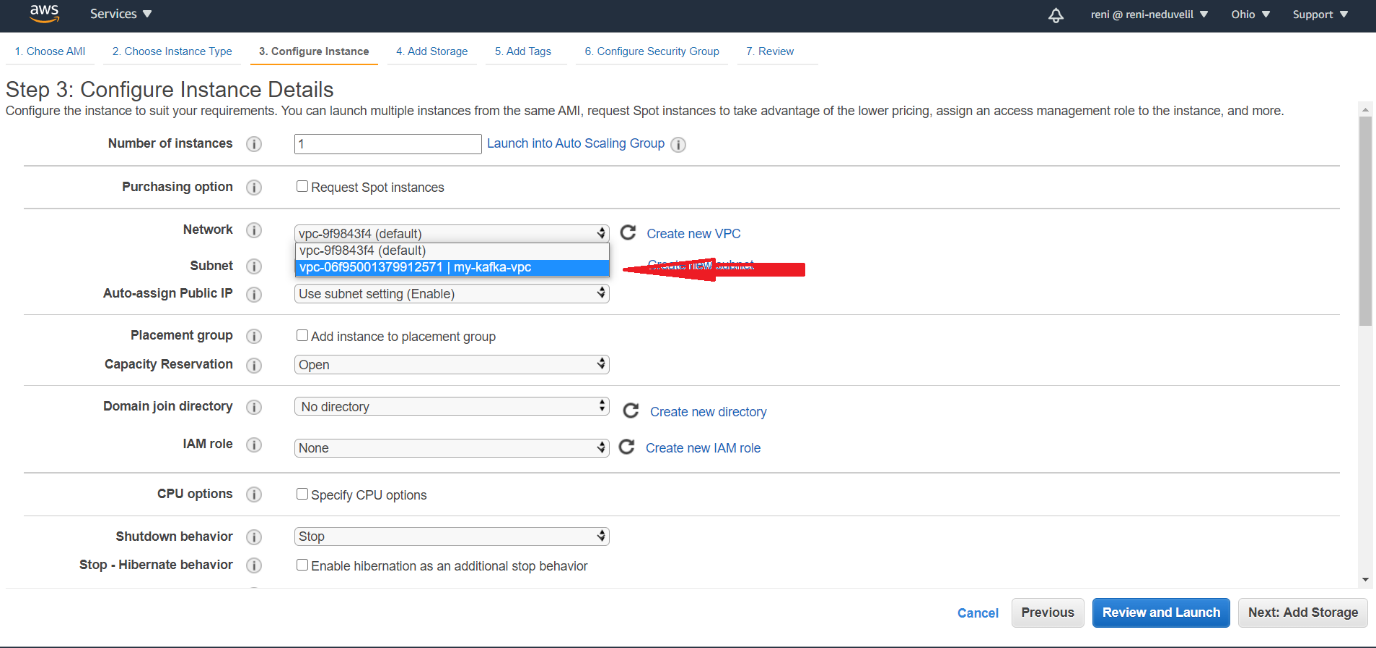
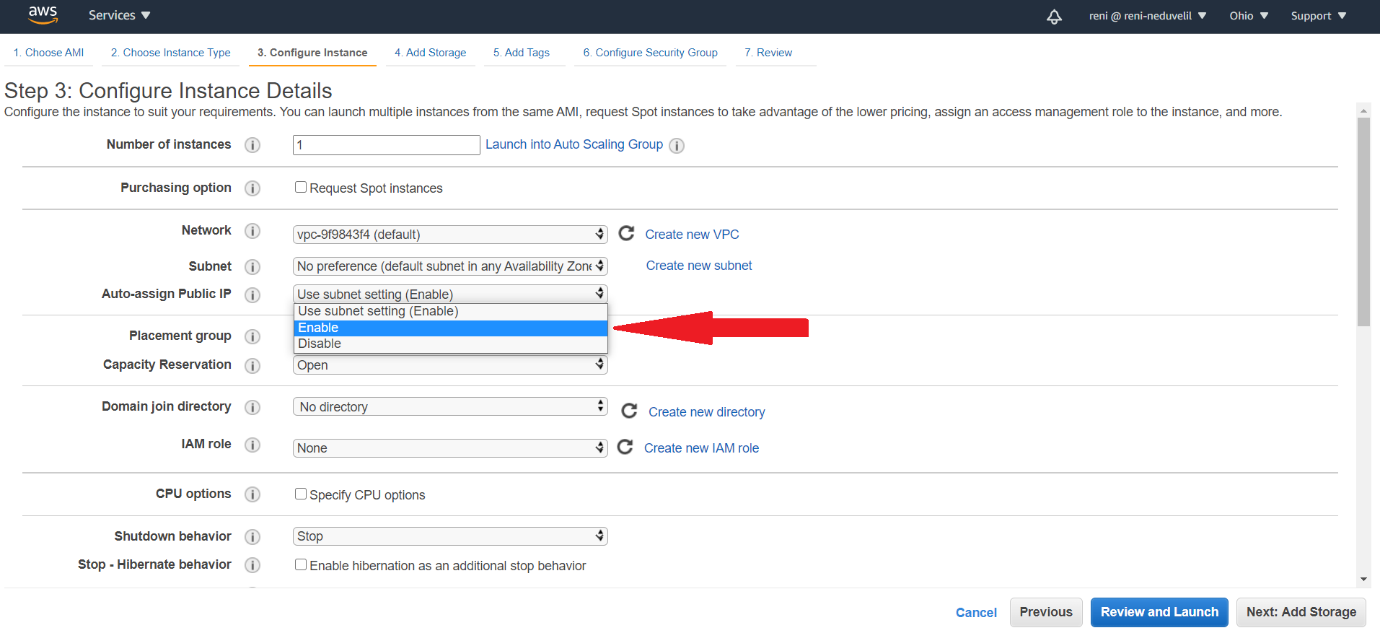
Handson 08 -**Solution**

