

Table of Contents

1.	<i>Install Spark in centos Linux – 60 Minutes.....</i>	2
2.	<i>Spark Two Nodes Cluster – 90 Minutes</i>	7
3.	<i>Launching Spark Job on a YARN Cluster– 120 Minutes (D).....</i>	16
4.	<i>Jobs Monitoring : Using Web UI. – 45 Minutes.....</i>	41

1. Install Spark in centos Linux – 60 Minutes

Start the Linux server open a terminal to the host using putty. Perform the following activities in the console.

Download the required software

Install JDK if not done earlier

Untar spark-X.X.o-bin-hadoop.X.tar to /opt

```
#cd /Software  
# tar -xvf spark-* -C /opt/
```

It should create a folder inside spark.X, rename to spark folder.

```
#cd /opt  
#mv spark* spark
```

Set the JAVA_HOME and initialize the PATH variable.

Open the profile and update with the following statements.

```
#vi ~/.bashrc
```

```
export JAVA_HOME=/opt/jdk  
export PATH=$PATH:$JAVA_HOME/bin  
export PATH=$PATH:/opt/spark/bin
```

Type bash, to initialize the profile.

```
#bash
```

Go to the installation directory And execute:

```
# cd /opt/spark/bin
```

```
#spark-shell
```

```
[root@master spark]# cd spark-2.1/
[root@master spark-2.1]# ls
bin  data    jars   licenses  python  README.md  sbin
conf examples LICENSE NOTICE   R        RELEASE   yarn
[root@master spark-2.1]# bin/spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).

15/05/15 23:22:19 INFO Executor: Using REPL class URI: http://192.168.188.1:4967
2
15/05/15 23:22:19 INFO AkkaUtils: Connecting to HeartbeatReceiver: akka.tcp://sparkDriver@ht:49686/user/HeartbeatReceiver
15/05/15 23:22:19 INFO NettyBlockTransferService: Server created on 49706
15/05/15 23:22:19 INFO BlockManagerMaster: Trying to register BlockManager
15/05/15 23:22:19 INFO BlockManagerMasterActor: Registering block manager localhost:49706 with 265.1 MB RAM, BlockManagerId<<driver>, localhost, 49706>
15/05/15 23:22:19 INFO BlockManagerMaster: Registered BlockManager
15/05/15 23:22:19 INFO SparkILoop: Created spark context..
Spark context available as sc.
15/05/15 23:22:20 INFO SparkILoop: Created sql context <with Hive support>..
SQL context available as sqlContext.

scala>
```

Enter the following command in the scala console to create a data set of 1...10 integers

```
#val data = 1 to 10
#data.foreach(println)
#println(data(2))
```

```
scala> val data = 1 to 10
data: scala.collection.immutable.Range.Inclusive = Range 1 to 10

scala> data.foreach(println)
1
2
3
4
5
6
7
8
9
10

scala> println(data(2))
3

scala> 
```

As seen above, you have initialized data variable with range from 1 to 10. Then print its value using foreach method of scala. You can also access the data using index.

Web UI of the spark shell can be accessed using the following URL. You can click on each tab and verify the screen.

<http://localhost:4040/jobs/>

The screenshot shows the Apache Spark 2.2.0 Web UI. At the top, there is a header bar with navigation icons (back, forward, search) and a URL field containing "10.10.20.27:4040/jobs/". Below the header is a navigation menu with tabs: "Jobs" (which is selected), "Stages", "Storage", "Environment", "Executors", and "SQL". To the left of the menu is the Apache Spark logo. The main content area is titled "Spark Jobs (?)". It displays system information: "User: root", "Total Uptime: 43 s", and "Scheduling Mode: FIFO". There is also a link "▶ Event Timeline".

Further details on this web UI will be discussed later.

You have successfully installed spark for local development.

-----Lab Ends Here -----

2. Spark Two Nodes Cluster – 90 Minutes

Architecture

master – sparko and Slave – spark1

	Master	Slave
Hostname	Sparko Hadoopo	Spark1 Hadoop1

Let us start the Spark master.

Install the spark. You can refer the first lab.

Execute the following in Sparko/hadoopo

Setup a Spark master node

Change the Master Port to 8090, there is issue when it is executed in 7077 in docker environment.

Execute on the sparko – Terminal.

```
#export SPARK_MASTER_PORT=8090
#spark-class org.apache.spark.deploy.master.Master
```

```
@5d486484cd1c:/software (docker) 961 bash 962
20/09/12 02:33:29 INFO Master: Started daemon with process name: 35@5d486484cd1c
20/09/12 02:33:29 INFO SignalUtils: Registered signal handler for TERM
20/09/12 02:33:29 INFO SignalUtils: Registered signal handler for HUP
20/09/12 02:33:29 INFO SignalUtils: Registered signal handler for INT
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/opt/spark/jars/spark-unsafe_2.12-3.0.1.jar) to constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
20/09/12 02:33:30 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
20/09/12 02:33:30 INFO SecurityManager: Changing view acls to: root
20/09/12 02:33:30 INFO SecurityManager: Changing modify acls to: root
20/09/12 02:33:30 INFO SecurityManager: Changing view acls groups to:
20/09/12 02:33:30 INFO SecurityManager: Changing modify acls groups to:
20/09/12 02:33:30 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(root); groups with view permissions: Set(); users with modify permissions: Set(root); groups with modify permissions: Set()
20/09/12 02:33:31 INFO Utils: Successfully started service 'sparkMaster' on port 7077.
20/09/12 02:33:31 INFO Master: Starting Spark master at spark://172.17.0.2:7077
20/09/12 02:33:31 INFO Master: Running Spark version 3.0.1
20/09/12 02:33:32 INFO Utils: Successfully started service 'MasterUI' on port 8080.
20/09/12 02:33:32 INFO MasterWebUI: Bound MasterWebUI to 0.0.0.0, and started at http://5d486484cd1c:8080
20/09/12 02:33:32 INFO Master: I have been elected leader! New state: ALIVE
```

Start a worker process On Node/spark o: In a separate terminal.

Attach a worker node to the cluster, execute the following in the sparko container.

```
#spark-class org.apache.spark.deploy.worker.Worker -c 1 -m 1G spark://172.18.0.2:8090
```

You need to replace with the IP of the sparko node. It can be retrieve using ifconfig command.

```
20/09/12 02:55:17 INFO Worker: Starting Spark worker 172.17.0.2:38483 with 1 cores, 2.0 GiB RAM
20/09/12 02:55:17 INFO Worker: Running Spark version 3.0.1
20/09/12 02:55:17 INFO Worker: Spark home: /opt/spark
20/09/12 02:55:17 INFO ResourceUtils: -----
20/09/12 02:55:17 INFO ResourceUtils: Resources for spark.worker:

20/09/12 02:55:17 INFO ResourceUtils: -----
20/09/12 02:55:17 INFO Utils: Successfully started service 'WorkerUI' on port 8081.
20/09/12 02:55:17 INFO WorkerWebUI: Bound WorkerWebUI to 0.0.0.0, and started at http://224e200bfc56:8081
20/09/12 02:55:17 INFO Worker: Connecting to master 172.17.0.2:7077...
20/09/12 02:55:18 INFO TransportClientFactory: Successfully created connection to /172.17.0.2:7077 after 40 ms (0 ms spent in bootstraps)
20/09/12 02:55:18 INFO Worker: Successfully registered with master spark://172.17.0.2:7077
20/09/12 02:55:18 INFO Worker: Asked to launch executor app-20200912025310-0000/0 for Spark shell
20/09/12 02:55:18 INFO SecurityManager: Changing view acls to: root
20/09/12 02:55:18 INFO SecurityManager: Changing modify acls to: root
20/09/12 02:55:18 INFO SecurityManager: Changing view acls groups to:
20/09/12 02:55:18 INFO SecurityManager: Changing modify acls groups to:
20/09/12 02:55:18 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(root); groups with view permissions: Set(); users with modify permissions: Set(root); groups with modify permissions: Set()
20/09/12 02:55:18 INFO ExecutorRunner: Launch command: "/opt/java/bin/java" "-cp" "/opt/spark/conf/:/opt/spark/jars/*" "-Xmx1024M" "-Dspark.driver.port=35927" "org.apache.spark.executor.CoarseGrainedExecutorBackend" "--driver-url" "spark://CoarseGrainedScheduler@224e200bfc56:35927" "--executor-id" "0" "--hostname" "172.17.0.2" "--cores" "1" "--app-id" "app-20200912025310-0000" "--worker-url" "spark://Worker@172.17.0.2:38483"
```

If unable to connect to localhost replace it with the container IP or the container alias i.e sparko.

