

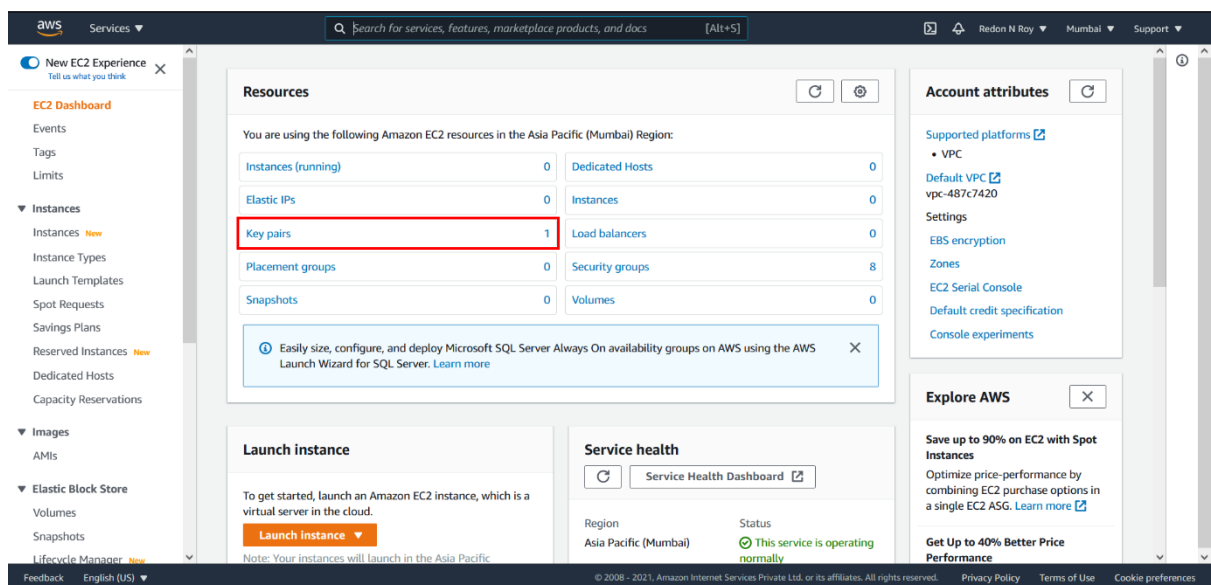
AWS EMR Demo

This file will help you to set up and perform hands-on with AWS EMR. We'll be creating the cluster and then run spark-submit in the EMR cluster.

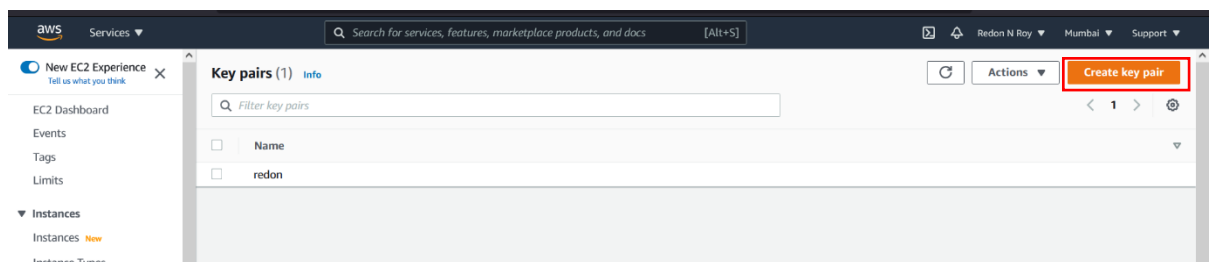
1. Creating key pair

Before starting with the creation of the cluster, we will be creating the key pair so that we can use it to access the cluster via SSH (Secure Shell). The steps to create a key pair are: -

- a. In the **AWS dashboard**, search for “**EC2**” in the search bar and navigate to the EC2 dashboard.



- b. On the EC2 dashboard, choose the **key pairs** option under the **resources** section. Inside that, choose the **Create key pair** option on the top right corner.



- c. Give a name to the key pair. Leave the type as **RSA**. The file format can be either **.pem** (Privacy Enhanced Mail) or **.ppk**.

If you are using **PuTTY**, then you must go with .ppk format or if you are using **OpenSSH**, then choose the .pem format.

For this demo, I'll be using OpenSSH (Instructions for PuTTY can be found at the end of the document). Then click on **Create key pair**.

Key pair
A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

Name
redon
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type [Info](#)
☒ RSA
☐ ED25519

Private key file format
☒ .pem
For use with OpenSSH
☐ .ppk
For use with PuTTY

Tags (Optional)
No tags associated with the resource.
[Add tag](#)
You can add 50 more tags.

Cancel [Create key pair](#)

- d. This will generate a key pair which we will be using while cluster creation.

2. Creating EMR cluster

Now that we have the key pair setup, we can proceed with the cluster creation. The steps for that are:-

- a. Go to the **EMR dashboard** and select the **Create cluster** option on the top left corner.

Name	ID	Status	Creation time (UTC+5:30)	Elapsed time	Normalized instance hours
My cluster	j-7LZ4RG0X2VYC	Terminated User request	2021-09-01 09:17 (UTC+5:30)	34 minutes	16
My cluster	j-2JEWOFCHL8SGQ	Terminated with errors Validation error	2021-09-01 09:14 (UTC+5:30)	26 seconds	0
demo_cluster	j-U3MKKB97A0P	Terminated User request	2021-09-01 09:09 (UTC+5:30)	1 minute	0

- b. Now provide the **cluster name** for the EMR cluster. You can turn **off** the **Logging** option. Select the **Launch mode** as **Cluster**. Then select the EMR configuration, here we'll be using the cluster for operations on **Spark**.

Create Cluster - Quick Options [Go to advanced options](#)

General Configuration

Cluster name: demo_spark

☐ Logging ⓘ

Launch mode: ☒ Cluster ⓘ ☐ Step execution ⓘ

Software configuration

Release: emr-5.33.0 ⓘ

Applications:

- ☐ Core Hadoop: Hadoop 2.10.1, Hive 2.3.7, Hue 4.9.0, Mahout 0.13.0, Pig 0.17.0, and Tez 0.9.2
- ☐ HBase: HBase 1.4.13, Hadoop 2.10.1, Hive 2.3.7, Hue 4.9.0, Phoenix 4.14.3, and ZooKeeper 3.4.14
- ☐ Presto: Presto 0.245.1 with Hadoop 2.10.1 HDFS and Hive 2.3.7 Metastore
- ☒ Spark: Spark 2.4.7 on Hadoop 2.10.1 YARN and Zeppelin 0.9.0
- ☐ Use AWS Glue Data Catalog for table metadata ⓘ

- c. Select the **Instance type** as per the requirement. Here we'll go with the default one. Set the **number of instances** according to the need. For the demo I'll go with 2. The select the key pair that we have created earlier in the **EC2 key pair**. Leave everything else at default. Finally click on Create cluster.

