Big Data Wrangling With Google Books Ngrams

Table of Contents

Introduction

The scope of this data processing and analysis report is to document the workflow involved in filtering and reducing big data of Google Ngrams down to a manageable size, and then doing some analysis locally on our machine after extracting data.

The Google Ngrams dataset was created by Google's research team by analyzing all of the content in Google Books - these digitized texts represent approximately 4% of all books ever printed, and span a time period from the 1800s into the 2000s.
The dataset is hosted in a public S3 bucket as part of the Amazon S3 Open Data Registry. This data has been converted to CSV and hosted on a public S3 bucket.

1. Spin up a new EMR cluster on AWS for using Spark and EMR notebooks.
2. Copy data from S3 to HDFS
3. Analyze and filter data using Spark.
4. Read filtered data on local machine
5. Plot the number of occurrences of the token (the frequency column) of data over the years.

Goal

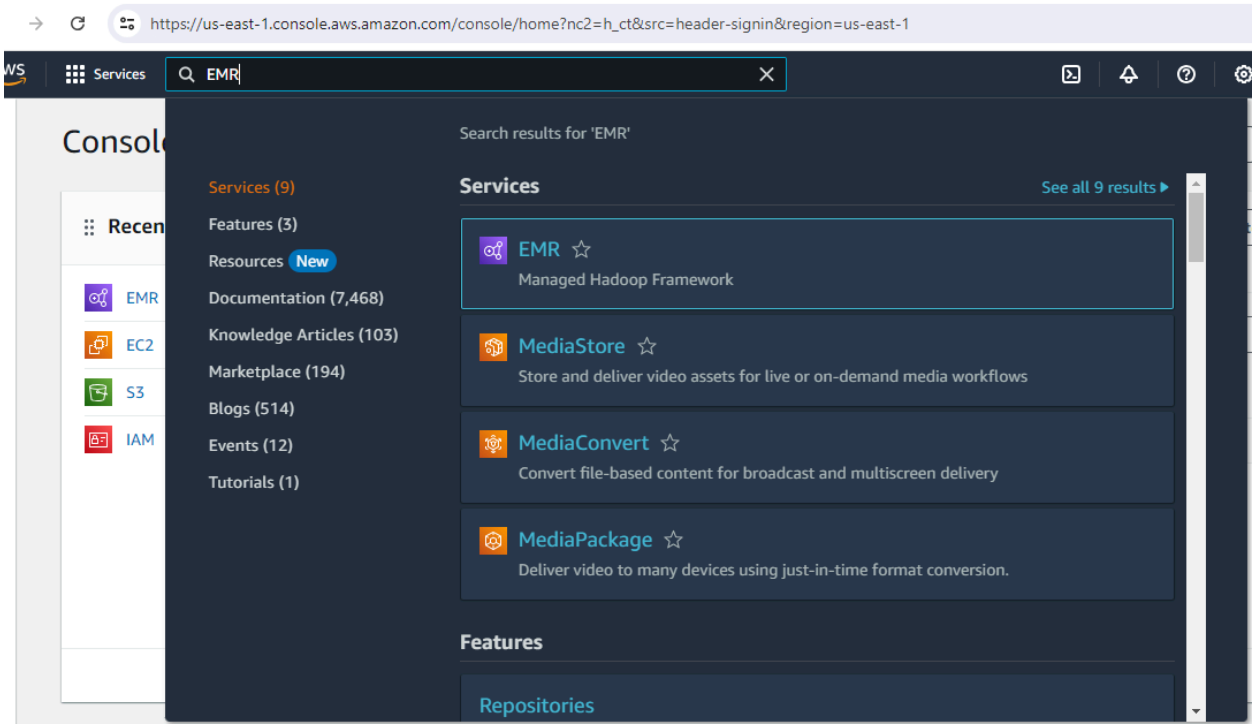
We will set up an EMR cluster, SSH into it, and copy the relevant file directly onto the HDFS.

We will also investigate the pros and cons of Hadoop vs Spark.

