take input of total size 10 gigabyte and count the number of lines in the file.

- On the basis of number of lines, we further split the data into equal no. of chunks.
- The file name of each chunk is fetched using a user defined function called *fileGeneration()*. The *fileGeneration()* has a dual functionality of fetching the file name and removing from the list once it is processed.
- In the function *count()*, each chunk is opened in the read mode and every line of chunk is read and every line is tokenized and then regular that it reads is stored into the map and it is tokenized into words with delimiters as “ ”.
- A pattern match is run on each of these tokens to find the leading and trailing special symbols. Once these special symbols are determined the word is stored into the map (*m1*). If the word has occurred for first time 1 is assigned as key for that particular word. The value of this key is incremented with subsequent occurrence of the token.
- The above mentioned point c and d, are executed under function called *run()*. The threads are started and execute one file per thread. Numbers of threads are dependent on the number of chunks of a file.
- The point d is executed within the semaphore block to achieve the synchronization and hence avoid deadlocks in threads.
- Finally, we execute sorting. In sorting the hash map is converted into array list and this list is sorted using inbuilt sort function.
- The value of map is stored in the text file name *output.txt*.

Outputs:

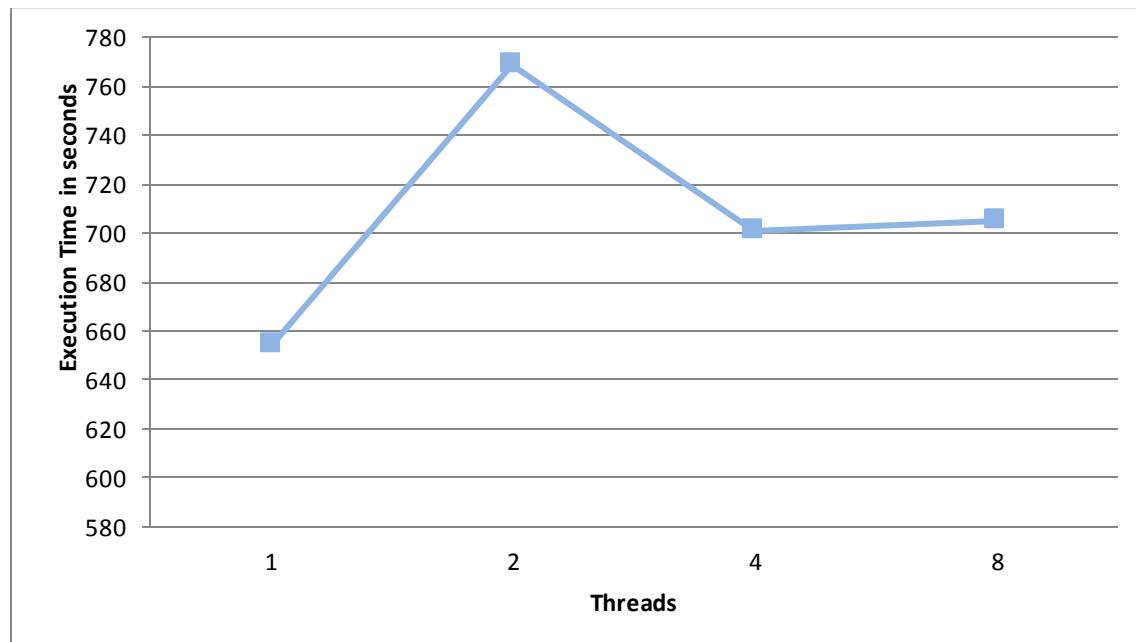
10 GB DATASET ON c3.large instance (Only Word Count with Frequency no sorting):

| Number of threads | Execution Time in Seconds |
|-------------------|---------------------------|
| 1 | 654.552 |
| 2 | 768.757 |
| 4 | 700.781 |
| 8 | 704.746 |

EXPLANATION:

Every thread is responsible for reading a file and processing it achieves the word count functionality

- Best performance was at thread 1



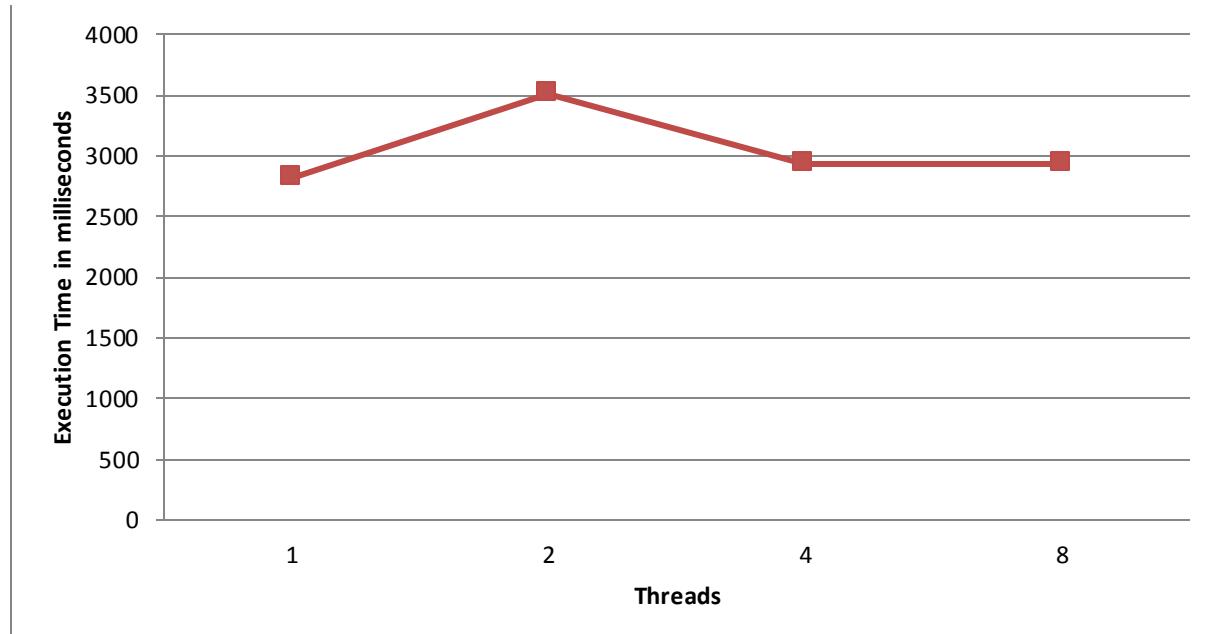
10GB DATASET ON c3.large instance(Sorted word count without frequency based on the ASCII) :

| Number of threads | Execution Time in Seconds |
|-------------------|---------------------------|
| 1 | 2819.673 |
| 2 | 3503.585 |
| 4 | 2935.958 |
| 8 | 2927.552 |

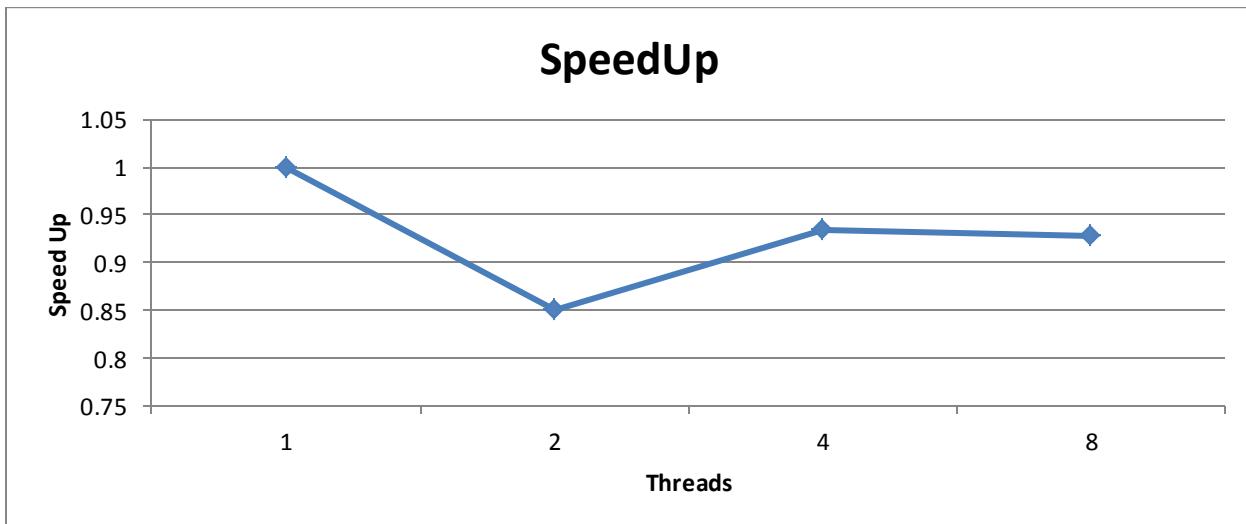
EXPLANATION:

In a multithreaded process on a single processor, the processor can switch execution resources between threads, resulting in concurrent execution .

- Best performance was at thread 1. Thread 4 and 8 were very close in terms of execution time.



| Number of threads | Execution Time in Seconds | Execution Time in Seconds for 1 Node | Speed Up |
|-------------------|---------------------------|--------------------------------------|-------------|
| 1 | 654.552 | 654.552 | 1 |
| 2 | 768.757 | 654.552 | 0.851442003 |
| 4 | 700.781 | 654.552 | 0.934032173 |
| 8 | 704.746 | 654.552 | 0.928777176 |



Screenshots Word Count with Frequency no sorting :

The screenshot shows the AWS EC2 Management Console interface. On the left, there's a sidebar with various services like EC2 Dashboard, Instances, Images, and Network & Security. The main area shows a list of instances. One instance, with Instance ID i-10492dda and Public DNS ec2-54-183-153-22, is selected. A terminal window is open on this instance, displaying the output of a word count script. The script asks for the number of threads (1) and then lists 16 files (file1.txt to file16.txt) with their assigned threads. The total execution time is 654.552 milliseconds.

```
ubuntu@ip-172-31-3-140:~$ Enter number of Threads:  
1  
Thread-0 ---> assigned to ---> file1.txt  
Thread-0 ---> assigned to ---> file2.txt  
Thread-0 ---> assigned to ---> file3.txt  
Thread-0 ---> assigned to ---> file4.txt  
Thread-0 ---> assigned to ---> file5.txt  
Thread-0 ---> assigned to ---> file6.txt  
Thread-0 ---> assigned to ---> file7.txt  
Thread-0 ---> assigned to ---> file8.txt  
Thread-0 ---> assigned to ---> file9.txt  
Thread-0 ---> assigned to ---> file10.txt  
Thread-0 ---> assigned to ---> file11.txt  
Thread-0 ---> assigned to ---> file12.txt  
Thread-0 ---> assigned to ---> file13.txt  
Thread-0 ---> assigned to ---> file14.txt  
Thread-0 ---> assigned to ---> file15.txt  
Thread-0 ---> assigned to ---> file16.txt  
ubuntu@ip-172-31-3-140:~$ Total time taken: 654552.0 milliseconds  
ubuntu@ip-172-31-3-140:~$ Execution Time(s): 654.552  
ubuntu@ip-172-31-3-140:~$ Execution Time(min): 10.9092  
ubuntu@ip-172-31-3-140:~$ Execution Time(hr): 0.18182
```

This screenshot is similar to the first one, but it shows the output for 2 threads instead of 1. The terminal window displays the same word count script running with 2 threads. It lists 16 files assigned to Thread-0 and Thread-1. The total execution time is 763.757 milliseconds.

```
ubuntu@ip-172-31-14-222:~$ Enter number of Threads:  
2  
Thread-0 ---> assigned to ---> file1.txt  
Thread-1 ---> assigned to ---> file2.txt  
Thread-0 ---> assigned to ---> file3.txt  
Thread-1 ---> assigned to ---> file4.txt  
Thread-0 ---> assigned to ---> file5.txt  
Thread-1 ---> assigned to ---> file6.txt  
Thread-0 ---> assigned to ---> file7.txt  
Thread-1 ---> assigned to ---> file8.txt  
Thread-0 ---> assigned to ---> file9.txt  
Thread-1 ---> assigned to ---> file10.txt  
Thread-0 ---> assigned to ---> file11.txt  
Thread-1 ---> assigned to ---> file12.txt  
Thread-0 ---> assigned to ---> file13.txt  
Thread-1 ---> assigned to ---> file14.txt  
Thread-0 ---> assigned to ---> file15.txt  
Thread-1 ---> assigned to ---> file16.txt  
ubuntu@ip-172-31-14-222:~$ Total time taken: 763757.0 milliseconds  
ubuntu@ip-172-31-14-222:~$ Execution Time(s): 763.757  
ubuntu@ip-172-31-14-222:~$ Execution Time(min): 12.729283  
ubuntu@ip-172-31-14-222:~$ Execution Time(hr): 0.21215472
```

The screenshot shows the AWS EC2 Management Console interface. On the left, there's a sidebar with various services like EC2 Dashboard, Events, Tags, Reports, Limits, Instances, AMIs, and Network & Security. The main area displays a list of instances. One instance, i-61482cab, is selected and shown in more detail. The instance details pane shows the following information:

| Instance ID | Instance Type | Availability Zone | Instance State | Status Checks | Alarm Status | Public DNS |
|-------------|---------------|-------------------|----------------|---------------|--------------|-------------------------------------|
| i-61482cab | c3.large | us-west-1a | running | Passing | N/A | ec2-51-198-182-125.usr.ec2.internal |

Below the instance details, a terminal window is open with the command:

```
ubuntu@ip-172-31-13-253: ~
```

The terminal output shows the execution of a multi-threaded task:

```
Enter number of Threads: 4
Thread-0 ----> assigned to ----> file1.txt
Thread-1 ----> assigned to ----> file2.txt
Thread-2 ----> assigned to ----> file3.txt
Thread-3 ----> assigned to ----> file4.txt
Thread-0 ----> assigned to ----> file5.txt
Thread-1 ----> assigned to ----> file6.txt
Thread-2 ----> assigned to ----> file7.txt
Thread-3 ----> assigned to ----> file8.txt
Thread-0 ----> assigned to ----> file9.txt
Thread-1 ----> assigned to ----> file10.txt
Thread-2 ----> assigned to ----> file11.txt
Thread-3 ----> assigned to ----> file12.txt
Thread-0 ----> assigned to ----> file13.txt
Thread-1 ----> assigned to ----> file14.txt
Thread-2 ----> assigned to ----> file15.txt
Thread-3 ----> assigned to ----> file16.txt
Total time taken: 700781.0 milliseconds
Execution Time(s): 700.781
Execution Time(min): 11.679684
Execution Time(hr): 0.1946614
```

This screenshot is similar to the one above, showing the AWS EC2 Management Console. The instance details pane for instance i-944a2e5e shows the following information:

| Instance ID | Instance Type | Availability Zone | Instance State | Status Checks | Alarm Status | Public DNS |
|-------------|---------------|-------------------|----------------|---------------|--------------|-------------------------------------|
| i-944a2e5e | c3.large | us-west-1a | running | Passing | N/A | ec2-51-198-182-116.usr.ec2.internal |

The terminal window shows the same multi-threaded task execution as the previous instance:

```
ubuntu@ip-172-31-3-116: ~
```

```
Enter number of Threads: 8
Thread-0 ----> assigned to ----> file1.txt
Thread-1 ----> assigned to ----> file2.txt
Thread-2 ----> assigned to ----> file3.txt
Thread-3 ----> assigned to ----> file4.txt
Thread-4 ----> assigned to ----> file5.txt
Thread-5 ----> assigned to ----> file6.txt
Thread-6 ----> assigned to ----> file7.txt
Thread-7 ----> assigned to ----> file8.txt
Thread-5 ----> assigned to ----> file9.txt
Thread-0 ----> assigned to ----> file10.txt
Thread-1 ----> assigned to ----> file11.txt
Thread-7 ----> assigned to ----> file12.txt
Thread-6 ----> assigned to ----> file13.txt
Thread-4 ----> assigned to ----> file14.txt
Thread-3 ----> assigned to ----> file15.txt
Thread-5 ----> assigned to ----> file16.txt
Total time taken: 704746.0 milliseconds
Execution Time(s): 704.746
Execution Time(min): 11.745767
Execution Time(hr): 0.19576278
```

Running Instance:

The screenshot shows the AWS EC2 Management Console interface. On the left, there's a sidebar with various service icons and a navigation menu. The main area displays a table of running instances. The table includes columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS. One instance, with the Name '116' and Instance ID 'i-944a2e5e', is selected. Below the table, a detailed view of this instance is shown with tabs for Description, Status Checks, Monitoring, and Tags. The 'Description' tab is active, displaying information such as Instance ID, Public DNS, Instance state, Public IP, Instance type, Private DNS, Availability zone, Security groups, Secondary private IPs, VPC ID, and AMI ID.

Screen Shots Sorted word count without frequency based on the ASCII:

This screenshot shows the same EC2 Management Console interface as above, but with a terminal window open on the right side. The terminal window has a dark background and displays the output of a command. It starts with 'ubuntu@ip-172-31-3-140: ~' followed by 'Enter number of Threads:'. Then it lists multiple lines of text starting with 'Thread-0' followed by a series of file names: file1.txt, file2.txt, file3.txt, file4.txt, file5.txt, file6.txt, file7.txt, file8.txt, file9.txt, file10.txt, file11.txt, file12.txt, file13.txt, file14.txt, file15.txt, file16.txt. Below this, it says 'The total number of unique words are: 3368742'. At the bottom, it provides performance metrics: 'Total time taken: 2819673.0 milliseconds', 'Execution Time(s): 2819.673', and 'Execution Time(min): 46.9945'. The rest of the interface is identical to the first screenshot, showing the list of instances and the sidebar.

