

Instructions for setting up and exploring the operation of a single VM version of Kafka

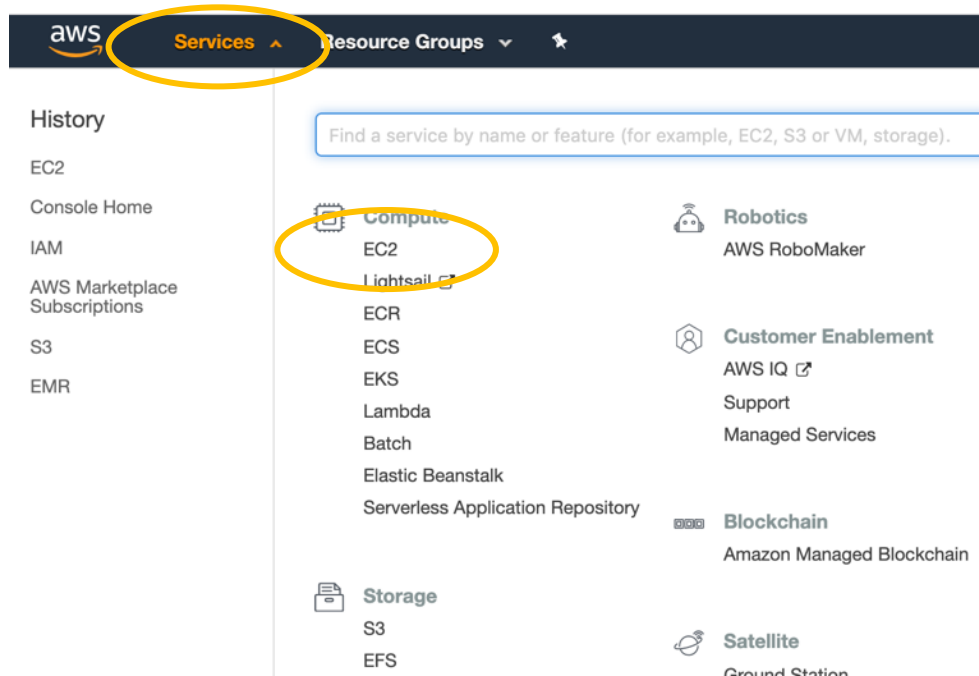
The AWS cloud offers a managed version of Kafka but it is complex to configure and potentially expensive to run. Instead we will use a well-supported open source version of Kafka and install and manage it ourselves on a single virtual machine.

Note, for those of you who might want to include Kafka in a big data processing pipeline, in the future, you might try to install it onto an EMR/Hadoop master node, following the instructions below, but for the following assignment, we will use a separate virtual machine.

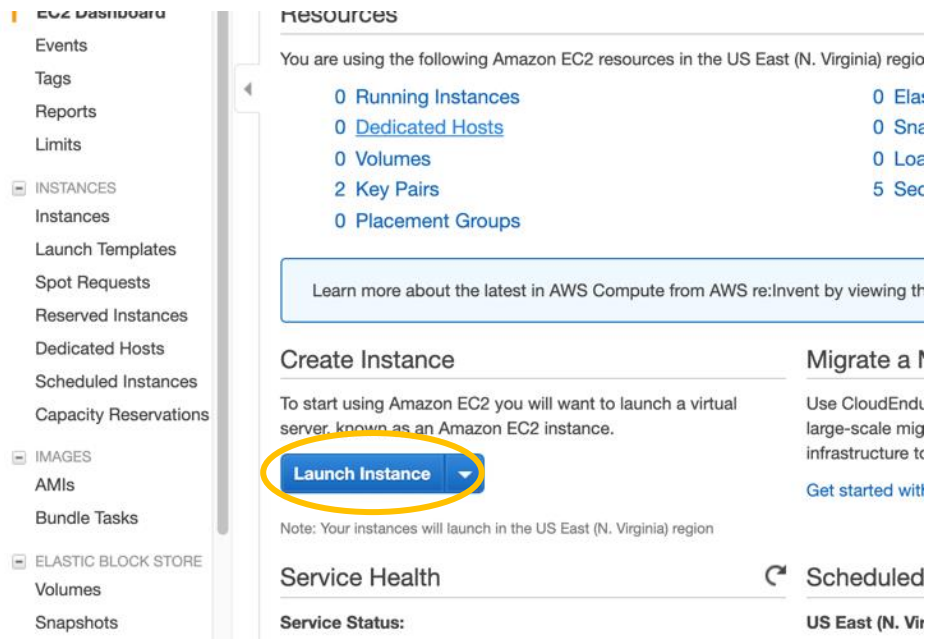
These instructions are divided into five sections. In the first we will set up an AWS EC2 instance (virtual machine). In the second section we will configure the virtual machine with the software and properties necessary to execute Kafka. In the third section we will learn how to start up Kafka and then, in the fourth section explore some of its capabilities. Finally, in the fifth section we describe how to stop, start or terminate your EC2 instance.