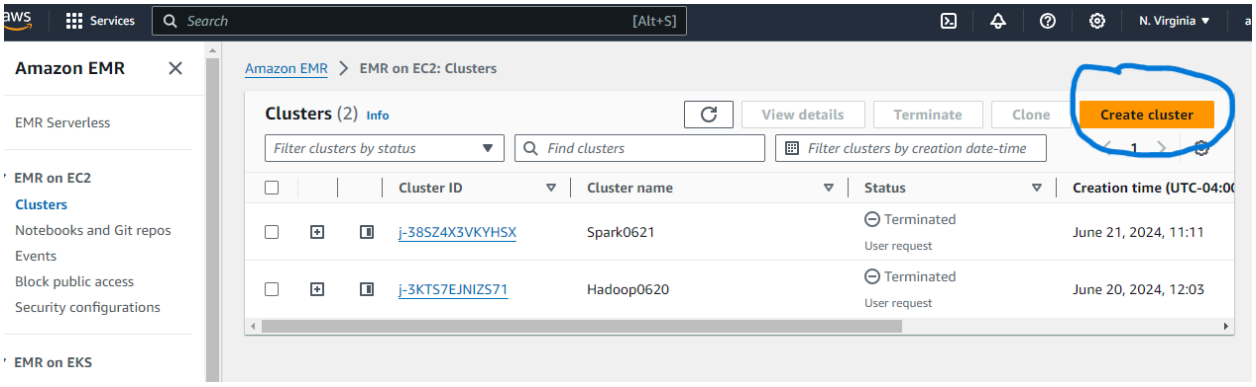
Create a Cluster

Create a new EMR cluster on AWS for using Spark and EMR notebooks.

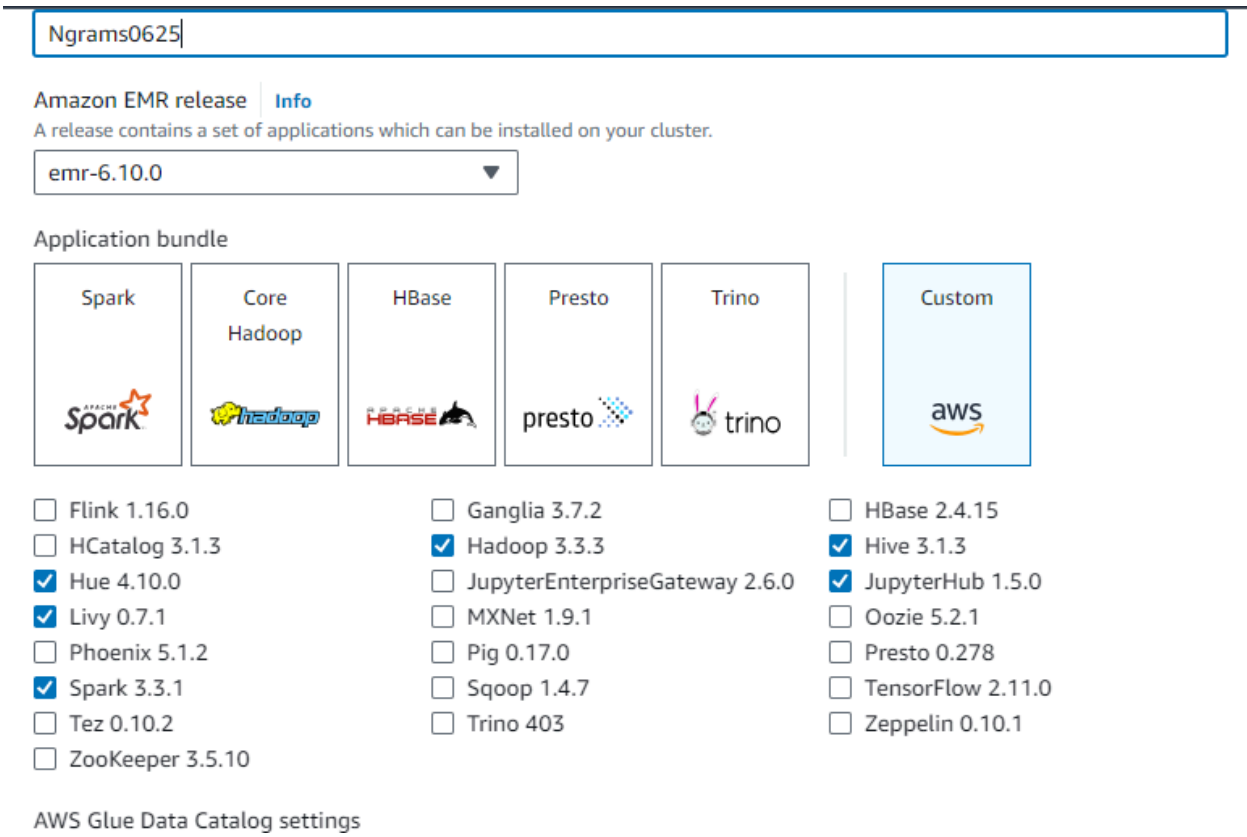
1. Create an AWS account if you don't already have one.
2. Sign in to the AWS console
3. On the Console Home, type 'EMR' in the Search bar and select EMR from the results



4. On the EMR home page, click on Create Cluster, or if you have an old cluster with the settings you want, you can select that cluster and click on Clone.