The screenshot shows the AWS EC2 Management Console interface. On the left, there's a sidebar with various services like EC2 Dashboard, Events, Tags, Reports, Limits, Instances, AMIs, and more. The Instances section is currently selected. In the main pane, a terminal window is open on instance **i-d00d691a**. The terminal output shows a program running with the following parameters:

```
ubuntu@ip-172-31-14-222: ~
Enter number of Threads:
2
Thread-0 ---> assigned to ---> file1.txt
Thread-1 ---> assigned to ---> file2.txt
Thread-0 ---> assigned to ---> file3.txt
Thread-1 ---> assigned to ---> file4.txt
Thread-1 ---> assigned to ---> file5.txt
Thread-0 ---> assigned to ---> file6.txt
Thread-1 ---> assigned to ---> file7.txt
Thread-0 ---> assigned to ---> file8.txt
Thread-1 ---> assigned to ---> file9.txt
Thread-0 ---> assigned to ---> file10.txt
Thread-1 ---> assigned to ---> file11.txt
Thread-0 ---> assigned to ---> file12.txt
Thread-1 ---> assigned to ---> file13.txt
Thread-0 ---> assigned to ---> file14.txt
Thread-1 ---> assigned to ---> file15.txt
Thread-0 ---> assigned to ---> file16.txt
The total number of unique words are: 3368742
Total time taken: 3503585.0 milliseconds
Execution Time(s): 3503.585
Execution Time(min): 58.39308
```

This screenshot is similar to the one above, showing the AWS EC2 Management Console. The terminal window is now on instance **i-61482cab**. The terminal output shows a program running with the following parameters:

```
ubuntu@ip-172-31-13-253: ~
Enter number of Threads:
4
Thread-0 ---> assigned to ---> file1.txt
Thread-1 ---> assigned to ---> file2.txt
Thread-2 ---> assigned to ---> file3.txt
Thread-3 ---> assigned to ---> file4.txt
Thread-1 ---> assigned to ---> files.txt
Thread-0 ---> assigned to ---> file6.txt
Thread-3 ---> assigned to ---> file7.txt
Thread-0 ---> assigned to ---> file8.txt
Thread-2 ---> assigned to ---> file9.txt
Thread-1 ---> assigned to ---> file10.txt
Thread-3 ---> assigned to ---> file11.txt
Thread-2 ---> assigned to ---> file12.txt
Thread-0 ---> assigned to ---> file13.txt
Thread-1 ---> assigned to ---> file14.txt
Thread-3 ---> assigned to ---> file15.txt
Thread-2 ---> assigned to ---> file16.txt
The total number of unique words are: 3368742
Total time taken: 2935958.0 milliseconds
Execution Time(s): 2935.958
Execution Time(min): 48.932632
```

The screenshot shows the AWS EC2 Management Console interface. On the left, there's a sidebar with various services like EC2 Dashboard, Events, Tags, Reports, Limits, Instances, AMIs, and Network & Security. The main area shows a list of instances, with one instance selected (ID: i-944a2e5e). A terminal window is open on this instance, displaying the following output:

```
ubuntu@ip-172-31-3-116: ~
Enter number of Threads:
8
Thread-0 ----> assigned to ----> file1.txt
Thread-1 ----> assigned to ----> file2.txt
Thread-2 ----> assigned to ----> file3.txt
Thread-3 ----> assigned to ----> file4.txt
Thread-4 ----> assigned to ----> file5.txt
Thread-5 ----> assigned to ----> file6.txt
Thread-6 ----> assigned to ----> file7.txt
Thread-7 ----> assigned to ----> file8.txt
Thread-8 ----> assigned to ----> file9.txt
Thread-9 ----> assigned to ----> file10.txt
Thread-10 ----> assigned to ----> file11.txt
Thread-11 ----> assigned to ----> file12.txt
Thread-12 ----> assigned to ----> file13.txt
Thread-13 ----> assigned to ----> file14.txt
Thread-14 ----> assigned to ----> file15.txt
Thread-15 ----> assigned to ----> file16.txt
The total number of unique words are: 3368742

Total time taken: 2927552.0 milliseconds
Execution Time(s): 2927.552
Execution Time(min): 48.792534
```

Best Performance:

Normal Word Count with Frequency no sorting: 1 Thread

Sorting Sorted word count without frequency based on the ASCII: 1 Thread

Setting Up a Virtual Cluster of 16 Nodes + 1 Master:

Creating an instance in Amazon(AWS):

- Log in into the Amazon AWS service website (<http://aws.amazon.com/>).
- Select EC2 from the network and services section.
- Select launch instance. Then configure the service and chose the best AMI to meet the requirements.
- Then choose the instance type which can be micro, small, medium, large Xlarge, 2Xlarge, 4Xlarge or 8Xlarge. For our assignment it is c3.large
- Then we configure the instance i.e we can choose the instance to be either spot instance or on Demand instance.
- Add storage to our selected instance.
- Configure the security group. We add TCP, UDP and ICMP protocols from the security group. We then create the instance. We access the instance by connecting to the instance by suing the public IP address of it and later on we can configure the settings of the AMI that we are using.

To Connect to the Node:

chmod 400 keygiven(This is to change the permission to be read by owner.)

ssh -i key IP address

In our case we use:

1. chmod 400 newkey.pem
2. ssh -i newkey.pem Ubuntu@54.69.130.130

After the connection is established:

- After getting connected to the AMI we now have to install java on the virtual node.
- Install java by using sudo apt-get install openjdk-7-jreheadless.
- Install javac by using sudo apt-get install openjdk-7-jdk.
- Now update all the installed applications by using sudo apt-get update
- Now check the java version by using java –version.
- Now right click the instance we created and select “create image” option. This will create an image for what all we did till now and reflect the same in the other instances.
- Go to the images tab in the left panel of the AWS webpage and select AMI images. You will see the name of the image we just created. Select that and launch the other 16 nodes.

Screen Shots of 17 nodes (1 Master and the other 16 are slaves):

EC2 Management Console - Google Chrome

EC2 Management C × Hadoop v2 overview × Hadoop 2.4-Instal... ×

https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:sort=launchTime

Apps Hadoop 2.4-Inst... Setting up Hadoop Hadoop Tutorial... Apache Hadoop Hadoop v2 over...

Services Edit

Launch Instance Connect Actions

Filter by tags and attributes or search by keyword

| Name | Instance ID | Instance Type | Availability Zone | Instance State | Status Checks | Alarm Status | Public DNS |
|---------|-------------|---------------|-------------------|----------------|----------------|--------------|--------------------------|
| Master | i-8c789680 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-69-130-130.us... |
| Slave1 | i-784ea074 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-111-103.us... |
| Slave2 | i-ad739da1 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-112-169.us... |
| Slave3 | i-27719f2b | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-187-214-27.us... |
| Slave4 | i-99709e95 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-187-156-49.us... |
| Slave5 | i-ae739da2 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-68-182-246.us... |
| Slave6 | i-98709e94 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-69-241-219.us... |
| Slave7 | i-ac739da0 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-124-168.us... |
| Slave8 | i-87709e8b | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-111-212.us... |
| Slave9 | i-b3739dbf | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-56-160.us... |
| Slave10 | i-b2739dbe | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-132-113.us... |
| Slave11 | i-86709e8a | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-127-242.us... |
| Slave12 | i-24719f28 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-112-159.us... |
| Slave13 | i-20719f2c | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-187-99-109.us... |
| Slave14 | i-fd719ff1 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-69-7-191.us-w... |
| Slave15 | i-454ea049 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-187-248-111.us... |
| Slave16 | i-85709e89 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-117-106.us... |

Instance: i-85709e89 (Slave16) Public DNS: ec2-54-191-117-106.us-west-2.compute.amazonaws.com

Description Status Checks Monitoring Tags

Feedback

EC2 Management Console - Google Chrome

EC2 Management C × Hadoop v2 overview × Hadoop 2.4-Instal... ×

https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:sort=launchTime

Apps Hadoop 2.4-Inst... Setting up Hadoop Hadoop Tutorial... Apache Hadoop Hadoop v2 over...

Services Edit

Launch Instance Connect Actions

Filter by tags and attributes or search by keyword

| Name | Instance ID | Instance Type | Availability Zone | Instance State | Status Checks | Alarm Status | Public DNS |
|---------|-------------|---------------|-------------------|----------------|----------------|--------------|--------------------------|
| Slave6 | i-98709e94 | c3.large | us-west-2a | running | Initializing | None | ec2-54-69-241-219.us... |
| Slave7 | i-ac739da0 | c3.large | us-west-2a | running | Initializing | None | ec2-54-191-124-168.us... |
| Slave8 | i-87709e8b | c3.large | us-west-2a | running | Initializing | None | ec2-54-191-111-212.us... |
| Slave9 | i-b3739dbf | c3.large | us-west-2a | running | Initializing | None | ec2-54-191-56-160.us... |
| Slave10 | i-b2739dbe | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-132-113.us... |
| Slave11 | i-86709e8a | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-127-242.us... |
| Slave12 | i-24719f28 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-112-159.us... |
| Slave13 | i-20719f2c | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-187-99-109.us... |
| Slave14 | i-fd719ff1 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-69-7-191.us-w... |
| Slave15 | i-454ea049 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-187-248-111.us... |
| Slave16 | i-85709e89 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-117-106.us... |

Instance: i-85709e89 (Slave16) Public DNS: ec2-54-191-117-106.us-west-2.compute.amazonaws.com

Description Status Checks Monitoring Tags

Feedback

Hadoop:

Installation:

Creating an instance in Amazon(AWS):

- Log in into the Amazon AWS service website (<http://aws.amazon.com/>).
- Select EC2 from the network and services section.
- Select launch instance. Then configure the service and chose the best AMI to meet the requirements.
- Then choose the instance type which can be micro, small, medium, large Xlarge, 2Xlarge, 4Xlarge or 8Xlarge. For our assignment it is c3.large
- Then we configure the instance i.e we can choose the instance to be either spot instance or on Demand instance.
- Add storage to our selected instance.
- Configure the security group. We add TCP, UDP and ICMP protocols from the security group. We then create the instance. We access the instance by connecting to the instance by suing the public IP address of it and later on we can configure the settings of the AMI that we are using.

To Connect to the Node:

chmod 400 keygiven(This is to change the permission to be read by owner.)

ssh -i key IP address

In our case we use:

3. chmod 400 dj_key.pem
4. ssh -i dj_key.pem Ubuntu@54.69.130.130

After the connection is established:

- After getting connected to the AMI we now have to install java on the virtual node.
- Hadoop Frame Work is written in java, so we need to install java on all the nodes.
- Install java by using sudo apt-get install openjdk-7-jreheadless.
- Install javac by using sudo apt-get install openjdk-7-jdk.
- Now update all the installed applications by using sudo apt-get update
- Now check the java version by using java –version.

To Install Hadoop:

- In the terminal do a wget <http://mirrors.sonic.net/apache/hadoop/common/hadoop-2.4.1/hadoop-2.4.1.tar.gz>
- Unzip it Using “tar xvzf hadoop-2.5.1.tar.gz”
- Create a folder installs in hadoop and then move the contents into this folder.

The following files will have to be modified to complete the Hadoop setup:

- `~/.bashrc`
- `/conf/hadoop-env.sh`
- `/conf/core-site.xml`
- `/conf/yarn-site.xml`
- `/conf/hdfs-site.xml`
- `/conf/mapred-site.xml.template`

1. `~/.bashrc`:

```
#HADOOP VARIABLES START
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
export HADOOP_INSTALL=/usr/ubuntu/install/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
eval $(ssh-agent)
ssh-add/home/ubuntu/dj_key.pem
#HADOOP VARIABLES END
```

Reboot the connection now for the changes to be reflected.

2. `/conf/hadoop-env.sh`:

This is to set the JAVA Path by modifying **hadoop-env.sh** file

```
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
```

3. / conf /core-site.xml:

The **/usr/local/hadoop/etc/hadoop/core-site.xml** file contains configuration properties that Hadoop uses when starting up. This file can be used to override the default settings that Hadoop starts with.

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://<172.31.22.90>:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/home/ubuntu/install/hadoop/hadoop_tmp/tmp</value>
</property>
</configuration>
```

4. / conf /yarn-site.xml:

```
<configuration>
<property>
<name>yarn.node.manager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.node.manager.aux-services.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
<name>yarn.resource.manager.resource-tracker.address</name>
<value><172.31.22.90>:8025</value>
</property>
<property>
<name>yarn.resource.manager.scheduler.address</name>
<value><172.31.22.90>:8030</value>
</property>
<property>
<name>yarn.resource.manager.address</name>
<value><172.31.22.90>:8040</value>
</property>
</configuration>
```

5. / conf /hdfs-site.xml:

```
<configuration>
<property>
<name>dfs.replication</name>
<value>16</value>
</property>
<property>
<name>dfs.permissions</name>
<value>false</value>
</property>
</configuration>
```

6. /conf /mapred-site.xml:

By default, the /usr/local/hadoop/etc/hadoop/ contains the /usr/local/hadoop/etc/hadoop/mapred-site.xml.template file which has to be renamed/copied with the name mapred-site.xml:

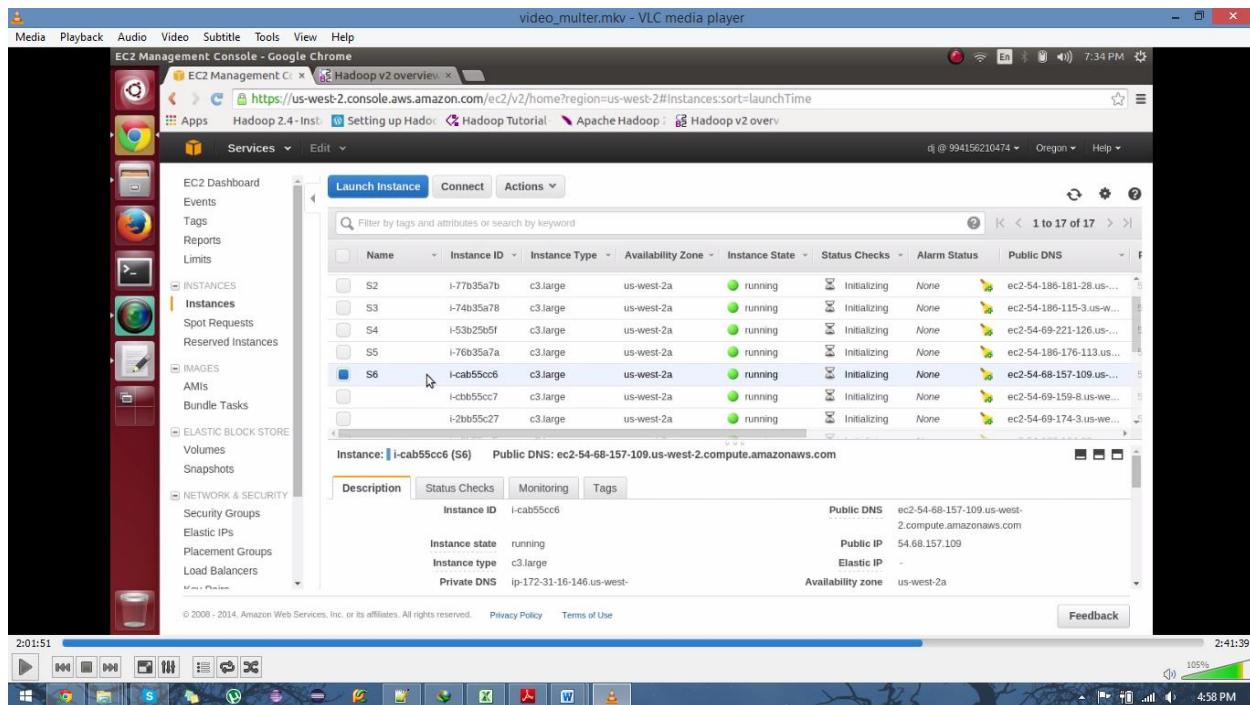
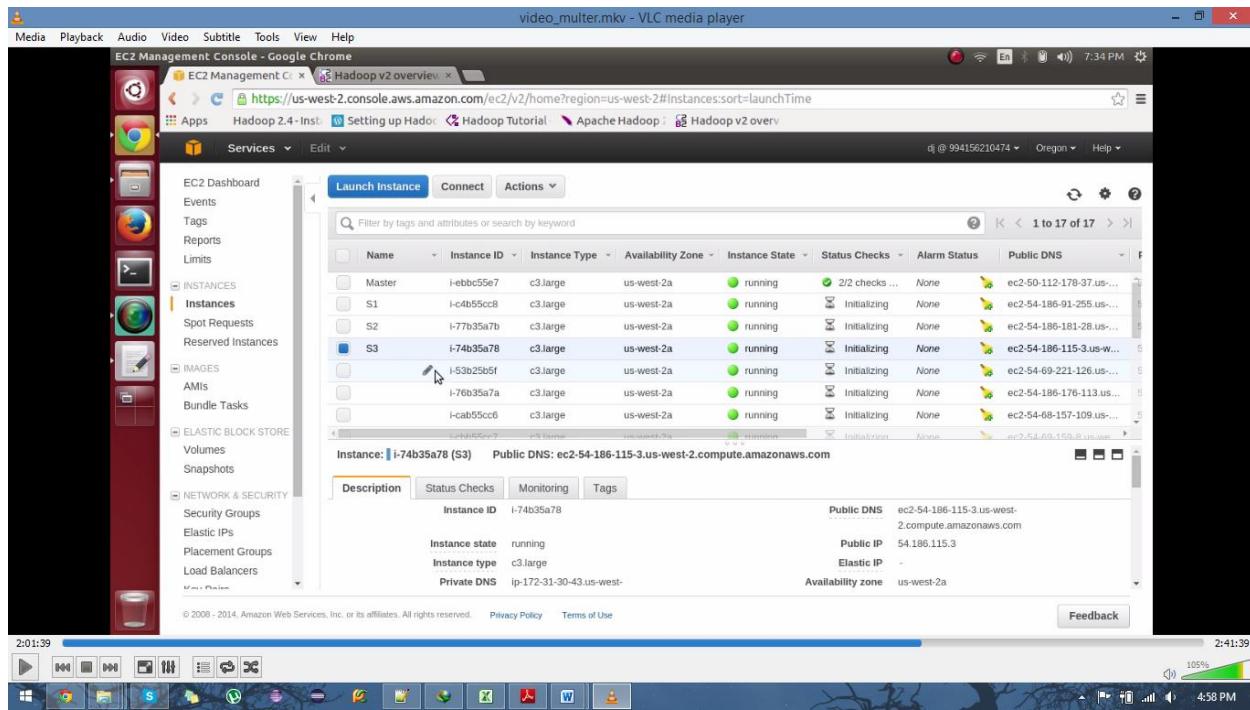
“cp /usr/local/hadoop/etc/hadoop/mapred-site.xml.template/usr/local/hadoop/etc/hadoop/mapred-site.xml”

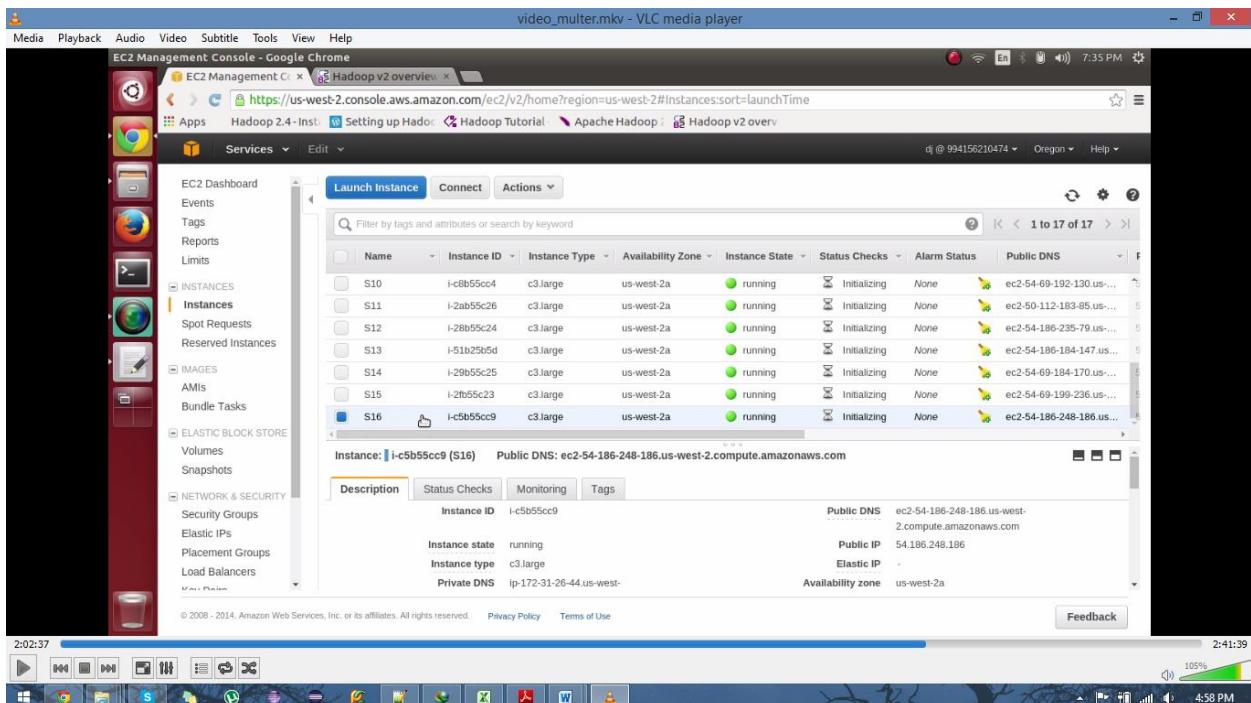
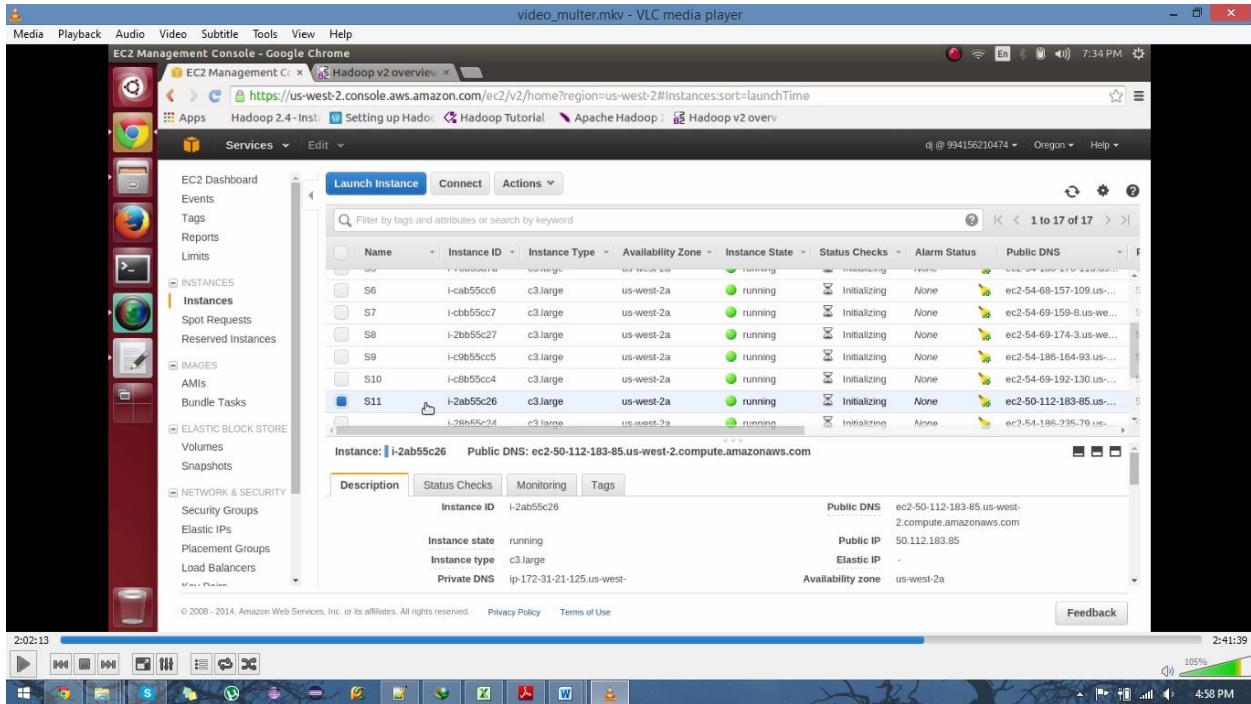
The **mapred-site.xml** file is used to specify which framework is being used for MapReduce. We need to enter the following content in between the <configuration></configuration> tag:

```
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>
```

Create an IMAGE for all this so that it can be reflected in all the nodes. This would be the basic configuration for hadoop for 1 node setup. For 16 node setup we have to edit 2 more configuration files the conf/hosts, conf/slaves on the **MASTER NODE ONLY**.

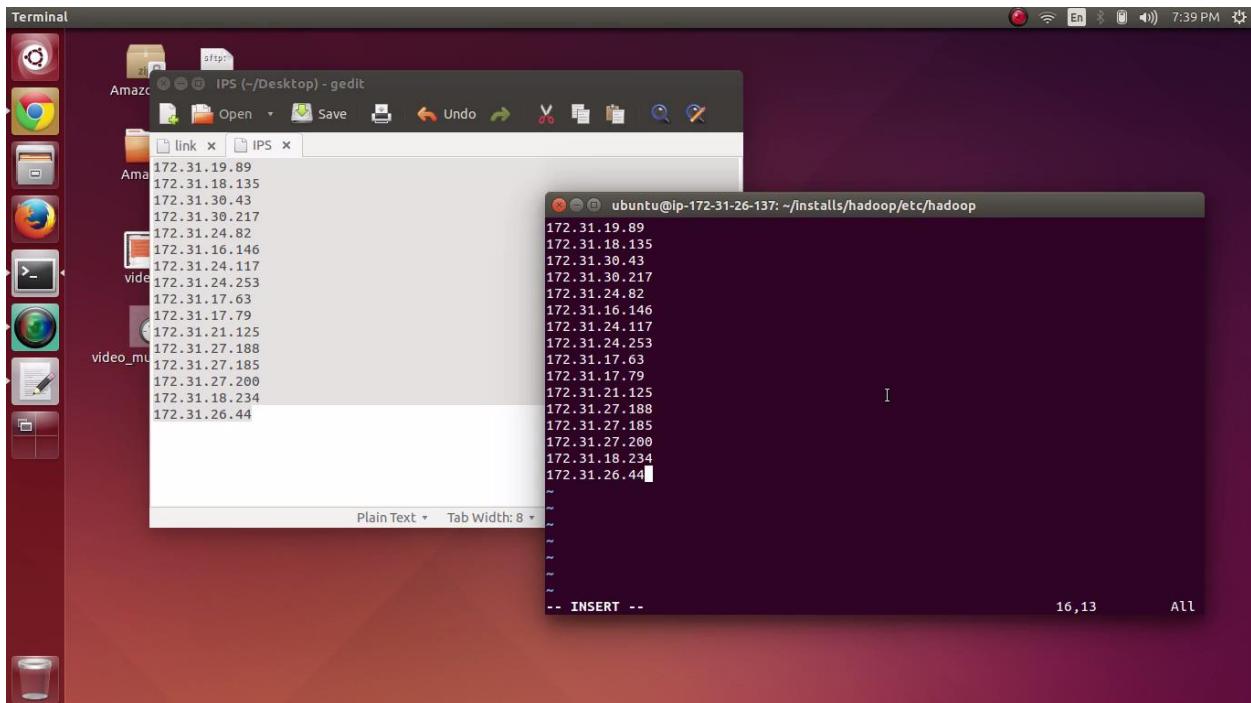
Master and Slave Nodes Connected:





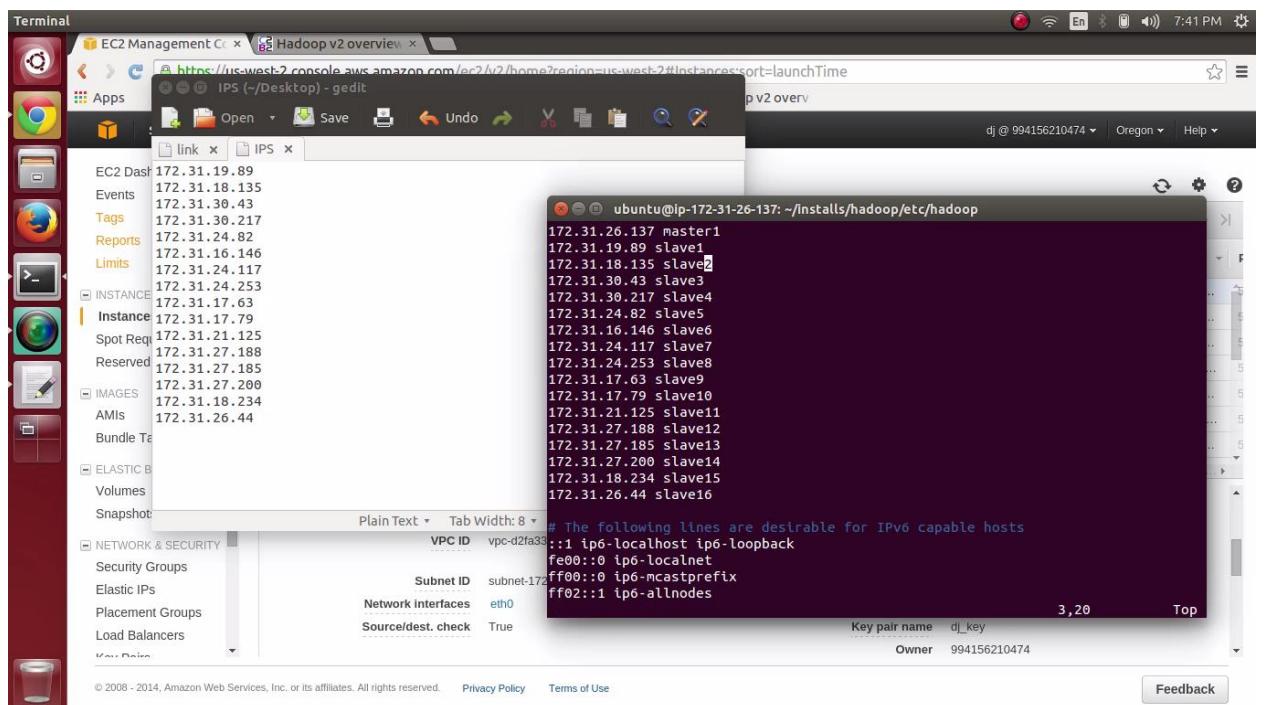
7. / conf /slaves:

Add all the IP address to the slaves file



8. conf/hosts:

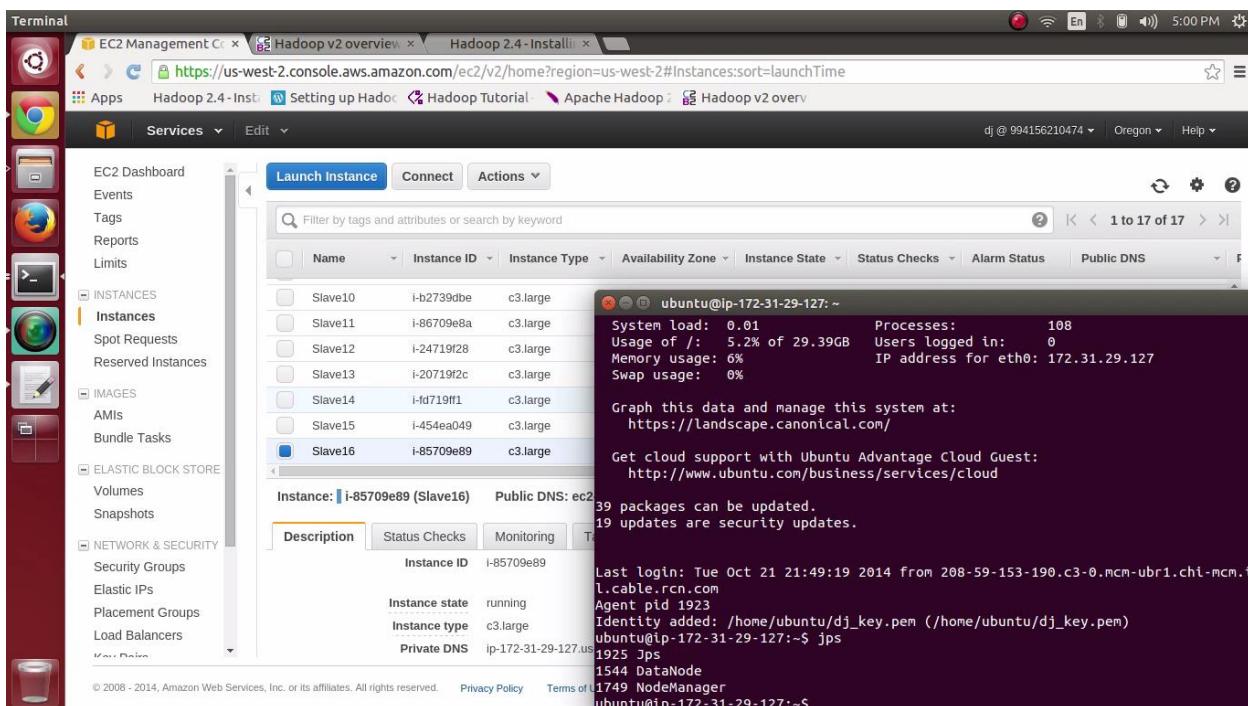
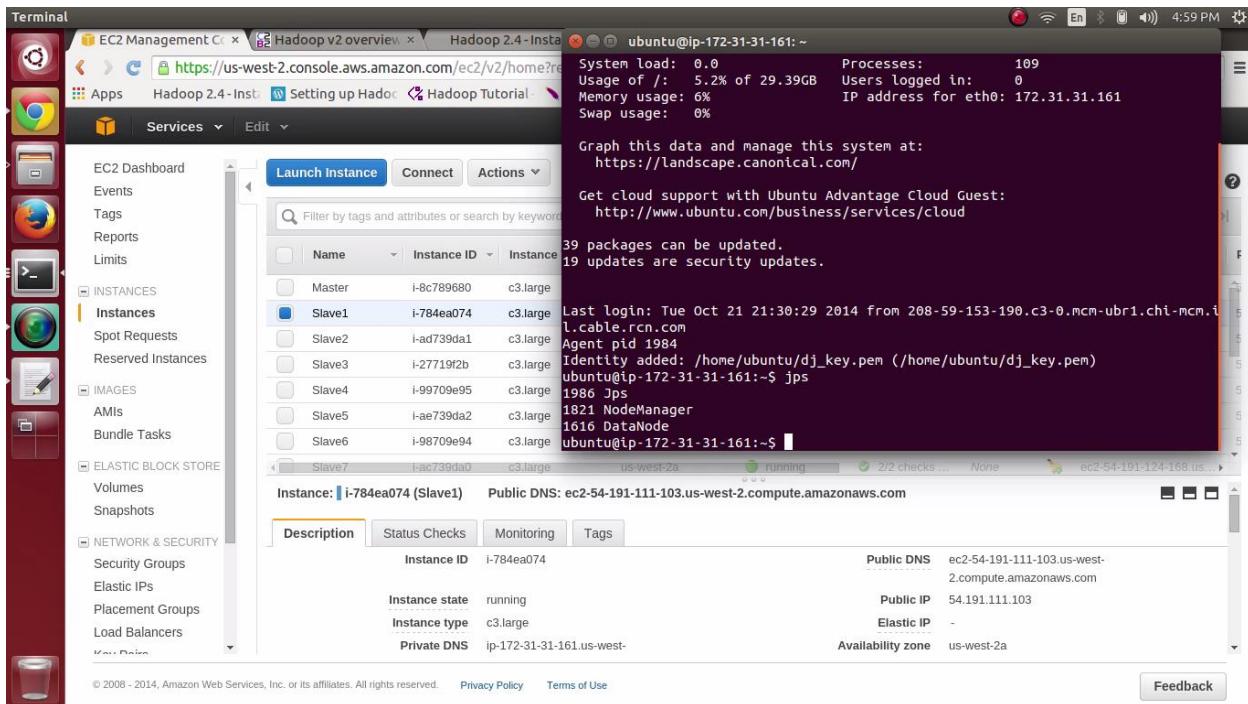
Add all the IP's of the the slaves in the file including the master IP.



1. The namenode will run only on the master in our test setup and is started by executing the following command:
 - `$<HADOOP_HOME>/sbin/hadoop-daemon.sh start namenode`
 2. Start the datanodes which will run on both the slave nodes:
 - `$<HADOOP_HOME>/hadoop-daemons.sh start datanode`
 3. The resource manager process will run only on the master and is started using the following command:
 - `$<HADOOP_HOME>/yarn-daemon.sh start resourcemanager`
 4. The node manager process runs on each of the slave nodes and is started using this command.:
 - `$<HADOOP_HOME>/yarn-daemons.sh start nodemanager`
 5. The job history server process runs on the master node and is started using the following command:
 - `$<HADOOP_HOME>/mr-jobhistory-daemon.sh start historyserver`

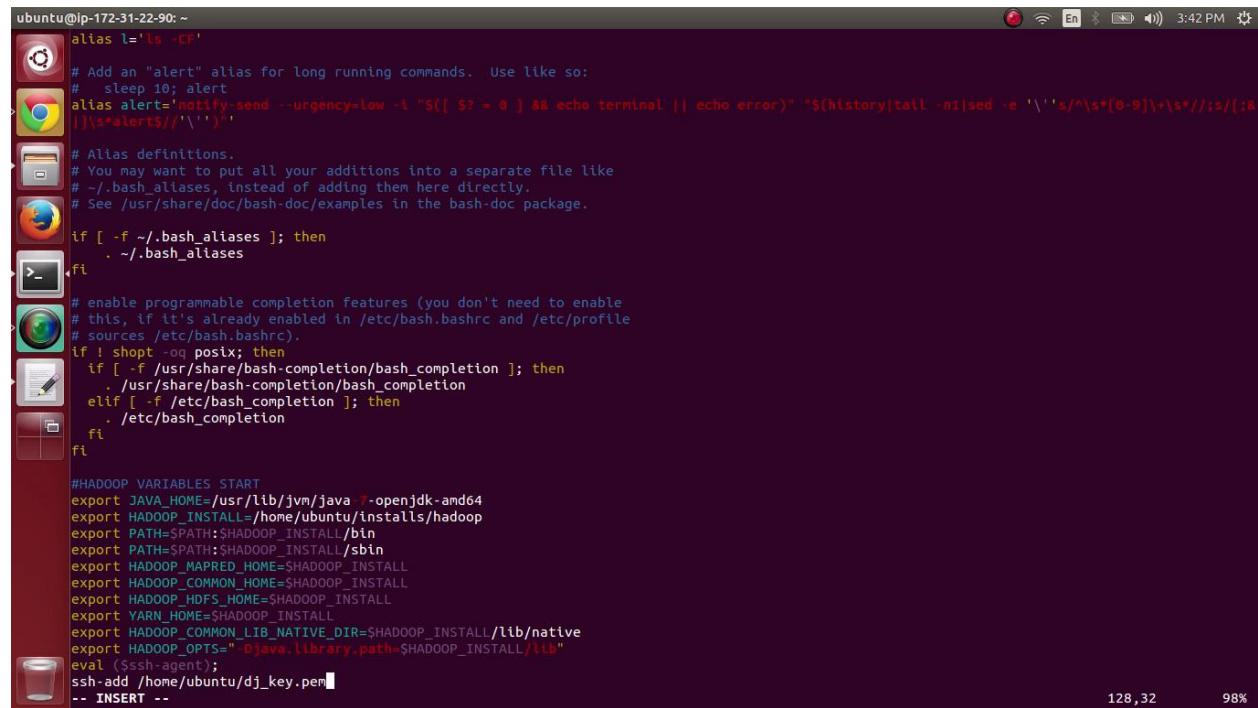
```
ubuntu@lp-172-31-22-90:~  
172.31.27.227: datanode running as process 1543. Stop it first.  
172.31.27.176: datanode running as process 1553. Stop it first.  
172.31.19.217: datanode running as process 1545. Stop it first.  
172.31.22.129: datanode running as process 1540. Stop it first.  
172.31.27.73: datanode running as process 1520. Stop it first.  
172.31.24.213: datanode running as process 1543. Stop it first.  
172.31.29.127: datanode running as process 1544. Stop it first.  
172.31.20.99: datanode running as process 1546. Stop it first.  
172.31.16.211: datanode running as process 1542. Stop it first.  
ubuntu@ip-172-31-22-90:~$ jps  
1644 NameNode  
1955 Jps  
ubuntu@ip-172-31-22-90:~$ yarn-daemons.sh start nodemanager  
172.31.24.59: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-24-59.out  
172.31.31.248: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-31-248.out  
172.31.22.129: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-22-129.out  
172.31.26.199: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-26-199.out  
172.31.24.213: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-24-213.out  
172.31.29.127: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-29-127.out  
172.31.25.232: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-25-232.out  
172.31.27.176: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-27-176.out  
172.31.25.127: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-25-127.out  
172.31.16.211: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-16-211.out  
172.31.27.73: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-27-73.out  
172.31.19.217: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-19-217.out  
172.31.18.210: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-18-210.out  
172.31.27.227: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-27-227.out  
172.31.20.99: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-20-99.out  
172.31.31.161: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-31-161.out  
ubuntu@ip-172-31-22-90:~$ jps  
1644 NameNode  
2028 Jps  
ubuntu@ip-172-31-22-90:~$ mr-jobhistory-daemon.sh start historyserver  
starting historyserver, logging to /home/ubuntu/install/hadoop/logs/mapred-ubuntu-historyserver-ip-172-31-22-90.out  
ubuntu@ip-172-31-22-90:~$ jps  
2099 Jps  
1644 NameNode  
2059 JobHistoryServer  
ubuntu@ip-172-31-22-90:~$ ls  
dj_key.pem hadoop-2.5.1.tar.gz installs  
ubuntu@ip-172-31-22-90:~$
```

To Check if the nodes are connected we login into another node and do jps:



Screen Shots of the Configuration FILES:

~/.bashrc:



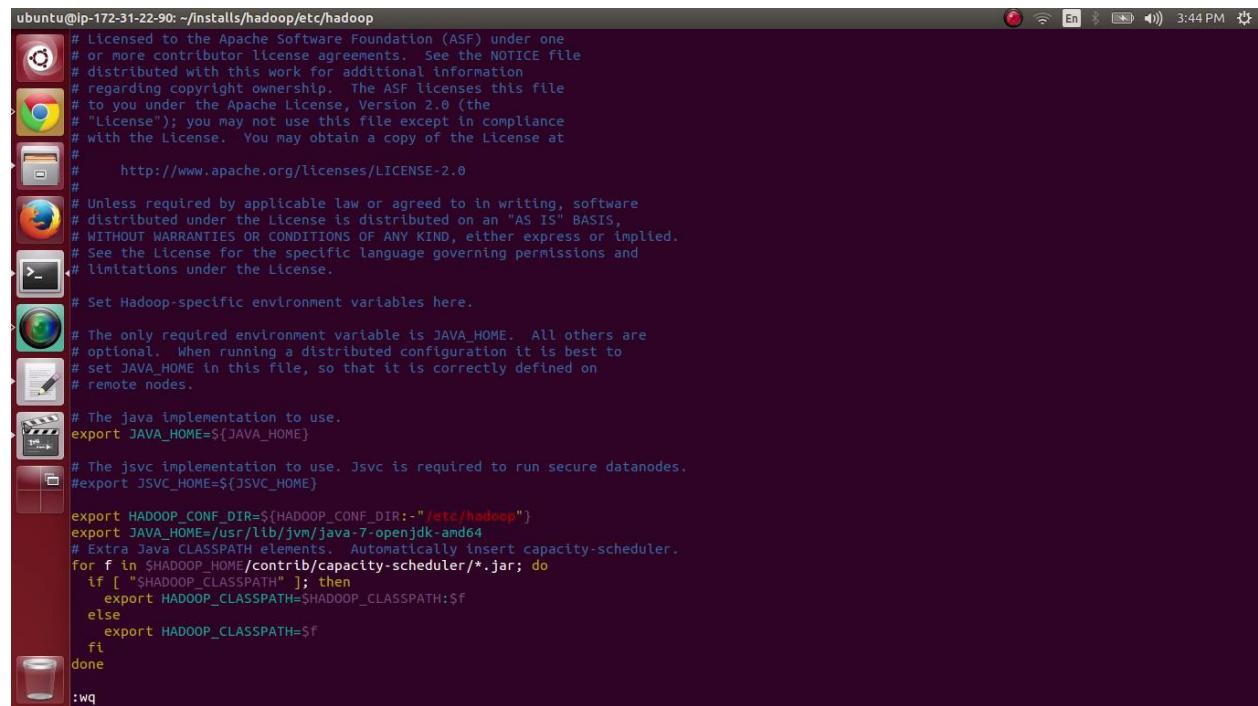
```
ubuntu@ip-172-31-22-90: ~
alias l='ls -CF'
# Add an "alert" alias for long running commands.  Use like so:
# sleep 10; alert
alias alert='notify-send --urgency=low -t "$([ $? = 0 ] && echo terminal || echo error)" "$(history|tail -n1|sed -e '\''$s/^\s*[0-9]\+\s*//;s/[[:space:]$/''')"'"
# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/._.bash_aliases ]; then
. ./_.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -q posix; then
if [ -f /usr/share/bash-completion/bash_completion ]; then
. /usr/share/bash-completion/bash_completion
elif [ -f /etc/bash_completion ]; then
. /etc/bash_completion
fi
fi