Refresh the web ui, ensure that you can see a worker as shown below.

<http://127.0.0.1:8080>

Apache Spark 3.0.1		Spark Master at spark://172.17.0.2:8090													
URL: spark://172.17.0.2:8090															
Alive Workers: 1															
Cores in use: 1 Total, 0 Used															
Memory in use: 1024.0 MiB Total, 0.0 B Used															
Resources in use:															
Applications: 0 Running, 0 Completed															
Drivers: 0 Running, 0 Completed															
Status: ALIVE															
- Workers (1)															
Worker Id			Address	State	Cores	Memory	Resources								
worker-20210118134218-172.17.0.2-34667			172.17.0.2:34667	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)									
- Running Applications (0)															
Application ID	Name	Cores	Memory per Executor	Resources Per Executor		Submitted Time	User	State	Duration						
- Completed Applications (0)															
Application ID	Name	Cores	Memory per Executor	Resources Per Executor		Submitted Time	User	State	Duration						

Start the Node on **spark1/hadoop1**

Ensure that you have install spark in the Spark1 too.

```
# spark-class org.apache.spark.deploy.worker.Worker -c 1 -m 1G spark://172.18.0.2:8090
```

Ensure to replace the IP of the master node. i.e **hadoopo**

At the end of this step, you should have 2 worker nodes as shown below:

Spark 3.0.1 **Spark Master at spark://172.19.0.3:8090**

URL: spark://172.19.0.3:8090
 Alive Workers: 2
 Cores in use: 2 Total, 0 Used
 Memory in use: 2.0 GiB Total, 0.0 B Used
 Resources in use:
 Applications: 0 Running, 0 Completed
 Drivers: 0 Running, 0 Completed
 Status: ALIVE

Workers (2)

Worker Id	Address	State	Cores	Memory	Resources
worker-20200912142723-172.19.0.3-35677	172.19.0.3:35677	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	
worker-20200912142924-172.19.0.2-38199	172.19.0.2:38199	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

You can also verify from the master console.

```
23/02/22 17:37:49 INFO Utils: Successfully started service 'sparkMaster' on port 8090.
23/02/22 17:37:49 INFO Master: Starting Spark master at spark://172.18.0.2:8090
23/02/22 17:37:50 INFO Master: Running Spark version 3.3.0
23/02/22 17:37:51 INFO Utils: Successfully started service 'MasterUI' on port 8080.
23/02/22 17:37:51 INFO MasterWebUI: Bound MasterWebUI to 0.0.0.0, and started at http://hadoop0:8080
23/02/22 17:37:52 INFO Master: I have been elected leader! New state: ALIVE
23/02/22 17:40:33 INFO Master: Registering worker 172.18.0.2:43915 with 1 cores, 1024.0 MiB RAM
23/02/22 17:46:28 INFO Master: Registering worker 172.18.0.3:43643 with 1 cores, 1024.0 MiB RAM
```

Open a scala shell and connect to the Spark cluster – hadoop1

```
# spark-shell --master spark://172.18.0.2:8090
```

Or

```
#spark-shell --conf spark.executor.memory=512mb --conf spark.executor.cores=1 --  
master spark://hadoop0:8090
```

Run an example job in the interactive scala shell

```
val myRange = spark.range(100).toDF("number")  
val divisBy2 = myRange.where("number % 2 = 0")  
divisBy2.count()
```

Check the application UI by navigating to <http://localhost:4040>. You should see the following

Spark Jobs (?)

User: root

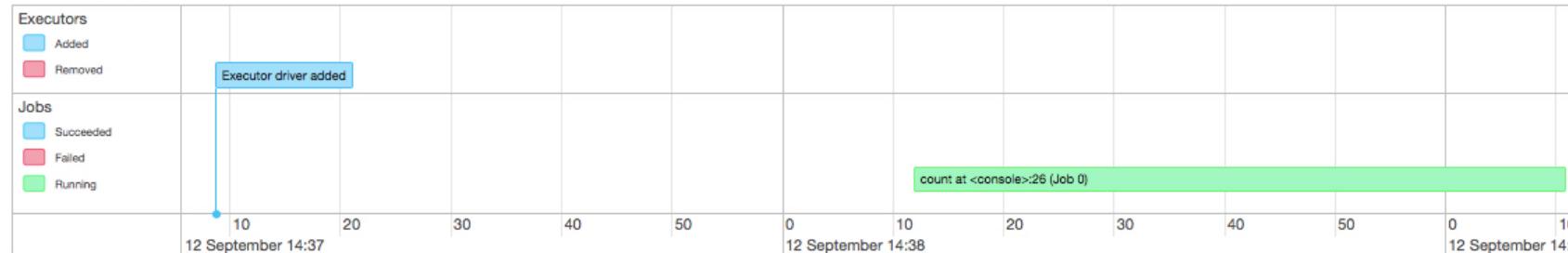
Total Uptime: 2.1 min

Scheduling Mode: FIFO

Active Jobs: 1

Event Timeline

Enable zooming



Active Jobs (1)

Page:

1 Pages. Jump to . Show items in a page.

Job Id ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	count at <console>:26 count at <console>:26 (kill)	2020/09/12 14:38:11	59 s	0/2	0/3

Page:

1 Pages. Jump to . Show items in a page.

The console should display the result as shown below:

```
Spark session available as 'spark'.
Welcome to

    __
   / \_ \
  \ V - V - \_ \_ \
 / \_ . \_, \_ / \_ \_ \
    / \_ \

```

version 3.0.1

Using Scala version 2.12.10 (Java HotSpot(TM) 64-Bit Server VM, Java 14.0.2)

Type in expressions to have them evaluated.

Type :help for more information.

```
scala> val myRange = spark.range(100).toDF("number")
myRange: org.apache.spark.sql.DataFrame = [number: bigint]
```

```
scala> val divisBy2 = myRange.where("number % 2 = 0")
divisBy2: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [number: bigint]
```

```
scala> divisBy2.count()
res0: Long = 50
```

```
scala> 
```

----- Lab Ends Here -----

3. Launching Spark Job on a YARN Cluster— 120 Minutes (D)

In this lab, you will deploy Spark Application on Hadoop Cluster.

By now, you should have two nodes hadoop cluster : ***hadoop0*** and ***hadoop1***.

You can run a MapReduce job on YARN in a pseudo-distributed mode

Start HDFS and YARN on the YARN Node.