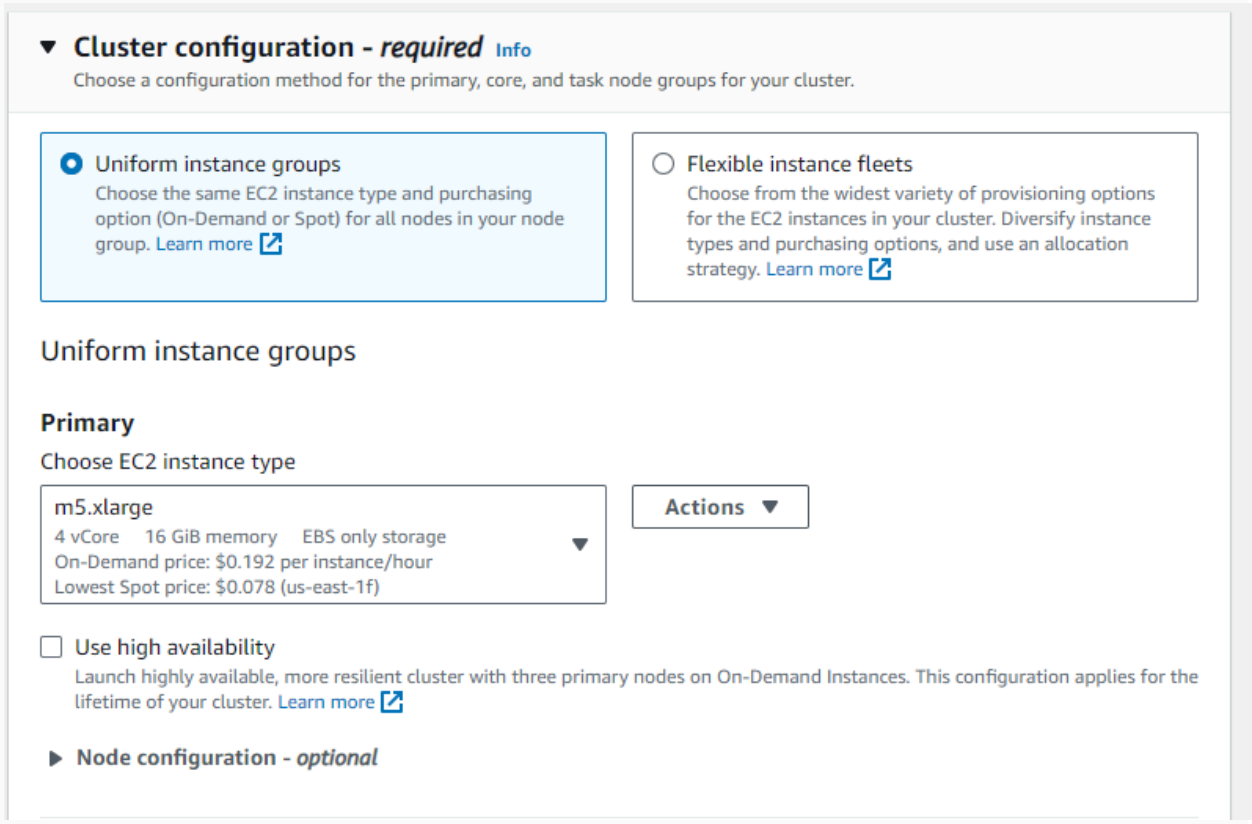


5. Cluster Settings
   - Give your cluster a name.
   - In the 'Release' dropdown, select emr-6.10.0.
   - Select the Custom application bundle, and tick the boxes for Hadoop, Hue, JupyterHub, Livy, Hive, and Spark.



Instance Groups

6. Remove the task instance group.
7. Allocate 2 nodes to the core instance group.



Cluster Termination

- Set cluster termination to 4h idle time.
- Turn termination protection off.