Section 1: Instructions for setting up an EC2 instance (virtual machine)

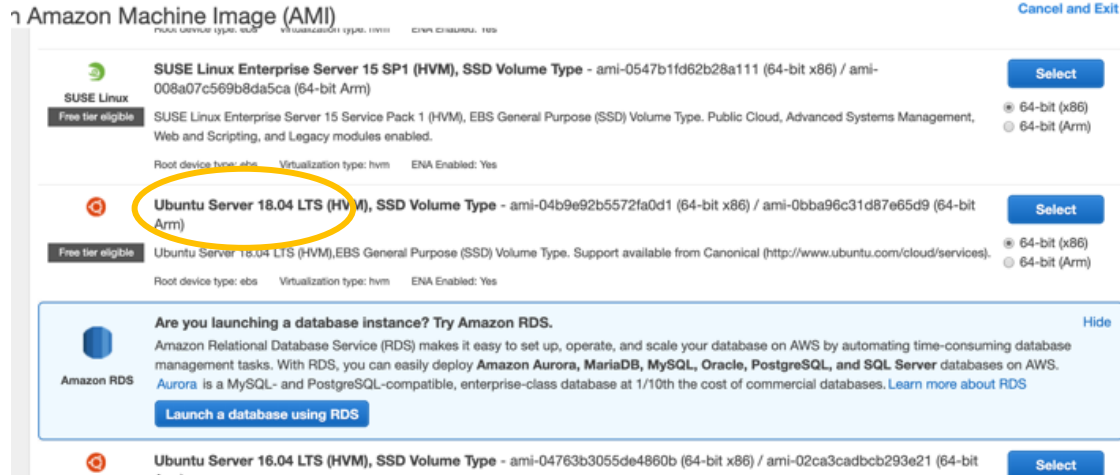
- 1) Select “Services” on the AWS console. Then select “EC2”



2) Launch a new EC2 instance by clicking on “Launch Instance”



3) Select the Ubuntu 18.04 Amazon Machine Image (AMI)



Select an instance (VM) type of m4.xlarge (or if for some reason this is not available to you, try m4.large):

Step 2: Choose an Instance Type

<input type="checkbox"/>	General purpose	m5.4xlarge	16	64	EBS only	Yes	Up to 10 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.8xlarge	32	128	EBS only	Yes	10 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.12xlarge	48	192	EBS only	Yes	10 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.16xlarge	64	256	EBS only	Yes	20 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.24xlarge	96	384	EBS only	Yes	25 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.metal	96	384	EBS only	Yes	25 Gigabit	Yes
<input type="checkbox"/>	General purpose	m4.xlarge	2	8	EBS only	Yes	Moderate	Yes
<input type="checkbox"/>	General purpose	m4.2xlarge	4	16	EBS only	Yes	High	Yes
<input type="checkbox"/>	General purpose	m4.4xlarge	8	32	EBS only	Yes	High	Yes
<input type="checkbox"/>	General purpose	m4.8xlarge	16	64	EBS only	Yes	High	Yes
<input type="checkbox"/>	General purpose	m4.10xlarge	40	160	EBS only	Yes	10 Gigabit	Yes
<input type="checkbox"/>	General purpose	m4.16xlarge	64	256	EBS only	Yes	25 Gigabit	Yes

Cancel Previous **Review and Launch** Next: Configure Instance Details

4) Select “Next: Configure Instance Details”