At this point, you should have services and processes as shown below:

On *hadoop0*:

jps

2321 ResourceManager

1075 DataNode

1524 SecondaryNameNode

2632 NodeManager

761 NameNode

On hadoop1

jps

883 NodeManager

931 Jps

332 DataNode

Browse the web interface for the ResourceManager; by default it is available at:

ResourceManager –

<http://localhost:8088/>

Cluster Metrics																					
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Vcores Used	Vcores Total	Vcores Reserved											
0	0	0	0	0	0 B	12 GB	0 B	0	16	0											
Cluster Nodes Metrics																					
Active Nodes		Decommissioning Nodes		Decommissioned Nodes		Lost Nodes		Unhealthy Nodes		Rebooted Nodes											
2	0		0		0	0	0	0	0	0											
Scheduler Metrics																					
Scheduler Type		Scheduling Resource Type		Minimum Allocation		Maximum Allocation		Maximum Cluster Application Priority													
Capacity Scheduler		[memory-mb (unit=Mi), vcores]		<memory:1024, vCores:1>		<memory:8192, vCores:4>		0													
Show 20 entries																					
Search:																					
Node Labels	Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Allocation Tags	Mem Used	Mem Avail	Vcores Used	Vcores Avail	Version								
/default-rack	RUNNING	hadoop1:39505	hadoop1:8042		Fri Feb 24 14:46:49 +0530 2023		0		0 B	4 GB	0	8	3.1.2								
/default-rack	RUNNING	hadoop0:42991	hadoop0:8042		Fri Feb 24 14:46:29 +0530 2023		0		0 B	8 GB	0	8	3.1.2								

Showing 1 to 2 of 2 entries

First Previous 1 Next Last

You should have two Nodemanager as shown above.

Let us create a temporary folder, that will be used for working space for the YARN cluster.

hdfs dfs -mkdir /tmp

hdfs dfs -chmod -R 1777 /tmp

hdfs dfs -mkdir /tmp/in

hdfs dfs -ls /tmp

You can get the README.md file from the installation folder, which is provided along with the training. Copy the file on the hdfs cluster.

```
hdfs dfs -copyFromLocal /opt/hadoop/README.txt /tmp/in  
hdfs dfs -ls /tmp/in
```

```
#hdfs dfs -mkdir /tmp/spark-events
```

Congrats! You have successfully configure Yarn Cluster for spark job.

Let us copy the program, ***simple-project_2.12-1.0.jar*** to the spark node (***hadoop1***). It was developed earlier.

Create the following file in /opt on hadoop node and copy the file to hdfs /user/root/input folder.

File **names.json** contains:

```
{"firstName":"Grace","lastName":"Hopper"}  
 {"firstName":"Alan","lastName":"Turing"}  
 {"firstName":"Ada","lastName":"Lovelace"}  
 {"firstName":"Charles","lastName":"Babbage"}
```

```
#hdfs dfs -mkdir /user/root/input  
# hdfs dfs -copyFromLocal /opt/names.json /user/root/input/
```

You can verify the file using the following command:

```
# hdfs dfs -ls -R /user/root/input
```

Output:

```
[root@hadoop1 logs]# hdfs dfs -ls -R /user/root/input  
-rw-r--r--  2 root supergroup    171 2023-02-24 14:56 /user/root/input/names.json
```

Open a console on **hadoop1** to execute the Spark jobs to submit on YARN.

```
#export HADOOP_CONF_DIR=/opt/hadoop/etc/hadoop  
#export YARN_CONF_DIR=/opt/hadoop/etc/hadoop  
#spark-submit --class "NameList" --master yarn --deploy-mode cluster  
/Software/simple-project_2.12-1.0.jar \  
hdfs://hadoop0:8020/user/root/input hdfs://hadoop0:8020/user/root/output
```

Note --> Application Jar should be in the local file system, In folder is in HDFS and Out should not be present in the HDFS,it will be automatically created.

You can verify the progress of the application using <http://hp.com:8088/cluster>

Click on Cluster-->Application --> accepted

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Vcores Used	Vcores Total	Vcores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Re
3	0	1	2	1	1 GB	8 GB	0 B	1	8	0	1	0	0	0	0
Show 20 ▾ entries Search:															
ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking					
application_1434297324587_0003	root	com.ht.sparks.WordCount	SPARK	default	Sun, 14 Jun 2015 17:55:58 GMT	N/A	ACCEPTED	UNDEFINED	<div style="width: 0%; height: 10px; background-color: #ccc;"></div>	UNASSI					
application_1434297324587_0002	root	com.ht.sparks.WordCount	SPARK	default	Sun, 14 Jun 2015 16:27:58	Sun, 14 Jun 2015 16:30:02	FINISHED	SUCCEEDED	<div style="width: 100%; height: 10px; background-color: #5cb85c;"></div>	History					

Wait for few moment and verify on Running tab.

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebo Nod
3	0	1	2	4	7 GB	8 GB	0 B	4	8	0	1	0	0	0	0
Show 20 ▾ entries														Search:	
ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI					
application_1434297324587_0003	root	com.ht.sparks.WordCount	SPARK	default	Sun, 14 Jun 2015 17:55:58 GMT	N/A	RUNNING	UNDEFINED	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	ApplicationMa					

Showing 1 to 1 of 1 entries

First Previous 1 Next Last

Click on the Application ID --> Logs

The screenshot shows the Hadoop Cluster UI at hp.com:8088/cluster/app/application_1434297324587_0003. The page is titled "hadoop".

Application Overview:

User:	root
Name:	com.ht.sparks.WordCount
Application Type:	SPARK
Application Tags:	
State:	RUNNING
FinalStatus:	UNDEFINED
Started:	14-Jun-2015 23:25:58
Elapsed:	3mins, 57sec
Tracking URL:	ApplicationMaster
Diagnostics:	

Application Metrics:

Total Resource Preempted:	<memory:0, vCores:0>
Total Number of Non-AM Containers Preempted:	0
Total Number of AM Containers Preempted:	0
Resource Preempted from Current Attempt:	<memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt:	0
Aggregate Resource Allocation:	701453 MB-seconds, 459 vcore-seconds

ApplicationMaster:

Attempt Number	Start Time	Node	Logs
1	14-Jun-2015 23:25:58	hp.com:8042	logs

Logs --> Stderr to get details as follows:

The screenshot shows a web browser window with the URL http://hp.com:8042/node/containerlogs/container_1434297324587_0003_01_000001/root. The page title is "Logs for container_143429732458". On the left, there is a sidebar with links for ResourceManager (RM Home), NodeManager, and Tools. The main content area displays log entries:
stderr : Total file length is 16271 bytes.
stdout : Total file length is 0 bytes.

After sometimes click on Finished, after the console exit the program.

The screenshot shows a web browser window with the URL <http://hp.com:8088/cluster/apps/FINISHED>. The page title is "FINISHED Applications". On the left, there is a sidebar with links for Cluster (About Nodes, Applications: NEW, NEW SAVING, SUBMITTED, ACCEPTED, RUNNING, FINISHED, FAILED, KILLED), Cluster Metrics, and a table showing cluster metrics. The main content area displays a table of finished applications:
application_1434297324587_0003 root com.ht.sparks.WordCount SPARK default Sun, 14 Jun 2015 17:55:58 Sun, 14 Jun 2015 18:00:50 FINISHED SUCCEEDED History

On the job submit console.

```
2021-01-22 16:36:11,589 INFO yarn.Client: Application report for application_1611331123305_0002 (state: RUNNING)
2021-01-22 16:36:12,662 INFO yarn.Client: Application report for application_1611331123305_0002 (state: RUNNING)
2021-01-22 16:36:13,765 INFO yarn.Client: Application report for application_1611331123305_0002 (state: RUNNING)
2021-01-22 16:36:14,896 INFO yarn.Client: Application report for application_1611331123305_0002 (state: RUNNING)
2021-01-22 16:36:16,011 INFO yarn.Client: Application report for application_1611331123305_0002 (state: RUNNING)
2021-01-22 16:36:17,296 INFO yarn.Client: Application report for application_1611331123305_0002 (state: RUNNING)
2021-01-22 16:36:18,725 INFO yarn.Client: Application report for application_1611331123305_0002 (state: RUNNING)
2021-01-22 16:36:19,914 INFO yarn.Client: Application report for application_1611331123305_0002 (state: RUNNING)
2021-01-22 16:36:21,041 INFO yarn.Client: Application report for application_1611331123305_0002 (state: FINISHED)
2021-01-22 16:36:21,401 INFO yarn.Client:
    client token: N/A
    diagnostics: N/A
    ApplicationMaster host: hadoop0
    ApplicationMaster RPC port: 38603
    queue: default
    start time: 1611333080086
    final status: SUCCEEDED
    tracking URL: http://hadoop0:8088/proxy/application_1611331123305_0002/
    user: root
2021-01-22 16:36:23,265 INFO util.ShutdownHookManager: Shutdown hook called
2021-01-22 16:36:23,442 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-7b20b446-2b75-4550-9e2d-c84ad6
7c9b3b
2021-01-22 16:36:23,532 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-89770943-febc-4bdb-bf6e-404103
88026d
[root@303b68143644 bin]#
```