Hardware configuration

Instance type: **m5.xlarge** (The selected instance type adds 64 GiB of GP2 EBS storage per instance by default. [Learn more](#))

Number of instances: **2** (1 master and 1 core nodes)

Cluster scaling: ☐ scale cluster nodes based on workload

Security and access

EC2 key pair: **redon** ([Learn how to create an EC2 key pair.](#))

Permissions: ☒ Default ☐ Custom
Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.

EMR role: **EMR_DefaultRole** [Use EMR_DefaultRole_V2](#)

EC2 instance profile: **EMR_EC2_DefaultRole**

[Cancel](#) [Create cluster](#)

- d. Now the cluster will be created and started. It will take a **few minutes** to get the cluster up and running. We will be able to use the cluster once the status changes from **Starting** to **Waiting**.

Amazon EMR

Cluster: **demo_spark** **Waiting** Cluster ready after last step completed.

Summary | Application user interfaces | Monitoring | Hardware | Configurations | Events | Steps | Bootstrap actions

Summary

ID: j-2VK4IANM4ZS11
Creation date: 2021-09-01 19:30 (UTC+5:30)
Elapsed time: 5 minutes
After last step completes: Cluster waits
Termination protection: Off [Change](#)
Tags: [View All / Edit](#)
Master public DNS: ec2-13-233-207-49.ap-south-1.compute.amazonaws.com
[Connect to the Master Node Using SSH](#)

Configuration details

Release label: emr-5.33.0
Hadoop distribution: Amazon
Applications: Spark 2.4.7, Zeppelin 0.9.0
Log URI: --
EMRFS consistent view: Disabled
Custom AMI ID: --

Application user interfaces

Persistent user interfaces: [Spark history server](#), [YARN timeline server](#)
On-cluster user interfaces: [Not Enabled](#) [Enable an SSH Connection](#)

Network and hardware

Availability zone: ap-south-1a
Subnet ID: [subnet-79685911](#)
Master: **Bootstrapping** 1 m5.xlarge
Core: **Provisioning** 1 m5.xlarge
Task: --
Cluster scaling: Not enabled

Security and access

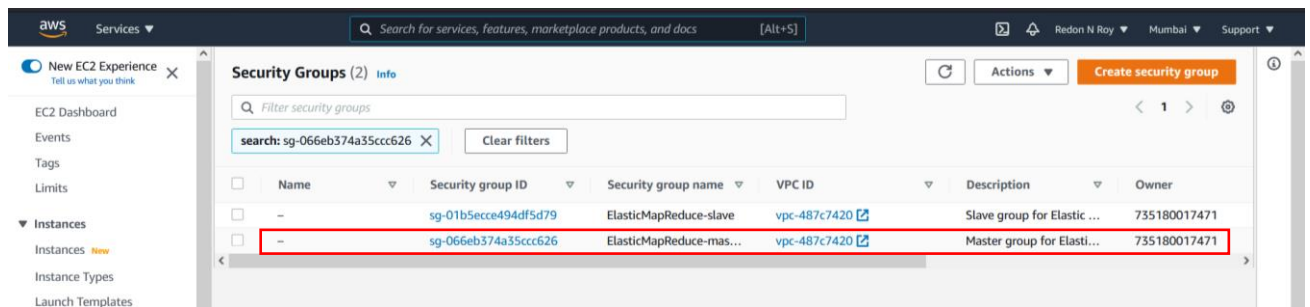
Key name: redon
EC2 instance profile: EMR_EC2_DefaultRole
EMR role: EMR_DefaultRole
Visible to all users: [All](#) [Change](#)
Security groups for Master: sg-066eb374a35ccc626 [\(ElasticMapReduce-master\)](#)
Security groups for Core & Task: [sg-01b5ecce494df5d79](#) [\(ElasticMapReduce-slave\)](#)

- e. Now we need to configure the Security groups to allow connection over SSH. For that go to the bottom on the cluster dashboard and choose the **Security group for Master**.

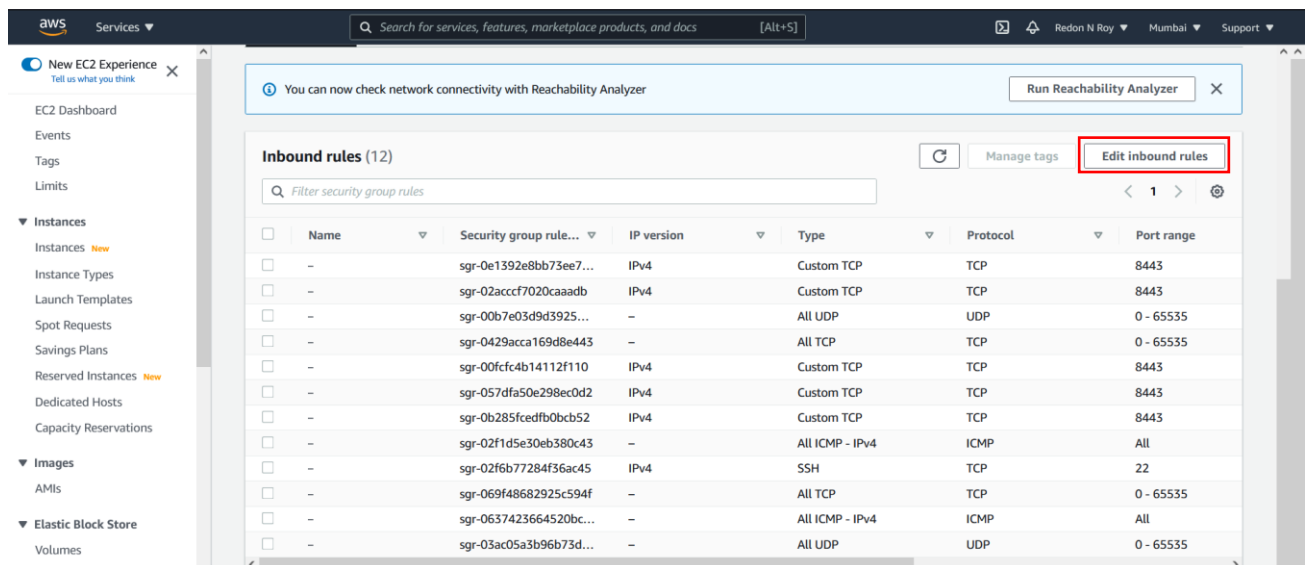
Security and access

Key name: redon
EC2 instance profile: EMR_EC2_DefaultRole
EMR role: EMR_DefaultRole
Visible to all users: [All](#) [Change](#)
Security groups for Master: sg-066eb374a35ccc626 [\(ElasticMapReduce-master\)](#)
Security groups for Core & Task: [sg-01b5ecce494df5d79](#) [\(ElasticMapReduce-slave\)](#)