Step 2: Choose an Instance Type

<input type="checkbox"/>	General purpose	m5.4xlarge	16	64	EBS only	Yes	Up to 10 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.8xlarge	32	128	EBS only	Yes	10 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.12xlarge	48	192	EBS only	Yes	10 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.16xlarge	64	256	EBS only	Yes	20 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.24xlarge	96	384	EBS only	Yes	25 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5.metal	96	384	EBS only	Yes	25 Gigabit	Yes
<input type="checkbox"/>	General purpose	m4.xlarge	2	8	EBS only	Yes	Moderate	Yes
<input type="checkbox"/>	General purpose	m4.2xlarge	4	16	EBS only	Yes	High	Yes
<input type="checkbox"/>	General purpose	m4.4xlarge	8	32	EBS only	Yes	High	Yes
<input type="checkbox"/>	General purpose	m4.8xlarge	16	64	EBS only	Yes	High	Yes
<input type="checkbox"/>	General purpose	m4.10xlarge	40	160	EBS only	Yes	10 Gigabit	Yes
<input type="checkbox"/>	General purpose	m4.16xlarge	64	256	EBS only	Yes	25 Gigabit	Yes

Cancel Previous **Review and Launch** Next: Configure Instance Details

5) Select “Next: Add Storage”

Step 3: Configure Instance Details

Purchasing option ☐ Request Spot instances

Network [Create new VPC](#)

Subnet [Create new subnet](#)

Auto-assign Public IP

Placement group ☐ Add instance to placement group

Capacity Reservation [Create new Capacity Reservation](#)

IAM role [Create new IAM role](#)

CPU options ☐ Specify CPU options

Shutdown behavior

Enable termination protection ☐ Protect against accidental termination

Monitoring ☐ Enable CloudWatch detailed monitoring
[Additional charges apply.](#)

EBS-optimized instance ☒ Launch as EBS-optimized instance

Cancel Previous **Review and Launch** Next: Add Storage

- 6) Change the root storage size from 8 GiB to 30 GiB (if possible, otherwise as much as you can between 8 and 30 GiB). Then select “Review and Launch”

Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

Volume Type ⓘ	Device ⓘ	Snapshot ⓘ	Size (GiB) ⓘ	Volume Type ⓘ	IOPS ⓘ	Throughput (MB/s) ⓘ	Delete on Termination ⓘ	Encryption ⓘ
Root	/dev/sda1	snap-02e105f837cd927	8	General Purpose SSD (gp2)	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypted

[Add New Volume](#)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Tags](#)

7 Now “Launch” the instance

Step 7: Review Instance Launch

eligible Root Device Type: ebs Virtualization type: hvm

▼ Instance Type

[Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t3a.xlarge	Variable	4	16	EBS only	Yes	Up to 5 Gigabit

▼ Security Groups

[Edit security groups](#)

Security group name launch-wizard-2
Description launch-wizard-2 created 2019-10-23T13:12:36.320-05:00

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ	Description ⓘ
--------	------------	--------------	----------	---------------

This security group has no rules

▶ Instance Details

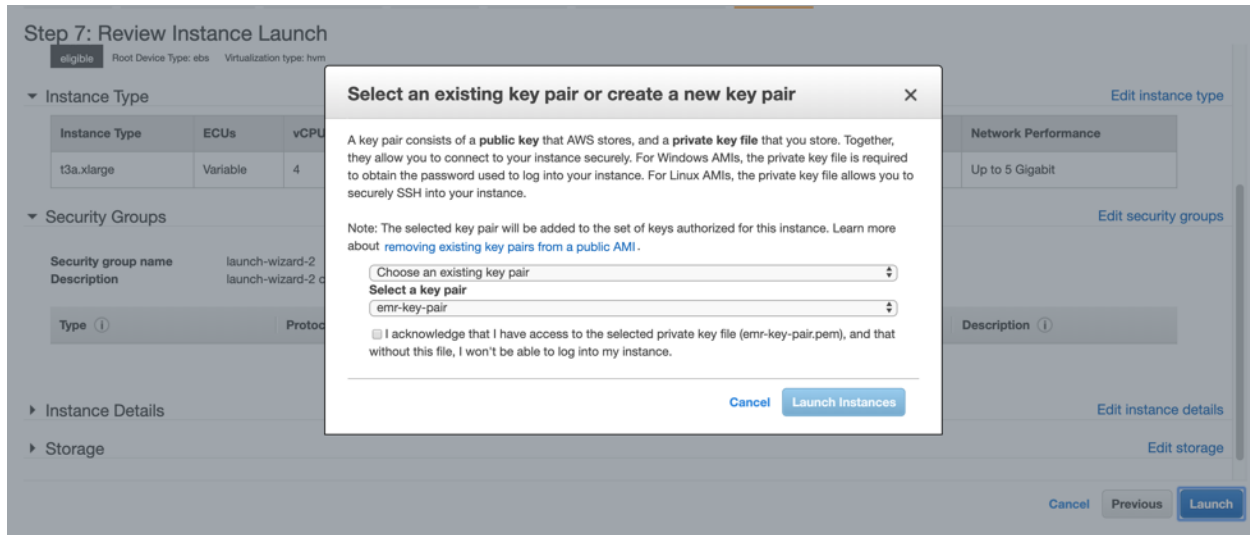
[Edit instance details](#)

▶ Storage

[Edit storage](#)

[Cancel](#) [Previous](#) [Launch](#)

- 8) Select an existing key pair (or create a new pair). You can use the same key pair you created for your EMR instances. Make sure to check the “I acknowledge...” checkbox.