Create an AWS MSK cluster and deploy a producer and consumer and produce and consume data by making use of AWS MSK cluster.

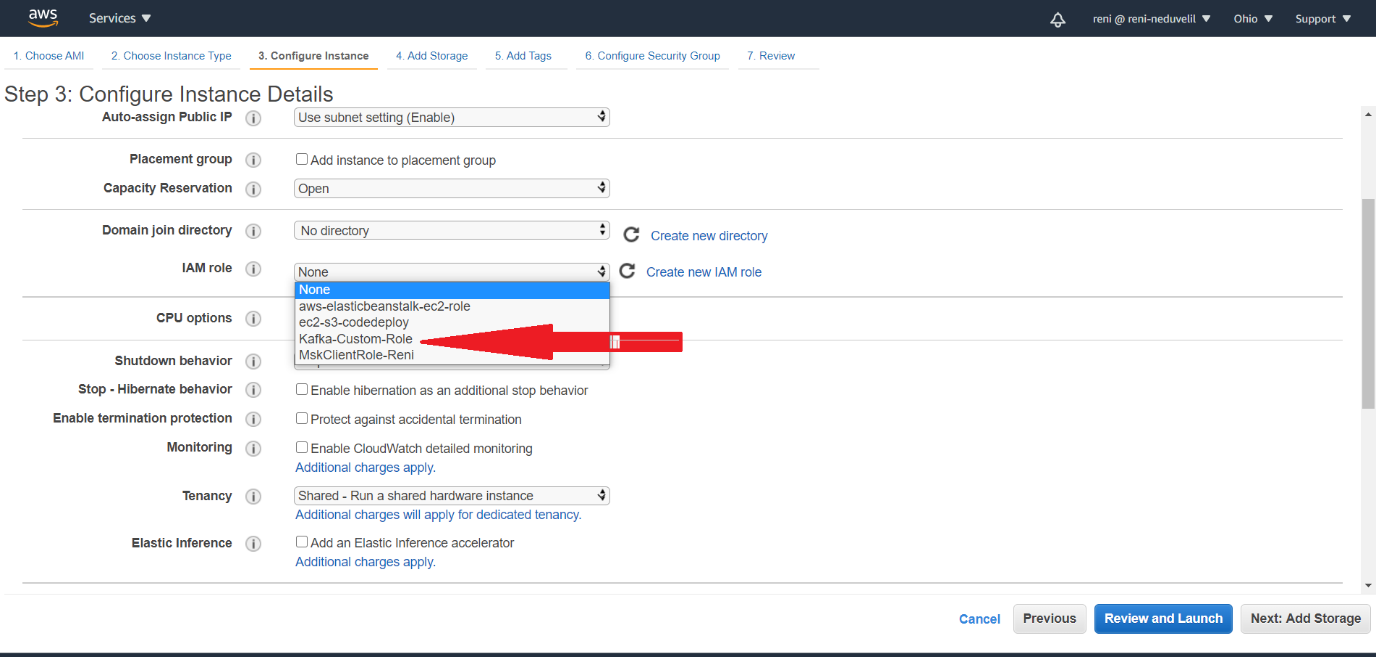
1. Create an AWS cluster as discussed in the previous hands-on
2. Create an EC2 Instance. While selecting the VPC, make sure that you select the same VPC selected for your MSK cluster.