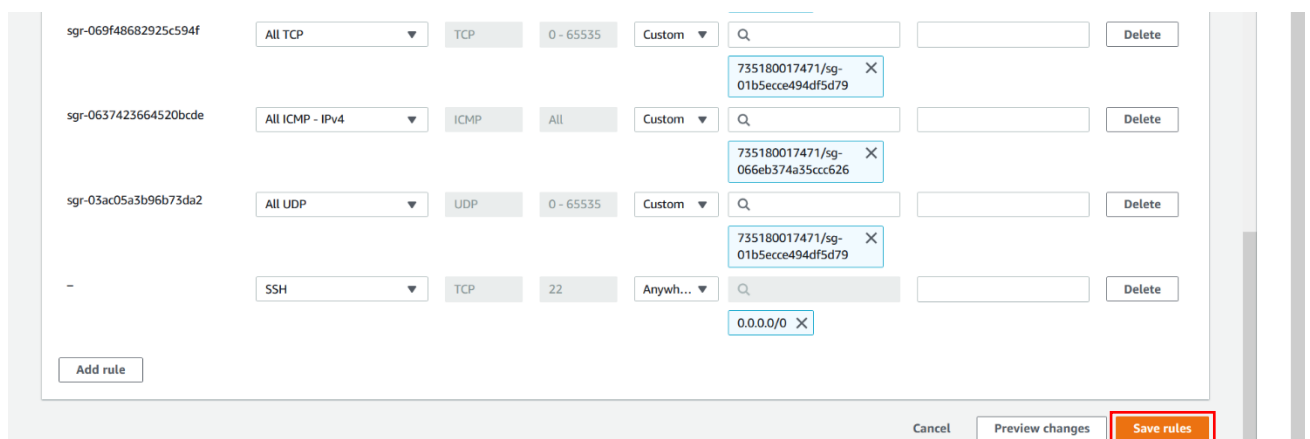
- f. In there, choose the security group for the **master node**.



- g. In the Inbound rules section, choose **Edit inbound rules** option on the right corner.



- h. Go to the bottom of the Inbound rules and click on **Add rules**. Now select the **Type** as **SSH** and the **connection** to be **IPv4 – Anywhere**. Finally click on **Save rules**.



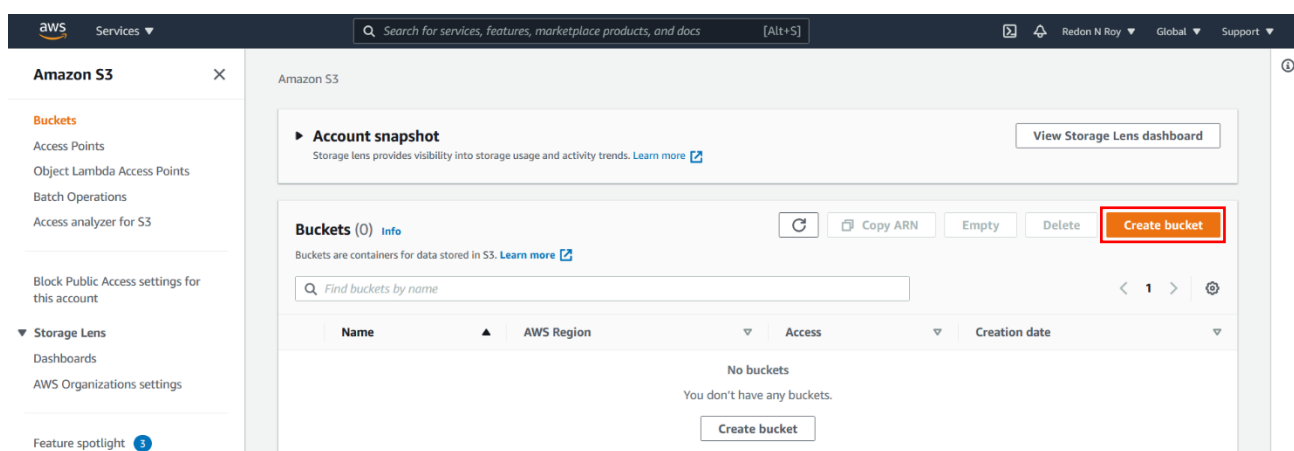
- i. That's it. The EMR cluster is fully configured.

3. Uploading Dataset and Program to S3

Before we can start processing the data on the EMR cluster, we need to put the dataset and the pyspark program into the S3 bucket so that it can be accessed easily by the EMR cluster. Since we need the file path to read the CSV file into the dataframe in pyspark program, we'll be uploading the dataset followed by the pyspark program. The steps include: -

Note: - The dataset and Pyspark programs can be found here <https://tinyurl.com/aws-emr-demo>

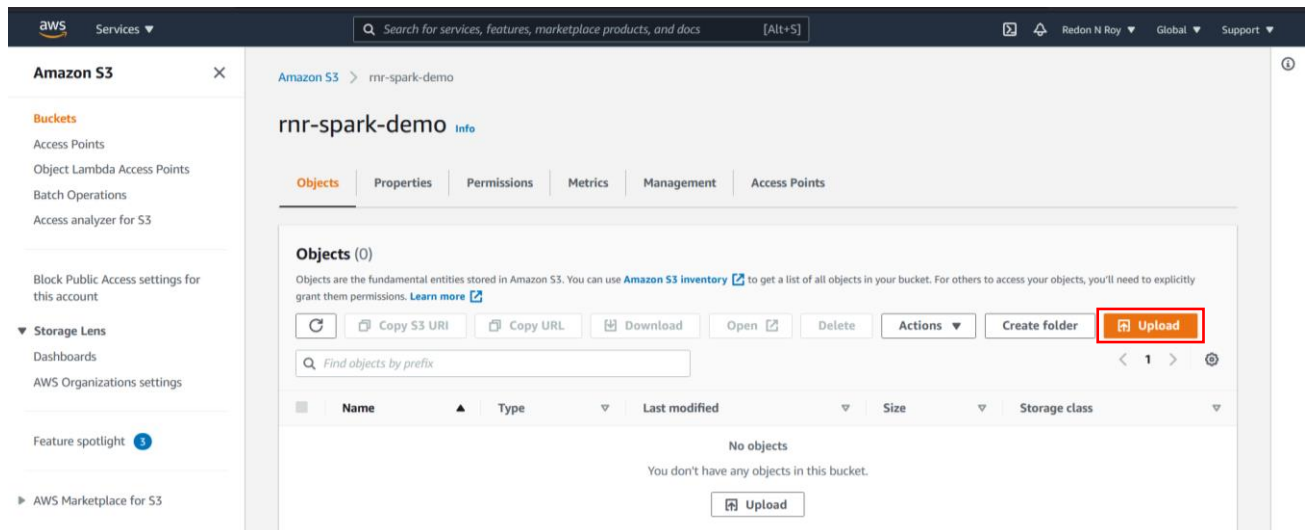
- a. We need to go to the S3 dashboard and setup a bucket there to hold all our data. Click on Create bucket option to create a new S3 bucket.