- 9) You should see this. Select “View Instances”

Launch Status

The following instance launches have been initiated: [i-03dd5eb8288beffca](#) [View launch log](#)

Get notified of estimated charges
Create billing alerts to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier).

How to connect to your instances

Your instances are launching, and it may take a few minutes until they are in the **running** state, when they will be ready for you to use. Usage hours on your new instances will start immediately and continue to accrue until you stop or terminate your instances.

Click **View Instances** to monitor your instances' status. Once your instances are in the **running** state, you can **connect** to them from the Instances screen. [Find out](#) how to connect to your instances.

Here are some helpful resources to get you started

- [How to connect to your Linux instance](#)
- [Amazon EC2: User Guide](#)
- [Learn about AWS Free Usage Tier](#)
- [Amazon EC2: Discussion Forum](#)

While your instances are launching you can also

- [Create status check alarms](#) to be notified when these instances fail status checks. (Additional charges may apply)
- [Create and attach additional EBS volumes](#) (Additional charges may apply)
- [Manage security groups](#)

[View Instances](#)

10) Wait until the instance state is “running”. Note that the public address (Public DNS) is provided for use in connecting via ssh or scp to the instance.

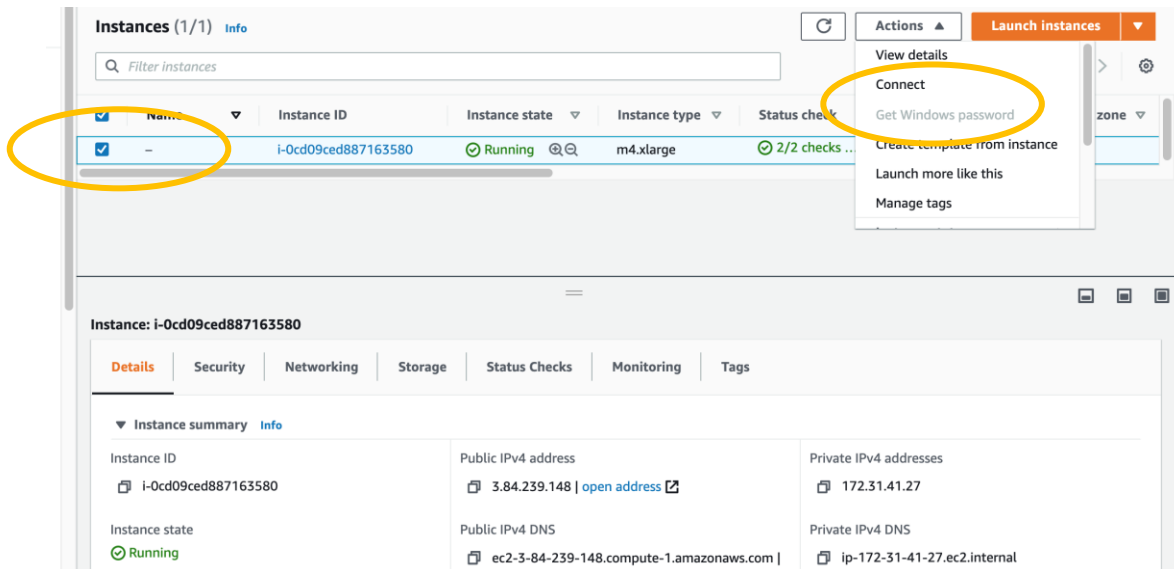
The screenshot displays the AWS Management Console interface for EC2 instances. At the top, there's a header for 'Instances (1/1)' with a search bar and a 'Launch instances' button. Below this is a table listing instances. The first instance, with ID 'i-0cd09ced887163580', is in the 'Running' state, which is circled in yellow. Its instance type is 'm4.xlarge'. Below the table, the 'Instance: i-0cd09ced887163580' details page is shown. The 'Instance summary' section contains a table with the following information:

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0cd09ced887163580	3.84.239.148 open address	172.31.41.27

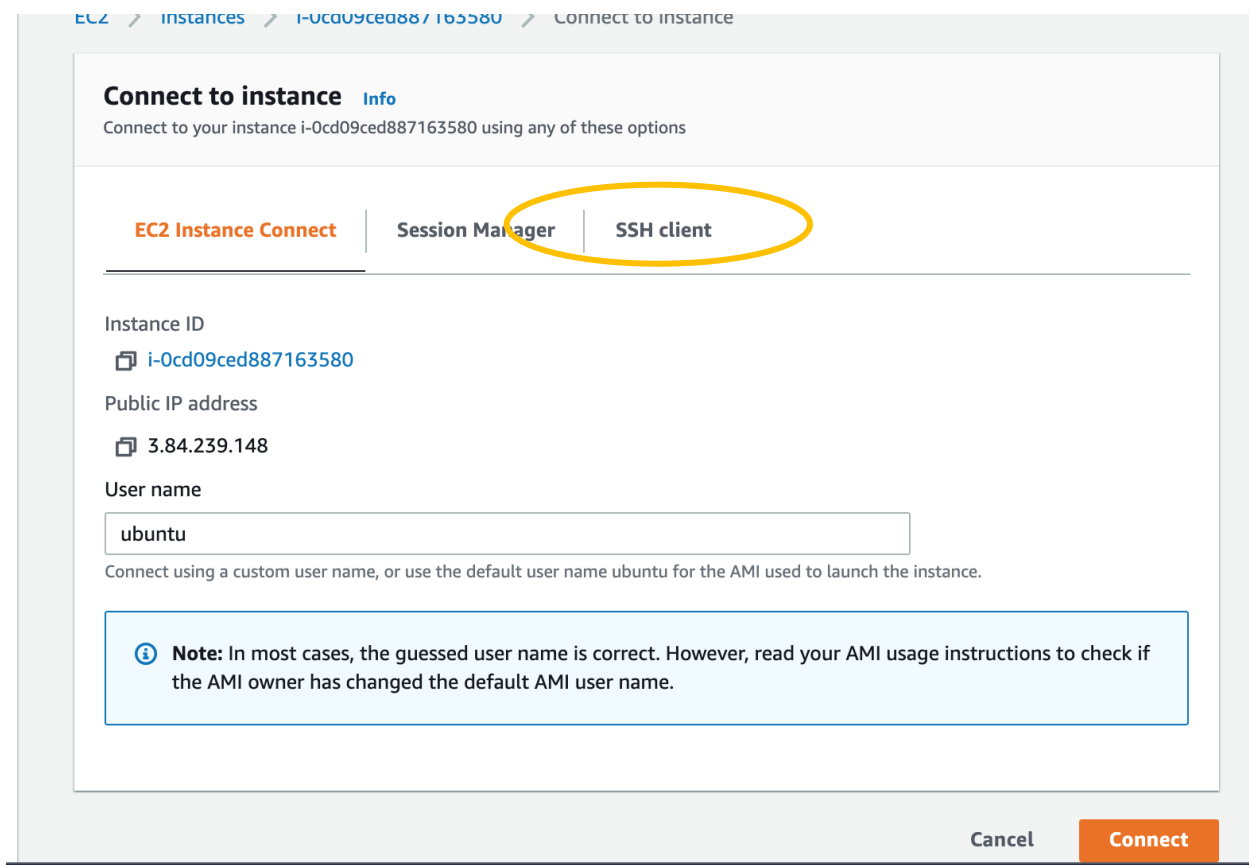
Below this, the 'Instance state' is shown as 'Running'. The 'Public IPv4 DNS' is listed as 'ec2-3-84-239-148.compute-1.amazonaws.com', and the 'Private IPv4 DNS' is 'ip-172-31-41-27.ec2.internal'. The public IPv4 address '3.84.239.148' and the public IPv4 DNS address are both circled in yellow.