1. Select “Auto-assign public IP” is enabled



1. IAM role selected should have “[AmazonMSKFullAccess](https://console.aws.amazon.com/iam/home?region=us-east-2#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonMSKFullAccess)” policy attached.

****

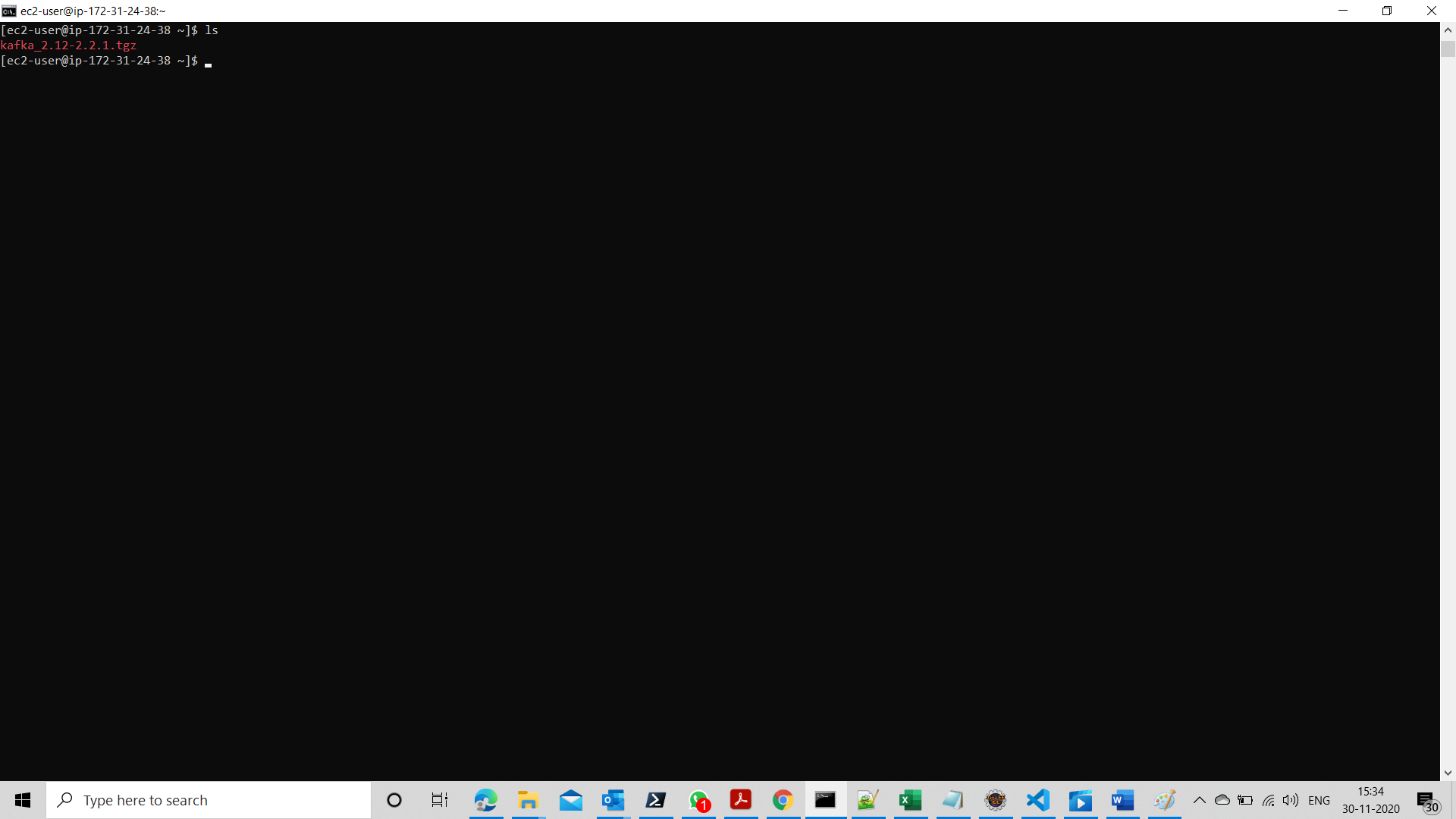
1. Once you created the EC2 instance, connect to the instance using SSH
2. Install java by issuing the below mentioned command.

