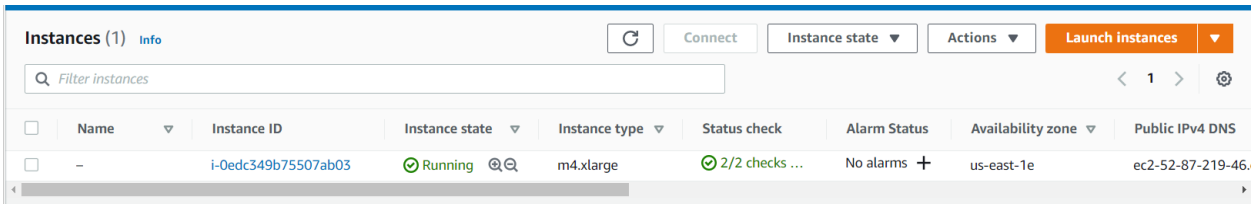
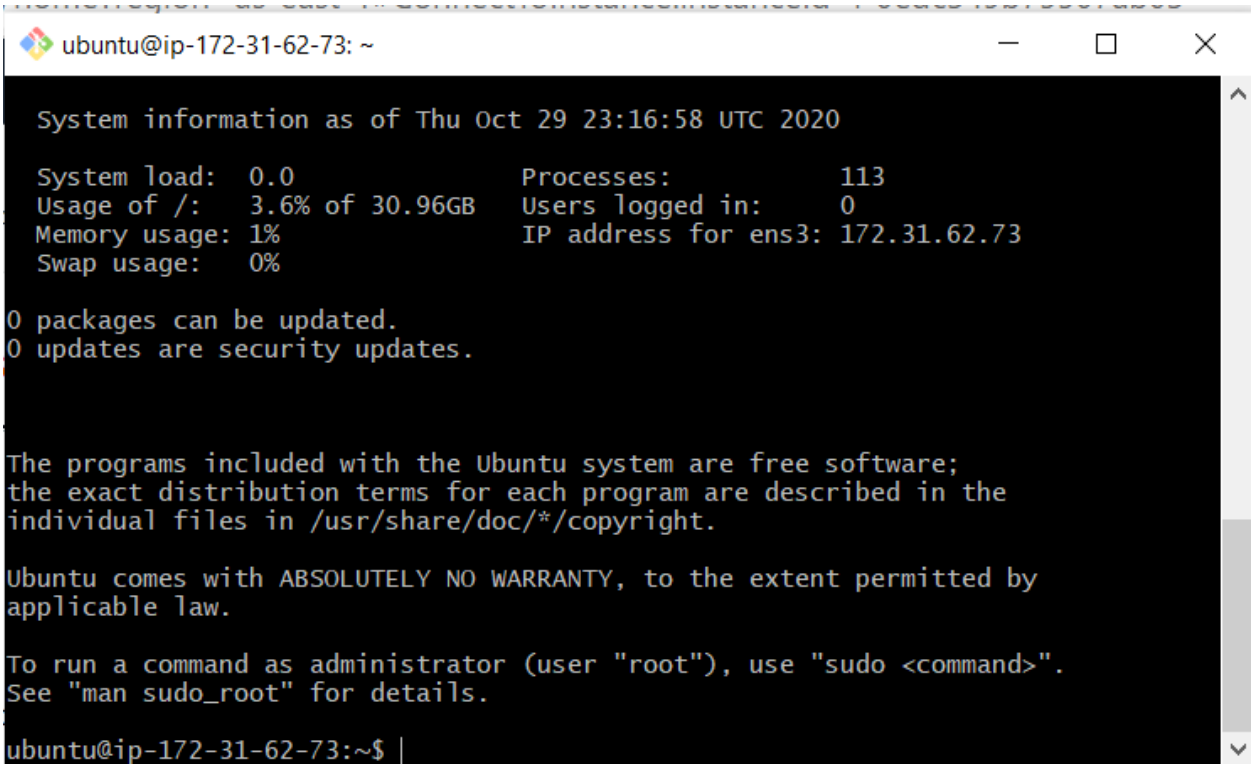
Exercise 2)