10 Now you can ssh to the instance the easy way as follows.

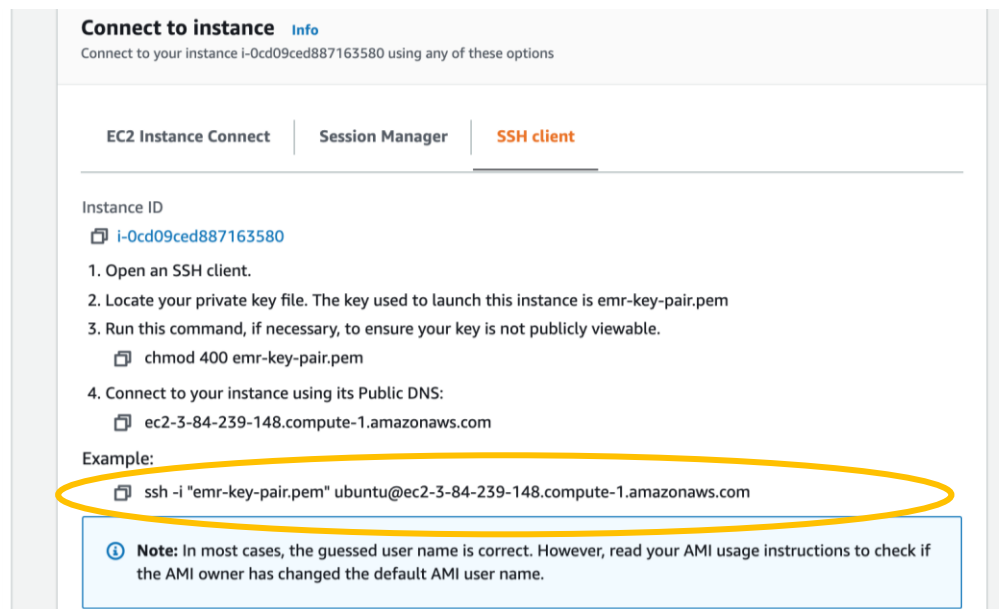
- Make sure you check the EC2 instance
- Click on the “Connect” button.



c) Select the “SSH Client” tab



d) Cut and paste the example ssh command that appears into your terminal to connect to your instance



Note that in this case the user name of your account is “ubuntu” and the address of your instance <Public DNS (IPv4)> is also used. So the ssh command about is built as follows. Note <Public DNS (IPv4)> is replaced with the actual value listed in the previously shown instance information pane.

11) About terminal windows...

You will need to open up several terminal windows to set up and interact with Kafka. We will refer to these terminal windows by a unique name, which will, we hope, keep things clearer and more organized.

Terminal Window “Name”	Connected to...	Purpose
KT-1	Local computer	Used to scp files from your local computer to the AWS virtual machine
KT-2	AWS virtual machine using ssh	Used to execute the basic commands on the AWS virtual machine necessary to install, configure and start Kafka. Also, used to execute Kafka data producers.
KT-3	AWS virtual machine using ssh	Used to execute Kafka data producers.
KT-4	AWS virtual machine using ssh	Used to execute Kafka data consumers and output the data they consume.

12) Now you can ssh to the EC2 instance as follows using the ssh command you just copied above:


```
ssh -i emr-key-pair.pem ubuntu@<Public DNS (IPv4)>
```

Section 2: Installing basic software onto your EC2 instance

- 1) Download the Kafka software (kafka_2.12-2.3.0.tgz) from the assignment to your PC or MAC
- 2) In a terminal window (KT-1) “scp” the confluent Kafka software to your EC2 instance:

```
scp -i emr-key-pair.pem /path/to/ kafka_2.12-2.3.0.tgz ubuntu@<Public DNS (IPv4)>:/home/ubuntu
```

- 3) If you haven’t already done so, open a terminal window and “ssh” to your EC2 instance. We will call this window, now connected to the EC2 instance, the KT-2 terminal window

```
ssh -i emr-key-pair.pem ubuntu@<Public DNS (IPv4)>
```

- 4) Now into the EC2 console accessible through your “KT-2” terminal window enter the following commands one at a time. As each executes, they may ask you to respond with “Yes” or “Y”. Do so each time. On your instance console enter the following:

- I. `sudo apt update`
- II. `sudo apt install zip`
- III. `sudo apt install default-jre`
- IV. `sudo apt install default-jdk`
- V. `sudo apt install python3-pip`
- VI. `pip3 install kafka-python`

- 5) Then through your “KT-2” terminal window enter the following commands

```
tar -xvf kafka_2.12-2.3.0.tgz
```

- 6) Add the install location of the Kafka “bin” directory to your PATH in the KT-2 terminal window as follows:

```
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
```

Section 3: Starting Kafka

We need to have three windows connected to the EC2 instance for this section. If you have one already recall we refer to it as the “KT-2” terminal but we will need to open and connect two more windows to the EC2 instance.