**sudo yum install java-1.8.0 -y**

1. Get the Apache Kafka distributable by issuing the below mentioned command

**wget** [**https://archive.apache.org/dist/kafka/2.2.1/kafka\_2.12-2.2.1.tgz**](https://archive.apache.org/dist/kafka/2.2.1/kafka_2.12-2.2.1.tgz)

1. Issue ls command and you will get the screen below



1. Extract the tgz file by issuing the command below.

**tar -xzf kafka\_2.12-2.2.1.tgz**

1. Issue the ls command, you can see that one folder named kafka\_2.12-2.2.1 is created.
2. Issue the command below to set the path to the bin folder of Kafka

**export PATH=$PATH:/home/ec2-user/kafka\_2.12-2.2.1/bin**

1. Issue the command aws kafka describe-cluster --cluster-arn arn:aws:kafka:us-east-2:929084036303:cluster/demo-kafka-cluster/0df170a6-9da9-40f4-8ba2-0f7bdecba222-4 --region us-east-2

Cluster-arn🡺 is the arn of the AWS MSK cluster

1. You might get an error while issuing this command. In that case you have to provide your access key, secret key and the region as shown below

Unable to locate credentials. You can configure credentials by running "aws configure".

[ec2-user@ip-172-31-24-38 ~]$ aws configure

AWS Access Key ID [None]: AKIA5QUNZKTHWGIL2QDT

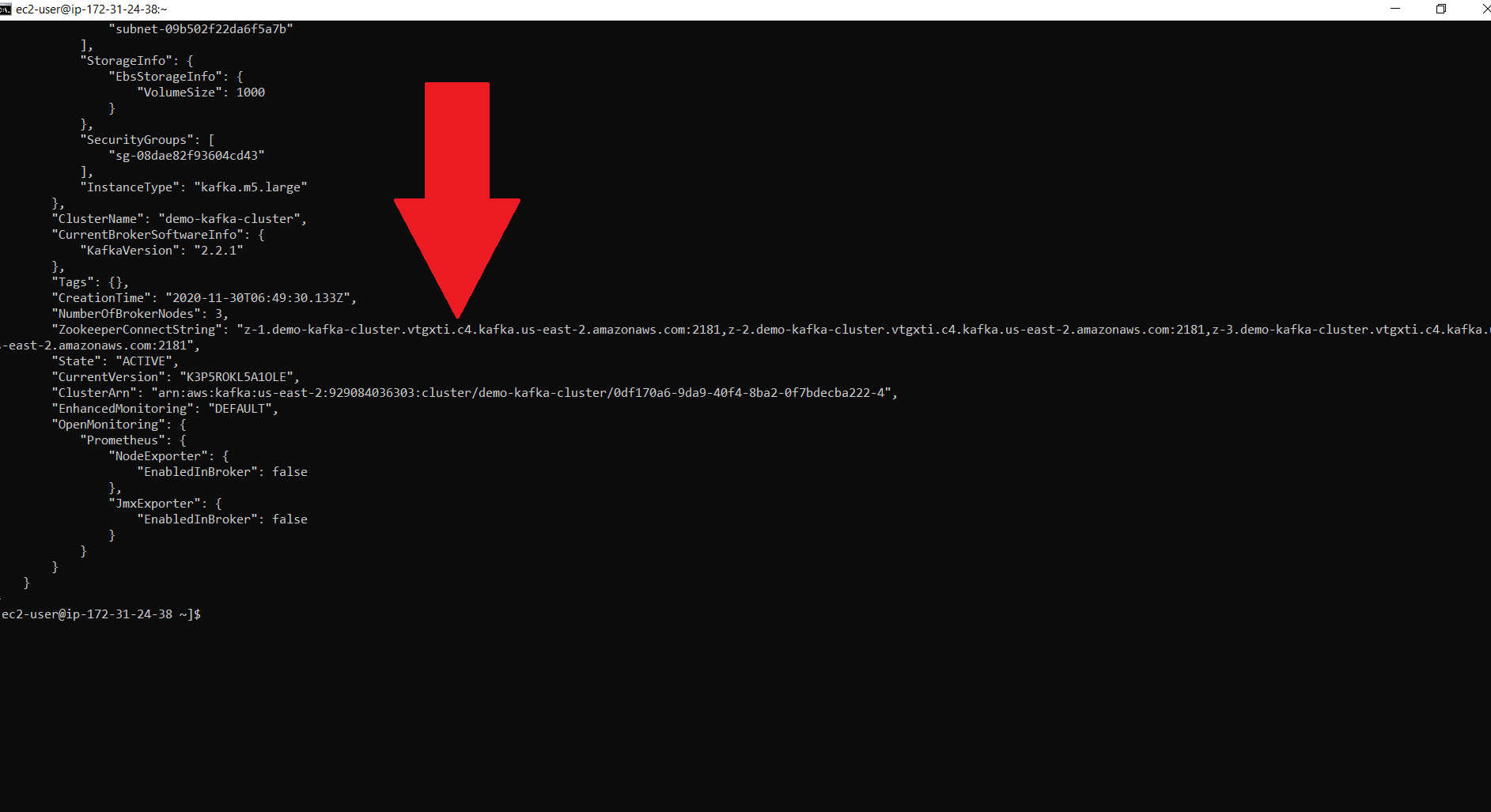
AWS Secret Access Key [None]: 8eSk0MAweF1VYWv4vjXLSlcGeC77i0O9fKyVWn+8

