# CSP554—Big Data Technologies

## Assignment #9

**Exercise 1) 5 points**

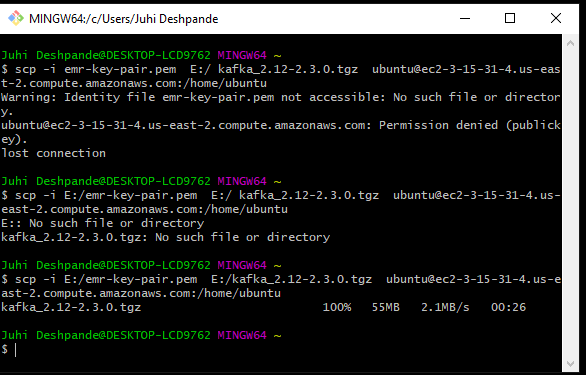
**Read and provide a half page summary and analysis of this article available on the blackboard in the ‘Articles’ section: Real-time stream processing for Big Data**

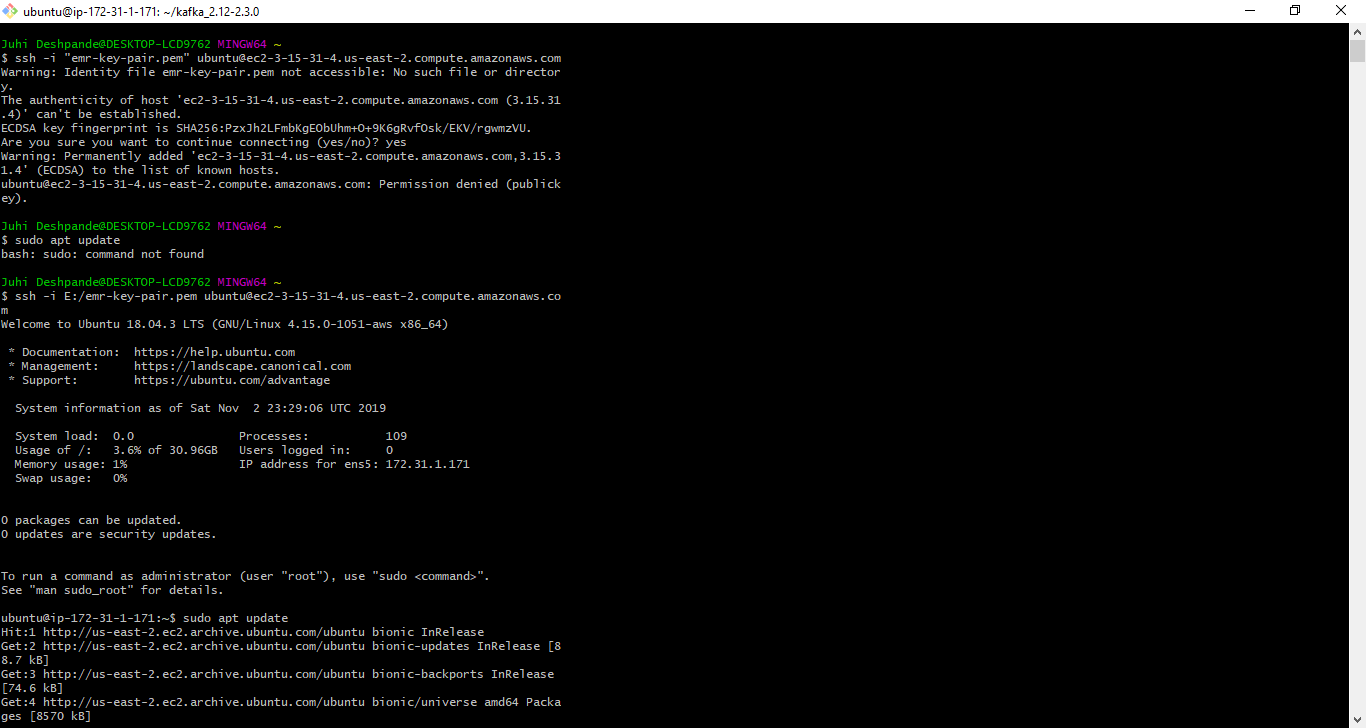
From the paper

The paper focuses on implementing Storm and its abstraction layer Trident, Samza and Spark Streaming for a task and their trade-offs instead of batch processing to improve the efficiency and avoid latency in Big data. The paper explains about the streamlining operation that the data is collected and then set into Kafka, Kinesis, etc. so that it becomes available for streamline processors and the output is passed to web analytics GUI. So, one solution is to combine the batch streaming and real time processing lambda architecture is used. It says that the latency of batch processing can me compensated by using Spark that does real time and batch processing. The other solution is Kappa architecture that does the batch processing along with the real time processing and only when logic is changed it substitutes by historical data. The paper also describes about Storm, a real time processing system that supports various languages, has a ZooKeeper and a Nimbus that makes it scalable, fault tolerant and can be assigned during run time. The second real time system stated is Samza that is like Storm where streams are partitioned and the data items inside a stream are