- 1) Open a third terminal window and enter the following. We will refer to this as the “KT-3” terminal window.

```
ssh -i emr-key-pair.pem ubuntu@<Public DNS (IPv4)>
```

Add the install location of the Kafka “bin” directory to your PATH in the KT-3 terminal window as follows:

```
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
```

- 2) Open a fourth terminal window and enter the following. We will refer to this as the “KT-4” terminal

```
ssh -i emr-key-pair.pem ubuntu@<Public DNS (IPv4)>
```

Add the install location of the Kafka “bin” directory to your PATH in the KT-1 window as follows:

```
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
```

- 3) Kafka uses Zookeeper so you need to first start a local ZooKeeper server in the KT-2 terminal window. You use the following command packaged with kafka to get a quick-and-dirty single-node Zookeeper instance. Notice the “&” at the end of the command. It instructs the command shell to run this command in the background. This will allow you to enter further commands into the KT-2 window:

```
zookeeper-server-start.sh config/zookeeper.properties &
```

Note: you may need to press “Enter” a time or two for the Linux prompt to reappear.

- 4) Now start the Kafka server itself in the KT-2 terminal window as follows. Again, notice the “&” at the end of the command:

```
kafka-server-start.sh config/server.properties &
```

Note: you may need to press “Enter” a time or two for the Linux prompt to reappear.

Section 4: Working with Kafka

- 1) Let's create a topic named "test" with a single partition and only one replica. Do so in the KT-3 terminal window:

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

- 2) We can now see that topic if we run the list topic command in the KT-3 terminal window:

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

- 3) Kafka comes with a command line client that will take input from a file or standard input and send it out as messages to the Kafka cluster. By default, each line will be sent as a separate message. Run the producer in the KT-3 terminal window and then type a few messages into the console to send to the server:

```
kafka-console-producer.sh --broker-list localhost:9092 --topic test
```

Upon execution of this command you should see a prompt (“>”) in a few moments. When you see the prompt, enter two or three lines of text.

- 4) Kafka also has a command line consumer that will dump out messages to standard output. Run the consumer in the KT-4 terminal window as follows:

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

If you have each of the above commands running in a different terminal then you should now be able to see messages you previously typed in and also type new messages into the producer terminal and see them appear in the consumer terminal.

- 5) To end this part of the demo type control-c into the KT-3 terminal window and KT-4 terminal window to stop the producer and consumer. But keep the KT-3 terminal window and KT-4 terminal window connected to your kafka VM.

- 6) Writing data from the console and writing it back to the console is a convenient place to start, but you'll probably want to use data from other sources or export data from Kafka to other systems. For many systems, instead of writing custom integration code you can use Kafka Connect to import or export data. Kafka Connect is a tool included with Kafka that imports and exports data to Kafka. It is an extensible tool that runs *connectors*, which implement the custom logic for interacting with an external system. Here we will see how to run Kafka Connect with simple connectors that import data from a file to a Kafka topic and export data from a Kafka topic to a file.

- 7) Now, we'll start by quickly creating a file to test with using the KT-3 terminal window:

```
echo -e "foo\nbar" > test.txt
```

- 8) Next, we'll start two connectors running in standalone mode, using the following command below in the KT-3 window, which means they run in a single, local, dedicated process. We provide three configuration files as parameters. The first is always the configuration for the Kafka Connect process, containing common configuration such as the Kafka brokers to connect to and the serialization format for data. The remaining configuration files each specify a connector to create. These files include a unique connector name, the connector class to instantiate, and any other configuration required by the connect

```
connect-standalone.sh config/connect-standalone.properties config/connect-file-source.properties config/connect-file-sink.properties
```

These sample configuration files, included with Kafka, use the default local cluster configuration you started earlier and create two connectors: the first is a source connector that reads lines from an input file and produces each to a Kafka topic and the second is a sink connector that reads messages from a Kafka topic and produces each as a line in an output file.