Cluster termination and node replacement Info

Choose termination settings and protect your cluster from accidental shutdown.

Termination option

- ○ Manually terminate cluster
- ○ Automatically terminate cluster after last step ends
- ◉ Automatically terminate cluster after idle time (Recommended)

Idle time

Enter the time until your cluster terminates.

0 days ▼ | 04:00:00

Choose a time that is greater than 1 minute (00:01:00) and less than 7 days. The time is in hh:mm:ss (24-hour) format.

☐ Use termination protection

Protects your cluster from accidental termination. If on, you must first turn off protection to terminate the cluster. We recommend turning on termination protection for your long running clusters.

Unhealthy node replacement - *new* Info

- ◉ Turn on

  Amazon EMR gracefully stops processes on unhealthy nodes to minimize data loss and job interruptions. It quickly replaces unhealthy nodes with new EC2 instances to keep your jobs running smoothly.
- ○ Turn off

  Amazon EMR adds unhealthy nodes to a denylist while keeping them in the cluster, allowing you continued access for troubleshooting.

Security and Access Management

- Select your key pair (keys are associated to geographies so if you switched recently, you might need to create a new key pair).
- In Identity and Access Management, choose the EMR_DefaultRole and EMR_EC2_DefaultRole.

Security configuration and EC2 key pair Info

Choose a security configuration or create a new one that you can reuse with other clusters.