Default region name [None]: us-east-2

Default output format [None]:

[ec2-user@ip-172-31-24-38 ~]$

1. From the output of the command issued at point 12, locate “ZookeeperConnectionString” and copy its value.



1. Issue the command given below

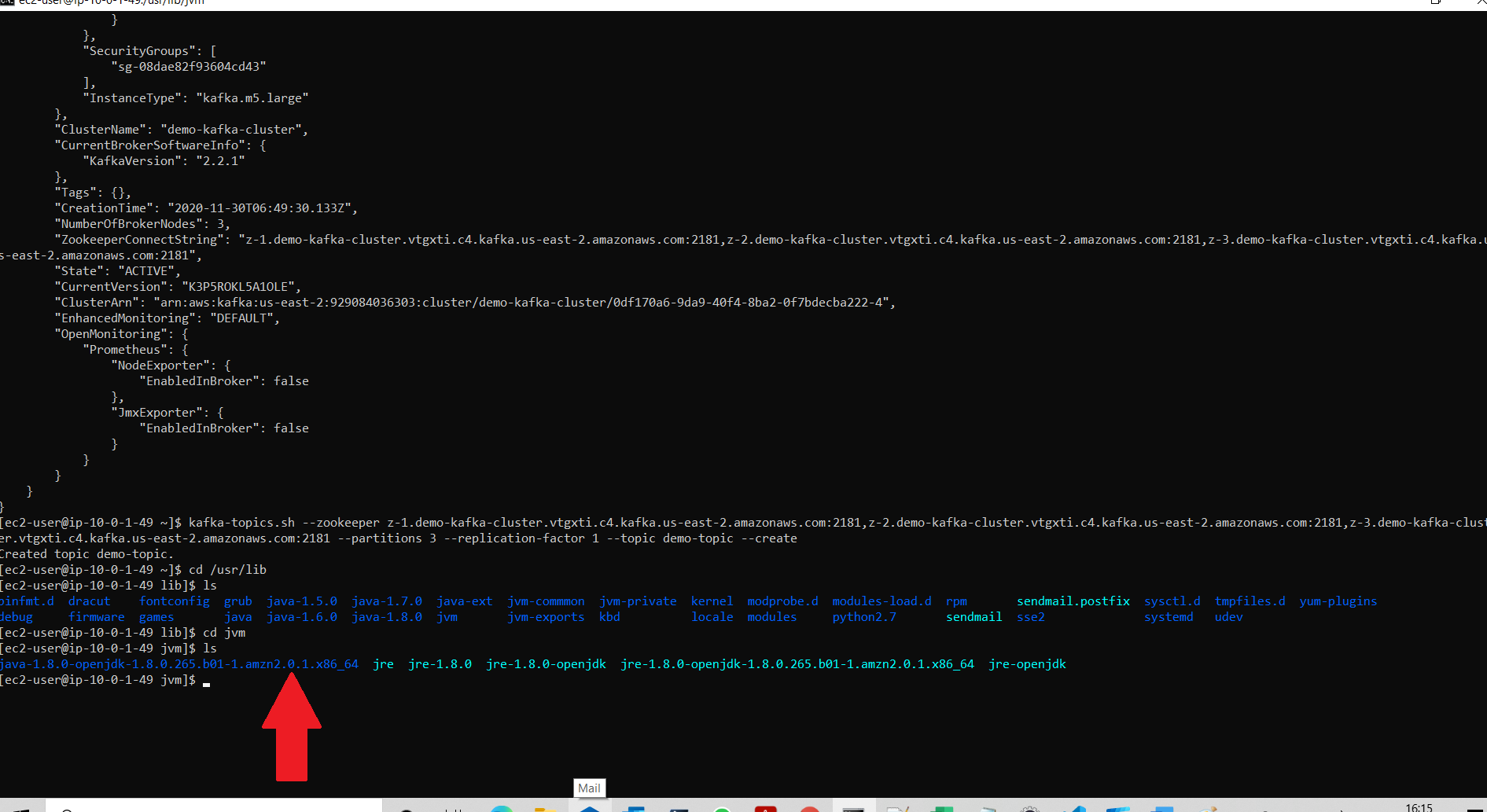
kafka-topics.sh --zookeeper z-1.demo-kafka-cluster.vtgxti.c4.kafka.us-east-2.amazonaws.com:2181,z-2.demo-kafka-cluster.vtgxti.c4.kafka.us-east-2.amazonaws.com:2181,z-3.demo-kafka-clust