You can verify the output file in the hdfs as shown below:

```
#hdfs dfs -ls -R /user/root/output
```

Substitute the file as listed from the above command.

```
#hdfs dfs -cat /user/root/output/part-00000-f9b9bf86-3d82-4341-b167-7a237950c754-c000.csv
```

```
[root@hadoop0 opt]# hdfs dfs -ls -R /user/root/output
-rw-r--r-- 1 root supergroup          0 2021-01-22 16:36 /user/root/output/_SUCCESS
-rw-r--r-- 1 root supergroup      73 2021-01-22 16:36 /user/root/output/part-00000-f9b9bf86-3d82-4341-b167-7a237950c754-c000.csv
[root@hadoop0 opt]# hdfs dfs -cat /user/root/output/part-00000-f9b9bf86-3d82-4341-b167-7a237950c754-c000.csv
firstName,lastName
Grace,Hopper
Alan,Turing
Ada,Lovelace
Charles,Babbage
[root@hadoop0 opt]#
```

Great! You have successfully executed spark job in YARN cluster. Let us verify the output now using HDFS browser.

<http://localhost:9870/explorer.html#/>

Click Utilities --> Browse the file system → /user/root/output (Enter in the Directory) - Go

Browse Directory

/user/root/output								Go!			
Show 25 entries								Search:			
	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name			
<input type="checkbox"/>	-rw-r--r--	root	supergroup	0 B	Jan 22 22:06	1	128 MB	_SUCCESS			
<input type="checkbox"/>	-rw-r--r--	root	supergroup	73 B	Jan 22 22:06	1	128 MB	part-00000-f9b9bf86-3d82-4341-b167-7a237950c754-c000.csv			

Showing 1 to 2 of 2 entries

Previous 1 Next

Hadoop, 2021.

Click on the above mark file.

Browse Directory

/tmp/out						Go!
Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	root	supergroup	0 B	3	128 MB	_SUCCESS
-rw-r--r--	root	supergroup	60.71 KB	3	128 MB	part-00000
-rw-r--r--	root	supergroup	61.17 KB	3	128 MB	part-00001
-rw-r--r--	root	supergroup	60.76 KB	3	128 MB	part-00002
-rw-r--r--	root	supergroup	57.2 KB	3	128 MB	part-00003

Let us view one of the output. click on part-0001 --> Download.
Replace the hostname with localhost in the url if you can't access the hostname.

```
--+---1---+---2---+---3---+---4---+---5---+---6---+---7---+---8---+---9---+
1 (Kakrafoon,,1)
2 (pitifully,1)
3 (mattered,2)
4 (propded,1)
5 (House,2)
6 (bone,8)
7 ("Conceited,1)
8 (afternoon's,1)
9 (screen.",7)
10 (five.,1)
11 (gaping,2)
12 (tents,1)
13 (2~Imports:,1)
14 (envelope,1)
15 (tone.,4)
16 (consideration,,1)
17 (conclusively,3)
18 (activation.,1)
^~
```

Congrats!.

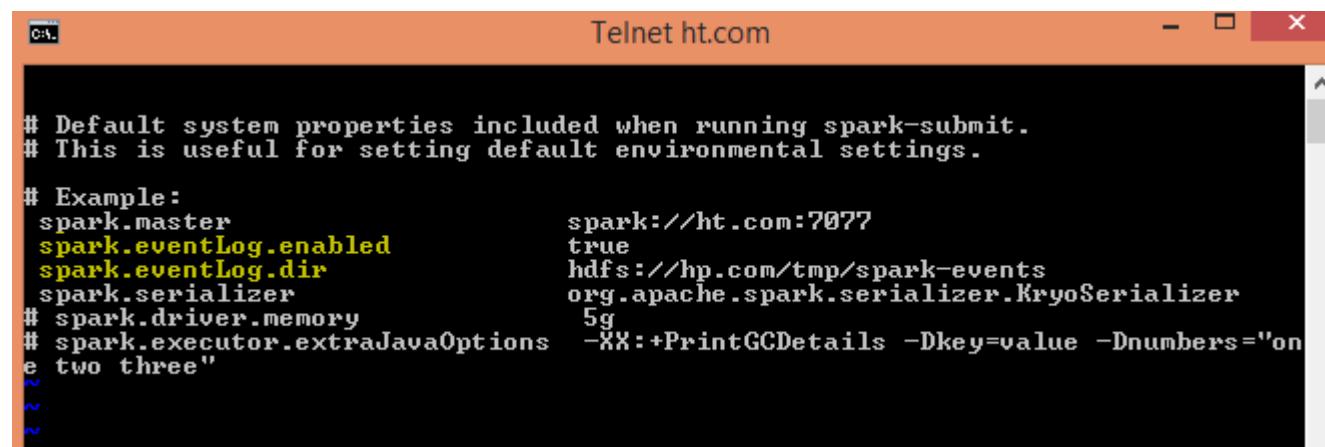
----- End of Lab -----

Errata:

1) 15/06/14 21:50:46 INFO storage.BlockManagerMasterActor: Registering block manager ht.com:50475 with 267.3 MB RAM, BlockManagerId(<driver>, ht.com, 50475)
15/06/14 21:50:46 INFO storage.BlockManagerMaster: Registered BlockManager
Exception in thread "main" java.lang.IllegalArgumentException: Wrong FS: file:/tmp/spark-events/application_1434297324587_0001.inprogress, expected: hdfs://hp.com at
org.apache.hadoop.fs.FileSystem.checkPath(FileSystem.java:643)
at
org.apache.hadoop.hdfs.DistributedFileSystem.getPathName(DistributedFileSystem.java:191)
at
org.apache.hadoop.hdfs.DistributedFileSystem.access\$ooo(DistributedFileSystem.java:102)
at
org.apache.hadoop.hdfs.DistributedFileSystem\$22.doCall(DistributedFileSystem.java:1266)
at
org.apache.hadoop.hdfs.DistributedFileSystem\$22.doCall(DistributedFileSystem.java:1262)
at org.apache.hadoop.fs.FileSystemLinkResolver.resolve(FileSystemLinkResolver.java:81)
at
org.apache.hadoop.hdfs.DistributedFileSystem.setPermission(DistributedFileSystem.java:1262)
at org.apache.spark.scheduler.EventLoggingListener.start(EventLoggingListener.scala:128)

```
15/06/14 21:50:46 INFO storage.BlockManagerMasterActor: Registering block manager ht.com:50475 with 267.3 MB RAM, BlockManagerId<driver>, ht.com, 50475
15/06/14 21:50:46 INFO storage.BlockManagerMaster: Registered BlockManager
Exception in thread "main" java.lang.IllegalArgumentException: Wrong FS: file:/tmp/spark-events/application_1434297324587_0001.inprogress, expected: hdfs://hp.com
        at org.apache.hadoop.fs.FileSystem.checkPath(FileSystem.java:643)
        at org.apache.hadoop.hdfs.DistributedFileSystem.getPathName(DistributedFileSystem.java:191)
        at org.apache.hadoop.hdfs.DistributedFileSystem.access$000(DistributedFileSystem.java:102)
        at org.apache.hadoop.hdfs.DistributedFileSystem$22.doCall(DistributedFileSystem.java:1266)
        at org.apache.hadoop.hdfs.DistributedFileSystem$22.doCall(DistributedFileSystem.java:1262)
        at org.apache.hadoop.fs.FileSystemLinkResolver.resolve(FileSystemLinkResolver.java:81)
        at org.apache.hadoop.hdfs.DistributedFileSystem.setPermission(DistributedFileSystem.java:1262)
        at org.apache.spark.scheduler.EventLoggingListener.start(EventLoggingListener.scala:128)
        at org.apache.spark.SparkContext.<init>(SparkContext.scala:399)
        at org.apache.spark.api.java.JavaSparkContext.<init>(JavaSparkContext.scala:61)
```