During startup you'll see a number of log messages, including some indicating that the connectors are being instantiated. Once the Kafka Connect process has started, the source connector should start reading lines from test.txt and producing them to the topic connect-test, and the sink connector should start reading messages from the topic connect-test and write them to the file test.sink.txt. We can verify the data has been delivered through the entire pipeline by examining the contents of the output file entering the following command in the KT-4 window

```
more test.sink.txt
```

And you should see the following:

```
foo
```

bar

Now type control-c into the KT-4 terminal window to stop the “more”

- 9) 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 KT-4 terminal 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
```

And you should see the following:

```
{"schema":{"type":"string","optional":false},"payload":"foo"}
```

```
{"schema":{"type":"string","optional":false},"payload":"bar"}
```

Now type control-c into the KT-4 window to stop the consumer.

- 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 KT-3 terminal 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
```

Section 5: Managing your EC2 instance

Your instance:

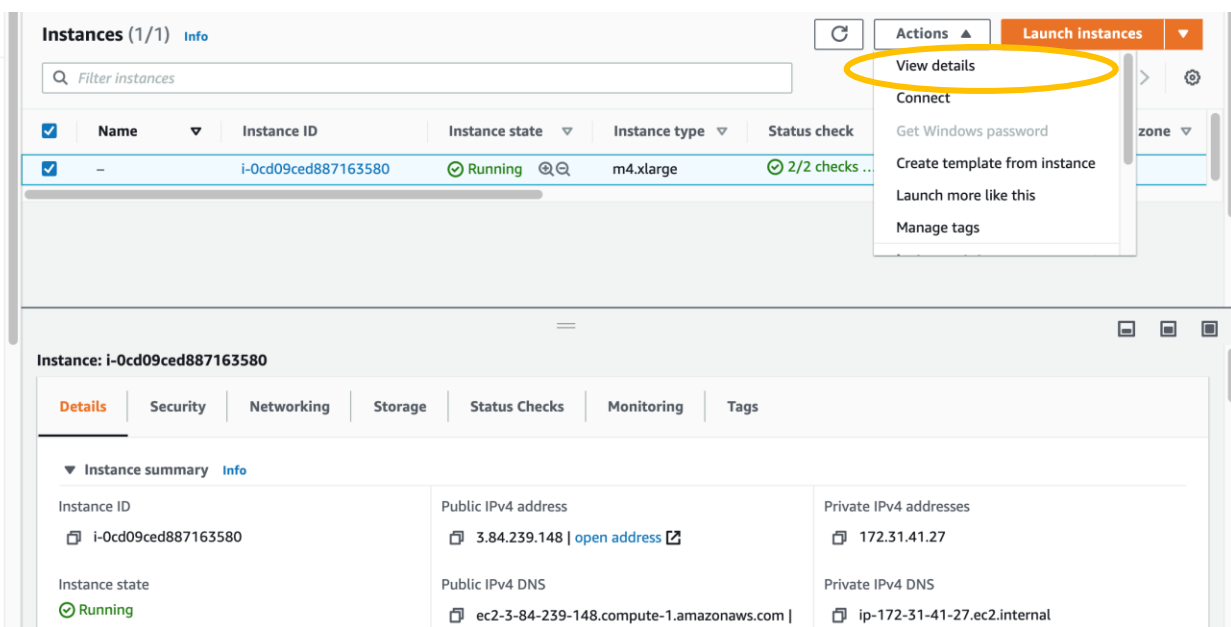
- Some basic facts about managing your instance. If you stop the instance you do not lose any of the software you installed but you will lose all topics and their contents. You will also stop the execution of zookeeper, kafka and any connectors and other related processes.

So, when you start the EC2 instance again, you will need to restart these processes and redefine any topics.

- When you terminate the instance you completely lose everything, including the software you installed.
- When you (re)start an instance it will get a new public DNS address, so make sure to find out what it is.

To manage an EC2 instance

- a) Click on the actions button
- b) Click “View Details”



Then select the “Actions” button and a list of actions will be displayed. Now select “Stop instance” “Start instance” or “Terminate instance” as required.

Instance summary for i-0cd09ced887163580 [Info](#)

Updated less than a minute ago



Connect

Actions

- Stop instance
- Start instance
- Hibernate instance
- Reboot instance
- Terminate instance

Instance ID

i-0cd09ced887163580

Instance state

Running

Instance type

m4.xlarge

IAM Role

-

Public IPv4 address

3.84.239.148 | [open address](#)

Public IPv4 DNS

ec2-3-84-239-148.compute-1.amazonaws.com
| [open address](#)

Elastic IP addresses

-

Subnet ID

subnet-9605f9aa

Private IPv4 addresses

172.31.41.27

Private IPv4 DNS

ip-172-31-41-27.ec2.internal

VPC ID

vpc-b02c12d7



AWS Compute Optimizer

Opt-in to AWS Compute Optimizer for recommendations.

[Learn more](#)