er.vtgxti.c4.kafka.us-east-2.amazonaws.com:2181 --partitions 3 --replication-factor 1 --topic demo-topic –create

1. --Zookeeper specifies the zookeeper connectionstring.
2. --topic specifies the name of the topic to be created.
3. –partitions specified the number of partitions to be created in the topic
4. –replication-factor specified the replication factor of the broker.
5. Go to the /usr/lib folder by issuing the command given below.

Cd /usr/lib

1. Move to the jvm folder by issuing the command “cd jvm”
2. Issue the ls command and copy the java folder name (shown in the below screen)



1. Now move back into the user home directory by issuing the command below

Cd /home/ec2-user

1. Issue the below mentioned command

cp /usr/lib/jvm/JDKFolder/jre/lib/security/cacerts /tmp/kafka.client.truststore.jks

The highlighted JDKFolder should be replaced with the java folder name we copied in step 18.

**cp /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.265.b01-1.amzn2.0.1.x86\_64/jre/lib/security/cacerts /tmp/kafka.client.truststore.jks**

1. Move to the bin folder inside the Kafka folder by issuing the below given command

**cd kafka\_2.12-2.2.1/bin**

1. Create a file named client.properties inside this folder by using the text editor vim
2. Type vim client.properties in the terminal
3. Press i for insert inside the editor and type the following lines of code

