Assignment 2
Group 10: Shweta Anchan & Srinath Sridhar

1. **Kafka wordcount- Scala** (requires spark 1.4.0)

    - Install Kafka through the following link
      http://apache.mirrors.pair.com/kafka/0.8.2.1/kafka_2.11-0.8.2.1.tgz

    - Untar this and cd into the folder

      The following should be done each on different terminals to see the results.

    - Start zookeeper

      ```
      $ bin/zookeeper-server-start.sh config/zookeeper.properties
      ```

      Zookeeper starts at localhost:2181

    - Start Kafka Broker

      ```
      $ bin/kafka-server-start.sh config/server.properties
      ```

      KafkaBroker starts on localhost:9092

    - Create Kafka Topic

      ```
      $ bin/kafka-topic.sh --create --zookeeper localhost:2181 --
      replication-factor 1 --partitions 1 --topic kafkatopic
      ```

      Creates a topic by name Kafkatopic

    - Start a Producer

      ```
      $ bin/kafka-console-producer.sh --broker-list localhost:9092 --
      topic kafkatopic
      ```

    - Go to your spark folder and run the following

      ```
      $ bin/run-example
      org.apache.spark.examples.streaming.KafkaWordCount localhost:2181
      my-consumer-group kafkatopic
      ```

2. Flume wordcount- Scala

- Use the command to download the compressed flume file onto EMR.

```
wget http://www.apache.org/dyn/closer.cgi/flume/1.6.0/apache-flume-1.6.0-bin.tar.gz
```

- Then use the compressed file to untar

```
tar -xvf apache-flume-1.6.0-bin.tar.gz
```

- You can remove the compressed file using the command

```
rm -rf apache-flume-1.6.0-bin.tar.gz
```

- Now, you change into the flume's directory

```
cd apache-flume-1.6.0-bin
```

- Create a config file inside the directory by using the command

```
touch filename.conf
```

- The information about the conf file can be seen here

  https://github.com/abhinavg6/spark-flume-stream

- Then inside the config file, give out the details of the sources, sink and channels.

- Once the *conf* file is ready, run the following command:

```
bin/flume-ng agent --conf conf --conf-file filename.conf --name agent_name mentioned in conf file -Dflume.root.avro=INFO,console
```

  This gets the flume server up and running.

- Now, open another EMR terminal and type:

```
telnet localhost 12345
```
--(same number as the one given in conf file under source)

- Again, open another terminal and go to Spark 1.4.0 folder and type the command

```
bin/run-example org.apache.spark.examples.streaming.FlumeEventCount localhost 54321
```

Note: The *conf* has been attached as *avro.conf*.

3. Kinesis- clickstream analysis

- Go to the Spark 1.4.1 folder in your EMR and type

```
bin/run-example
org.apache.spark.examples.streaming.clickstream.PageViewGenerator
44444 10
```

  This starts the page generator

- Then type the command

```
bin/run-example
org.apache.spark.examples.streaming.clickstream.PageViewStream
errorRatePerZipCode localhost 44444
```

  This line processes the generated stream.

4. HDFS wordcount- Scala

- Creating a S3 bucket:
  Go to the link given below and follow the instructions given on the page to create a s3 bucket.

  https://console.aws.amazon.com/s3

  Then click on the properties -> permission to make the bucket "public"

- Once a bucket is created, you can place the files from your local machine to s3.

- Open the EMR cluster and type the command. This copies the text file from the S3 bucket to the EMR instance.

  ```
  aws s3 cp s3://my_bucket/my_folder/my_file.txt my_copied_file.txt
  ```

- Now use the command hadoop fs -mkdir /dirname to create a new directory.

  ```
  hadoop fs –put my_copied_file.txt
  ```

  Don't press enter before running the *spark-submit* command

- Now, go to the spark folder and type the following command

  ```
  bin/spark–submit
  examples/src/main/python/streaming/hdfs_wordcount.py /dirname
  ```

- After running the spark-submit command, run the command mentioned on line 5.

- Run the following command

  ```
  bin/run–example org.apache.spark.examples.streaming.HdfsWordCount
  /dirname
  ```

5. **MQTT wordcount & Hello World** (EMR 3.8)

WordCount

- Log into EMR and install mosquito (MQTT broker)

```
$ cd /etc/yum.repos.d/
$ sudo wget
http://download.opensuse.org/repositories/home:/oojah:/mqtt/RedHat_RHEL-7/home:oojah:mqtt.repo
```
(the above link is the RHEL 7 link from http://mosquitto.org/download/)
```
$ cd ~
$ sudo yum install mosquitto
```

- Start the broker, mosquitto on this terminal call it #1

```
$mosquitto
```

- Now, open up another terminal #2 and you may run the publisher

```
$ bin/run-example
org.apache.spark.examples.streaming.MQTTPublisher
tcp://localhost:1883 foo
```

- Opening up terminal #3, you run the spark example

```
$ bin/run-example
org.apache.spark.examples.streaming.MQTTWordCount
tcp://localhost:1883 foo
```

- In terminal #2 you will notice the word count.