Solution: Verify default configuration of the spark installation. /spark/spark-1.3.0-bin-hadoop2.4/conf/spark-defaults.conf



```
# Default system properties included when running spark-submit.
# This is useful for setting default environmental settings.

# Example:
spark.master          spark://ht.com:7077
spark.eventLog.enabled true
spark.eventLog.dir    hdfs://hp.com/tmp/spark-events
spark.serializer      org.apache.spark.serializer.KryoSerializer
# spark.driver.memory   5g
# spark.executor.extraJavaOptions -XX:+PrintGCDetails -Dkey=value -Dnumbers="one two three"
~
```

Check that spark.eventLog.dir is begin with hdfs:// and the /tmp/spark-events is already created in HDFS system

You can verify on the YARN cluster only.

Verify using : hadoop fs -ls -R /tmp/

```
[root@hp ~]# hadoop fs -ls -R /tmp/
Java HotSpot(TM) Client VM warning: You have loaded library /YARN/hadoop-2.6.0/lib/native/libhadoop.so.1.0.0 which might have disabled stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with '-z noexecstack'.
15/06/14 22:04:11 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
drwxr-xr-x  - root supergroup          0 2015-06-14 16:25 /tmp/in
-rw-r--r--  1 root supergroup  3629 2015-06-14 16:25 /tmp/in/README.md
drwxr-xr-x  - root supergroup          0 2015-06-14 17:47 /tmp/out
-rw-r--r--  3 root supergroup          0 2015-06-14 17:47 /tmp/out/_SUCCESS
-rw-r--r--  3 root supergroup  1956 2015-06-14 17:47 /tmp/out/part-00000
-rw-r--r--  3 root supergroup  1717 2015-06-14 17:47 /tmp/out/part-00001
drwxr-xr-x  - root supergroup          0 2015-06-14 17:50 /tmp/out1
-rw-r--r--  3 root supergroup          0 2015-06-14 17:50 /tmp/out1/_SUCCESS
-rw-r--r--  3 root supergroup  1956 2015-06-14 17:50 /tmp/out1/part-00000
-rw-r--r--  3 root supergroup  1717 2015-06-14 17:50 /tmp/out1/part-00001
drwxr-xr-x  - root supergroup          0 2015-06-14 17:52 /tmp/out3
-rw-r--r--  3 root supergroup          0 2015-06-14 17:52 /tmp/out3/_SUCCESS
-rw-r--r--  3 root supergroup  1956 2015-06-14 17:52 /tmp/out3/part-00000
-rw-r--r--  3 root supergroup  1717 2015-06-14 17:52 /tmp/out3/part-00001
drwxr-xr-x  - root supergroup          0 2015-06-14 21:59 /tmp/out5
-rw-r--r--  3 root supergroup          0 2015-06-14 21:59 /tmp/out5/_SUCCESS
-rw-r--r--  3 root supergroup  1956 2015-06-14 21:59 /tmp/out5/part-00000
-rw-r--r--  3 root supergroup  1717 2015-06-14 21:59 /tmp/out5/part-00001
drwxr-xr-x  - root supergroup          0 2015-06-14 21:58 /tmp/spark-events
-rw-rwx---  3 root supergroup  20287 2015-06-14 21:59 /tmp/spark-events/application_1434297324587_0002.inprogress
```

2) 15/06/14 00:41:20 INFO client.RMProxy: Connecting to ResourceManager at /o.o.o.o:8032

15/06/14 21:58:27 INFO yarn.Client: Application report for application_1434297324587_0002
(state: ACCEPTED)

Indefinite loop of above

Solution: Ensure that all configuration file are map to appropriate hostname and hostname to ip details are modified in the hosts file , Increase the resources and wait for few minutes to convert the status to running.

Yarn-site.xml

yarn.scheduler.minimum-allocation-mb: 256m
yarn.scheduler.increment-allocation-mb: 256m

```
<property>
<name>yarn.nodemanager.resource.memory-mb</name>
<value>4096</value>
</property>
<property>
<name>yarn.scheduler.minimum-allocation-mb</name>
<value>512</value>
</property>
<property>
<name>yarn.scheduler.increment-allocation-mb</name>
<value>256</value>
</property>
```

yarn-site.xml – Spark Node. (Specify the hostname of the Hadoop RM Node as shown below)

```
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>hadoopo</value>
  <description>The hostname of the RM.</description>
</property>
<property>
  <name>yarn.resourcemanager.address</name>
  <value>hadoopo:8032</value>
  <description>The hostname of the RM.</description>
</property>
```

3) If unhealthy nodes is being displayed in the cluster URL with 1/1 local-dirs are bad:
/tmp/hadoop-yarn/nm-local-dir;

Solution: delete the folder using the following command and ensure that nodes are healthy before proceeding forward, you can restart if require

```
hadoop fs -rm -fr /tmp/hadoop-yarn/*
```

Try the following command:
yarn application -list

```
yarn application -kill [application name]
```

You can view jps in YARN cluster when spark job is executing. It will start Coarse* processes.

```
[root@hp ~]# jps
5082 DataNode
9633 CoarseGrainedExecutorBackend
9670 Jps
4980 NameNode
5345 NodeManager
9625 CoarseGrainedExecutorBackend
5029 SecondaryNameNode
9420 ApplicationMaster
5294 ResourceManager
9629 CoarseGrainedExecutorBackend
[root@hp ~]#
```

Even, cluster mode works

```
15/06/14 23:43:46 INFO yarn.Client: Application report for application_143429732
4587_0004 (state: FINISHED)
15/06/14 23:43:46 INFO yarn.Client:
  client token: N/A
  diagnostics: N/A
  ApplicationMaster host: hp.com
  ApplicationMaster RPC port: 0
  queue: default
  start time: 1434305512588
  final status: SUCCEEDED
  tracking URL: http://hp.com:8088/proxy/application_1434297324587_0004/
  user: root
[root@ht spark-1.3.0-bin-hadoop2.4]# ./bin/spark-submit --master yarn-cluster -
--class com.ht.sparks.WordCount --num-executors 3 --executor-cores 1 /spark/Sp
arkWC-0.0.1-SNAPSHOT.jar /tmp/in /tmp/out7
```

Issue:

```
UpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
21/01/23 05:38:08 INFO Client: Retrying connect to server: 0.0.0.0/0.0.0.0:8032. Already tried 2 time(s); retry policy is Retry
UpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
21/01/23 05:38:09 INFO Client: Retrying connect to server: 0.0.0.0/0.0.0.0:8032. Already tried 3 time(s); retry policy is Retry
UpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
21/01/23 05:38:10 INFO Client: Retrying connect to server: 0.0.0.0/0.0.0.0:8032. Already tried 4 time(s); retry policy is Retry
UpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
21/01/23 05:38:11 INFO Client: Retrying connect to server: 0.0.0.0/0.0.0.0:8032. Already tried 5 time(s); retry policy is Retry
UpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
21/01/23 05:38:12 INFO Client: Retrying connect to server: 0.0.0.0/0.0.0.0:8032. Already tried 6 time(s); retry policy is Retry
UpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
21/01/23 05:38:13 INFO Client: Retrying connect to server: 0.0.0.0/0.0.0.0:8032. Already tried 7 time(s); retry policy is Retry
UpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
21/01/23 05:38:14 INFO Client: Retrying connect to server: 0.0.0.0/0.0.0.0:8032. Already tried 8 time(s); retry policy is Retry
UpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
21/01/23 05:38:15 INFO Client: Retrying connect to server: 0.0.0.0/0.0.0.0:8032. Already tried 9 time(s); retry policy is Retry
UpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
21/01/23 05:38:15 INFO RetryInvocationHandler: java.net.ConnectException: Your endpoint configuration is wrong; For more details see: http://wiki.apache.org/hadoop/UnsetHostnameOrPort, while invoking ApplicationClientProtocolPBClientImpl.getClusterMetrics over null after 1 failover attempts. Trying to failover after sleeping for 42760ms.
```