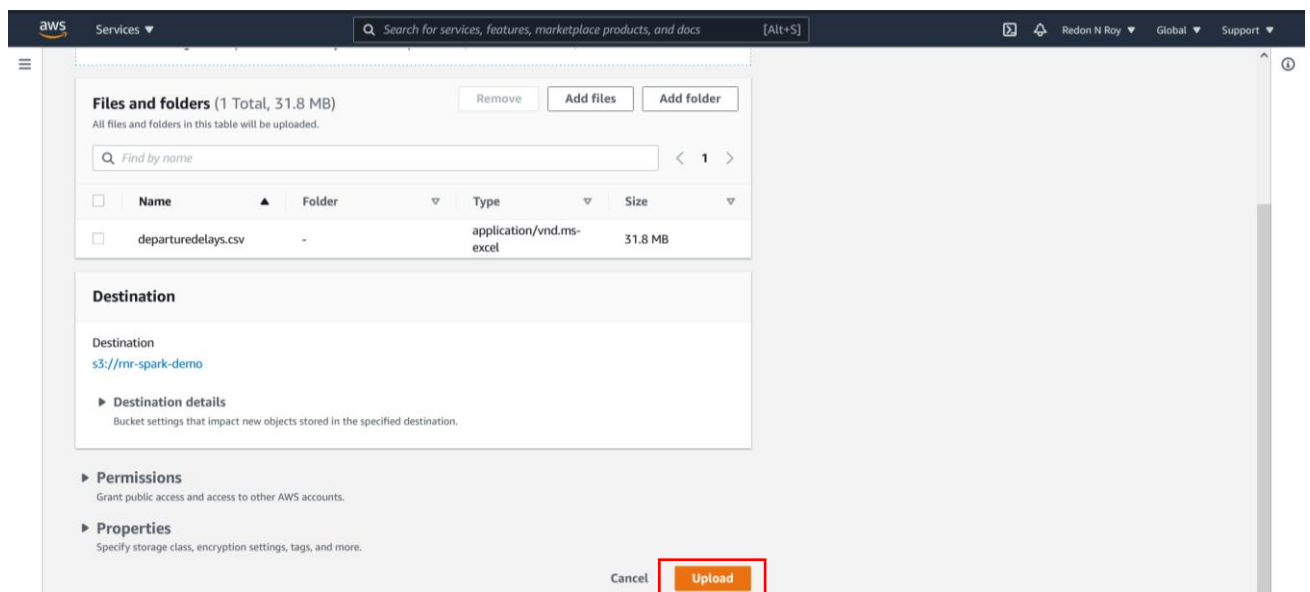
- b. In the create bucket option, we need to give a bucket name that is globally unique. Leave all the other options as default and press the Create bucket option at the bottom of the screen.

The screenshot displays the 'Create bucket' configuration page. Under 'General configuration', the 'Bucket name' is 'rnr-spark-demo' and the 'AWS Region' is 'Asia Pacific (Mumbai) ap-south-1'. The 'Default encryption' section shows 'Server-side encryption' set to 'Disable'. At the bottom, there is a 'Create bucket' button highlighted with a red box, and a 'Cancel' button next to it. A blue information box at the bottom states: 'After creating the bucket you can upload files and folders to the bucket, and configure additional bucket settings.'

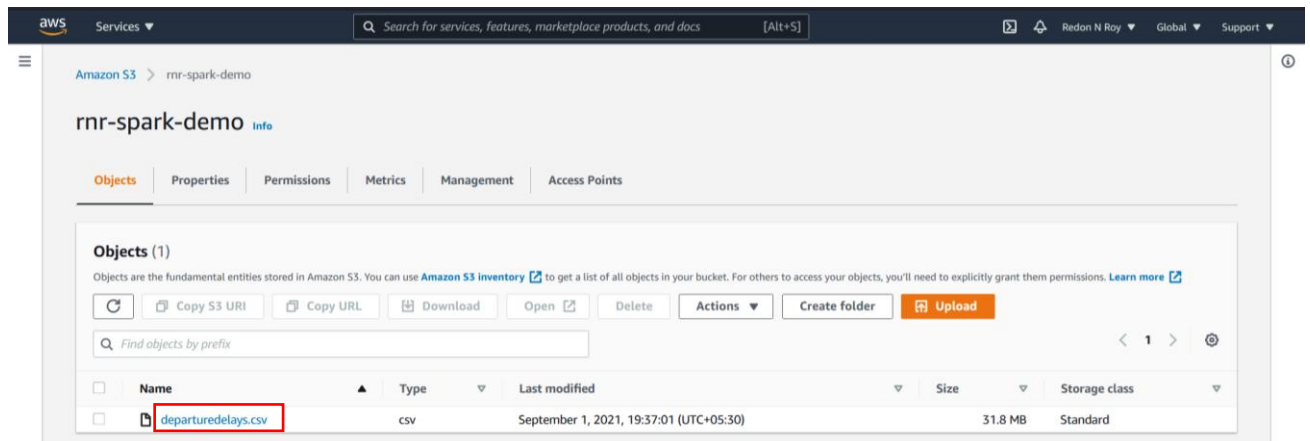
- c. Now that we have created a bucket, we can upload the dataset into the bucket. For that, we can click on the **Upload** button.



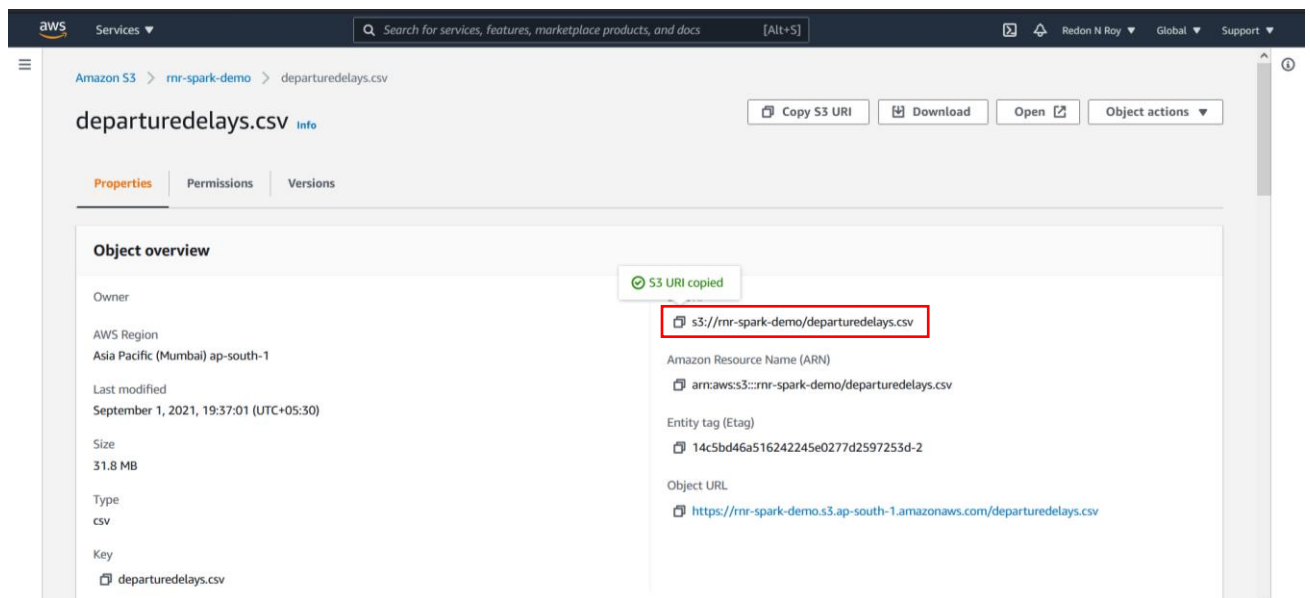
- d. Now choose the CSV file that you want to upload. Keep all other option as default. Finally click **Upload**.