ordered but are unordered with respect to another stream. It is mostly deployed with Kafka due to its capabilities. The next one is Spark Streaming that provides real time processing also Machine Learning libraries also resilient to machine failures. It takes the real time data converts them to batches and converts them into RDDs. The table showed that Storm provided low latency but offers no guarantee. Samza on the other sides provides guarantee when integrated with Kafka but has latency higher than Storm. Spark has rich set of libraries and processing real time and batch data efficiently has the disadvantage of latency. Hence, these real time systems have certain trade offs when it comes to latency. Further the paper suggests that there are certain systems developed that are available, fault tolerant and provide horizontal scalability The paper concludes by stating that Lambda architecture thus used by many systems fails to fill the gap between batch and real time processing and the Kappa architecture that avoids batch processing along with being good in processing the Big Data analytics is gaining less momentum as compared to the horizontal stream processors.

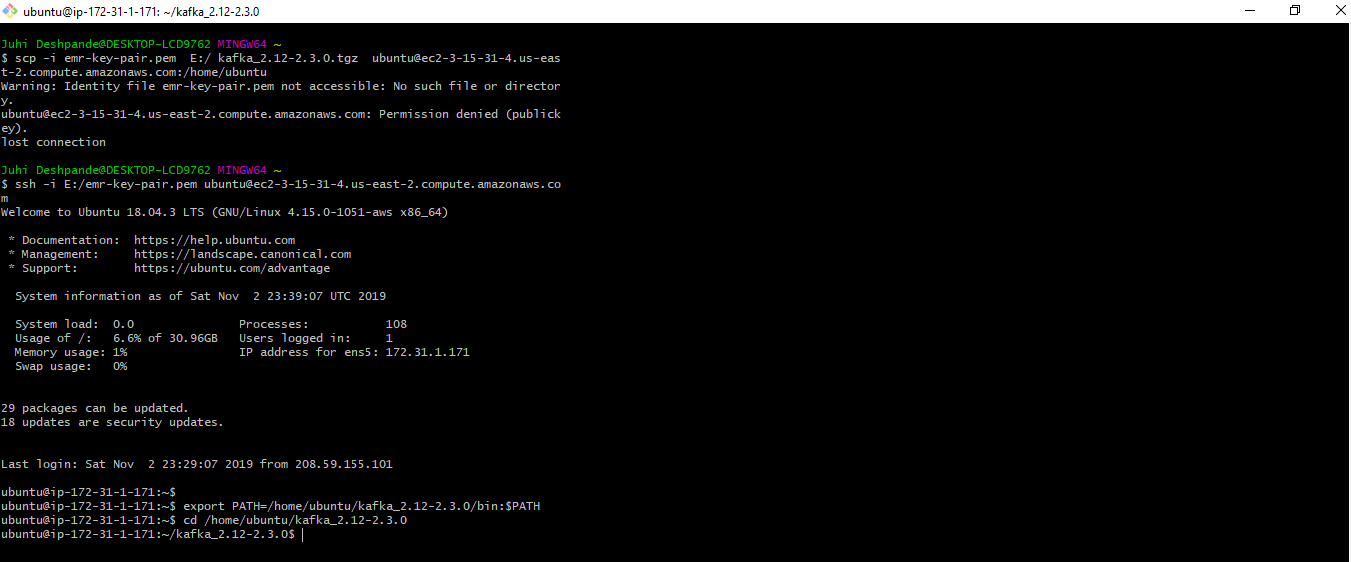
**Exercise 2) 5 points extra credit**

**Follow the document “Instructions for setting up a VM with Kafka” included with this assignment and execute the demo code. Provide enough screen shots to indicate you have completed the document through section 4. Then remember to terminate your VM.**