Verify the ip of the resource manager and set the home directory appropriately.
Try updating the following setting in yarn-site.xml of Hadoop and spark node too..

```
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>hadoopo</value>
  <description>The hostname of the RM.</description>
</property>
<property>
```

```
<name>yarn.resourcemanager.address</name>
<value>hadoop:8032</value>
<description>The hostname of the RM.</description>
</property>
```

Error:

Failing this attempt. Diagnostics: [2022-06-18 03:01:08.210]Container
[pid=1686,containerID=container_1655521013018_0001_02_000001] is running 32197120B
beyond the 'VIRTUAL' memory limit. Current usage: 198.8 MB of 1 GB physical memory used; 2.1
GB of 2.1 GB virtual memory used. Killing container.

Add following property in yarn-site.xml

```
<property>
<name>yarn.nodemanager.vmem-check-enabled</name>
<value>false</value>
<description>Whether virtual memory limits will be enforced for containers</description>
</property>
<property>
<name>yarn.nodemanager.vmem-pmem-ratio</name>
<value>4</value>
<description>Ratio between virtual memory to physical memory when setting memory limits  
for containers</description>
</property>
```

Complete yarn-site.xml

```
<configuration>

<property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
</property>
<property>
    <name>yarn.nodemanager.env-whitelist</name>

    <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PREPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
</property>
<property>
    <name>yarn.resourcemanager.hostname</name>
    <value>master0</value>
    <description>The hostname of the RM.</description>
</property>
<property>
    <name>yarn.resourcemanager.address</name>
    <value>master0:8032</value>
```

```
    <description>The hostname of the RM.</description>
  </property>
<property>
  <name>yarn.nodemanager.resource.memory-mb</name>
  <value>4096</value>
</property>
<property>
  <name>yarn.scheduler.minimum-allocation-mb</name>
  <value>512</value>
</property>
<property>
  <name>yarn.scheduler.increment-allocation-mb</name>
  <value>256</value>
</property>
<property>
  <name>yarn.nodemanager.vmem-check-enabled</name>
  <value>false</value>
  <description>Whether virtual memory limits will be enforced for containers</description>
</property>
<property>
  <name>yarn.nodemanager.vmem-pmem-ratio</name>
  <value>4</value>
  <description>Ratio between virtual memory to physical memory when setting memory limits
for containers</description>
```

```
</property>  
</configuration>
```

Various ways of job submission.

```
./spark-submit --class "NameList" --master yarn --driver-memory 1G --executor-memory 1G --num-executors 2 --deploy-mode cluster /opt/data/simple-project_2.12-1.0.jar \  
hdfs://hadoop0:8020/user/root/input1 hdfs://hadoop0:8020/user/root/output7
```

```
./spark-submit --class "NameList" --master yarn --driver-memory 512m --executor-memory 512m --num-executors 2 --deploy-mode cluster /opt/data/simple-project_2.12-1.0.jar \  
hdfs://master0:8020/user/root/input hdfs://master0:8020/user/root/output7
```

```
./spark-submit --class "NameList" --master yarn --driver-memory 512m --executor-memory 512m --num-executors 2 --deploy-mode client /opt/data/simple-project_2.12-1.0.jar \  
hdfs://hadoop0:8020/user/root/input1 hdfs://hadoop0:8020/user/root/output7
```

or using spark-shell

Ensure that `HADOOP_CONF_DIR` or `YARN_CONF_DIR` points to the directory which contains the (client side) configuration files for the Hadoop cluster.

```
./bin/spark-shell --master yarn --deploy-mode client
```

4. Jobs Monitoring : Using Web UI. – 45 Minutes

Prerequisite:

Start two nodes Spark cluster. On the spark master terminal console, you should have the following two nodes entries.

```
23/02/24 16:15:39 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(root);  
groups with view permissions: Set(); users with modify permissions: Set(root); groups with modify permissions: Set()  
23/02/24 16:15:43 INFO Utils: Successfully started service 'sparkMaster' on port 8090.  
23/02/24 16:15:43 INFO Master: Starting Spark master at spark://172.18.0.2:8090  
23/02/24 16:15:43 INFO Master: Running Spark version 3.3.0  
23/02/24 16:15:45 INFO Utils: Successfully started service 'MasterUI' on port 8080.  
23/02/24 16:15:46 INFO MasterWebUI: Bound MasterWebUI to 0.0.0.0, and started at http://hadoop0:8080  
23/02/24 16:15:47 INFO Master: I have been elected leader! New state: ALIVE  
23/02/24 16:17:17 INFO Master: Registering worker 172.18.0.2:41451 with 1 cores, 1024.0 MiB RAM  
23/02/24 16:19:58 INFO Master: Registering worker 172.18.0.3:45775 with 1 cores, 1024.0 MiB RAM
```

Open a terminal on **hadoop1**:

Note down the ip of the hadoop or the spark master and substitute below accordingly.

```
#spark-shell --master spark://172.18.0.2:8090
```

Copy the input file in both the nodes (**/Software/data/**), if you are executing on the Standalone cluster.

File **names.json** contains:

```
{"firstName":"Grace","lastName":"Hopper"}  
 {"firstName":"Alan","lastName":"Turing"}  
 {"firstName":"Ada","lastName":"Lovelace"}  
 {"firstName":"Charles","lastName":"Babbage"}
```

The file should be in all the Executing node.

Execute the following jobs:

Enter the following commands one at a time on the spark shell.

```
val ip="/software/data/names.json"  
val op="/software/data/nameop"  
spark.sparkContext.setLogLevel("WARN")  
val peopleDF = spark.read.json(ip)  
val namesDF = peopleDF.select("firstName","lastName")  
namesDF.write.option("header","true").csv(op)
```

You can access the spark Web UI using the following URL.

<http://127.0.0.1:8080>

Click on Application ID → Application Detail UI

The screenshot shows the 'Application Detail UI' for an application named 'Spark shell'. The application ID is app-20201113095544-0006. It was submitted by root on 2020/11/13 at 09:55:44 and is currently RUNNING. The UI displays two executors, each with 1 core and 1024 MB of memory, running on workers 172.18.0.2 and 172.18.0.3 respectively. Logs for both executors are available at stdout and stderr.