EC2 Creation

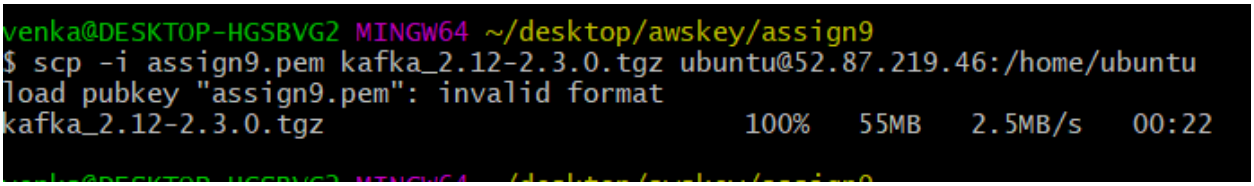


Connecting to UButntu EC2 Server:



3)

Command Used: *scp -i assign9.pem kafka_2.12-2.3.0.tgz ubuntu@52.87.219.46:/home/ubuntu*



```
ubuntu@ip-172-31-62-73:~$ ls
kafka_2.12-2.3.0.tgz
ubuntu@ip-172-31-62-73:~$
```

4)

I) Command Used: *sudo apt update*

```
ubuntu@ip-172-31-62-73:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates InRelease [8
8.7 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-backports InRelease
[74.6 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/universe amd64 Packa
ges [8570 kB]
Get:5 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
```

```
Fetched 21.6 MB in 4s (5446 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
59 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-62-73:~$
```

II) Command Used: *sudo apt install zip*

```
ubuntu@ip-172-31-62-73:~$ sudo apt install zip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  unzip
The following NEW packages will be installed:
  unzip zip
0 upgraded, 2 newly installed, 0 to remove and 59 not upg
```

```
Setting up unzip (6.0-21ubuntu1) ...
Setting up zip (3.0-11build1) ...
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
ubuntu@ip-172-31-62-73:~$
```

III) Command Used : *sudo apt install default-jre*

```
ubuntu@ip-172-31-62-73:~$ sudo apt install default-jre
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  at-spi2-core ca-certificates-java default-jre-headless fontconfig-config
  fonts-dejavu-core fonts-dejavu-extra java-common libasound2 libasound2-data
  libatk-bridge2.0-0 libatk-wrapper-java libatk-wrapper-java-jni libatk1.0-0
  libatk1.0-data libatspi2.0-0 libavahi-client3 libavahi-common-data
  libavahi-common3 libcups2 libdrm-amdgpu1 libdrm-intel1 libdrm-nouveau2
```

```
done.
done.
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for ureadahead (0.100.0-21) ...
ubuntu@ip-172-31-62-73:~$
```

IV) Command Used: *sudo apt install default-jdk*

```
ubuntu@ip-172-31-62-73:~$ sudo apt install default-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  default-jdk-headless libice-dev libpthread-stubs0-dev libsm-dev libx11-dev
  libx11-doc libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk
  openjdk-11-jdk-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools
  xtrans-dev
```

```
Setting up libxau-dev:amd64 (1:1.0.8-1ubuntu1) ...
Setting up libxcb1-dev:amd64 (1.13-2~ubuntu18.04) ...
Setting up libx11-dev:amd64 (2:1.6.4-3ubuntu0.3) ...
Setting up default-jdk (2:1.11-68ubuntu1~18.04.1) ...
Setting up libxt-dev:amd64 (1:1.1.5-1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
ubuntu@ip-172-31-62-73:~$
```

V) Command Used: *sudo apt install python3-pip*

```
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
ubuntu@ip-172-31-62-73:~$ sudo apt install python3-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential cpp cpp-7
```

VI) Command Used:*pip3 install kafka-python*

```
ubuntu@ip-172-31-62-73:~$ pip3 install kafka-python
Collecting kafka-python
  Downloading https://files.pythonhosted.org/packages/75/68/dcb0db055309f680ab29
31a3eeb22d865604b638acf8c914bedf4c1a0c8c/kafka_python-2.0.2-py2.py3-none-any.whl
 (246kB)
    100% |                                | 256kB 4.4MB/s
Installing collected packages: kafka-python
Successfully installed kafka-python-2.0.2
ubuntu@ip-172-31-62-73:~$
```