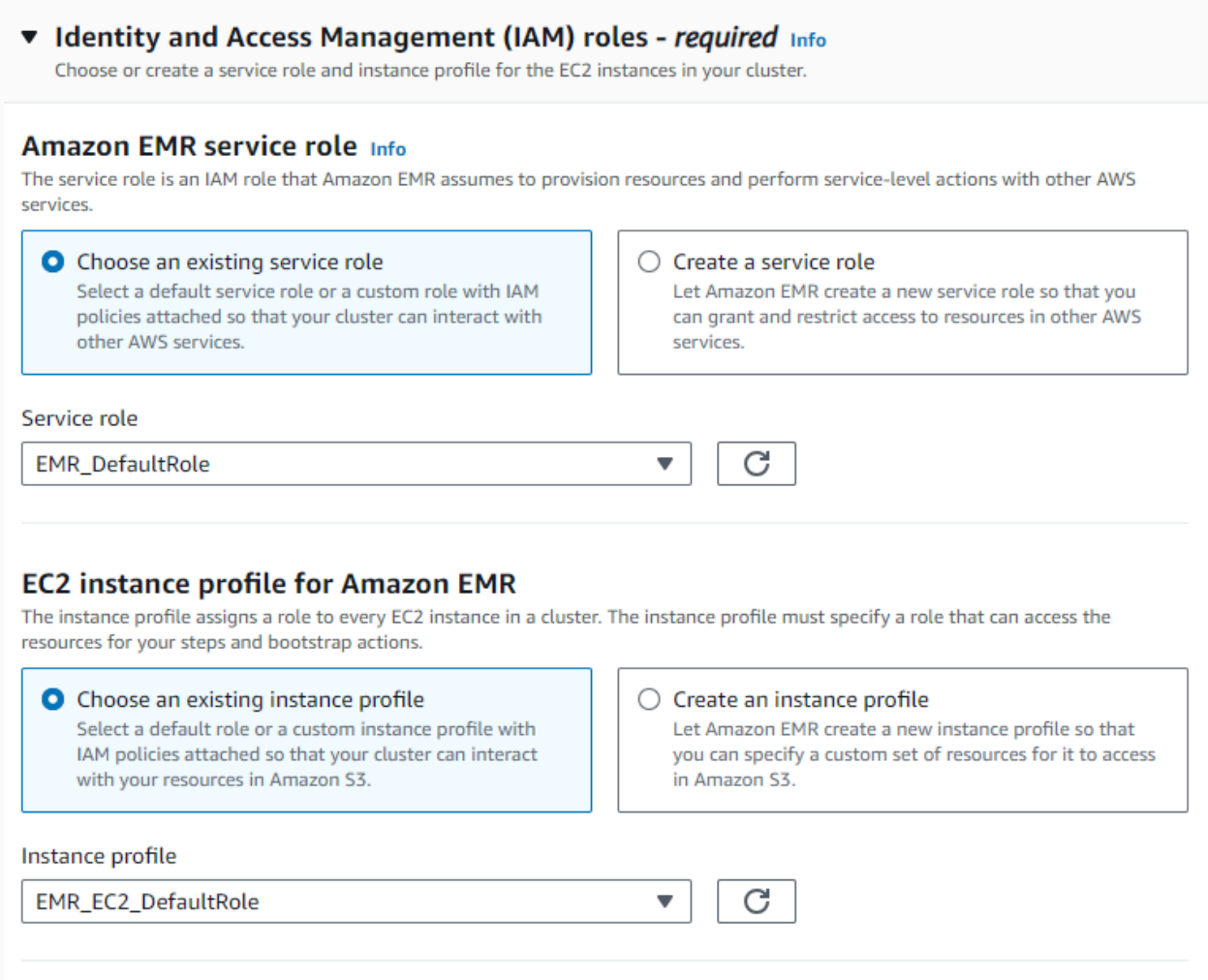
Security configuration

Select your cluster encryption, authentication, authorization, and instance metadata service settings.

Choose a security configuration | Browse | Create security configuration

Amazon EC2 key pair for SSH to the cluster Info

bstn_personal_aws | Browse | Create key pair



Create Cluster

- Click the 'Create Cluster' button.

The cluster creation process will start. It may take some time for the EC2 instances to spin up and all of the cluster software to be configured.

Once the cluster has been created it will enter a 'Waiting' state.

[Side Note] Create EC2 Roles

Creating the EC2 roles in case they are not found during the 'Identity and Access Management' block above.

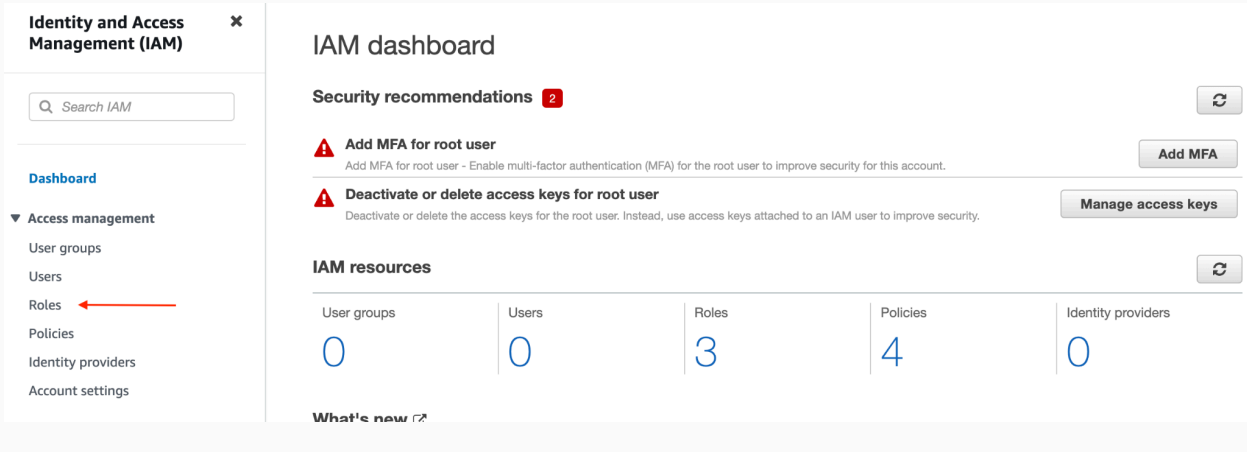
Identity and Access Management

These settings determine how our cluster is allowed to interact with other AWS services. The permissions come in pre-defined 'roles'. As this is our first time set-up, we will need double check (or create) the correct roles.