ExecutorID	Worker	Cores	Memory	Resources	State	Logs
1	worker-20201113080544-172.18.0.2-38441	1	1024		RUNNING	stdout stderr
0	worker-20201113080133-172.18.0.3-46519	1	1024		RUNNING	stdout stderr

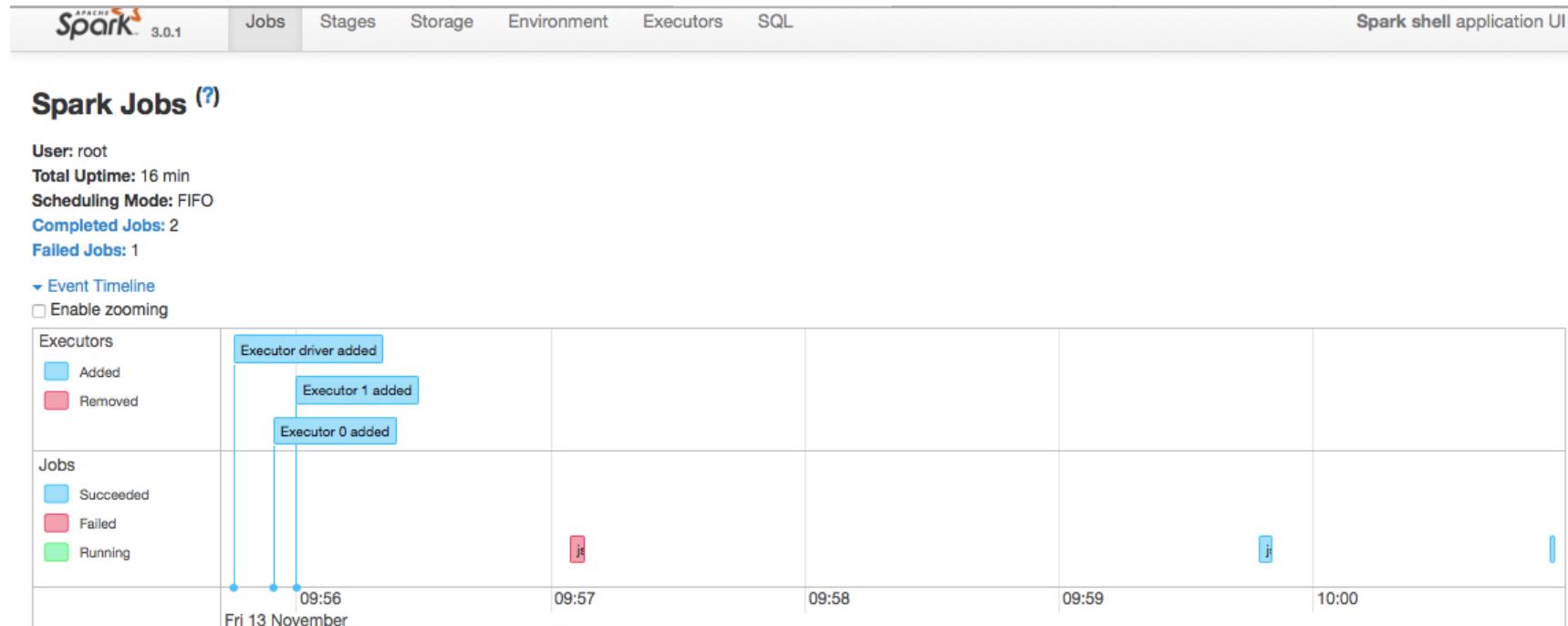
Or You can verify the Jobs details Using: <http://localhost:4040/jobs/>

web UI comes with the following tabs (which may not all be visible at once as they are lazily created on demand, e.g. Streaming tab):

- Jobs
- Stages
- Storage with RDD size and memory use

- Environment
- Executors
- SQL

The **Jobs Tab** shows [status of all Spark jobs](#) in a Spark application



Display the timeline of Executor being added in the Job. When the Job Get completed etc. In above, you can verify that 2 executors have been added.

When you hover over a job in Event Timeline not only you see the job legend but also the job is highlighted in the Summary section.

The Event Timeline section shows not only jobs but also executors.

You can verify the completed and failed jobs too:

- Completed Jobs (2)
Application: Spark shell

Page: 1 Pages. Jump to . Show items in a page.

Job Id ▲	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
1	json at <console>:25 json at <console>:25	2020/11/13 09:59:47	3 s	1/1	1/1
2	csv at <console>:28 csv at <console>:28	2020/11/13 10:00:55	1 s	1/1	1/1

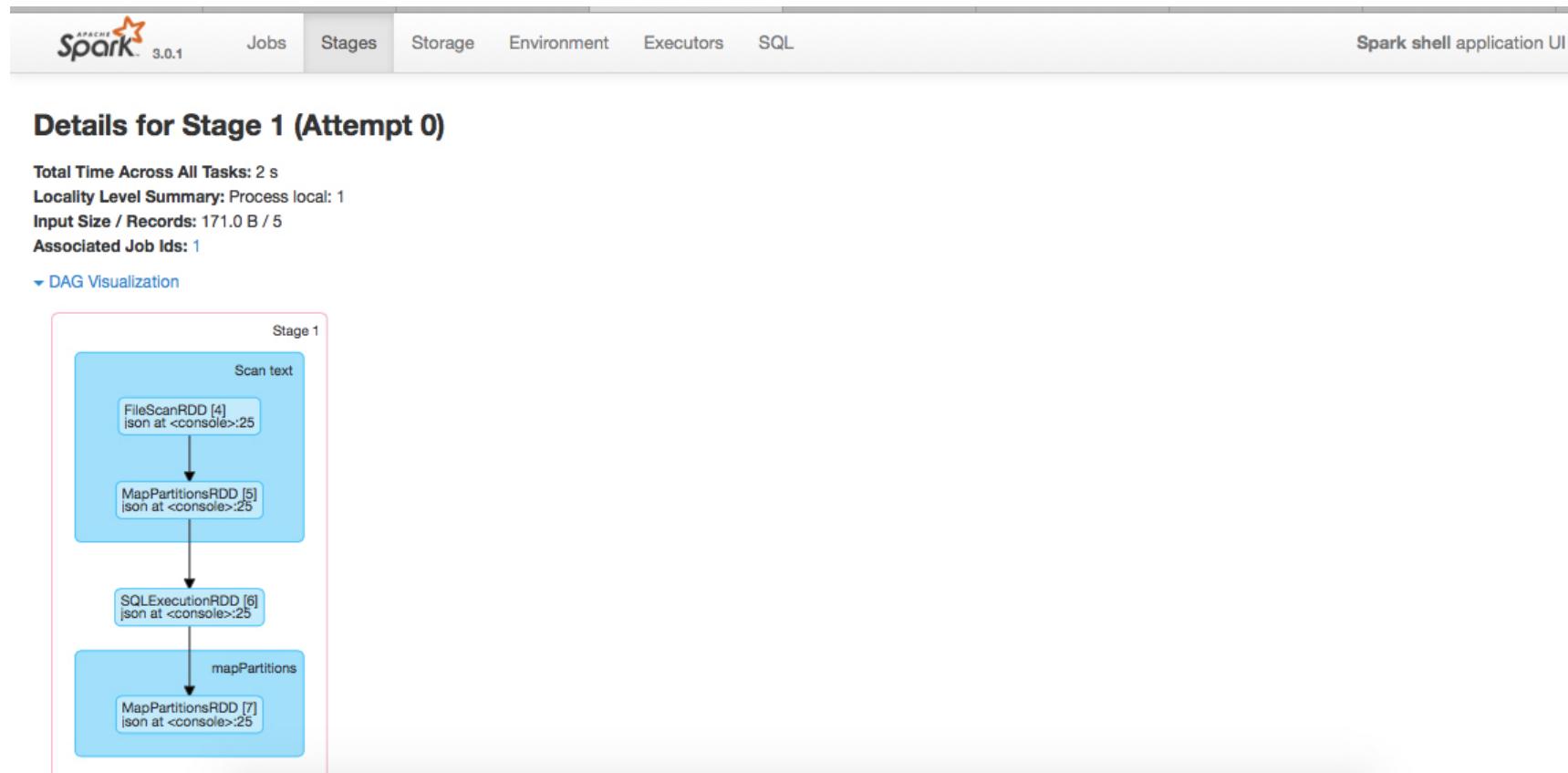
Page: 1 Pages. Jump to . Show items in a page.