5) ***Entering a Command in KT2***

Command Used: *tar -xvf kafka_2.12-2.3.0.tgz*

```
ubuntu@ip-172-31-62-73:~$ tar -xvf kafka_2.12-2.3.0.tgz
kafka_2.12-2.3.0/
kafka_2.12-2.3.0/LICENSE
kafka_2.12-2.3.0/NOTICE
kafka_2.12-2.3.0/bin/
kafka_2.12-2.3.0/bin/kafka-consumer-perf-test.sh
```

6)

Install location in KT2

Command Used:

export PATH=/home/ubuntu/kafka_2.12-2.3.0/bin:$PATH

Then change to the /home/ubuntu/kafka_2.12-2.3.0 directory:

cd /home/ubuntu/kafka_2.12-2.3.0

```
ubuntu@ip-172-31-62-73:~$ export PATH=/home/ubuntu/kafka_2.12-2.3.0/bin:$PATH
ubuntu@ip-172-31-62-73:~$
```

```
ubuntu@ip-172-31-62-73:~$ cd /home/ubuntu/kafka_2.12-2.3.0
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$
```

SECTION 3

1)

Install location in KT3

Command Used:

export PATH=/home/ubuntu/kafka_2.12-2.3.0/bin:$PATH

Then change to the /home/ubuntu/kafka_2.12-2.3.0 directory:

cd /home/ubuntu/kafka_2.12-2.3.0

```
ubuntu@ip-172-31-62-73:~$ export PATH=/home/ubuntu/kafka_2.12-2.3.0/bin:$PATH
ubuntu@ip-172-31-62-73:~$ cd /home/ubuntu/kafka_2.12-2.3.0
```

2) ***Install location in KT4***

Command Used:

export PATH=/home/ubuntu/kafka_2.12-2.3.0/bin:$PATH

Then change to the /home/ubuntu/kafka_2.12-2.3.0 directory:

cd /home/ubuntu/kafka_2.12-2.3.0

```
ubuntu@ip-172-31-62-73:~$ export PATH=/home/ubuntu/kafka_2.12-2.3.0/bin:$PATH
ubuntu@ip-172-31-62-73:~$ cd /home/ubuntu/kafka_2.12-2.3.0
```

3)

Command Used: *zookeeper-server-start.sh config/zookeeper.properties &*

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ zookeeper-server-start.sh config/zook
eeper.properties &
[1] 24439
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ [2020-10-30 00:00:23,087] INFO Readin
g configuration from: config/zookeeper.properties (org.apache.zookeeper.server.q
uorum.QuorumPeerConfig)
[2020-10-30 00:00:23,089] INFO autopurge.snapRetainCount set to 3 (org.apache.zo
okeeper.server.DatadirCleanupManager)
[2020-10-30 00:00:23,089] INFO autopurge.purgeInterval set to 0 (org.apache.zook
eeper.server.DatadirCleanupManager)
[2020-10-30 00:00:23,089] INFO Purge task is not scheduled. (org.apache.zookeepe
```

4)

Command Used : *kafka-server-start.sh config/server.properties &*

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ kafka-server-start.sh config/server.p
roperties &
[2] 24766
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ [2020-10-30 00:04:34,958] INFO Regist
ered kafka:type=kafka.Log4jController MBean (kafka.utils.Log4jControllerRegistra
tion$)
[2020-10-30 00:04:35,432] INFO Registered signal handlers for TERM, INT, HUP (or
g.apache.kafka.common.utils.LoggingSignalHandler)
[2020-10-30 00:04:35,433] INFO starting (kafka.server.KafkaServer)
[2020-10-30 00:04:35,434] INFO Connecting to zookeeper on localhost:2181 (kafka.
server.KafkaServer)
```

Section 4

1)

Command used: *kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic test*

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ kafka-topics.sh --create --bootstrap-
server localhost:9092 --replication-factor 1 --partitions 1 --topic test
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$
```

2)

Command Used*: kafka-topics.sh --list --bootstrap-server localhost:9092*

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ kafka-topics.sh --list --bootstrap-s
rver localhost:9092
test
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$
```

3)

Command Used: *kafka-console-producer.sh --broker-list localhost:9092 --topic test*