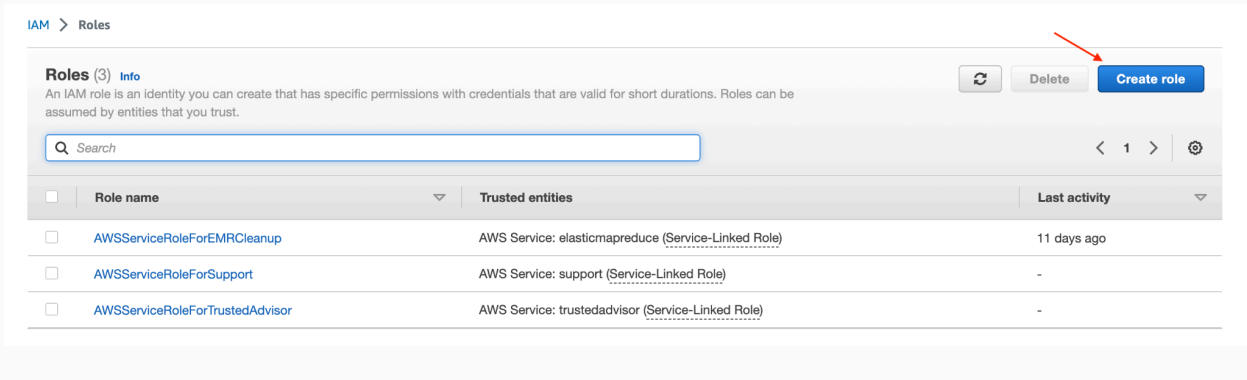
In the future, you can skip these steps and go straight to selecting a 'Service Role' and 'Instance Profile' from the drop-down menus.

---

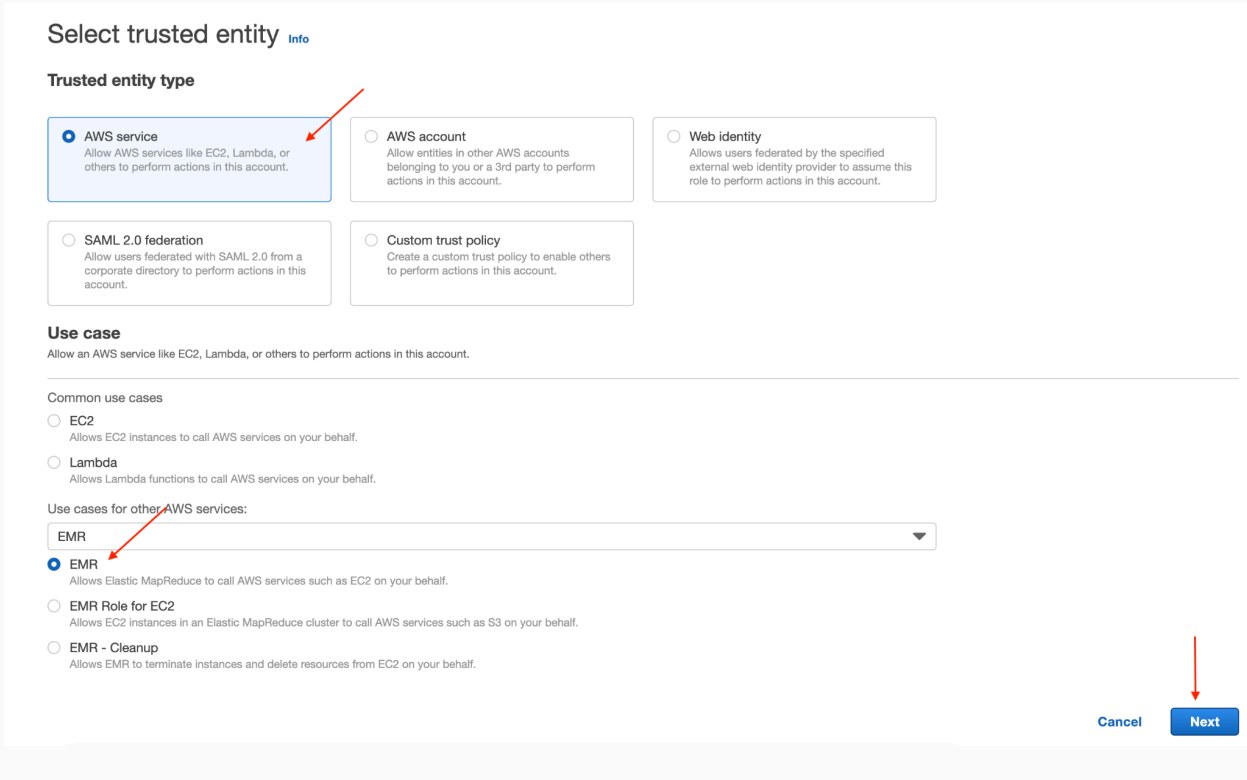
1. Leave the EMR cluster set-up window on hold for now.
2. In a new browser tab, open the AWS IAM page, by searching 'IAM' in the search bar.
3. Go to the 'Roles' tab through the left-side menu.