- e. You would be able to see the file in the S3 bucket that you have created. From here go into the dataset to get the S3 URI.



- f. Now copy the S3 URI on the right side so that the path to the dataset can be added to the pyspark program.



- g. Now let's add this URI to both of the pyspark program. The path should be added to the variable named "csv_file" in both the programs.



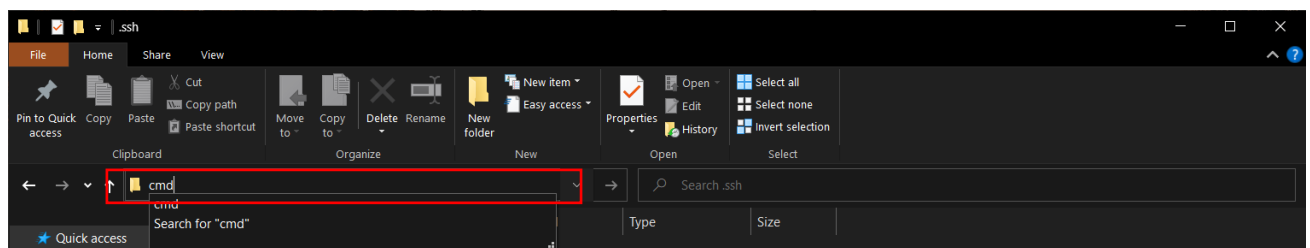
- h. Now we can upload both pyspark files to the S3 bucket in the same way we uploaded the dataset.

4. Running spark-submit on EMR cluster

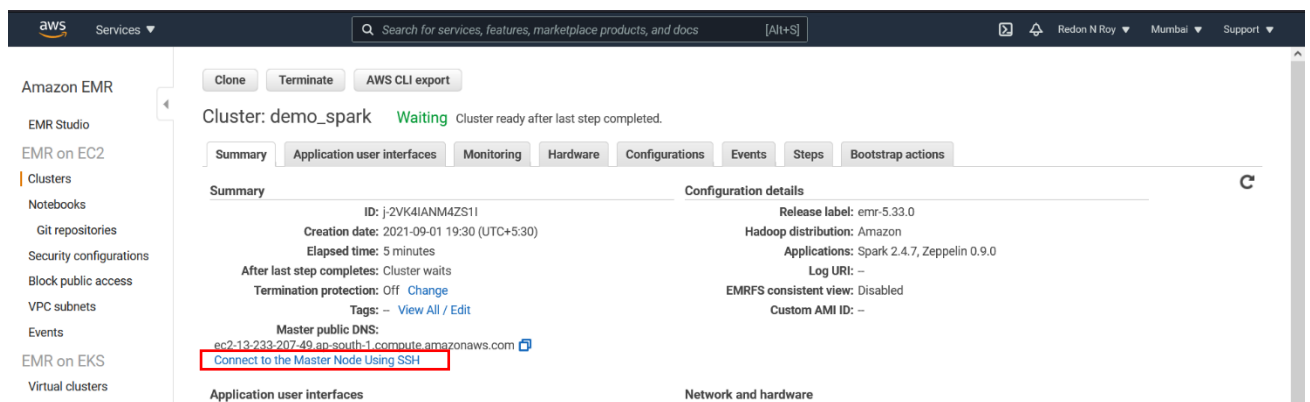
Now we are all set to run the pyspark program on the EMR cluster. For that we need to connect to the master node of the cluster via SSH. As discussed before, we can do it either through OpenSSH or with PuTTY. To connect to the cluster, follow the steps according to your preference of SSH connection.

Using OpenSSH

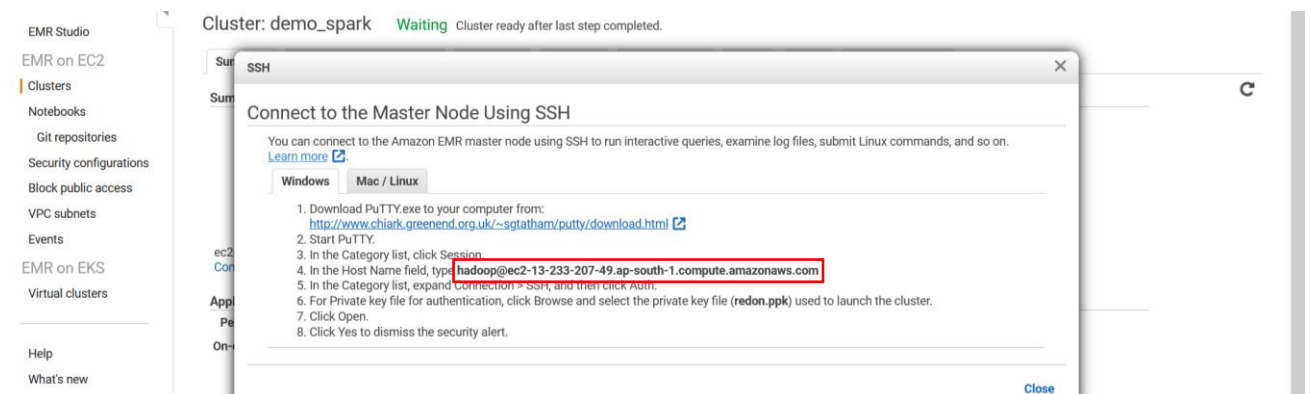
1. Open the folder where you have downloaded the key for the cluster (usually the downloads folder). Go to the address bar and type “cmd” and press enter. This will open a Command Prompt in that location. For me I had the key in a different folder.



2. Once you have the CMD open, go to the EMR cluster dashboard and click on the option “Connect to the Master Node Using SSH”.