#HADOOP VARIABLES START
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
export HADOOP_INSTALL=/home/ubuntu/install/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
eval $(ssh-agent)
ssh-add /home/ubuntu/dj_key.pem
-- INSERT --
```

conf /hadoop-env.sh



```
ubuntu@ip-172-31-22-90: ~/install/hadoop/etc/hadoop
# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements. See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership. The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License. You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# Set Hadoop-specific environment variables here.

# The only required environment variable is JAVA_HOME. All others are
# optional. When running a distributed configuration it is best to
# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The java implementation to use.
export JAVA_HOME=${JAVA_HOME}

# The jsvc implementation to use. Jsvc is required to run secure datanodes.
#export JSVC_HOME=${JSVC_HOME}

export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:"/etc/hadoop"}
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
# Extra Java CLASSPATH elements. Automatically insert capacity-scheduler.
for f in $HADOOP_HOME/contrib/capacity-scheduler/*.jar; do
  if [ "$HADOOP_CLASSPATH" ]; then
    export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$f
  else
    export HADOOP_CLASSPATH=$f
  fi
done

:wq
```

/ conf /core-site.xml:

```
ubuntu@ip-172-31-22-90: ~/installs/hadoop/etc/hadoop
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href= "configuration.xsl"?>
<!--
 Licensed under the Apache License, Version 2.0 (the "License");
 you may not use this file except in compliance with the License.
 You may obtain a copy of the License at

 http://www.apache.org/licenses/LICENSE-2.0

 Unless required by applicable law or agreed to in writing, software
 distributed under the License is distributed on an "AS IS" BASIS,
 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 See the License for the specific language governing permissions and
 limitations under the License. See accompanying LICENSE file.
-->
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
  <name>fs.default.name</name>
  <value>hdfs://172.31.22.90:9000</value>
</property>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/home/ubuntu/installs/hadoop_tmp/tmp</value>
</property>
</configuration>
~
~
```

/ conf /yarn-site.xml:

```
ubuntu@lp-172-31-22-90:~/Installs/hadoop/etc/hadoop
<?xml version="1.0"?>
<!--
 Licensed under the Apache License, Version 2.0 (the "License");
 you may not use this file except in compliance with the License.
 You may obtain a copy of the License at

 http://www.apache.org/licenses/LICENSE-2.0

 Unless required by applicable law or agreed to in writing, software
 distributed under the License is distributed on an "AS IS" BASIS,
 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 See the License for the specific language governing permissions and
 limitations under the License. See accompanying LICENSE file.
-->
<configuration>
<property>
 <name>yarn.nodemanager.aux-services</name>
 <value>mapreduce_shuffle</value>
</property>
<property>
 <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
 <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
 <name>yarn.resourcemanager.resource-tracker.address</name>
 <value>172.31.22.90:8025</value>
</property>
<property>
 <name>yarn.resourcemanager.scheduler.address</name>
 <value>172.31.22.90:8030</value>
</property>
<property>
 <name>yarn.resourcemanager.address</name>
 <value>172.31.22.90:8040</value>
</property>
</configuration>
~
```

/ conf /hdfs-site.xml:

```
ubuntu@ip-172-31-22-90: ~/installs/hadoop/etc/hadoop
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href= "configuration.xsl"?>
<!--
 Licensed under the Apache License, Version 2.0 (the "License");
 you may not use this file except in compliance with the License.
 You may obtain a copy of the License at

 http://www.apache.org/licenses/LICENSE-2.0

 Unless required by applicable law or agreed to in writing, software
 distributed under the License is distributed on an "AS IS" BASIS,
 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 See the License for the specific language governing permissions and
 limitations under the License. See accompanying LICENSE file.
-->
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
    <name>dfs.replication</name>
    <value>16</value>
</property>
<property>
    <name>dfs.permissions</name>
    <value>false</value>
</property>
</configuration>
~
~
```

/conf /mapred-site.xml :

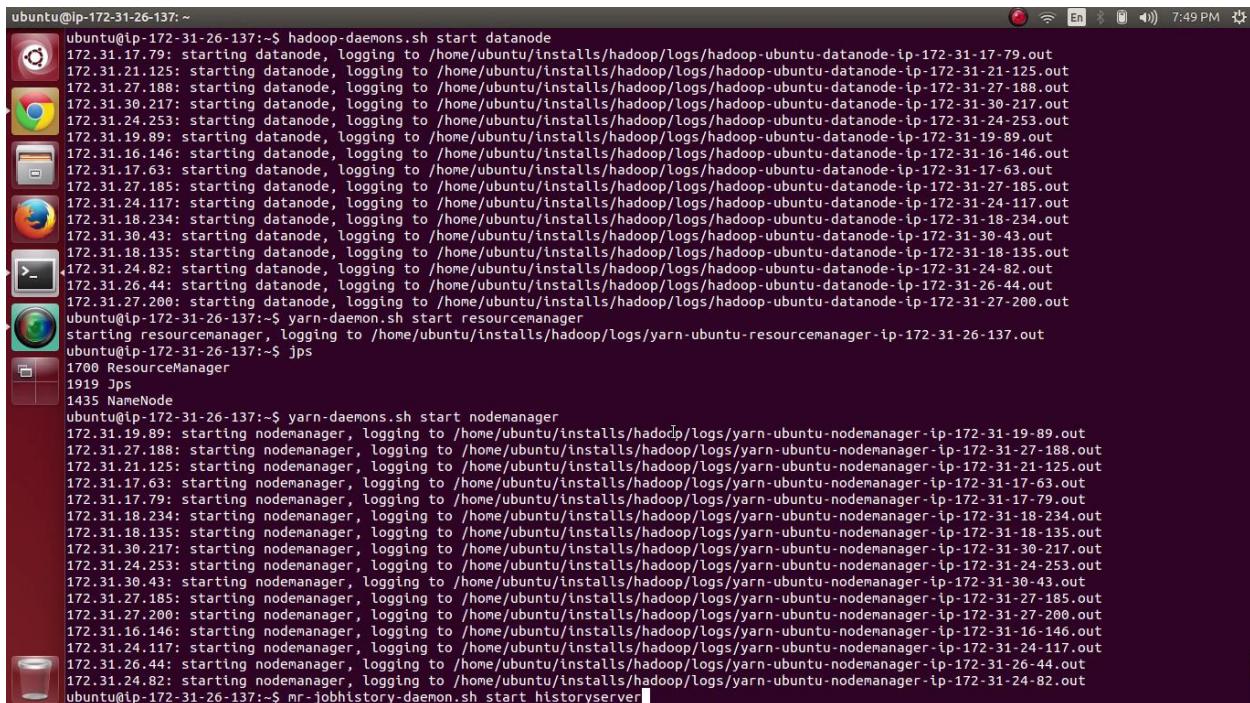
```
ubuntu@lp-172-31-22-90: ~/Installs/hadoop/etc/hadoop
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
    Licensed under the Apache License, Version 2.0 (the "License");
    you may not use this file except in compliance with the License.
    You may obtain a copy of the License at

        http://www.apache.org/licenses/LICENSE-2.0

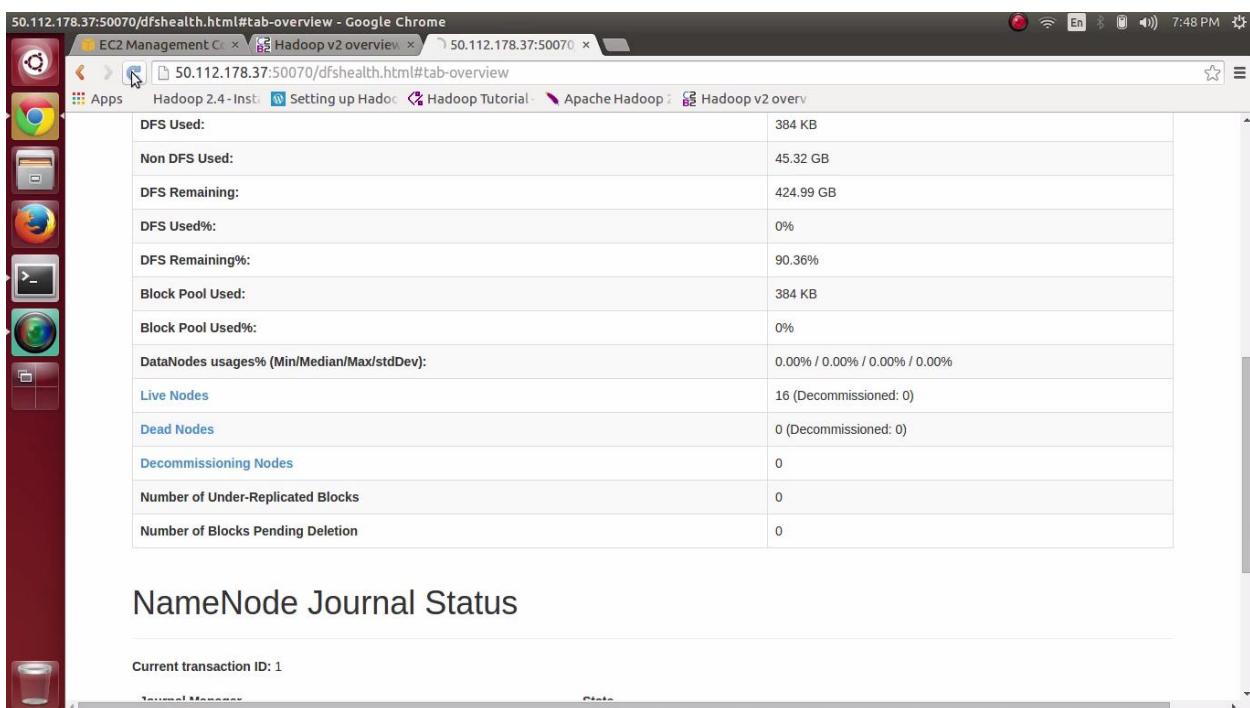
    Unless required by applicable law or agreed to in writing, software
    distributed under the License is distributed on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
    See the License for the specific language governing permissions and
    limitations under the License. See accompanying LICENSE file.
-->
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
</property>
</configuration>
~
```

Connecting to Data Nodes:

Screen Shots:



```
ubuntu@ip-172-31-26-137:~$ hadoop-daemons.sh start datanode
172.31.17.79: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-17-79.out
172.31.21.125: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-21-125.out
172.31.27.188: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-27-188.out
172.31.30.217: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-30-217.out
172.31.24.253: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-24-253.out
172.31.19.89: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-19-89.out
172.31.16.146: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-16-146.out
172.31.17.63: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-17-63.out
172.31.27.185: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-27-185.out
172.31.24.117: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-24-117.out
172.31.18.234: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-18-234.out
172.31.30.43: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-30-43.out
172.31.18.135: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-18-135.out
172.31.24.82: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-24-82.out
172.31.26.44: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-26-44.out
172.31.27.200: starting datanode, logging to /home/ubuntu/install/hadoop/logs/hadoop-ubuntu-datanode-ip-172-31-27-200.out
ubuntu@ip-172-31-26-137:~$ yarn-daemon.sh start resourcemanager
starting resourcemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-resourcemanager-ip-172-31-26-137.out
ubuntu@ip-172-31-26-137:~$ jps
1700 ResourceManager
1919 Jps
1435 NameNode
ubuntu@ip-172-31-26-137:~$ yarn-daemons.sh start nodemanager
172.31.19.89: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-19-89.out
172.31.27.188: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-27-188.out
172.31.21.125: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-21-125.out
172.31.17.63: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-17-63.out
172.31.17.79: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-17-79.out
172.31.18.234: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-18-234.out
172.31.18.135: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-18-135.out
172.31.30.217: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-30-217.out
172.31.24.253: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-24-253.out
172.31.30.43: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-30-43.out
172.31.27.185: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-27-185.out
172.31.27.200: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-27-200.out
172.31.16.146: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-16-146.out
172.31.24.117: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-24-117.out
172.31.26.44: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-26-44.out
172.31.24.82: starting nodemanager, logging to /home/ubuntu/install/hadoop/logs/yarn-ubuntu-nodemanager-ip-172-31-24-82.out
ubuntu@ip-172-31-26-137:~$ mr-jobhistory-daemon.sh start historyserver
```



50.112.178.37:50070/dfshealth.html#tab-overview - Google Chrome

EC2 Management C... Hadoop v2 overview 50.112.178.37:50070 x

Apps Hadoop 2.4-Inst. Setting up Hadoop Hadoop Tutorial Apache Hadoop Hadoop v2 over...

| | |
|--------------------------------------------|-------------------------------|
| DFS Used: | 384 KB |
| Non DFS Used: | 45.32 GB |
| DFS Remaining: | 424.99 GB |
| DFS Used%: | 0% |
| DFS Remaining%: | 90.36% |
| Block Pool Used: | 384 KB |
| Block Pool Used%: | 0% |
| DataNodes usages% (Min/Median/Max/stdDev): | 0.00% / 0.00% / 0.00% / 0.00% |
| Live Nodes | 16 (Decommissioned: 0) |
| Dead Nodes | 0 (Decommissioned: 0) |
| Decommissioning Nodes | 0 |
| Number of Under-Replicated Blocks | 0 |
| Number of Blocks Pending Deletion | 0 |

NameNode Journal Status

Current transaction ID: 1

Journal Manager

| Namenode Information - Google Chrome | | | | | | | | | | |
|-------------------------------------------------------------------|--------------|-------------|----------|-------|--------------|-----------|--------|-----------------|----------------|---------|
| 50.112.178.37:50070/dfshealth.html#tab-datanode | | | | | | | | | | |
| Node | Last contact | Admin State | Capacity | Used | Non DFS Used | Remaining | Blocks | Block pool used | Failed Volumes | Version |
| ip-172-31-27-185.us-west-2.compute.internal (172.31.27.185:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-17-63.us-west-2.compute.internal (172.31.17.63:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-27-200.us-west-2.compute.internal (172.31.27.200:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-18-135.us-west-2.compute.internal (172.31.18.135:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-30-217.us-west-2.compute.internal (172.31.30.217:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-27-188.us-west-2.compute.internal (172.31.27.188:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-26-44.us-west-2.compute.internal (172.31.26.44:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-18-234.us-west-2.compute.internal (172.31.18.234:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-30-43.us-west-2.compute.internal (172.31.30.43:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-21-125.us-west-2.compute.internal (172.31.21.125:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-17-79.us-west-2.compute.internal (172.31.17.79:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-24-253.us-west-2.compute.internal (172.31.24.253:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |

| Namenode information - Google Chrome | | | | | | | | | | |
|-------------------------------------------------------------------|--------------|-------------|----------|-------|--------------|-----------|--------|-----------------|----------------|---------|
| 50.112.178.37:50070/dfshealth.html#tab-datanode | | | | | | | | | | |
| Node | Last contact | Admin State | Capacity | Used | Non DFS Used | Remaining | Blocks | Block pool used | Failed Volumes | Version |
| ip-172-31-16-146.us-west-2.compute.internal (172.31.16.146:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-24-117.us-west-2.compute.internal (172.31.24.117:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-19-89.us-west-2.compute.internal (172.31.19.89:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-24-82.us-west-2.compute.internal (172.31.24.82:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-27-185.us-west-2.compute.internal (172.31.27.185:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-17-63.us-west-2.compute.internal (172.31.17.63:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-27-200.us-west-2.compute.internal (172.31.27.200:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-18-135.us-west-2.compute.internal (172.31.18.135:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-30-217.us-west-2.compute.internal (172.31.30.217:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-27-188.us-west-2.compute.internal (172.31.27.188:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-26-44.us-west-2.compute.internal (172.31.26.44:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |
| ip-172-31-18-234.us-west-2.compute.internal (172.31.18.234:50010) | 1 | In Service | 29.39 GB | 24 KB | 2.83 GB | 26.56 GB | 0 | 24 KB (0%) | 0 | 2.5.1 |

Namenode information - Google Chrome

EC2 Management Consoles Hadoop v2 overview Namenode information

50.112.178.37:50070/dfshealth.html#tab-datanode

Hadoop 2.4-Inst. Setting up Hadoop Hadoop Tutorial Apache Hadoop Hadoop v2 overv

| Node | Last contact | Under replicated blocks | Blocks with no live replicas | Under Replicated Blocks In files under construction |
|-------------------------------------------------------------------|--------------|-------------------------|---------------------------------------------|--------------------------------------------------------|
| ip-172-31-30-217.us-west-2.compute.internal (172.31.30.217:50010) | 1 | In Service 29.39 GB | 24 KB 2.83 GB 26.56 GB 0 24 KB (0%) 0 2.5.1 | |
| ip-172-31-27-188.us-west-2.compute.internal (172.31.27.188:50010) | 1 | In Service 29.39 GB | 24 KB 2.83 GB 26.56 GB 0 24 KB (0%) 0 2.5.1 | |
| ip-172-31-26-44.us-west-2.compute.internal (172.31.26.44:50010) | 1 | In Service 29.39 GB | 24 KB 2.83 GB 26.56 GB 0 24 KB (0%) 0 2.5.1 | |
| ip-172-31-18-234.us-west-2.compute.internal (172.31.18.234:50010) | 1 | In Service 29.39 GB | 24 KB 2.83 GB 26.56 GB 0 24 KB (0%) 0 2.5.1 | |
| ip-172-31-30-43.us-west-2.compute.internal (172.31.30.43:50010) | 1 | In Service 29.39 GB | 24 KB 2.83 GB 26.56 GB 0 24 KB (0%) 0 2.5.1 | |
| ip-172-31-21-125.us-west-2.compute.internal (172.31.21.125:50010) | 1 | In Service 29.39 GB | 24 KB 2.83 GB 26.56 GB 0 24 KB (0%) 0 2.5.1 | |
| ip-172-31-17-79.us-west-2.compute.internal (172.31.17.79:50010) | 1 | In Service 29.39 GB | 24 KB 2.83 GB 26.56 GB 0 24 KB (0%) 0 2.5.1 | |
| ip-172-31-24-253.us-west-2.compute.internal (172.31.24.253:50010) | 1 | In Service 29.39 GB | 24 KB 2.83 GB 26.56 GB 0 24 KB (0%) 0 2.5.1 | |

Decommissioning

| Node | Last contact | Under replicated blocks | Blocks with no live replicas | Under Replicated Blocks In files under construction |
|---------------|--------------|-------------------------|------------------------------|--------------------------------------------------------|
| Hadoop, 2014. | | | | |

Legacy UI

Word Count in Hadoop:

SYSTEM CONFIGURATION:

| | |
|---------------|--------------------|
| VERSION | : Hadoop 1.2.1 |
| INSTANCE TYPE | : c3. Large UBUNTU |
| RAM | : 3.75 GB |
| NO. OF CORES | : 2 virtual cores |
| STORAGE | : 32 GB |

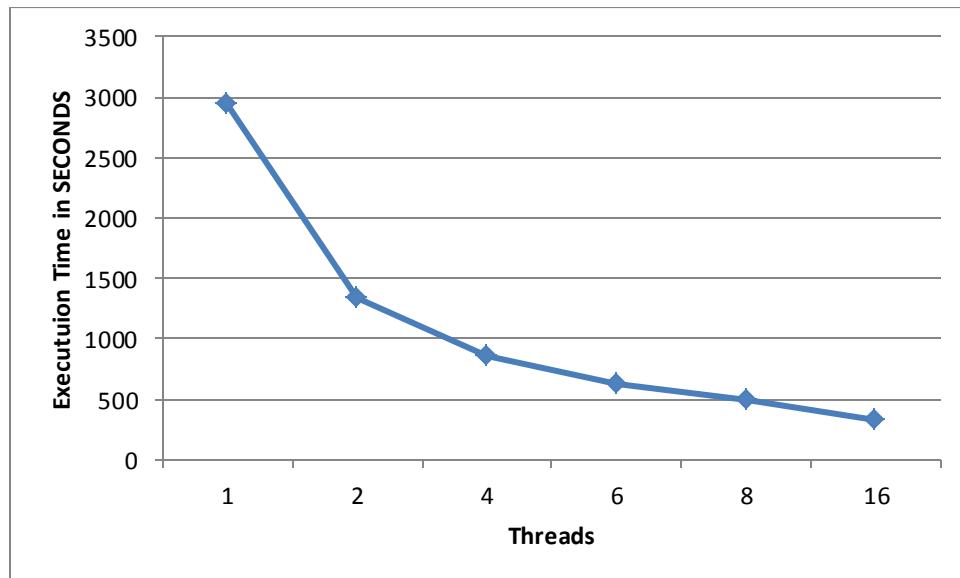
Methodology:

1. HADOOP framework is written in java.
2. Install java latest version (1.7) and check version by using java –version command.
3. Next is to create and to setup SSH certificates and install HADOOP.
4. Hadoop WordCount Map Reduce: MapReduce programs are designed to compute large volumes of data in a parallel fashion i.e. 10GB input. This requires dividing the workload across a two or single nodes.
5. List Processing: MapReduce programs transform lists of input data elements into lists of output data elements. A MapReduce program will do this twice, using two different list processing idioms: map, and reduce.
6. Mapping List: In the first phase called mapping, a list of data elements are provided, one at a time, to a function called the Mapper, which transforms each element individually to an output data element. As an example of the utility of map: We have a code to remove all the special characters (symbols) form the beginning and end of the word. We are not modifying the input string here: we are returning a new string that will form part of a new output list.
7. Reducing List: Reducing aggregate values together. A reducer function receives an iterator of input values from an input list. It then combines these values together, returning a single output value.
8. Now set up the configuration files and format the HADOOP file system and start HADOOP.
9. We have to access the output from the HDFS. This can be done using get command.
10. The output files stored are titled as wordcount-HADOOP.txt

Outputs:

10 GB DATASET ON c3.large instance (Only Word Count with Frequency no sorting):

| Nodes | Execution time in seconds |
|-------|---------------------------|
| 1 | 2950 |
| 2 | 1338 |
| 4 | 850.8 |
| 6 | 625.8 |
| 8 | 498 |
| 16 | 327.6 |

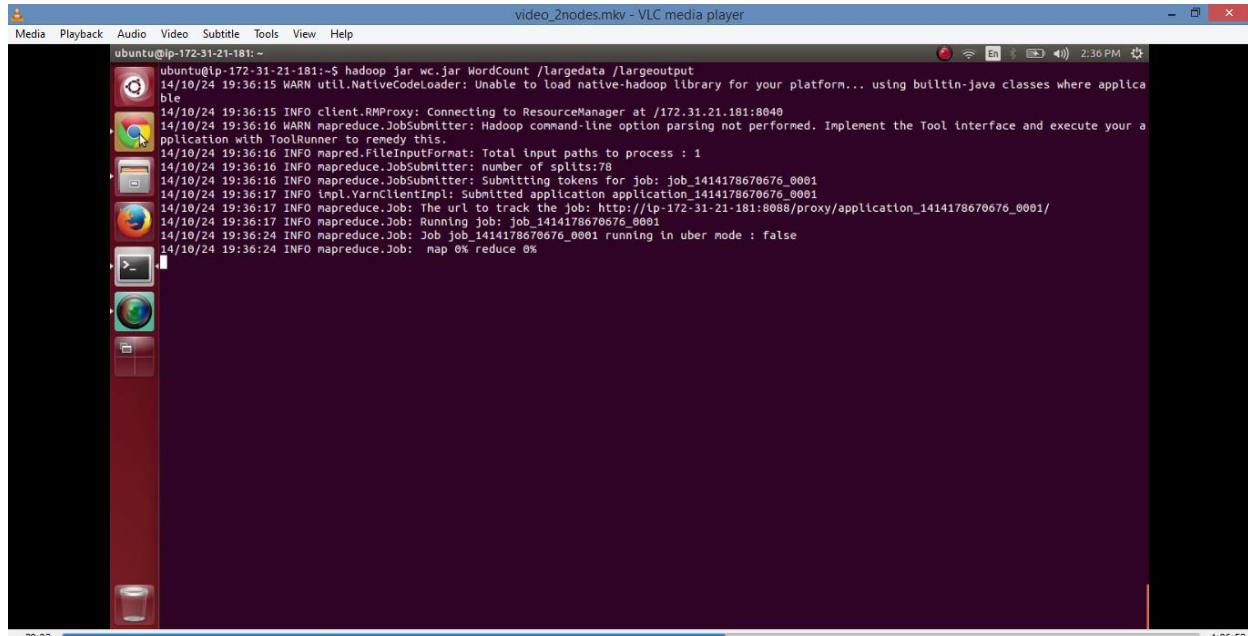


EXPLANATION:

1. The above graph clearly shows that as the number of nodes increases, the execution time decreases.

Screen Shots for setting up and running 2,4,6,8 and 16 nodes:

2 Nodes:



ubuntu@ip-172-31-21-181:~\$ hadoop jar wc.jar WordCount /largedata /largeoutput

14/10/24 19:36:15 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

14/10/24 19:36:15 INFO client.RMProxy: Connecting to ResourceManager at /172.31.21.181:8040

14/10/24 19:36:16 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.

14/10/24 19:36:16 INFO mapred.FileInputFormat: Total input paths to process : 1

14/10/24 19:36:16 INFO mapreduce.JobSubmitter: number of splits:8

14/10/24 19:36:16 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1414178670676_0001

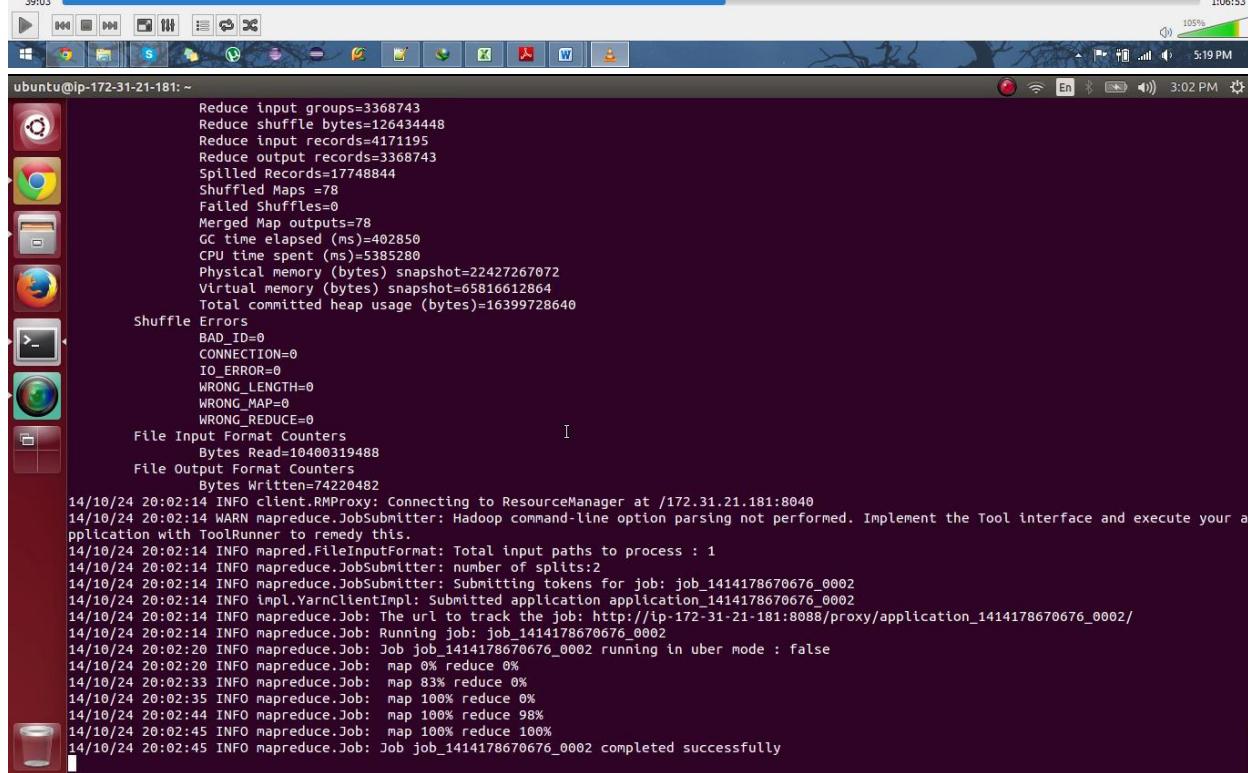
14/10/24 19:36:17 INFO yarnclientimpl.Submitted application application_1414178670676_0001

14/10/24 19:36:17 INFO mapreduce.Job: The url to track the job: http://ip-172-31-21-181:8088/proxy/application_1414178670676_0001/

14/10/24 19:36:24 INFO mapreduce.Job: Running job: job_1414178670676_0001

14/10/24 19:36:24 INFO mapreduce.Job: Job job_1414178670676_0001 running in uber mode : false

14/10/24 19:36:24 INFO mapreduce.Job: map 0% reduce 0%

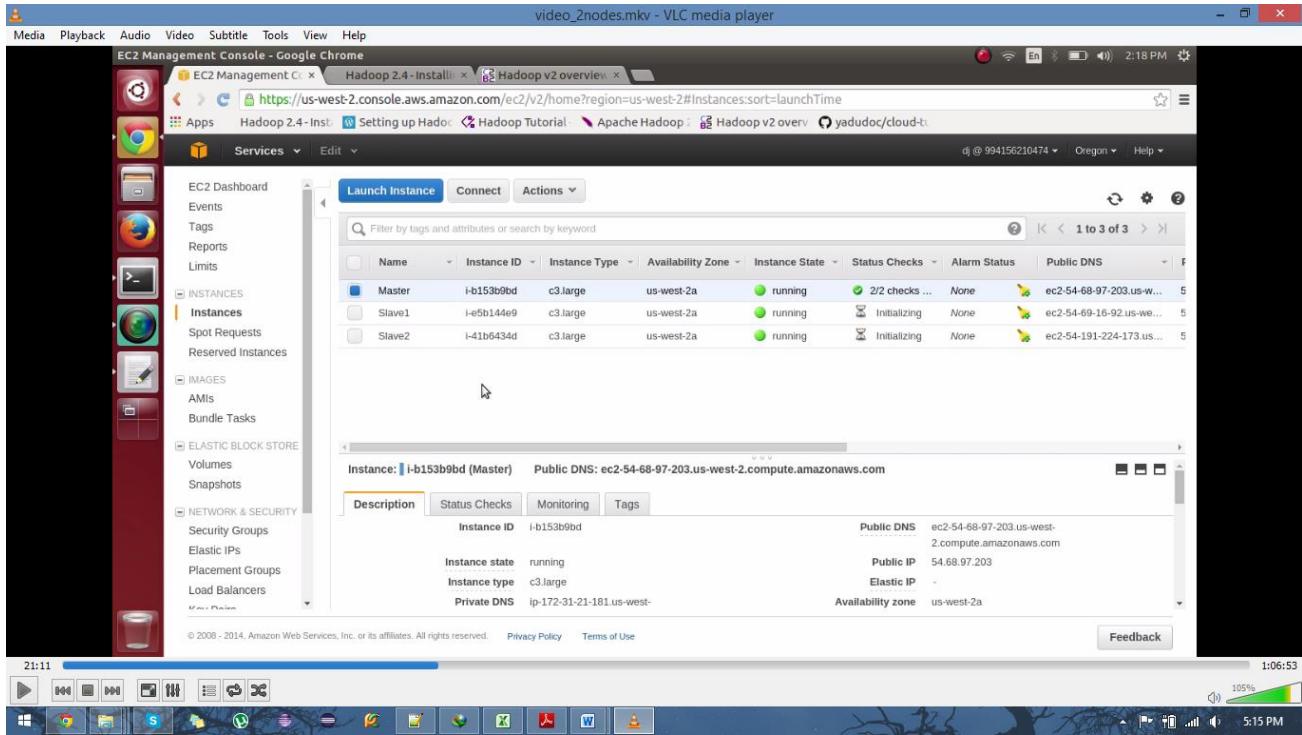


```
Reduce input groups=3368743
Reduce shuffle bytes=126434448
Reduce input records=4171195
Reduce output records=3368743
Spilled Records=17748844
Shuffled Maps =78
Failed Shuffles=0
Merged Map outputs=78
GC time elapsed (ms)=402850
CPU time spent (ms)=5385280
Physical memory (bytes) snapshot=22427267072
Virtual memory (bytes) snapshot=65816612864
Total committed heap usage (bytes)=16399728640

Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0

File Input Format Counters
Bytes Read=10400319488
File Output Format Counters
Bytes Written=74220482

14/10/24 20:02:14 INFO client.RMProxy: Connecting to ResourceManager at /172.31.21.181:8040
14/10/24 20:02:14 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
14/10/24 20:02:14 INFO mapred.FileInputFormat: Total input paths to process : 1
14/10/24 20:02:14 INFO mapreduce.JobSubmitter: number of splits:2
14/10/24 20:02:14 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1414178670676_0002
14/10/24 20:02:14 INFO yarnclientimpl.Submitted application application_1414178670676_0002
14/10/24 20:02:14 INFO mapreduce.Job: The url to track the job: http://ip-172-31-21-181:8088/proxy/application_1414178670676_0002/
14/10/24 20:02:14 INFO mapreduce.Job: Running job: job_1414178670676_0002
14/10/24 20:02:20 INFO mapreduce.Job: Job job_1414178670676_0002 running in uber mode : false
14/10/24 20:02:20 INFO mapreduce.Job: map 0% reduce 0%
14/10/24 20:02:33 INFO mapreduce.Job: map 83% reduce 0%
14/10/24 20:02:35 INFO mapreduce.Job: map 100% reduce 0%
14/10/24 20:02:44 INFO mapreduce.Job: map 100% reduce 98%
14/10/24 20:02:45 INFO mapreduce.Job: map 100% reduce 100%
14/10/24 20:02:45 INFO mapreduce.Job: Job job_1414178670676_0002 completed successfully
```



4 Nodes:

```
ubuntu@ip-172-31-21-181: ~
14/10/24 21:18:11 INFO mapreduce.Job: map 94% reduce 29%
14/10/24 21:18:27 INFO mapreduce.Job: map 95% reduce 29%
14/10/24 21:18:31 INFO mapreduce.Job: map 95% reduce 30%
14/10/24 21:18:32 INFO mapreduce.Job: map 96% reduce 30%
14/10/24 21:18:37 INFO mapreduce.Job: map 96% reduce 31%
14/10/24 21:18:39 INFO mapreduce.Job: map 97% reduce 31%
14/10/24 21:18:56 INFO mapreduce.Job: map 98% reduce 31%
14/10/24 21:19:01 INFO mapreduce.Job: map 99% reduce 32%
14/10/24 21:19:13 INFO mapreduce.Job: map 100% reduce 32%
14/10/24 21:19:14 INFO mapreduce.Job: map 100% reduce 33%
14/10/24 21:19:20 INFO mapreduce.Job: map 100% reduce 38%
14/10/24 21:19:23 INFO mapreduce.Job: map 100% reduce 74%
14/10/24 21:19:26 INFO mapreduce.Job: map 100% reduce 95%
14/10/24 21:19:27 INFO mapreduce.Job: map 100% reduce 100%
14/10/24 21:19:27 INFO mapreduce.Job: Job job_1414184420914_0001 completed successfully
14/10/24 21:19:27 INFO mapreduce.Job: Counters:
  File System Counters
    FILE: Number of bytes read=259955041
    FILE: Number of bytes written=394070514
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=10400320274
    HDFS: Number of bytes written=74220482
    HDFS: Number of read operations=237
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=2
  Job Counters
    Killed map tasks=6
    Launched map tasks=84
    Launched reduce tasks=1
    Data-local map tasks=84
    Total time spent by all maps in occupied slots (ms)=21742917
    Total time spent by all reduces in occupied slots (ms)=599279
    Total time spent by all map tasks (ms)=21742917
    Total time spent by all reduce tasks (ms)=599279
    Total vcore-seconds taken by all map tasks=21742917
    Total vcore-seconds taken by all reduce tasks=599279
    Total megabyte-seconds taken by all map tasks=22264747008
    Total megabyte-seconds taken by all reduce tasks=613661696
```