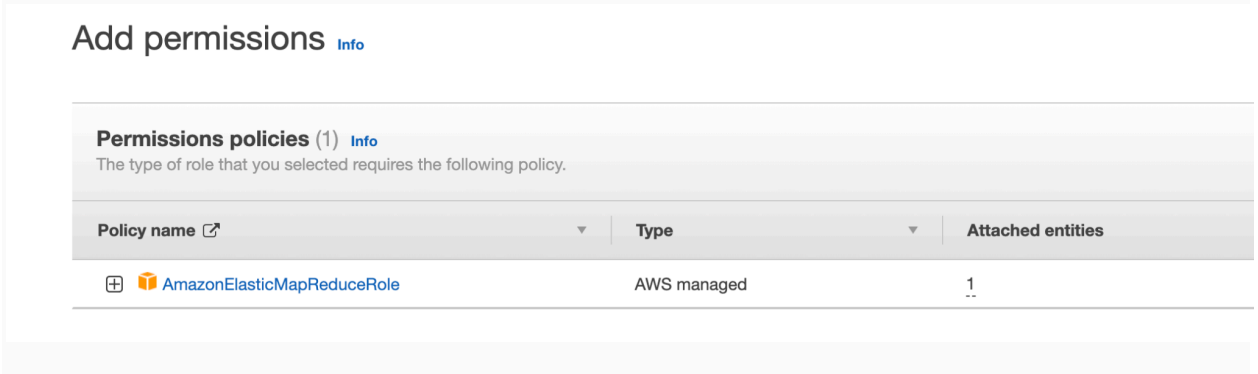
4. If 'EMR_DefaultRole' and 'EMR_EC2_DefaultRole' already show in the list of roles, you have the relevant roles. Otherwise, click 'Create Role'.



5. Select 'AWS service' as trusted entity type, and search and select 'EMR' as a use case, before clicking 'next'.



6. The 'Permissions Policies' should be pre-populated with "AmazonElasticMapReduceRole". Click 'Next'.

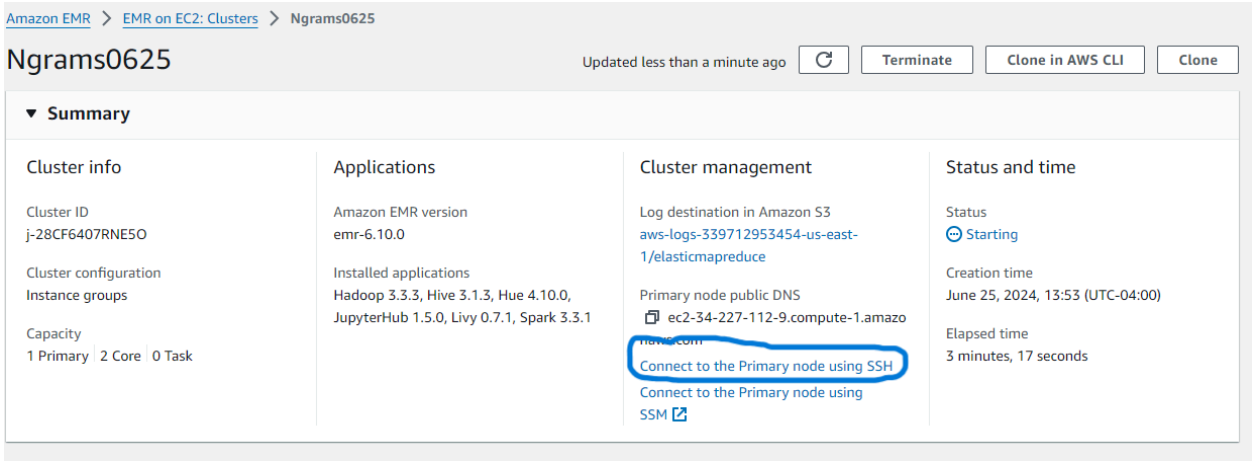


7. On the final page, fill in the 'Role Name' as 'EMR_DefaultRole', and at the bottom, click 'Create Role'. You should be brought back to the IAM page.
8. Repeat steps 4-7 to create another role, this time with 'EMR Role for EC2' as use case, which should pre-populate the "AmazonElasticMapReduceforEC2Role". Call it 'EMR_EC2_DefaultRole'.