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ kafka-console-producer.sh --broker-li
st localhost:9092 --topic test
>HI this is akshay
>hi kishore this is your name too
>
```

4)

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

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ kafka-console-consumer.sh --bootstrap
-server localhost:9092 --topic test --from-beginning
HI this is akshay
hi kishore this is your name too
```

5) ***Ending Demo in KT3***

Command Used: control +C

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ kafka-console-producer.sh --broker-li
st localhost:9092 --topic test
>HI this is akshay
>hi kishore this is your name too
>^Cubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$
```

Ending Demo in KT4

Command Used: control +C

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ kafka-console-consumer.sh --bootstrap
-server localhost:9092 --topic test --from-beginning
HI this is akshay
hi kishore this is your name too
^CProcessed a total of 2 messages
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$
```

7)

Command Used: *echo -e "foo\nbar" > test.txt*

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ echo -e "foo\nbar" > test.txt
ntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$
```

8)

Command used: *connect-standalone.sh config/connect-standalone.properties config/connect-file-source.properties config/connect-file-sink.properties*

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ connect-standalone.sh config/connect-
standalone.properties config/connect-file-source.properties config/connect-file-
sink.properties
[2020-10-30 00:34:03,925] INFO Kafka Connect standalone worker initializing ...
(org.apache.kafka.connect.cli.ConnectStandalone:69)
[2020-10-30 00:34:03,940] INFO WorkerInfo values:
        jvm.args = -Xms256M, -Xmx2G, -XX:+UseG1GC, -XX:MaxGCPauseMillis=20, -XX:
InitiatingHeapOccupancyPercent=35, -XX:+ExplicitGCInvokesConcurrent, -Djava.awt.
headless=true, -Dcom.sun.management.jmxremote, -Dcom.sun.management.jmxremote.au
thenticate=false, -Dcom.sun.management.jmxremote.ssl=false, -Dkafka.logs.dir=/ho
me/ubuntu/kafka_2.12-2.3.0/bin/../logs, -Dlog4j.configuration=file:/home/ubuntu/
kafka_2.12-2.3.0/bin/../config/connect-log4j.properties
        jvm.spec = Ubuntu, OpenJDK 64-Bit Server VM, 11.0.9, 11.0.9+11-Ubuntu-0u
buntu1.18.04.1
```

Command Used: *more test.sink.txt*

```
Processed a total of 2 messages
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ more test.sink.txt
foo
bar
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$
```

9)

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

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ kafka-console-consumer.sh --bootstrap
-server localhost:9092 --topic connect-test --from-beginning
{"schema":{"type":"string","optional":false},"payload":"foo"}
{"schema":{"type":"string","optional":false},"payload":"bar"}
```

10)

Command Used: *echo Another line>> test.txt*

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ echo Another line>> test.txt
```

Command Used: *more test.sink.txt*

```
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$ more test.sink.txt
foo
bar
line
Another line
ubuntu@ip-172-31-62-73:~/kafka_2.12-2.3.0$
```

Exercise 1)

a) Kappa architecture is a stream processing system and to achieve this it employs a powerful stream processor capable of coping with data at a far greater rate than it is incoming. In lambda architecture all data is periodically recomputed in batch layer and in kappa architecture the only perform computation when the business logic changes by replaying historical data.

b)

Advantages:

1. Purely stream-oriented systems provide very low latency and relatively high per-item cost. streaming provides scalable streaming system for data retention.
2. It restricts batch size to reduce latency.

Drawbacks:

1. Batch-oriented systems achieve unparalleled resource-efficiency at the expense of latency that is prohibitively high for real-time applications.
2. Trident processes one -at -time which is in favor of throughput than latency.

c) The pipeline in storm is called topology. The nodes that ingest data and initiate the data flow is called spouts. Spouts emit tuples to the nodes at downstream which are called bolts and do the processing. Storm distributes spouts and bolts across the nodes in the cluster in a round robin fashion. Storm does not provide any guarantee on the order in which tuples are processed, it provides the option of at-least-once processing through acknowledgement feature that tracks the processing.

d) Spark Streaming shifts Spark's batch-processing approach towards real-time requirements by chunking the stream of incoming data items into small batches, transforming them into RDDs and processing them as usual. It further takes care of data flow and distribution automatically. Data is ingested and transformed into a sequence of RDDs which is called DStream before processing through workers.