Hello World

- Start up EMR and install node.js & npm
- Install mosquitto (as given above)
- Start broker on #1
- On terminal #2, run

```
$ sudo npm install mqtt -g
```

- Then on #2, you subscribe to the topic with the following command

```
$ mqtt sub –t ‘topic’ –h ‘127.0.0.1’ –v
```

- Fire up terminal #3, and publish to the topic

```
$ mqtt pub –t ‘hello’ –h ‘127.0.0.1’ –m ‘hello world’
```

- You will get *topic hello world* on #2

6. Twitter popular tags Scala

- You need to create a twitter application first in order to get the twitter credentials. Go to the link mentioned below and fill out the details to generate the required keys.

  ```
  https://dev.twitter.com/oauth/tools/signature-generator/8631720
  ```

- On creating the application, you will get 4 keys. Now, create an EMR cluster with the EMR version 3.8. Go to the spark folder and type the following command. This will give you the popular twitter tags.

  ```
  bin/run-example
  org.apache.spark.examples.streaming.TwitterPopularTags ConsumerKey
  ConsumerSecretKey AccessToken AccessTokenSecret
  ```

7. ZeroMQ wordcount- Scala

- Performing ZeroMQ on an EC2 instance
- Once you have it fired up, the next step will be to install ZeroMQ and we use ZeroMQ 2.1.10 version.
- Install using the following command. We get the link from and pick `zeromq-2.1.10.tar.gz`

```
$ sudo wget http://download.zeromq.org/zeromq-2.1.10.tar.gz
```

- Untar the file and configure it. You need to have `automake, autoconf, libtool, uuid-dev installed`
- Once you have all the packages installed, run the following to configure

```
$ ./configure
$ make
$ sudo make install
$ sudo ldconfig
```

- ZeroMQ is now installed.
- Once you are done, go into your spark directory and start the publisher

```
bin/run-example
org.apache.spark.examples.streaming.SimpleZeroMQPublisher
tcp://127.0.1.1:1234 foo.bar
```

- And then run the example

```
bin/run-example
org.apache.spark.examples.streaming.ZeroMQWordCount
tcp://127.0.1.1:1234 foo
```

8. Kafka wordcount – Python

- Get the *spark-streaming-kafka-assembly* jar from (use *wget*)

  http://mvnrepository.com/artifact/org.apache.spark/spark-streaming-kafka-assembly_2.10/1.4.0

- Install Kafka through the following link
  http://apache.mirrors.pair.com/kafka/0.8.2.1/kafka_2.11-0.8.2.1.tgz

- Untar this and cd into the folder

  The following should be done each on different terminals to see the results.

- Start zookeeper

```
$ bin/zookeeper-server-start.sh config/zookeeper.properties
```

  Zookeeper starts at localhost:2181

- Start Kafka Broker

```
$ bin/kafka-server-start.sh config/server.properties
```

  KafkaBroker starts on localhost:9092

- Create Kafka Topic

```
$ bin/kafka-topic.sh --create --zookeeper localhost:2181 --
replication-factor 1 --partitions 1 --topic kafkatopic
```

  Creates a topic by name Kafkatopic

- Start a Producer

```
$ bin/kafka-console-producer.sh --broker-list localhost:9092 --
topic kafkatopic
```

- Run the example

```
$ bin/spark-submit --jars spark-streaming-kafka-assembly.jar
examples/src/main/python/streaming/kafka_wordcount.py
localhost:2181 kafkatopic
```

9. Flume wordcount- Python

- Instead of flume python, use the SQL network count program available in the spark examples folder as *sql_network_wordcount.py*. In your EMR, go to spark 1.4.0. If you don't have spark 1.4.0 already installed, perform a *wget* and *untar* the compressed spark folder to get spark 1.4.0

- Then, type the following command and run it,

```
bin/spark-submit
examples/src/main/python/streaming/sql_network_wordcount.py
localhost 9999
```

- Then, open another terminal and run the command

```
nc -lk 9999
```

10. HDFS wordcount- Python

- Follow same steps as the HDFS Scala above for initial steps
- Go to the spark folder and type the following command

```
bin/spark-submit
examples/src/main/python/streaming/hdfs_wordcount.py /dirname
```