3. In there, find and copy the Host name as we need it to connect to the cluster.



4. Now back in the CMD type the command in the following format and press enter.

ssh -i key_file_name.pem host_name

E.g.: `ssh -i redon.pem hadoop@ec2-13-233-207-49.ap-south-1.compute.amazonaws.com`

When connecting for the first time, it will ask permission to add to known host. Type yes and press enter.

```
C:\Users\redon\.ssh>ssh -i redon.pem hadoop@ec2-13-233-207-49.ap-south-1.compute.amazonaws.com
```

5. Now you'll be welcomed with a similar screen that confirms that you are connected to the EMR cluster.

```
  _|_  _|_  )
 _|_ ( _|_ /
 _|\_|_|_|

Amazon Linux 2 AMI

https://aws.amazon.com/amazon-linux-2/
63 package(s) needed for security, out of 106 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRRRRRRRRRRR
E::::::::::::::::::::E M::::::::M M::::::::M R::::::::::::R
EE::::::::::::::::::::E M::::::::M M::::::::M R::::::::RRRRRR::::R
E:::E EEEEE M::::::::M M::::::::M RR:::R R:::R
E:::E M::::::::M M::M M::M M::M R:::R R:::R
E:::EEEEEEEEEE M::M M::M M::M M::M R::RRRRRR::::R
E::: M::M M::M M::M M::M R::RRRRRR::::RR
E:::EEEEEEEEEE M::M M::M M::M R::RRRRRR::::R
E:::E M::M M::M M::M R:::R R:::R
E:::E EEEEE M::M M::M M::M R:::R R:::R
EE:::EEEEEEEEEE E M::M M::M R:::R R:::R
E::: M::M M::M RR:::R R:::R
EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRR RRRRRR

[hadoop@ip-172-31-43-144 ~]$
```

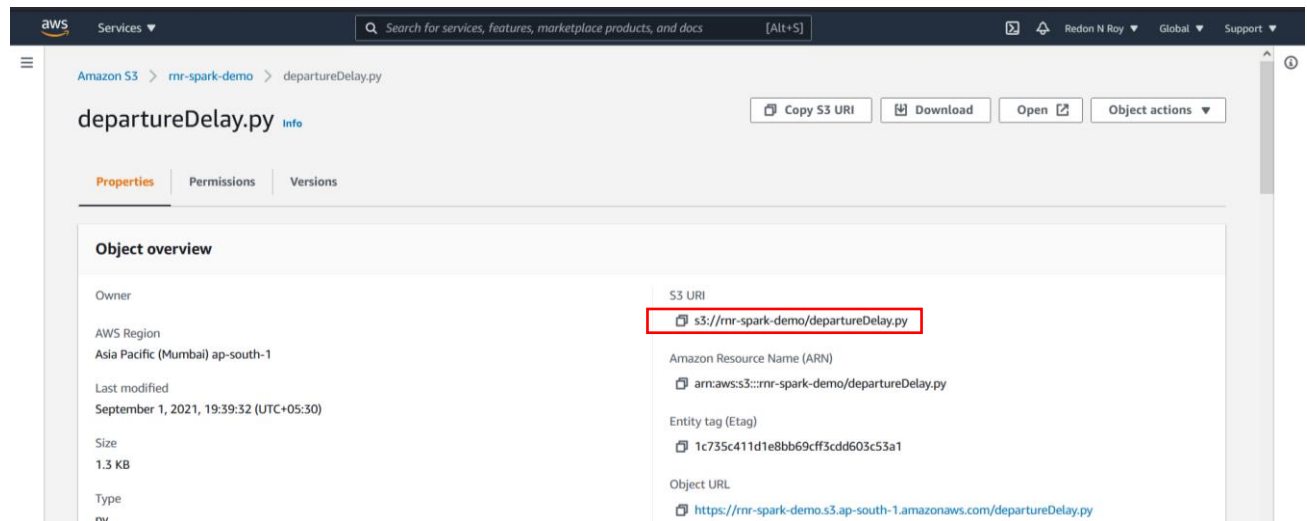
Using PuTTY

(I haven't tried this method personally; the instructions are being provided by AWS.)

1. Open PuTTY.
2. In the Category list, click Session.
3. In the Host Name field, type your host name, for example:
`hadoop@ec2-13-233-207-49.ap-south-1.compute.amazonaws.com`
4. In the Category list, expand Connection and choose SSH. Then click on Auth.
5. Now click on browse and select the .ppk file downloaded earlier. Then click open to add the key to PuTTY.
6. Finally click Yes to dismiss the security alert.
7. This should connect you to the cluster through PuTTY.

Once connected to the cluster by any one of the two methods, we can now run the spark-submit through the terminal. For that the steps are: -

- a. First go to the S3 bucket and get the S3 URI of the pyspark program so that we can download the same into the cluster for running it.



- b. Now go back to the terminal and use the following command to download the pyspark program into the cluster.

Aws s3 cp S3_URI .

E.g. : `aws s3 cp s3://rnr-spark-demo/departureDelay.py .`

(DON'T FORGET TO PUT A SPACE AND A DOT(.) AFTER THE S3 URI)

```
[hadoop@ip-172-31-43-144 ~]$ aws s3 cp s3://rnr-spark-demo/departureDelay.py .
download: s3://rnr-spark-demo/departureDelay.py to ./departureDelay.py
```

- c. Now run spark-submit with the pyspark program and it will produce the output of the queries.

```
[hadoop@ip-172-31-43-144 ~]$ spark-submit departureDelay.py
```



Getting output to S3

- d. Now we can run the pyspark program to get output to the S3 bucket. For that let's get the second pyspark program in the same way as the first program using S3 URI.

```
hadoop@ip-172-31-43-144:~$ aws s3 cp s3://rnr-spark-demo/outputdepartureDelay.py .
download: s3://rnr-spark-demo/outputdepartureDelay.py to ./outputdepartureDelay.py
hadoop@ip-172-31-43-144 ~]$
hadoop@ip-172-31-43-144 ~]$
hadoop@ip-172-31-43-144 ~]$
hadoop@ip-172-31-43-144 ~]$
hadoop@ip-172-31-43-144 ~]$ spark-submit outputdepartureDelay.py
```

- e. It will store the output into the S3 bucket in the folder we have mentioned. The file will be stored in parts if the coalesce(1) is not given along with the write command.