```

ubuntu@ip-172-31-21-181: ~
Error: No command named '--fs' was found. Perhaps you meant 'hadoop fs'
ubuntu@ip-172-31-21-181:~$ hadoop fs -copyFromLocal wiki10gb /largedata
14/10/24 21:01:15 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
ubuntu@ip-172-31-21-181:~$ hadoop jar wc.jar WordCount /largedata /largeoutput
14/10/24 21:05:09 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/10/24 21:05:10 INFO client.RMProxy: Connecting to ResourceManager at /172.31.21.181:8040
14/10/24 21:05:10 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
14/10/24 21:05:10 INFO mapred.FileInputFormat: Total input paths to process : 1
14/10/24 21:05:10 INFO mapreduce.JobSubmitter: number of splits:78
14/10/24 21:05:11 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1414184420914_0001
14/10/24 21:05:11 INFO impl.YarnClientImpl: Submitted application application_1414184420914_0001
14/10/24 21:05:11 INFO mapreduce.Job: The url to track the job: http://ip-172-31-21-181:8088/proxy/application_1414184420914_0001/
14/10/24 21:05:11 INFO mapreduce.Job: Running job: job_1414184420914_0001
14/10/24 21:05:19 INFO mapreduce.Job: Job job_1414184420914_0001 running in uber mode : false
14/10/24 21:05:19 INFO mapreduce.Job: map 0% reduce 0%
14/10/24 21:05:56 INFO mapreduce.Job: map 1% reduce 0%
14/10/24 21:06:03 INFO mapreduce.Job: map 2% reduce 0%
14/10/24 21:06:10 INFO mapreduce.Job: map 3% reduce 0%
14/10/24 21:06:16 INFO mapreduce.Job: map 4% reduce 0%
14/10/24 21:06:26 INFO mapreduce.Job: map 5% reduce 0%
14/10/24 21:06:39 INFO mapreduce.Job: map 6% reduce 0%
14/10/24 21:06:57 INFO mapreduce.Job: map 7% reduce 0%
14/10/24 21:07:04 INFO mapreduce.Job: map 8% reduce 0%
14/10/24 21:07:10 INFO mapreduce.Job: map 9% reduce 0%
14/10/24 21:07:19 INFO mapreduce.Job: map 10% reduce 0%
14/10/24 21:07:41 INFO mapreduce.Job: map 11% reduce 0%
14/10/24 21:07:48 INFO mapreduce.Job: map 12% reduce 0%
14/10/24 21:07:54 INFO mapreduce.Job: map 13% reduce 0%
14/10/24 21:08:01 INFO mapreduce.Job: map 14% reduce 0%
14/10/24 21:08:18 INFO mapreduce.Job: map 15% reduce 0%
14/10/24 21:08:30 INFO mapreduce.Job: map 16% reduce 0%
14/10/24 21:08:37 INFO mapreduce.Job: map 17% reduce 0%
14/10/24 21:08:46 INFO mapreduce.Job: map 18% reduce 0%
14/10/24 21:08:51 INFO mapreduce.Job: map 19% reduce 0%
14/10/24 21:08:58 INFO mapreduce.Job: map 20% reduce 0%
14/10/24 21:09:08 INFO mapreduce.Job: map 21% reduce 0%
14/10/24 21:09:21 INFO mapreduce.Job: map 22% reduce 0%

```

Namenode information - Google Chrome

EC2 Management Hadoop 2.4-Inst... Hadoop v2 over... Namenode info... Nodes of the cl... JobHistory namenode not... Hadoop - name...

54.68.97.203:50070/dfshealth.html#tab-datanode

Apps Hadoop 2.4-Inst... Setting up Hado... Hadoop Tutorial... Apache Hadoop Hadoop v2 over... yadudoc/cloud-t...

In operation

| Node | Last contact | Admin State | Capacity | Used | Non DFS Used | Remaining | Blocks | Block pool used | Failed Volumes | Version |
|----------------------------------------------------------------------|--------------|-------------|----------|---------|--------------|-----------|--------|------------------|----------------|---------|
| ip-172-31-201.us-west-2.compute.internal (172.31.30.201:50010) | 2 | In Service | 19.55 GB | 9.76 GB | 2.43 GB | 7.35 GB | 78 | 9.76 GB (49.94%) | 0 | 2.5.1 |
| ip-172-31-30-190.us-west-2.compute.internal (172.31.30.190:50010) | 2 | In Service | 19.55 GB | 9.76 GB | 2.44 GB | 7.35 GB | 78 | 9.76 GB (49.94%) | 0 | 2.5.1 |
| ip-172-31-26-95.us-west-2.compute.internal (172.31.26.95:50010) | 0 | In Service | 19.55 GB | 9.76 GB | 2.44 GB | 7.35 GB | 78 | 9.76 GB (49.94%) | 0 | 2.5.1 |
| ip-172-31-16-20.us-west-2.compute.internal (172.31.16.20:50010) | 0 | In Service | 19.55 GB | 9.76 GB | 2.43 GB | 7.35 GB | 78 | 9.76 GB (49.94%) | 0 | 2.5.1 |

Decommissioning

| Node | Last contact | Under replicated blocks | Blocks with no live replicas | Under Replicated Blocks In files under construction |
|------|--------------|-------------------------|------------------------------|--------------------------------------------------------|
|------|--------------|-------------------------|------------------------------|--------------------------------------------------------|

Hadoop, 2014.

Legacy UI

6 nodes:

```
ubuntu@ip-172-31-21-181:~$ hadoop fs -copyFromLocal wiki10gb /largedata
14/10/24 21:32:03 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
ubuntu@ip-172-31-21-181:~$ hadoop jar wc.jar WordCount /largedata /largeoutput
14/10/24 21:35:46 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/10/24 21:35:46 INFO client.RMProxy: Connecting to ResourceManager at /172.31.21.181:8040
14/10/24 21:35:47 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
14/10/24 21:35:47 INFO mapred.FileInputFormat: Total input paths to process : 1
14/10/24 21:35:47 INFO mapreduce.JobSubmitter: number of splits:78
14/10/24 21:35:48 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1414186288251_0001
14/10/24 21:35:48 INFO impl.YarnClientImpl: Submitted application application_1414186288251_0001
14/10/24 21:35:48 INFO mapreduce.Job: The url to track the job: http://ip-172-31-21-181:8088/proxy/application_1414186288251_0001/
14/10/24 21:35:48 INFO mapreduce.Job: Running job: job_1414186288251_0001
14/10/24 21:35:55 INFO mapreduce.Job: Job job_1414186288251_0001 running in uber mode : false
14/10/24 21:35:55 INFO mapreduce.Job: map 0% reduce 0%
14/10/24 21:36:34 INFO mapreduce.Job: map 1% reduce 0%
14/10/24 21:36:44 INFO mapreduce.Job: map 3% reduce 0%
14/10/24 21:36:48 INFO mapreduce.Job: map 4% reduce 0%
14/10/24 21:36:52 INFO mapreduce.Job: map 5% reduce 0%
14/10/24 21:36:57 INFO mapreduce.Job: map 6% reduce 0%
14/10/24 21:37:02 INFO mapreduce.Job: map 7% reduce 0%
14/10/24 21:37:10 INFO mapreduce.Job: map 8% reduce 0%
14/10/24 21:37:24 INFO mapreduce.Job: map 9% reduce 0%
14/10/24 21:37:35 INFO mapreduce.Job: map 10% reduce 0%
14/10/24 21:37:40 INFO mapreduce.Job: map 11% reduce 0%
14/10/24 21:37:44 INFO mapreduce.Job: map 12% reduce 0%
14/10/24 21:37:49 INFO mapreduce.Job: map 13% reduce 0%
14/10/24 21:37:53 INFO mapreduce.Job: map 14% reduce 0%
14/10/24 21:38:00 INFO mapreduce.Job: map 15% reduce 0%
14/10/24 21:38:18 INFO mapreduce.Job: map 16% reduce 0%
14/10/24 21:38:23 INFO mapreduce.Job: map 17% reduce 0%
14/10/24 21:38:28 INFO mapreduce.Job: map 18% reduce 0%
14/10/24 21:38:32 INFO mapreduce.Job: map 19% reduce 0%
14/10/24 21:38:37 INFO mapreduce.Job: map 20% reduce 0%
14/10/24 21:38:44 INFO mapreduce.Job: map 21% reduce 0%
14/10/24 21:38:53 INFO mapreduce.Job: map 22% reduce 0%
```

EC2 Management Console - Google Chrome

EC2 Management Console - Google Chrome https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:sort=launchTime

EC2 Dashboard Services Edit

1 to 7 of 7

| Name | Instance ID | Instance Type | Availability Zone | Instance State | Status Checks | Alarm Status | Public DNS | F |
|--------|-------------|---------------|-------------------|----------------|------------------|--------------|--------------------------|---|
| Master | i-b153b9bd | c3.large | us-west-2a | running | ✓ 2/2 checks ... | None | ec2-54-68-97-203.us-w... | 5 |
| Slave1 | i-e5b144e9 | c3.large | us-west-2a | running | ✓ 2/2 checks ... | None | ec2-54-69-16-92.us-w... | 5 |
| Slave2 | i-41b6434d | c3.large | us-west-2a | running | ✓ 2/2 checks ... | None | ec2-54-191-224-173.us... | 5 |
| Slave3 | i-2da45121 | c3.large | us-west-2a | running | ✓ 2/2 checks ... | None | ec2-54-191-212-235.us... | 5 |
| Slave4 | i-2ca45120 | c3.large | us-west-2a | running | ✓ 2/2 checks ... | None | ec2-54-191-152-181.us... | 5 |
| Slave5 | i-bb9663b7 | c3.large | us-west-2a | running | ✓ 2/2 checks ... | None | ec2-54-191-181-240.us... | 5 |
| Slave6 | i-d99762d5 | c3.large | us-west-2a | running | ✓ 2/2 checks ... | None | ec2-54-191-15-29.us-w... | 5 |

Instance: i-d99762d5 (Slave6) Public DNS: ec2-54-191-15-29.us-west-2.compute.amazonaws.com

Description Status Checks Monitoring Tags

| | | | |
|----------------|-----------------------|-------------------|--------------------------------------------------|
| Instance ID | i-d99762d5 | Public DNS | ec2-54-191-15-29.us-west-2.compute.amazonaws.com |
| Instance state | running | Public IP | 54.191.15.29 |
| Instance type | c3.large | Elastic IP | - |
| Private DNS | ip-172-31-21.us-west- | Availability zone | us-west-2a |

Feedback

```
ubuntu@ip-172-31-21-181:~$ 
Combine input records=1527348214
Combine output records=16008369
Reduce input groups=3368743
Reduce shuffle bytes=126434448
Reduce input records=4210816
Reduce output records=3368743
Spilled Records=17788465
Shuffled Maps =78
Failed Shuffles=0
Merged Map outputs=78
GC time elapsed (ms)=438319
CPU time spent (ms)=5457350
Physical memory (bytes) snapshot=22229140544
Virtual memory (bytes) snapshot=65845633024
Total committed heap usage (bytes)=16472604672

Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=10400319488
File Output Format Counters
Bytes Written=74220482

14/10/24 21:45:57 INFO client.RMProxy: Connecting to ResourceManager at /172.31.21.181:8040
14/10/24 21:45:57 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
14/10/24 21:45:57 INFO mapred.FileInputFormat: Total input paths to process : 1
14/10/24 21:45:57 INFO mapreduce.JobSubmitter: number of splits:2
14/10/24 21:45:58 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1414186288251_0002
14/10/24 21:45:58 INFO impl.YarnClientImpl: Submitted application application_1414186288251_0002
14/10/24 21:45:58 INFO mapreduce.Job: The url to track the job: http://ip-172-31-21-181:8088/proxy/application_1414186288251_0002/
14/10/24 21:45:58 INFO mapreduce.Job: Running job: job_1414186288251_0002
14/10/24 21:46:04 INFO mapreduce.Job: Job job_1414186288251_0002 running in uber mode : false
14/10/24 21:46:04 INFO mapreduce.Job: map 0% reduce 0%
14/10/24 21:46:18 INFO mapreduce.Job: map 83% reduce 0%
14/10/24 21:46:19 INFO mapreduce.Job: map 100% reduce 0%
14/10/24 21:46:30 INFO mapreduce.Job: map 100% reduce 100%
```

8 Nodes:

```
ubuntu@ip-172-31-21-181:~$ 
ubuntu@ip-172-31-21-181:~$ hadoop fs -copyFromLocal wiki10gb /largedata
14/10/24 22:00:48 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
ubuntu@ip-172-31-21-181:~$ hadoop jar wc.jar WordCount /largedata /largeoutput
14/10/24 22:04:23 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/10/24 22:04:24 INFO Client.RMProxy: Connecting to ResourceManager at /172.31.21.181:8040
14/10/24 22:04:24 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
14/10/24 22:04:25 INFO mapred.FileInputFormat: Total input paths to process : 1
14/10/24 22:04:25 INFO mapreduce.JobSubmitter: number of splits:78
14/10/24 22:04:25 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1414187980895_0001
14/10/24 22:04:25 INFO impl.YarnClientImpl: Submitted application application_1414187980895_0001
14/10/24 22:04:25 INFO mapreduce.Job: The url to track the job: http://ip-172-31-21-181:8088/proxy/application_1414187980895_0001/
14/10/24 22:04:33 INFO mapreduce.Job: Running job: job_1414187980895_0001
14/10/24 22:04:33 INFO mapreduce.Job: map 0% reduce 0%
14/10/24 22:05:10 INFO mapreduce.Job: map 1% reduce 0%
14/10/24 22:05:14 INFO mapreduce.Job: map 2% reduce 0%
14/10/24 22:05:18 INFO mapreduce.Job: map 3% reduce 0%
14/10/24 22:05:21 INFO mapreduce.Job: map 4% reduce 0%
14/10/24 22:05:24 INFO mapreduce.Job: map 5% reduce 0%
14/10/24 22:05:27 INFO mapreduce.Job: map 6% reduce 0%
14/10/24 22:05:30 INFO mapreduce.Job: map 7% reduce 0%
14/10/24 22:05:34 INFO mapreduce.Job: map 8% reduce 0%
14/10/24 22:05:37 INFO mapreduce.Job: map 9% reduce 0%
14/10/24 22:05:43 INFO mapreduce.Job: map 10% reduce 0%
14/10/24 22:05:49 INFO mapreduce.Job: map 11% reduce 0%
14/10/24 22:06:05 INFO mapreduce.Job: map 12% reduce 0%
14/10/24 22:06:10 INFO mapreduce.Job: map 13% reduce 0%
14/10/24 22:06:15 INFO mapreduce.Job: map 14% reduce 0%
14/10/24 22:06:18 INFO mapreduce.Job: map 15% reduce 0%
14/10/24 22:06:22 INFO mapreduce.Job: map 16% reduce 0%
14/10/24 22:06:25 INFO mapreduce.Job: map 17% reduce 0%
14/10/24 22:06:28 INFO mapreduce.Job: map 18% reduce 0%
14/10/24 22:06:32 INFO mapreduce.Job: map 19% reduce 0%
14/10/24 22:06:37 INFO mapreduce.Job: map 20% reduce 0%
14/10/24 22:06:53 INFO mapreduce.Job: map 21% reduce 0%
14/10/24 22:06:58 INFO mapreduce.Job: map 22% reduce 0%
14/10/24 22:07:01 INFO mapreduce.Job: map 23% reduce 0%
```

```

ubuntu@ip-172-31-21-181: ~
14/10/24 22:11:18 INFO mapreduce.Job: map 93% reduce 29%
14/10/24 22:11:29 INFO mapreduce.Job: map 94% reduce 29%
14/10/24 22:11:30 INFO mapreduce.Job: map 95% reduce 29%
14/10/24 22:11:33 INFO mapreduce.Job: map 95% reduce 30%
14/10/24 22:11:37 INFO mapreduce.Job: map 96% reduce 30%
14/10/24 22:11:39 INFO mapreduce.Job: map 97% reduce 31%
14/10/24 22:11:42 INFO mapreduce.Job: map 97% reduce 32%
14/10/24 22:11:48 INFO mapreduce.Job: map 98% reduce 32%
14/10/24 22:12:32 INFO mapreduce.Job: map 99% reduce 32%
14/10/24 22:12:43 INFO mapreduce.Job: map 100% reduce 32%
14/10/24 22:12:45 INFO mapreduce.Job: map 100% reduce 33%
14/10/24 22:12:51 INFO mapreduce.Job: map 100% reduce 71%
14/10/24 22:12:54 INFO mapreduce.Job: map 100% reduce 91%
14/10/24 22:12:55 INFO mapreduce.Job: map 100% reduce 100%
14/10/24 22:12:55 INFO mapreduce.Job: Job job_1414187980895_0001 completed successfully
14/10/24 22:12:55 INFO mapreduce.Job: Counters: 50
    File System Counters
        FILE: Number of bytes read=259939266
        FILE: Number of bytes written=394054739
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=10400326274
        HDFS: Number of bytes written=74220482
        HDFS: Number of read operations=237
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
    Job Counters
        Killed map tasks=5
        Launched map tasks=83
        Launched reduce tasks=1
        Data-local map tasks=83
        Total time spent by all maps in occupied slots (ms)=21592589
        Total time spent by all reduces in occupied slots (ms)=256444
        Total time spent by all map tasks (ms)=21592589
        Total time spent by all reduce tasks (ms)=256444
        Total vcore-seconds taken by all map tasks=21592589
        Total vcore-seconds taken by all reduce tasks=256444
        Total megabyte-seconds taken by all map tasks=22110811136
        Total megabyte-seconds taken by all reduce tasks=262598656

```

EC2 Management Console - Google Chrome

https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:sort=launchTime

Services Edit

Launch Instance Connect Actions

| Name | Instance ID | Instance Type | Availability Zone | Instance State | Status Checks | Alarm Status | Public DNS |
|---------------|-------------------|-----------------|-------------------|----------------|-----------------------|--------------|---------------------------------|
| Slave2 | i-41b6434d | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-224-173.us... |
| Slave3 | i-2da45121 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-212-235.us... |
| Slave4 | i-2ca45120 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-152-181.us... |
| Slave5 | i-bb9663b7 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-181-240.us... |
| Slave6 | i-d99762d5 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-191-15-29.us-w... |
| Slave7 | i-f49c68f8 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-186-215-113.us... |
| Slave8 | i-df9c69d3 | c3.large | us-west-2a | running | 2/2 checks ... | None | ec2-54-69-52-29.us-we... |

Instance: i-df9c69d3 (Slave8) Public DNS: ec2-54-69-52-29.us-west-2.compute.amazonaws.com

Description Status Checks Monitoring Tags

| | | | |
|----------------|---------------------------|-------------------|-------------------------------------------------|
| Instance ID | i-df9c69d3 | Public DNS | ec2-54-69-52-29.us-west-2.compute.amazonaws.com |
| Instance state | running | Public IP | 54.69.52.29 |
| Instance type | c3.large | Elastic IP | - |
| Private DNS | ip-172-31-28-217.us-west- | Availability zone | us-west-2a |

Feedback

© 2008 - 2014, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

16 nodes:

```
ubuntu@ip-172-31-26-137:~$ hadoop jar wc.jar WordCount /largedata /output
14/10/22 01:28:46 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/10/22 01:28:47 INFO client.RMProxy: Connecting to ResourceManager at /172.31.26.137:8040
14/10/22 01:28:47 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
14/10/22 01:28:48 INFO mapred.FileInputFormat: Total input paths to process : 1
14/10/22 01:28:48 WARN split.JobSplitWriter: Max block location exceeded for split: hdfs://172.31.26.137:9000/largedata:0+134217728 splitsize: 16 maxsize: 10
14/10/22 01:28:48 WARN split.JobSplitWriter: Max block location exceeded for split: hdfs://172.31.26.137:9000/largedata:134217728+134217728 splitsize: 16 maxsize: 10
14/10/22 01:28:48 WARN split.JobSplitWriter: Max block location exceeded for split: hdfs://172.31.26.137:9000/largedata:268435456+134217728 splitsize: 16 maxsize: 10
14/10/22 01:28:48 WARN split.JobSplitWriter: Max block location exceeded for split: hdfs://172.31.26.137:9000/largedata:402653184+134217728 splitsize: 16 maxsize: 10
14/10/22 01:28:48 WARN split.JobSplitWriter: Max block location exceeded for split: hdfs://172.31.26.137:9000/largedata:536870912+134217728 splitsize: 16 maxsize: 10
14/10/22 01:28:48 WARN split.JobSplitWriter: Max block location exceeded for split: hdfs://172.31.26.137:9000/largedata:671888640+134217728 splitsize: 16 maxsize: 10
14/10/22 01:28:48 WARN split.JobSplitWriter: Max block location exceeded for split: hdfs://172.31.26.137:9000/largedata:805306368+134217728 splitsize: 16 maxsize: 10
14/10/22 01:28:48 WARN split.JobSplitWriter: Max block location exceeded for split: hdfs://172.31.26.137:9000/largedata:939524096+134217728 splitsize: 16 maxsize: 10
14/10/22 01:28:48 WARN split.JobSplitWriter: Max block location exceeded for split: hdfs://172.31.26.137:9000/largedata:1073741824+134217728 splitsize: 16 maxsize: 10