security.protocol=SSL

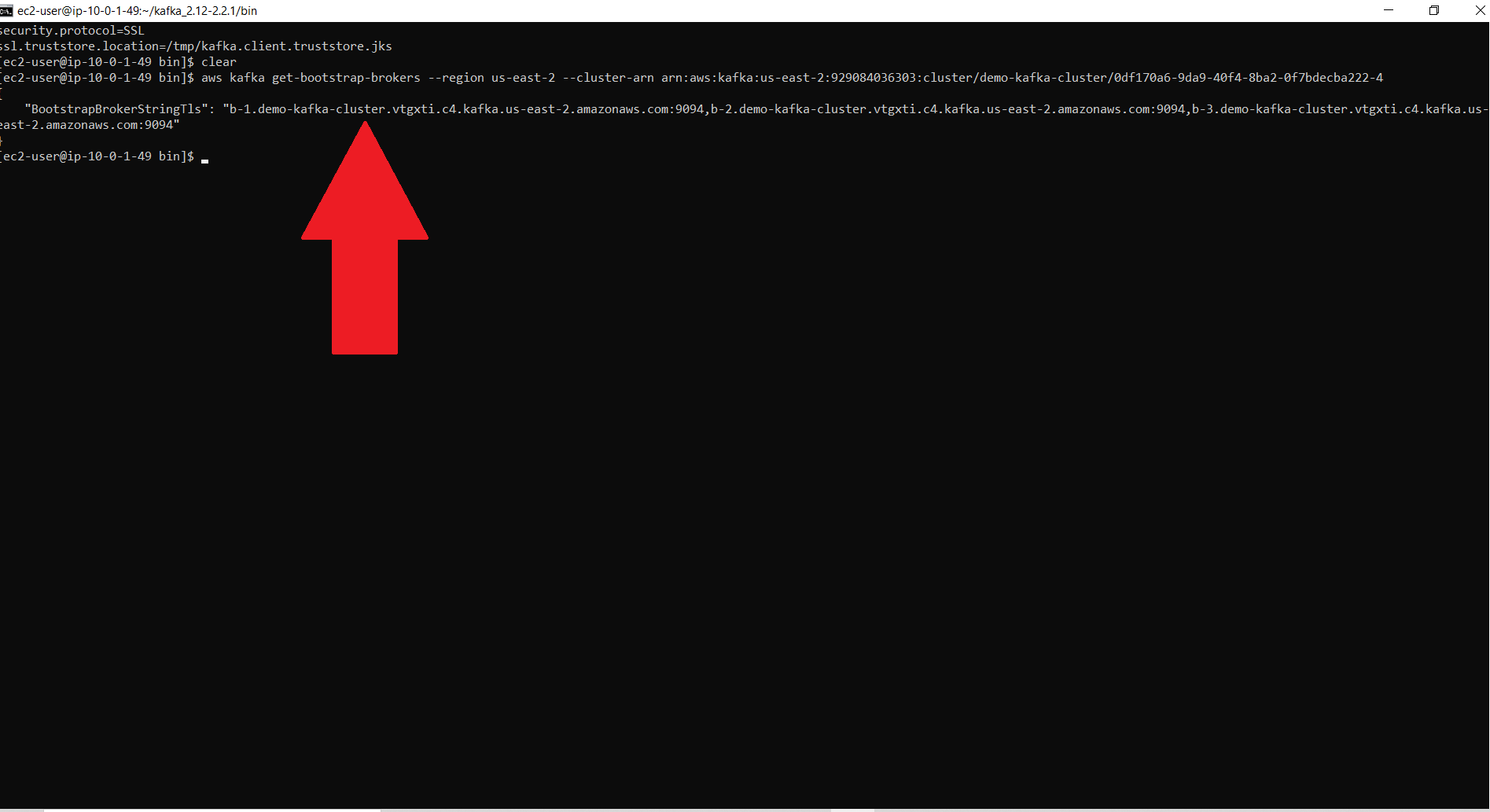
ssl.truststore.location=/tmp/kafka.client.truststore.jks

1. Press esc and type :wq to save and exit from the editor
2. Now you have to get the “BootstrapBrokerStringTls”. For that you need to issue the below given command

**aws kafka get-bootstrap-brokers --region us-east-2 --cluster-arn arn:aws:kafka:us-east-2:929084036303:cluster/demo-kafka-cluster/0df170a6-9da9-40f4-8ba2-0f7bdecba222-4**

* 1. –cluster-arn🡺 specifies you cluster arn
  2. –region 🡺 specifies your region

1. You will get the output as shown by the below screen

****

1. Now we need to create a Kafka producer by issuing the command given below

kafka-console-producer.sh --broker-list b-1.demo-kafka-cluster.vtgxti.c4.kafka.us-east-2.amazonaws.com:9094,b-2.demo-kafka-cluster.vtgxti.c4.kafka.us-east-2.amazonaws.com:9094,b-3.demo-kafka-cluster.vtgxti.c4.kafka.us-east-2.amazonaws.com:9094 --producer.config client.properties --topic demo-topic

--borker-list🡺 specifies the “BootstrapBrokerStringTls” you got from step 26 and 27.

--topic🡺 name of the topic to which you are sending the message.

--producer.config🡺points to the client.properties files which we created in step 22

1. You will get the screen shown below where you can type your messages.
2. Now you need to create the consumer. You can have separate EC2 instance as consumer. But for simplicity, we will create the consumer in the same EC2 instance in a separate terminal.
3. Open a new terminal and connect to the EC2 instance.
4. Set the path by issuing the command below

export PATH=$PATH:/home/ec2-user/kafka\_2.12-2.2.1/bin

1. Move to the folder kafka\_2.12-2.2.1/bin by issuing

cd kafka\_2.12-2.2.1/bin

1. Issue the below given command for creating a consumer

kafka-console-consumer.sh --bootstrap-server b-1.demo-kafka-cluster.vtgxti.c4.kafka.us-east-2.amazonaws.com:9094,b-2.demo-kafka-cluster.vtgxti.c4.kafka.us-east-2.amazonaws.com:9094,b

-3.demo-kafka-cluster.vtgxti.c4.kafka.us-east-2.amazonaws.com:9094 --topic demo-topic --consumer.config client.properties --from-beginning