Amazon S3 > rnr-spark-demo

rnr-spark-demo Info

Objects (4)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 Inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

Name	Type	Last modified	Size	Storage class
departureDelay.py	py	September 1, 2021, 19:39:32 (UTC+05:30)	1.3 KB	Standard
departuredelays.csv	csv	September 1, 2021, 19:37:01 (UTC+05:30)	31.8 MB	Standard
output/	Folder	-	-	-
outputdepartureDelay.py	py	September 1, 2021, 19:52:12 (UTC+05:30)	1.6 KB	Standard

Amazon S3 > rnr-spark-demo

rnr-spark-demo Info

Objects (27)

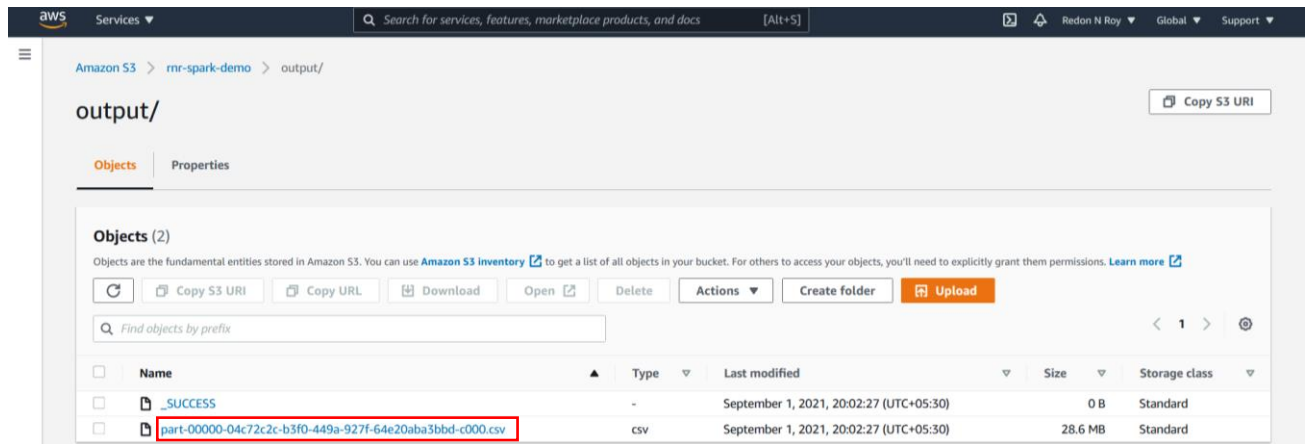
Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 Inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

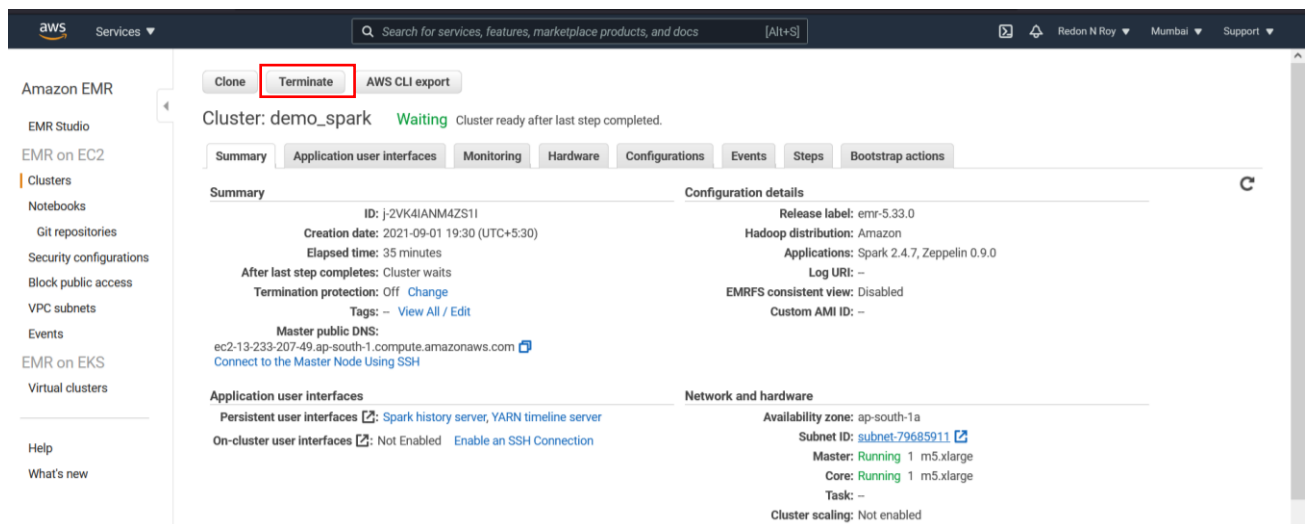
Name	Type	Last modified	Size	Storage class
_SUCCESS	-	September 1, 2021, 19:54:56 (UTC+05:30)	0 B	Standard
part-00000-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:54 (UTC+05:30)	1.3 MB	Standard
part-00001-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:54 (UTC+05:30)	1.0 MB	Standard
part-00002-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:54 (UTC+05:30)	1.1 MB	Standard
part-00003-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:54 (UTC+05:30)	1.1 MB	Standard
part-00004-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:55 (UTC+05:30)	1.1 MB	Standard
part-00005-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:55 (UTC+05:30)	1.2 MB	Standard
part-00006-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:55 (UTC+05:30)	1.2 MB	Standard
part-00007-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:55 (UTC+05:30)	1.1 MB	Standard
part-00008-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:55 (UTC+05:30)	1.1 MB	Standard
part-00009-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:55 (UTC+05:30)	1.1 MB	Standard
part-00010-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:55 (UTC+05:30)	1.2 MB	Standard
part-00011-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:55 (UTC+05:30)	1.1 MB	Standard
part-00012-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:56 (UTC+05:30)	1.1 MB	Standard
part-00013-b8a444b2-f597-4697-8b98-c9c9ae571a71-c000.csv	csv	September 1, 2021, 19:54:56 (UTC+05:30)	1.1 MB	Standard

- f. With the addition of coalesce(1), the entire output will be stored in a single file.

```
45 df2.coalesce(1).write.mode("overwrite").csv("s3://rnr-spark-demo/output")
46
```

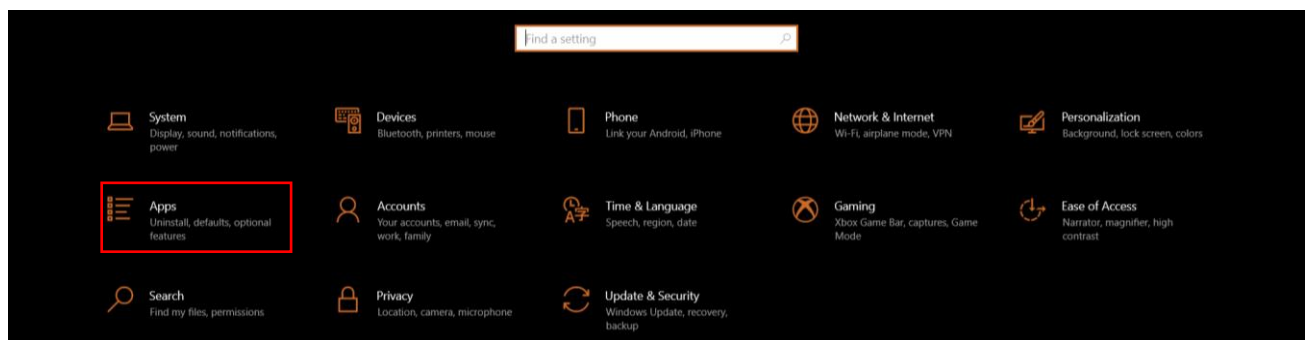


- g. Once you are done with working on the EMR cluster, you can go to the EMR cluster management dashboard and terminate the session.