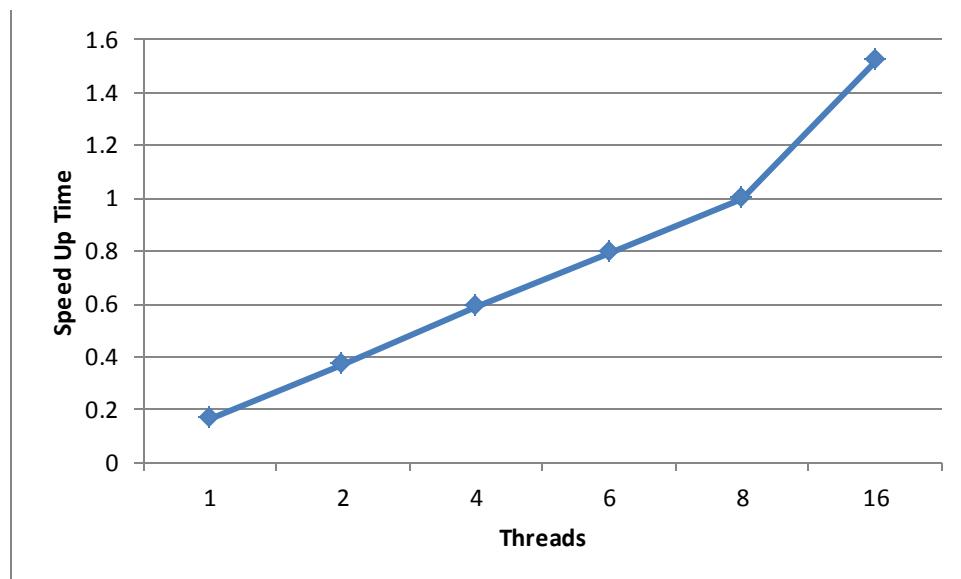
ubuntu@ip-172-31-26-137:~$ 
14/10/22 01:33:21 INFO mapreduce.Job: map 61% reduce 3%
14/10/22 01:33:28 INFO mapreduce.Job: map 62% reduce 3%
14/10/22 01:33:33 INFO mapreduce.Job: map 63% reduce 3%
14/10/22 01:33:37 INFO mapreduce.Job: map 64% reduce 3%
14/10/22 01:33:38 INFO mapreduce.Job: map 65% reduce 3%
14/10/22 01:33:40 INFO mapreduce.Job: map 66% reduce 3%
14/10/22 01:33:42 INFO mapreduce.Job: map 67% reduce 3%
14/10/22 01:33:44 INFO mapreduce.Job: map 68% reduce 4%
14/10/22 01:33:46 INFO mapreduce.Job: map 69% reduce 4%
14/10/22 01:33:47 INFO mapreduce.Job: map 69% reduce 5%
14/10/22 01:33:49 INFO mapreduce.Job: map 70% reduce 5%
14/10/22 01:33:50 INFO mapreduce.Job: map 71% reduce 6%
14/10/22 01:33:53 INFO mapreduce.Job: map 72% reduce 6%
14/10/22 01:33:57 INFO mapreduce.Job: map 74% reduce 6%
14/10/22 01:33:59 INFO mapreduce.Job: map 75% reduce 8%
14/10/22 01:34:01 INFO mapreduce.Job: map 76% reduce 8%
14/10/22 01:34:02 INFO mapreduce.Job: map 77% reduce 9%
14/10/22 01:34:03 INFO mapreduce.Job: map 79% reduce 9%
14/10/22 01:34:04 INFO mapreduce.Job: map 81% reduce 9%
14/10/22 01:34:05 INFO mapreduce.Job: map 82% reduce 15%
14/10/22 01:34:06 INFO mapreduce.Job: map 83% reduce 15%
14/10/22 01:34:07 INFO mapreduce.Job: map 85% reduce 15%
14/10/22 01:34:08 INFO mapreduce.Job: map 88% reduce 18%
14/10/22 01:34:09 INFO mapreduce.Job: map 90% reduce 18%
14/10/22 01:34:10 INFO mapreduce.Job: map 91% reduce 18%
14/10/22 01:34:11 INFO mapreduce.Job: map 92% reduce 18%
14/10/22 01:34:12 INFO mapreduce.Job: map 94% reduce 24%
14/10/22 01:34:13 INFO mapreduce.Job: map 95% reduce 24%
14/10/22 01:34:15 INFO mapreduce.Job: map 96% reduce 24%
14/10/22 01:34:17 INFO mapreduce.Job: map 97% reduce 24%
14/10/22 01:34:20 INFO mapreduce.Job: map 98% reduce 24%
14/10/22 01:34:22 INFO mapreduce.Job: map 99% reduce 24%
14/10/22 01:34:24 INFO mapreduce.Job: map 100% reduce 33%
14/10/22 01:34:27 INFO mapreduce.Job: map 100% reduce 68%
14/10/22 01:34:30 INFO mapreduce.Job: map 100% reduce 86%
14/10/22 01:34:32 INFO mapreduce.Job: map 100% reduce 100%
14/10/22 01:34:32 INFO mapreduce.Job: Job job_1413938955590_0009 completed successfully
14/10/22 01:34:32 INFO mapreduce.Job: Counters: 51
File System Counters
FILE: Number of bytes read=259697371
FILE: Number of bytes written=393811738
```

| Nodes of the cluster - Google Chrome | EC2 Management | NameNode Information | JobHistory | Nodes of the cluster |
|--------------------------------------|-------------------|----------------------|---------------------------|------------------------------------------------------------------------------|
| Apps | Hadoop 2.4-Inst-1 | Setting up Hadoop | Hadoop Tutorial | Apache Hadoop |
| FINISHED | /default- | RUNNING | ip-172-31-24-82.us-west- | ip-172-31-24-82.us-west- 2.compute.internal:8042 2014-10-22 0 0 B 8 GB 0 € |
| FAILED | rack | | ip-172-31-17-63.us-west- | ip-172-31-17-63.us-west- 2.compute.internal:8042 2014-10-22 0 0 B 8 GB 0 € |
| KILLED | /default- | RUNNING | ip-172-31-21-125.us-west- | ip-172-31-21-125.us-west- 2.compute.internal:8042 2014-10-22 8 8 GB 0 B 8 € |
| Scheduler | rack | | ip-172-31-21-125.us-west- | ip-172-31-21-125.us-west- 2.compute.internal:8042 2014-10-22 8 8 GB 0 B 8 € |
| Tools | /default- | RUNNING | ip-172-31-27-200.us-west- | ip-172-31-27-200.us-west- 2.compute.internal:8042 2014-10-22 8 8 GB 0 B 8 € |
| | rack | | ip-172-31-26-44.us-west- | ip-172-31-26-44.us-west- 2.compute.internal:41511 2014-10-22 8 8 GB 0 B 8 € |
| | /default- | RUNNING | ip-172-31-30-43.us-west- | ip-172-31-30-43.us-west- 2.compute.internal:36404 2014-10-22 8 8 GB 0 B 8 € |
| | rack | | ip-172-31-31-79.us-west- | ip-172-31-31-79.us-west- 2.compute.internal:46782 2014-10-22 8 8 GB 0 B 8 € |
| | /default- | RUNNING | ip-172-31-18-135.us-west- | ip-172-31-18-135.us-west- 2.compute.internal:36332 2014-10-22 8 8 GB 0 B 8 € |
| | rack | | ip-172-31-24-117.us-west- | ip-172-31-24-117.us-west- 2.compute.internal:40161 2014-10-22 7 8 GB 0 B 7 € |
| | /default- | RUNNING | ip-172-31-30-217.us-west- | ip-172-31-30-217.us-west- 2.compute.internal:40483 2014-10-22 8 8 GB 0 B 8 € |
| | rack | | ip-172-31-18-234.us-west- | ip-172-31-18-234.us-west- 2.compute.internal:48312 2014-10-22 0 0 B 8 GB 0 € |
| | /default- | RUNNING | ip-172-31-16-146.us-west- | ip-172-31-16-146.us-west- 2.compute.internal:56277 2014-10-22 8 8 GB 0 B 8 € |
| | rack | | ip-172-31-27-188.us-west- | ip-172-31-27-188.us-west- 2.compute.internal:58302 2014-10-22 0 0 B 8 GB 0 € |
| | /default- | RUNNING | ip-172-31-24-253.us-west- | ip-172-31-24-253.us-west- 2.compute.internal:50835 2014-10-22 8 8 GB 0 B 8 € |
| | rack | | ip-172-31-27-185.us-west- | ip-172-31-27-185.us-west- 2.compute.internal:53763 2014-10-22 0 0 B 8 GB 0 € |
| | /default- | RUNNING | ip-172-31-19-89.us-west- | ip-172-31-19-89.us-west- 2.compute.internal:58013 2014-10-22 0 0 B 8 GB 0 € |
| | rack | | ip-172-31-19-89.us-west- | ip-172-31-19-89.us-west- 2.compute.internal:8042 2014-10-22 0 0 B 8 GB 0 € |

Number of nodes and the speed up time:

Base :1 Node

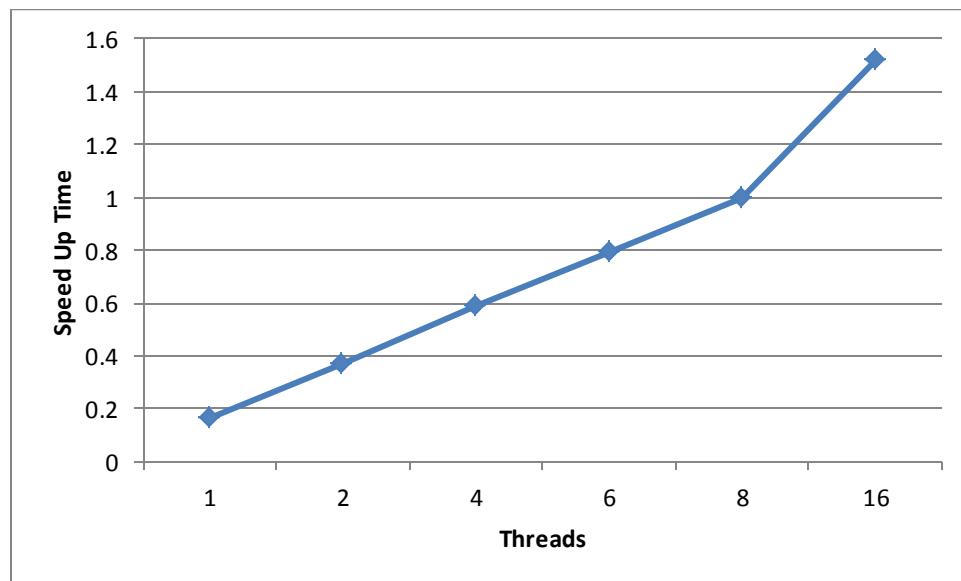
| Nodes | Execution time in seconds | Execution time for 1 Node | Speed Up |
|-------|---------------------------|---------------------------|-------------|
| 1 | 2950 | 2950 | 1 |
| 2 | 1338 | 2950 | 2.204783259 |
| 4 | 850.8 | 2950 | 3.467324871 |
| 6 | 625.8 | 2950 | 4.713966123 |
| 8 | 498 | 2950 | 5.923694779 |
| 16 | 327.6 | 2950 | 9.004884005 |



Number of nodes and the speed up time:

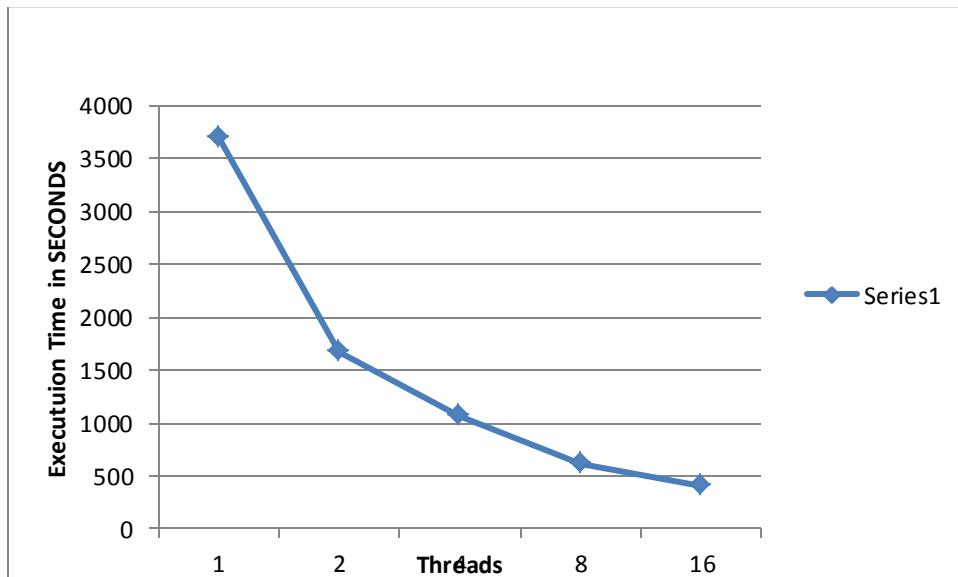
Base : 8 Node

| Nodes | Execution time in seconds | Execution time for 8 Node | Speed Up |
|-------|---------------------------|---------------------------|-------------|
| 1 | 2950 | 498 | 0.168813559 |
| 2 | 1338 | 498 | 0.372197309 |
| 4 | 850.8 | 498 | 0.585331453 |
| 6 | 625.8 | 498 | 0.7957814 |
| 8 | 498 | 498 | 1 |
| 16 | 327.6 | 498 | 1.52014652 |



Java 1 Thread Baseline:

| Nodes | Execution time in seconds | Execution time for 1 Thread java | Speed Up |
|-------|---------------------------|----------------------------------|-------------|
| 1 | 2950 | 654.552 | 0.221882034 |
| 2 | 1338 | 654.552 | 0.489201794 |
| 4 | 850.8 | 654.552 | 0.769337094 |
| 8 | 625.8 | 654.552 | 1.045944391 |
| 16 | 498 | 654.552 | 1.314361446 |



10GB DATASET ON c3.large instance(Sorted word count without frequency based on the ASCII) :

| Word Count | Execution time in seconds |
|------------|---------------------------|
| 1 | 2841 |
| 16 | 396 |

Best Performance is achieved at 16 nodes for both types .

SWIFT INSTALLATION:

Steps to get you IAM access keys:

1. Go to the IAM console.
2. From the navigation menu, click Users.
3. Select your IAM user name, or create a new one.
4. Click User Actions, and then click Manage Access Keys.
5. Click Create Access Key.
6. Click Download Credentials, and store the keys (a .csv file) in a secure location.

Create a Keypair to ssh to the ec2 instances:

1. Open the Amazon EC2 console .
2. In the top navigation bar, in the region selector, click **US West (Oregon)**.
3. In the left navigation pane, under Network and Security, click Key Pairs.
4. Click Create Key Pair. Type mykeypair in the new Key pair name box.
5. Download the private key file, which is named mykeypair.pem, and keep it in a safe place. You will need it to access any instances that you launch with this key pair.

Launch an EC2 instance with AMI as Ubuntu Server 14.04 LTS (HVM).

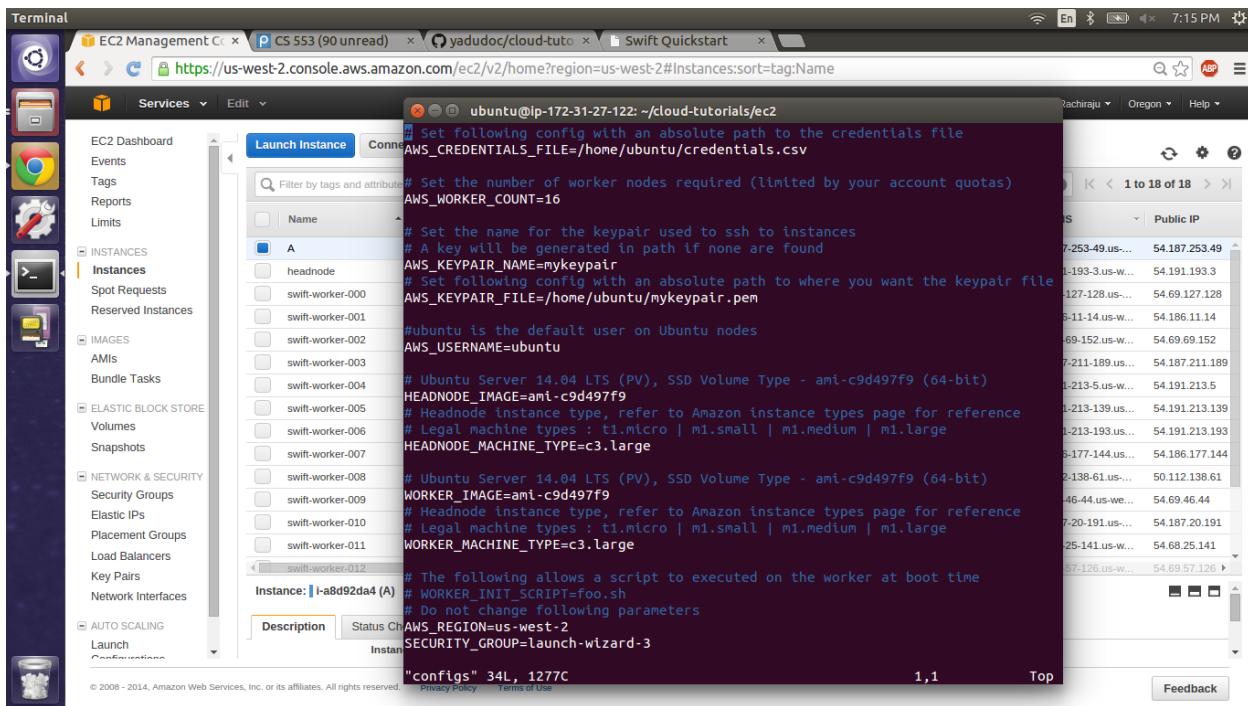
Do ssh-add mykey.pem. Then connect to the instance by using ssh -i @ubuntu:ipaddress

Then do a scp to send mykeypair.pem and credentials to the instance location.

Install the following:

1. sudo apt-get install -y python python-pip git
2. sudo pip install apache-libcloud
3. sudo apt-get install default-jdk
4. git clone <https://github.com/yadudoc/cloud-tutorials.git>
5. cd cloud-tutorials/ec2

Now configure the config file to the following shown in the screen shot:



```
# Set following config with an absolute path to the credentials file
AWS_CREDENTIALS_FILE=/home/ubuntu/credentials.csv

# Set the number of worker nodes required (limited by your account quotas)
AWS_WORKER_COUNT=16

# Set the name for the keypair used to ssh to instances
# A key will be generated in path if none are found
AWS_KEYPAIR_NAME=mykeypair
# Set following config with an absolute path to where you want the keypair file
AWS_KEYPAIR_FILE=/home/ubuntu/mykeypair.pem