- Failed Jobs (1)

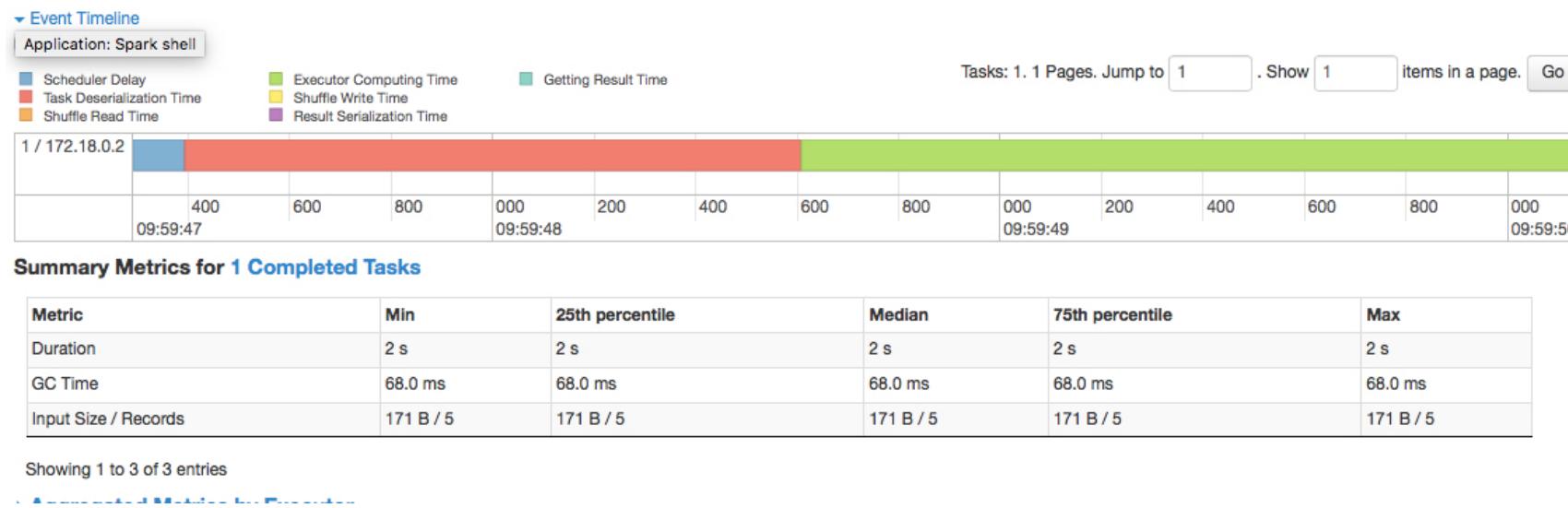
Page: 1 Pages. Jump to . Show items in a page.

Job Id ▼	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	json at <console>:25 json at <console>:25	2020/11/13 09:57:04	3 s	0/1 (1 failed)	0/1 (4 failed)

Click on On job -> for details on Description → stages to view DAG - When you click a job in All Jobs Page page, you see the **Details for Job** page.



It shows the DAG graph and the stage details.



Scroll down to view the task metrics: which task take majority of the time etc. How much byte consume or output by each task and its statistics.

Showing 1 to 1 of 1 entries

Application: Spark shell [Tasks by Executor](#)

Executor ID	Logs	Address	Task Time	Total Tasks	Failed Tasks	Killed Tasks	Succeeded Tasks	Blacklisted	Input Size / Records
1	stdout stderr	172.18.0.2:41887	3 s	1	0	0	1	false	171 B / 5

Showing 1 to 1 of 1 entries

[Previous](#) [1](#) [Next](#)

Tasks (1)

Show 20	entries	Search:										
Index	Task ID	Attempt	Status	Locality level	Executor ID	Host	Logs	Launch Time	Duration	GC Time	Input Size / Records	Errors
0	4	0	SUCCESS	PROCESS_LOCAL	1	172.18.0.2	stdout stderr	2020-11-13 15:29:47	2 s	68.0 ms	171 B / 5	

Executors Tab

Stats of each executor, how much time it spends on GC etc. How much data get shuffle?

Executors tab in web UI shows

Summary

RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write
Active(3) 1	23.4 KB / 1.3 GB	0.0 B	2	0	0	0	0	0 ms (0 ms)	0.0 B	0.0 B	0.0 B
Dead(4) 2	46.8 KB / 1.7 GB	0.0 B	4	0	0	7	7	1.0 min (6 s)	21 KB	0.0 B	0.0 B
Total(7) 3	70.2 KB / 3 GB	0.0 B	6	0	0	7	7	1.0 min (6 s)	21 KB	0.0 B	0.0 B

Executors

Show 20 ▾ entries

Search:

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)		Shuffle Input	Shuffle Read	Shuffle Write	Logs	Thread Dump	
											Time	GC						
driver	192.168.188.173:54112	Active	1	23.4 KB / 434 MB	0.0 B	0	0	0	0	0	0 ms (0 ms)	0	0.0 B	0.0 B	0.0 B		Thread Dump	
0	192.168.188.173:54931	Dead	1	23.4 KB / 434 MB	0.0 B	1	0	0	4	4	37 s (6 s)	9.5 KB	0.0 B	0.0 B	stdout	stderr	Thread Dump	
1	192.168.188.174:34320	Dead	1	23.4 KB / 434 MB	0.0 B	1	0	0	3	3	25 s (0.3 s)	11.5 KB	0.0 B	0.0 B	stdout	stderr	Thread Dump	
2	192.168.188.174:46874	Dead	0	0.0 B / 434 MB	0.0 B	1	0	0	0	0	0 ms (0 ms)	0	0.0 B	0.0 B	0.0 B	stdout	stderr	Thread Dump
3	192.168.188.173:34382	Dead	0	0.0 B / 434 MB	0.0 B	1	0	0	0	0	0 ms (0 ms)	0	0.0 B	0.0 B	0.0 B	stdout	stderr	Thread Dump
4	192.168.188.174:44548	Active	0	0.0 B / 434 MR	0.0 B	1	0	0	0	0	0 ms (0 ms)	0	0.0 B	0.0 B	0.0 B	stdout	stderr	Thread Dump

Click on SQL tab

Application: Spark shell

Query 0

Submitted Time: 2020/11/13 10:00:55

Duration: 2 s

Succeeded Jobs: 2

Show the Stage ID and Task ID that corresponds to the max metric

Scan json

number of files read: 1
metadata time: 0 ms
size of files read: 171.0 B
number of output rows: 4



Execute InsertIntoHadoopFsRelationCommand

number of written files: 1
written output: 73.0 B
number of output rows: 4
number of dynamic part: 0

[- Details](#)

Verify the plan

```
▼ Details
  Application: Spark shell
  Analyzed Logical Plan
InsertIntoHadoopFsRelationCommand file:/software/nameop, false, CSV, Map(header -> true, path -> /software/nameop), ErrorIfExists, [firstName, lastName]
+- Project [firstName#14, lastName#15]
  +- Relation[firstName#14,lastName#15] json

== Analyzed Logical Plan ==

InsertIntoHadoopFsRelationCommand file:/software/nameop, false, CSV, Map(header -> true, path -> /software/nameop), ErrorIfExists, [firstName, lastName]
+- Project [firstName#14, lastName#15]
  +- Relation[firstName#14,lastName#15] json

== Optimized Logical Plan ==
InsertIntoHadoopFsRelationCommand file:/software/nameop, false, CSV, Map(header -> true, path -> /software/nameop), ErrorIfExists, [firstName, lastName]
+- Relation[firstName#14,lastName#15] json

== Physical Plan ==
Execute InsertIntoHadoopFsRelationCommand file:/software/nameop, false, CSV, Map(header -> true, path -> /software/nameop), ErrorIfExists, [firstName, lastName]
+- FileScan json [firstName#14,lastName#15] Batched: false, DataFilters: , Format: JSON, Location: InMemoryFileIndex[file:/software/names.json], PartitionFilters: , PushedFilters: , ReadSchema: struct<firstName:string,lastName:string>
```

----- Lab Ends Here