**Spark Streaming:**

#### Intro:

Spark Streaming is an extension of the core Spark API that enables scalable, high-throughput, fault-tolerant stream processing of live data streams. This lab will cover the followings:

1. Pre-Requirements to start Spark Streaming
2. How to generate streaming data using Python
3. How to Access & Initialize dependencies for Spark Streaming on Spark shell
4. Files live stream example using Spark Streaming from Spark Shell

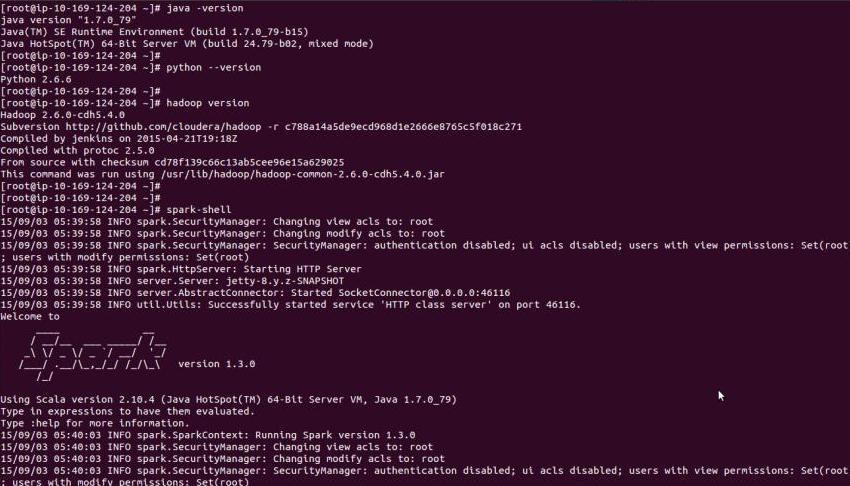
Let’s go!

#### Step-1. Pre-Requirements to start Spark Streaming:

You must have the followings on your server to start Spark Streaming.

1. Java 1.6+
2. Hadoop 1.X or 2.X - optional (for file storage or we can use local file system also)
3. Spark 1.X
4. Data source (we will be using file stream)
5. Python

Please check on your server if these exist. You can refer to the following Unix log for the exact code.



#### Step-2. How to generate streaming data using Python:

Here is the small snippet of code to generate data using python.

1. Create a python file name “generate\_data.py” to generate live streaming data. Make sure there is no error copying and pasting this code to your script.

#!/usr/bin/python

import threading

import time

import string

import random

import os

import uuid

class Generate\_events(threading.Thread):

def \_\_init\_\_(self, events\_count, file\_name):

threading.Thread.\_\_init\_\_(self)

if os.path.exists(file\_name):

self.file\_name = open(file\_name, 'a')

else:

self.file\_name = open(file\_name, 'w')

self.events\_count = events\_count

def \_\_gen\_sal(self):

while True:

sal = range(1000000)

yield random.choice(sal)

def \_\_gen\_emp\_number(self):

while True:

sal = range(1000000)

yield random.choice(sal)

def \_\_gen\_bonus(self):

while True:

bonus = range(500000)

yield random.choice(bonus)

def \_\_gen\_name(self):

alphabets = list(string.ascii\_lowercase)

while True:

yield ''.join(random.choice(alphabets) for \_ in range(6))

def \_get\_name(self):

return self.\_\_gen\_name().next()

def \_get\_sal(self):

return self.\_\_gen\_sal().next()

def \_get\_bonus(self):

return self.\_\_gen\_bonus().next()

def \_get\_emp\_number(self):

return self.\_\_gen\_emp\_number().next()

def run(self):

try:

for event in range(self.events\_count):

self.file\_name.write("\n" + str(self.\_get\_emp\_number()) + ',' + self.\_get\_name() + ',' + str(self.\_get\_sal()) + ',' + str(self.\_get\_bonus()))

self.file\_name.close()

except Exception, e:

print e

while(True):