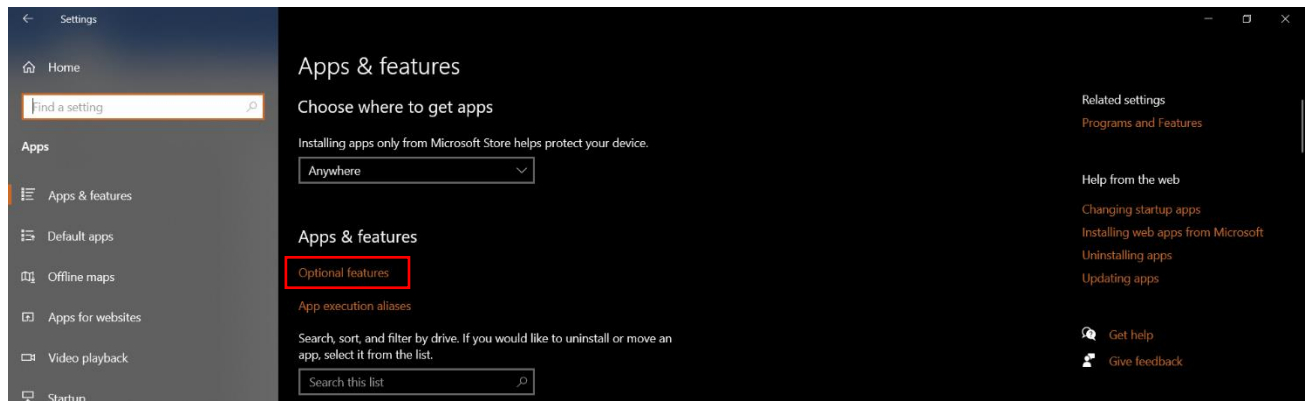


How to install OpenSSH in Windows 10

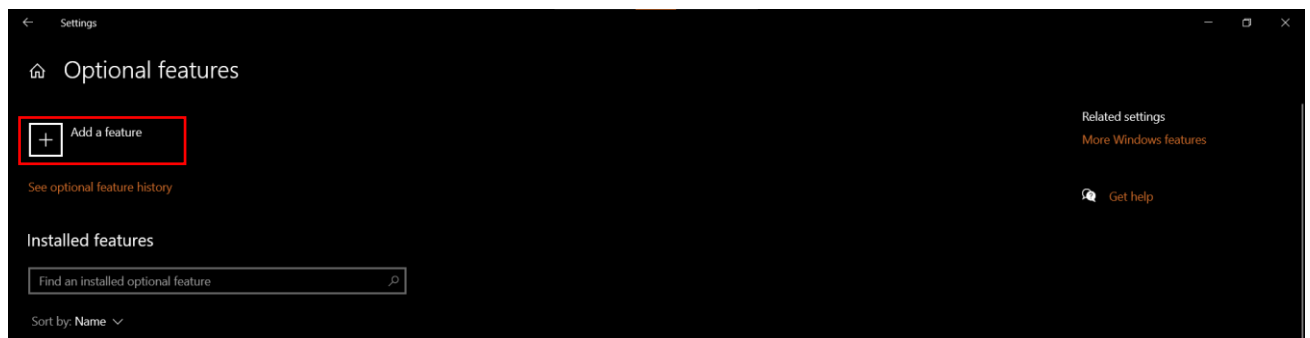
1. Go to setting in Windows 10 and select the Apps option.



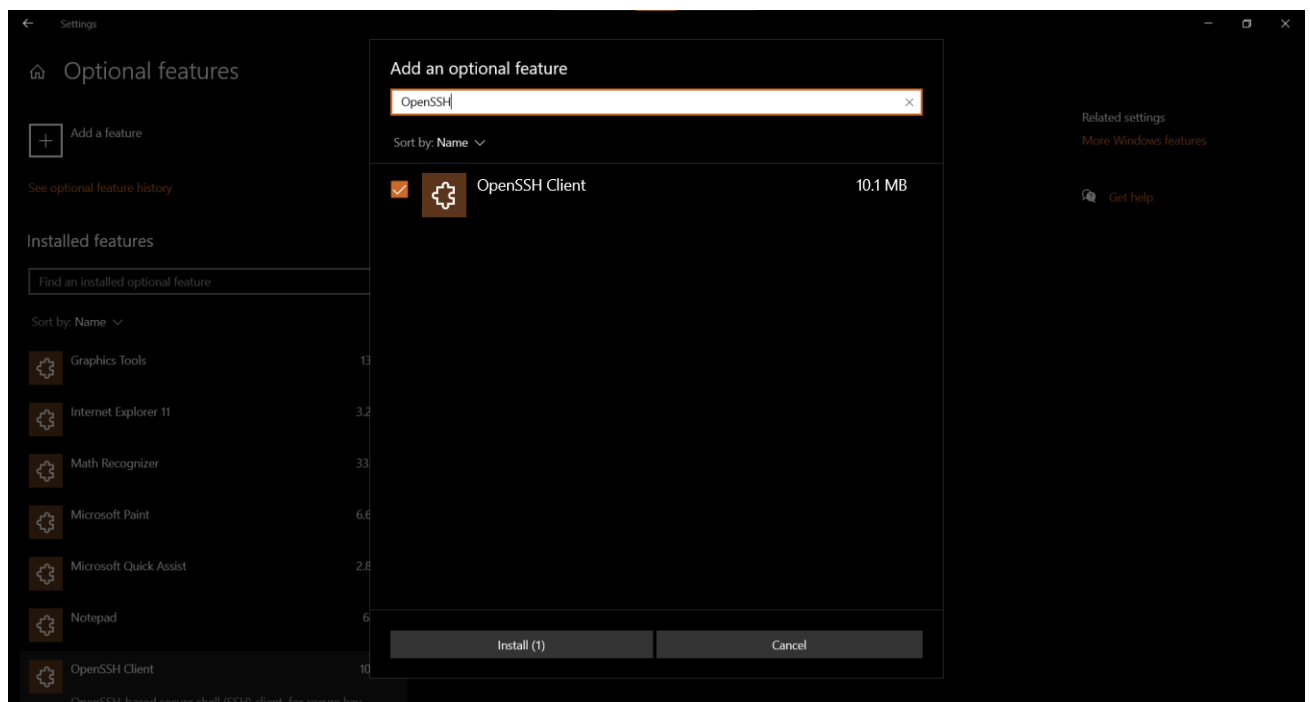
2. In Apps option, go to the optional features.



3. Click on Add a feature



4. Now search and install **OpenSSH client**.



5. Now OpenSSH is installed on your system successfully. You can use it directly from the CMD.
In case if it is not working, just restart your system and it will be available.