#ubuntu is the default user on Ubuntu nodes
AWS_USERNAME=ubuntu

# Ubuntu Server 14.04 LTS (PV), SSD Volume Type - ami-c9d497f9 (64-bit)
HEADNODE_IMAGE=ami-c9d497f9
# Headnode instance type, refer to Amazon instance types page for reference
# Legal machine types : t1.micro | m1.small | m1.medium | m1.large
HEADNODE_MACHINE_TYPE=c3.large

# Ubuntu Server 14.04 LTS (PV), SSD Volume Type - ami-c9d497f9 (64-bit)
WORKER_IMAGE=ami-c9d497f9
# Headnode instance type, refer to Amazon instance types page for reference
# Legal machine types : t1.micro | m1.small | m1.medium | m1.large
WORKER_MACHINE_TYPE=c3.large

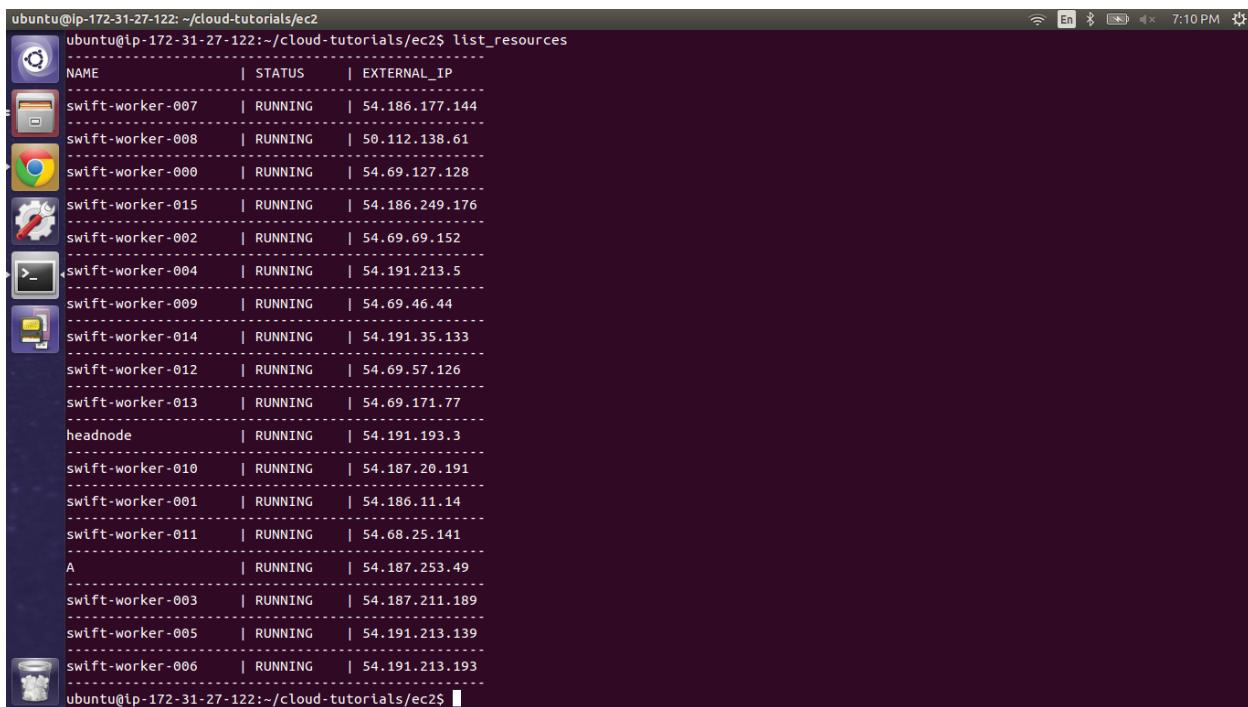
# The following allows a script to be executed on the worker at boot time
# WORKER_INIT_SCRIPT=foo.sh
# Do not change following parameters
AWS_REGION=us-west-2
SECURITY_GROUP=launch-wizard-3

"configs" 34L, 1277C
```

Now navigate to ec2 directory and do a source setup.sh.

To see if all the 16 workers are launched check by using list_resources.

This will give the following:



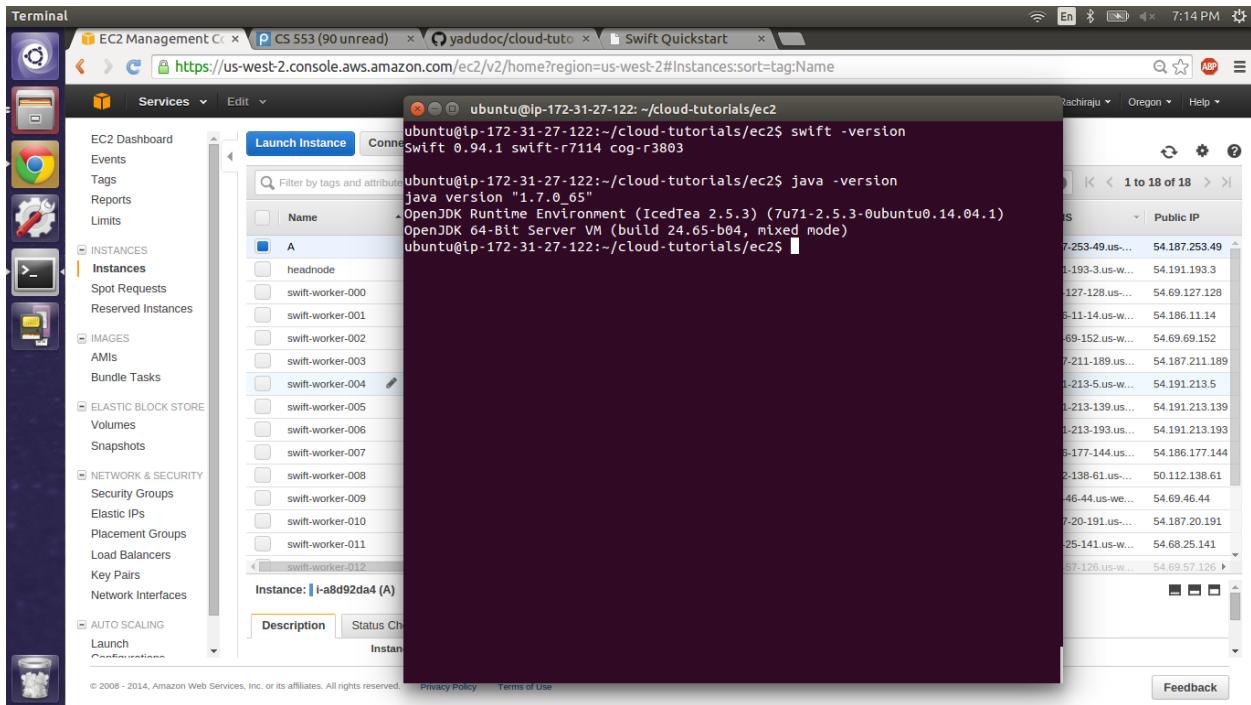
| NAME | STATUS | EXTERNAL_IP |
|------------------|---------|----------------|
| swift-worker-007 | RUNNING | 54.186.177.144 |
| swift-worker-008 | RUNNING | 50.112.138.61 |
| swift-worker-000 | RUNNING | 54.69.127.128 |
| swift-worker-015 | RUNNING | 54.186.249.176 |
| swift-worker-002 | RUNNING | 54.69.69.152 |
| swift-worker-004 | RUNNING | 54.191.213.5 |
| swift-worker-009 | RUNNING | 54.69.46.44 |
| swift-worker-014 | RUNNING | 54.191.35.133 |
| swift-worker-012 | RUNNING | 54.69.57.126 |
| swift-worker-013 | RUNNING | 54.69.171.77 |
| headnode | RUNNING | 54.191.193.3 |
| swift-worker-010 | RUNNING | 54.187.20.191 |
| swift-worker-001 | RUNNING | 54.186.11.14 |
| swift-worker-011 | RUNNING | 54.68.25.141 |
| A | RUNNING | 54.187.253.49 |
| swift-worker-003 | RUNNING | 54.187.211.189 |
| swift-worker-005 | RUNNING | 54.191.213.139 |
| swift-worker-006 | RUNNING | 54.191.213.193 |

Connect to the headnode by using the “connect headnode” command.

Java and Swift Version:

To see the java version: `java -version`

To see the swift version: `swift -version`



Head node and 1 Worker:

The screenshot shows the AWS EC2 Management Console interface. On the left, the navigation pane is visible with options like EC2 Dashboard, Events, Tags, Reports, Limits, Instances, Images, AMIs, and more. The 'Instances' section is currently selected. In the main content area, there is a table of instances. The table has columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS. Two instances are listed: 'headnode' (Instance ID: i-14d1bfe, Type: c3.large, Zone: us-west-2c, State: running, Status Checks: 2/2 checks, Alarm Status: None, Public DNS: ec2-54-191-193-3.us-west-2.compute.amazonaws.com) and 'swift-worker-000' (Instance ID: i-fbdb2ff7, Type: c3.large, Zone: us-west-2a, State: running, Status Checks: 2/2 checks, Alarm Status: None, Public DNS: ec2-54-69-127-128.us-west-2.compute.amazonaws.com). Below the table, there are tabs for Description, Status Checks, Monitoring, and Tags. The 'Description' tab is active, showing the instance IDs and public DNS for both instances.

Head Node and 16 Workers:

The screenshot shows the AWS EC2 Management Console interface. The navigation pane is identical to the previous screenshot. In the main content area, there is a table of instances. The table has columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, Public DNS, Public IP, and Key Name. There are 17 entries in the table, starting with 'A' and ending with 'swift-worker-015'. Each entry includes a checkbox, the instance name, instance ID, type, zone, state, status checks, alarm status, public DNS, public IP, and key name. The first entry, 'A', is highlighted. Below the table, there is a summary line: 'Instance: i-a8d92da4 (A) Public DNS: ec2-54-187-253-49.us-west-2.compute.amazonaws.com'. At the bottom, there are tabs for Description, Status Checks, Monitoring, and Tags, with 'Description' being the active tab. The 'Public DNS' field at the bottom right is set to 'ec2-54-187-253-49.us-west-2.compute.amazonaws.com'.

Swift Word Count:

SYSTEM CONFIGURATION:

| | |
|---------------|--------------------|
| VERSION | : Swift 0.94.1 |
| INSTANCE TYPE | : c3. Large UBUNTU |
| RAM | : 3.75 GB |
| NO. OF CORES | : 2 virtual cores |
| STORAGE | : 32 GB |

Methodology:

1. Install Swift on 17 nodes (including the POSIX layer over S3 via FUSE, as Swift expects a shared POSIX filesystem between all the compute nodes)
2. Install the following in the node:
 - Install python-pip, apache-libcloud
 - git clone cloud-tutorials
 - Configure and start the cluster
3. Create an Access key credentials.csv and store it in an secure location.
4. Connect to instance by ssh and save the credentials and key file in it.
5. Update the configs file.
6. Navigate to ec2 and do a source setup.sh
7. Connect to headnode.
8. Now run the program using swift wordCount.swift.
9. Make sure we have the config file present to run the main swift file.

Change the following in the configs file:

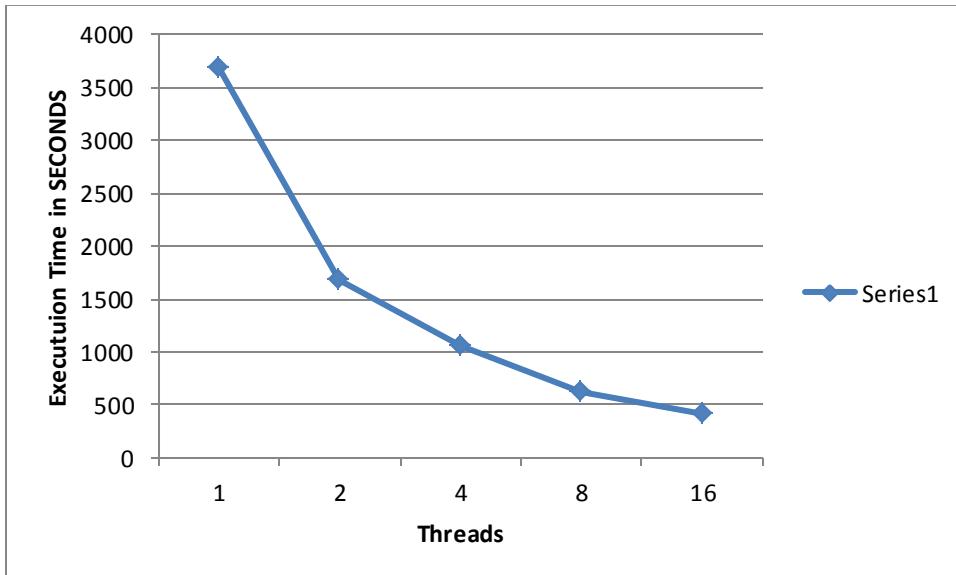
| | |
|-----------------------|------------------------------|
| AWS_CREDENTIALS_FILE | /home/Ubuntu/credentials.csv |
| AWS_WORKER_COUNT | Worker count is 17 |
| AWS_KEYPAIR_NAME | mykeypair.pem |
| AWS_KEYPAIR_FILE | /home/Ubuntu/cs553w.pem |
| HEADNODE_MACHINE_TYPE | c3.large |
| WORKER_MACHINE_TYPE | c3.large |

Execution results:

| Nodes | Execution time in seconds | Execution time for 1 Node | Speed Up |
|-------|---------------------------|---------------------------|-------------|
| 1 | 3687.5 | 3687.5 | 1 |
| 2 | 1672 | 3687.5 | 2.205442584 |
| 4 | 1061 | 3687.5 | 3.475494816 |
| 8 | 622.5 | 3687.5 | 5.923694779 |
| 16 | 409.2 | 3687.5 | 9.011485826 |

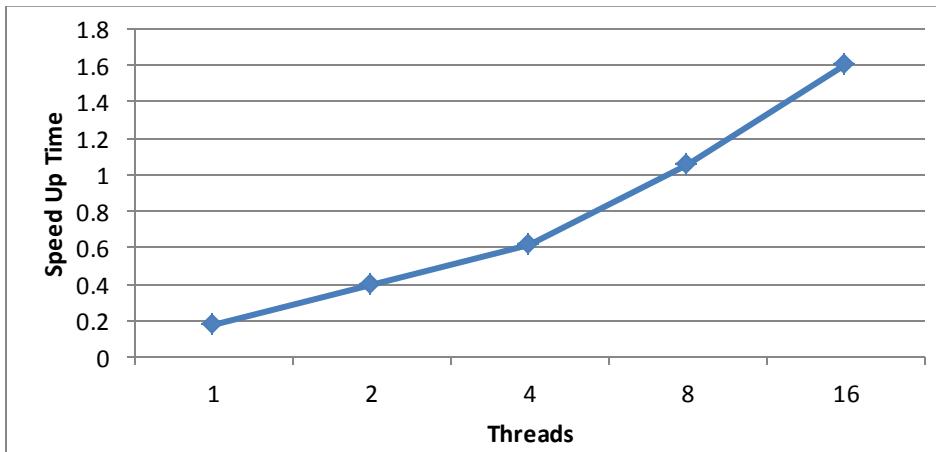
GRAPH

From the data available in the above table, a graph is plotted between number of nodes and execution time.



GRAPH :

Using the information from the table above a graph is plotted between number of nodes and speed up time. The baseline of this graph is value obtained at node 1.

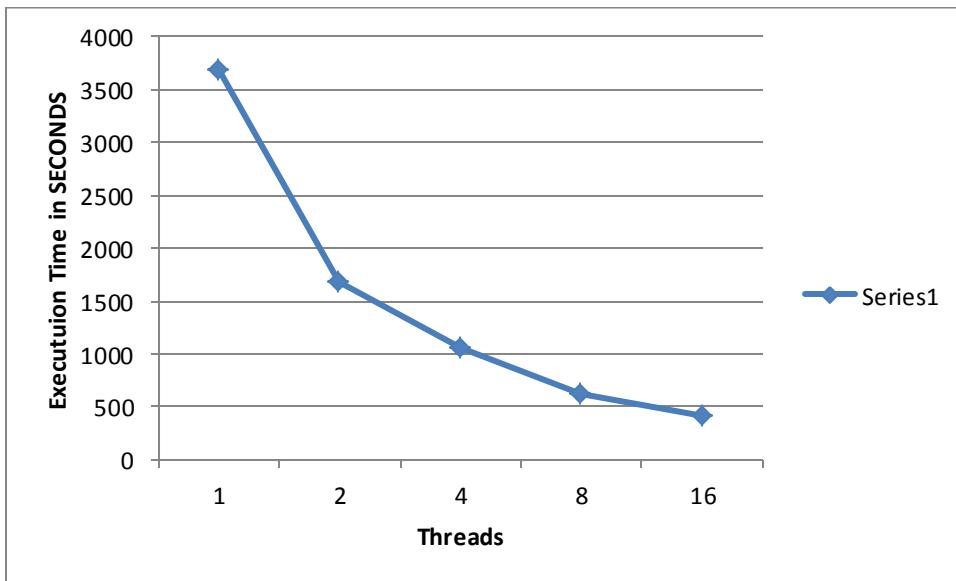


Best Performance is obtained at 16 nodes.

Java 1 Thread baseline:

| Nodes | Execution time in seconds | Execution time for 1 Thread java | Speed Up |
|-------|---------------------------|----------------------------------|-------------|
| 1 | 3687.5 | 654.552 | 0.177505627 |
| 2 | 1672 | 654.552 | 0.391478469 |
| 4 | 1061 | 654.552 | 0.616919887 |
| 8 | 622.5 | 654.552 | 1.051489157 |
| 16 | 409.2 | 654.552 | 1.599589443 |

Graph:



Sorting in SWIFT:

| Word Count | Execution time in seconds |
|------------|---------------------------|
| 1 | 2131 |
| 16 | 372 |

Best Performance is obtained at 16 nodes

Performance:

| PLATFORM | NUMBER OF NODES | EXECUTION TIME in seconds |
|----------|-----------------|---------------------------|
| | | Word Count |
| JAVA | 1 | 654.552 |
| HADOOP | 1 | 2950 |
| SWIFT | 1 | 3687 |
| JAVA | 1 | 654.552 |
| HADOOP | 16 | 327.6 |
| SWIFT | 16 | 409 |

Out of all the three **HADOOP** is said to give the best performance.

From the above table we can say that best performance at 16 nodes is offered by HADOOP and at 1 node is offered by JAVA. Compared to 1 node execution time of JAVA execution time of 16 nodes of SWIFT and HADOOP is faster. Let it be 100 nodes or 1000 nodes it doesn't make any difference as you scale from 16 nodes to higher because the execution time stays constant as there is resource saturation.

While Considering sorting we can see that swift gives the best performance. It gives almost 1/4th times that of hadoop. So while considering sorting we can say that SWIFT has the best performance.

MPI Sort:

Configuration:

| | |
|---------------|--------------------|
| VERSION | : MPI 3.1.3 |
| INSTANCE TYPE | : c3. Large UBUNTU |
| RAM | : 3.75 GB |
| NO. OF CORES | : 2 virtual cores |
| STORAGE | : 32 GB |

Methodology:

MPI INSTALL VERSION: mpich3.1.3

- 1.install mpich3.1.3 and configure it with command ./configure disable --fortran
2. build \$make

after the installation of mpich3.1.3 on the local node

3. run MPIsort.cpp using

```
mpicc MPIsort.cpp -o MPITsort
```

after compilation the resulting file is an executable file MPIsort

4. mpirun -np 2 /home/ubuntu/MPIsort wiki10gb.txt

mpirun executes the executable file with 2 processes sorting the wiki10gb.txt file.

Running MPI ON AMAZON EC2

run the below commands

```
sudo apt-get install pythonsetuptools
sudo apt-get install python-pip
sudo apt-get update
sudo easy_install -upgrade pip
sudo easy_install StarCluster0.95.5
and install pycrypto-2.6.1
```

after installing Starcluster0.95.5
configure starcluster in
.starcluster/config

and after configuring the start cluster is installed
now start the cluster on 16 nodes.