#id = os.urandom(32)

id = uuid.uuid4().int & (1<<64)-1

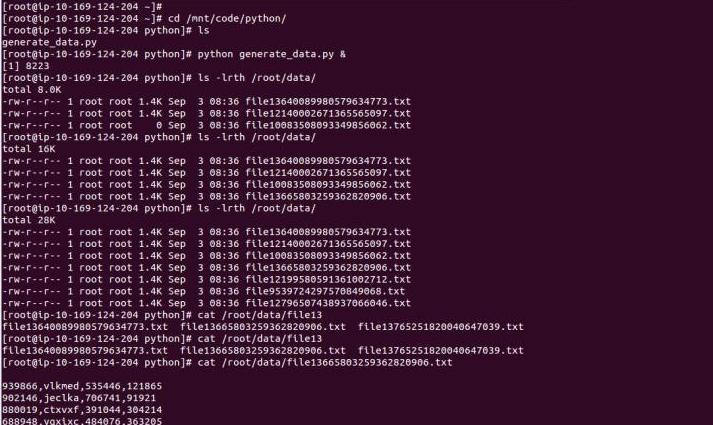
file = 'file'+str(id)+'.txt'

Generate\_events(50, '/root/data/'+file).start()

time.sleep(5)

1. Create a folder call “data” in /root/ directory
2. Input Data set contains these columns “emp\_number,name,salary,bonus,filename”
3. You can run this python file using this command “python generate\_data.py”
4. You can find result data in /root/data/. You can use this command to see your data “ls -lrth /root/data/”. Don’t keep running this script as it can fill up your space quickly.

Please refer to the Unix log of these steps.



#### Step-3. How to Access and Initialize Dependencies for Spark Streaming on Spark shell:

**How to Access:**

You can access spark shell using below command on terminal

spark-shell

You can refer to this Unix log for the same.



#### How to build Dependencies for Spark Streaming :

Below listed imports statements are necessary for Spark Streaming. You can directory use these commands and run in Spark-Shell.

import org.apache.hadoop.io.\_

import org.apache.hadoop.mapred.OutputFormat

import org.apache.spark.\_

import org.apache.spark.streaming.\_

import org.apache.spark.streaming.StreamingContext.\_

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat

You can refer to this Unix log for the same.



#### How to Initialize Spark Streaming:

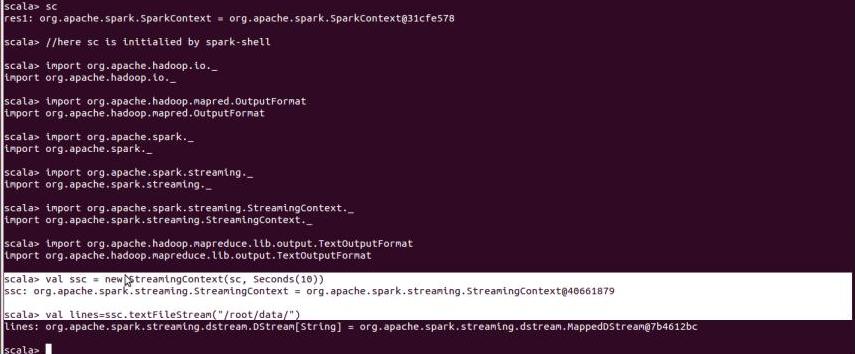
You can use the following commands to do the same.

val ssc = new StreamingContext(sc, Seconds(10))

val lines=ssc.textFileStream(“/root/data/”)

Here sc is sparkContext, which has been initialized by spark-shell.

You can refer below the Unix log for the same.



#### Step-4. Live streaming data processing example using Spark Streaming on Spark-Shell :

**on Spark shell:**

**Please follow the following steps:**

* Start the data generate process using this command  
   python generate\_data.py & (refer to step-2)
* Make a folder using “mkdir /root/outputdata” to store output results

The below code simply adds run time the salaries and bonuses and create a employee monthly income result set. You can use the following code on Spark-Shell and execute.

import org.apache.hadoop.io.\_

import org.apache.hadoop.mapred.OutputFormat

import org.apache.spark.\_

import org.apache.spark.streaming.\_

import org.apache.spark.streaming.StreamingContext.\_

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat

val ssc = new StreamingContext(sc, Seconds(10))

val lines=ssc.textFileStream("file:///root/data/")

val total= lines.map(line=>if(line.contains(",")){(line+","+(line.split(",")(2)).toLong+(line.split(",")(3)).toLong,null)})

total.saveAsTextFiles("file:///root/outputdata/","output")

ssc.start()

ssc.awaitTermination();

Once “sss.start ” processes started on spark shell, you can see result on “/root/outputdata” folder. You might want to open two termnials to see data generation process and spark streaming real time calculations.

You can refer to the following Unix log for the same. Also, please check in your output folder for the output results.