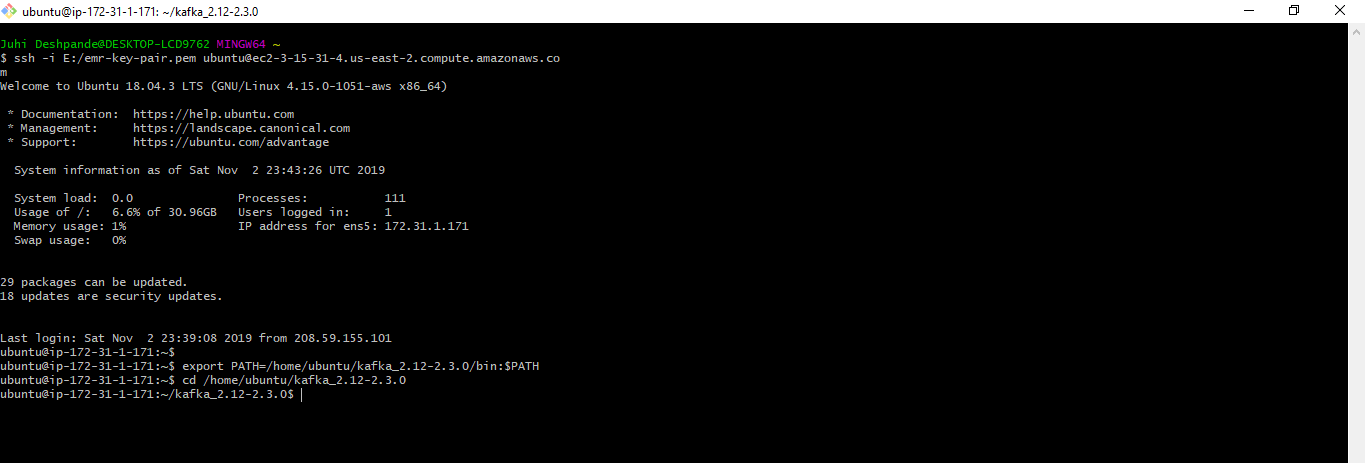


**EC-1 Window:**

**EC – 2 window:**



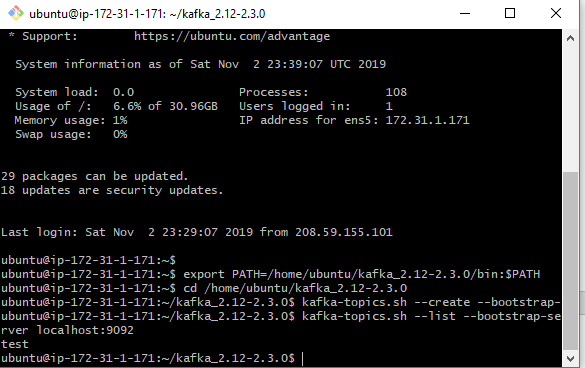
**EC-3 Window:**



## Section 4: Working with Kafka

1. We can now see that topic if we run the list topic command in the EC2-2 window (or actually any window connected to our EC2 instance:

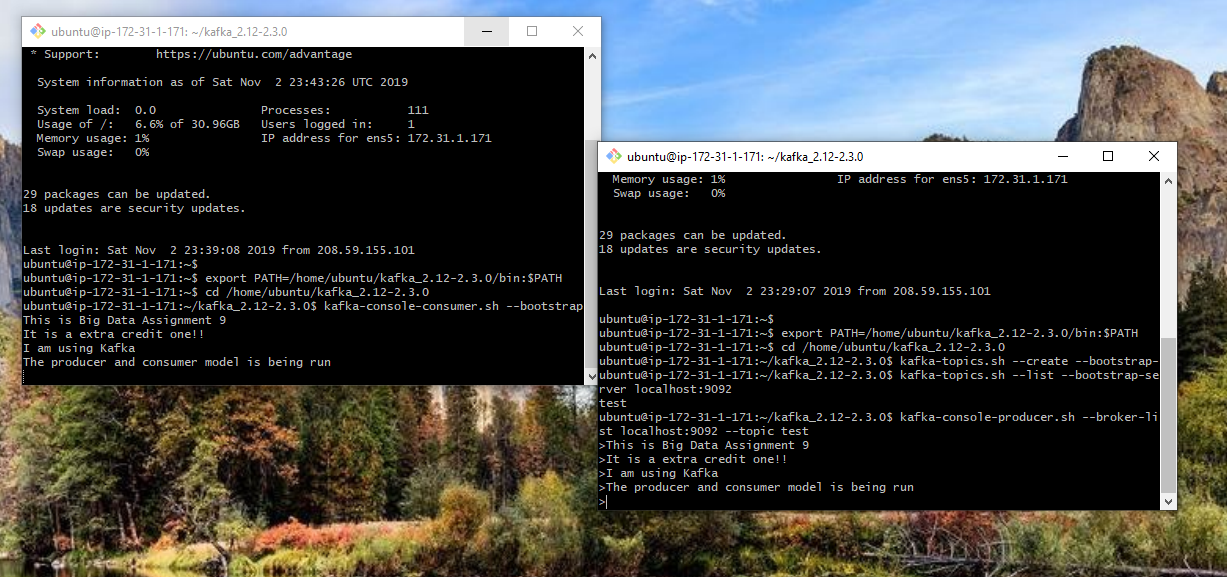
kafka-topics.sh --list --bootstrap-server localhost:9092



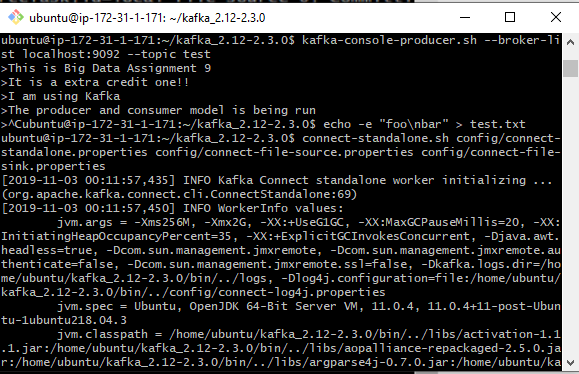
1. Kafka also has a command line consumer that will dump out messages to standard output. Run the consumer in the EC2-3 window as follows:

kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic test --from-beginning

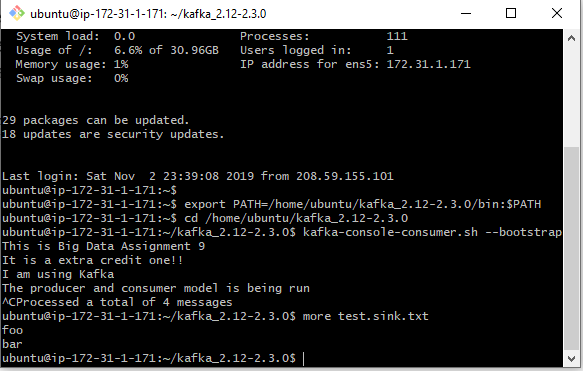
The messages were being received at the consumer side after clicking “Enter”



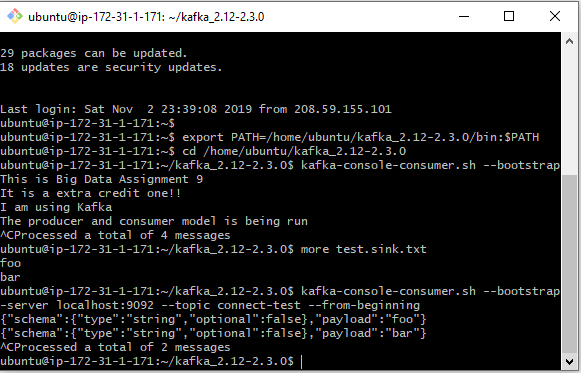
**Q6, 7, 8 combined**

**EC2 window:**

**EC3 window:**



**Q9)** Note that the data is being stored in the Kafka topic connect-test, so we can also run the following console consumer command in the EC2-3 window to see the data in the topic (or use custom consumer code to process it):

kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic connect-test --from-beginning

10)The connectors continue to process data, so we can add data to the file and see it move through the pipeline. Enter the following into the EC2-3 window

echo Another line>> test.txt

You should see the line appear in the console consumer output and in the sink file if you then do the following:

more test.sink.txt

And you should see the following:

foo

bar

Another line