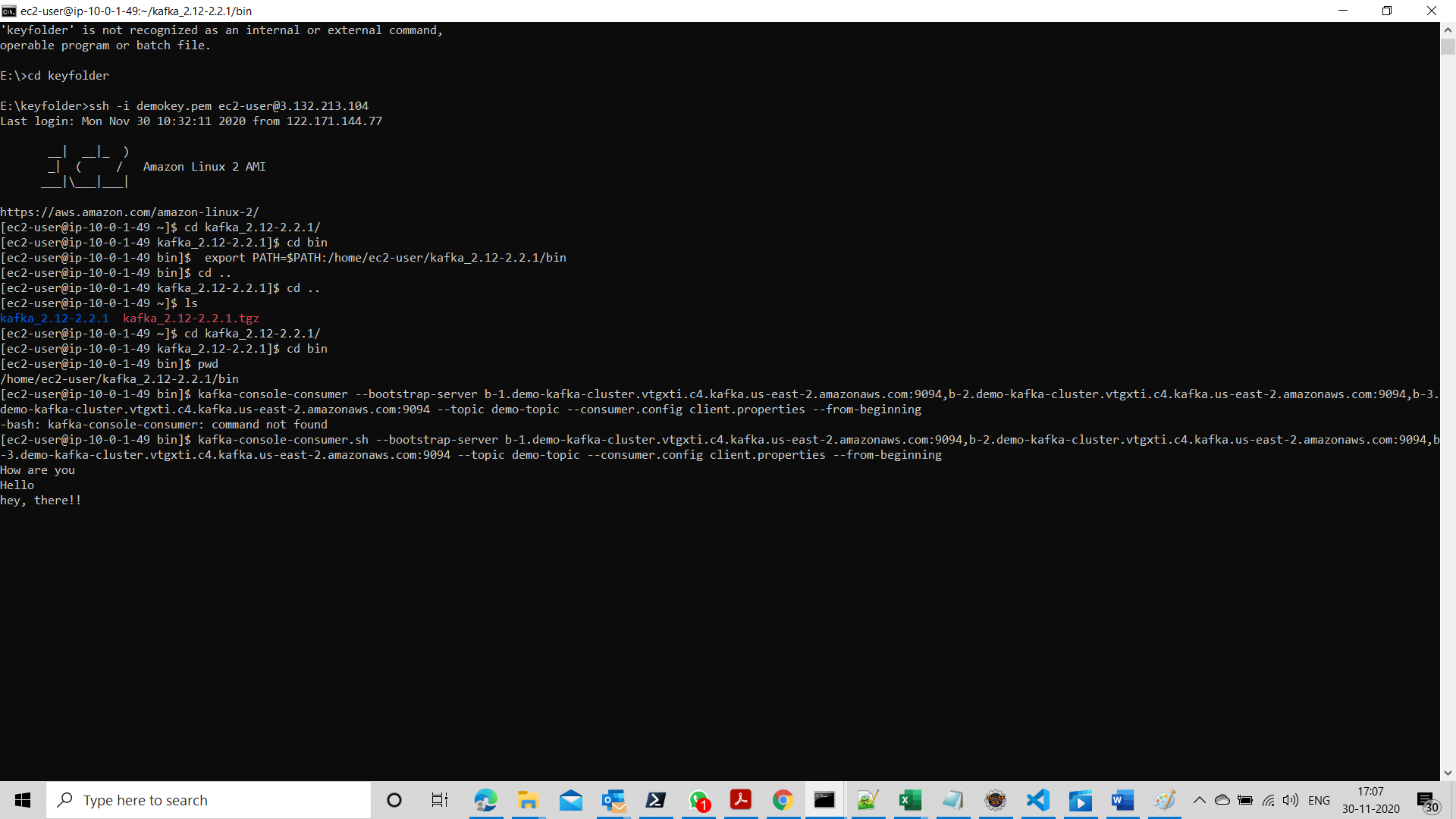
--bootstrap-server🡺 BootstrapBrokerStringTls” you got from step 26 and 27

--topic🡺 specifies the topic from where we are consuming messages.

--consumer.config🡺 points to the client.properties files which we created in step 22

--from-beginning🡺 specifies , read messages from the beginning.

1. You can see the output as shown below.



1. Now you can keep on typing in the producer terminal, you will be able the see the messages are immediately getting reflected in consumer terminal.