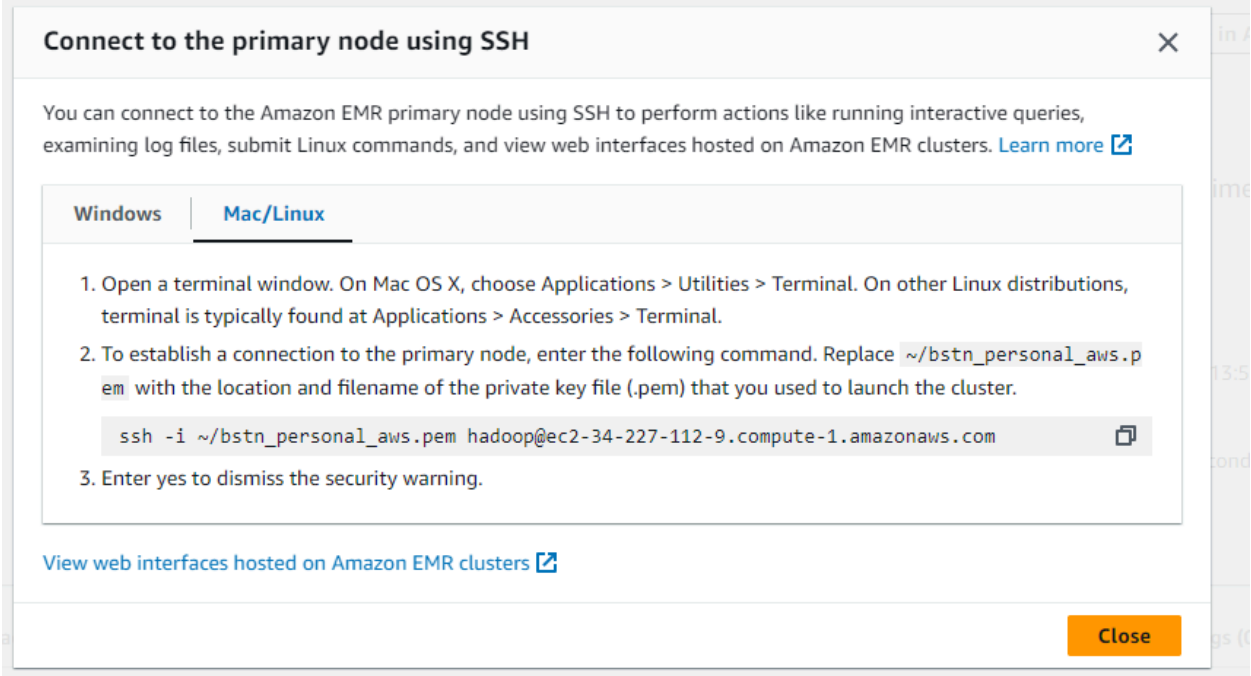
SSH into head node

Connect to the head node of the cluster using SSH.

1. You must know the location of the private key (.ppk or .pem) file on your local machine.
2. Once the cluster is created, give it a few minutes for the status to change from Starting to Waiting.
3. Click on Connect to Primary Node using SSH



4. This will give you the address of the machine to ssh into.



5. On Windows, open Putty

6. Under Hostname, put the name of the remote machine. e.g. - hadoop@ec2-34-227-112-9.compute-1.amazonaws.com

7. In Putty, in the Menu on the left side, go to Connection -> SSH -> Auth -> Credentials. Put the path to your private key file there.

8. Under Connection -> SSH-> Tunnels, do the following setting so that your jupyter on the remote machine can be seen on your localhost.
    a. Source Port - set to 9995
    b. Destination port - set to the remote address followed by 9443. E.g. ec2-34-227-112-9.compute-1.amazonaws.com:9443
    c. Click on 'Add'
    d. Make sure the two boxes "Local ports accept connections from other hosts" and "Remote ports do the same (SSH-2 only)" are checked.

9. Go to Session, give it a new name under Saved Sessions and hit Save.

10. Connect to the session. If you see a ssh session that says EMR, the connection is successful.

Copy data to HDFS

Copy the data folder from the S3 bucket *directly* into a directory on the Hadoop File System (HDFS)

Source path - s3://brainstation-dsft/eng_1M_1gram.csv
Destination folder- /user/hadoop/eng_1M_1gram
Command to run:
hadoop distcp s3://brainstation-dsft/eng_1M_1gram.csv /user/hadoop/eng_1M_1gram

If everything went right, you will see the file when you do hadoop fs -ls

Transfer file HDFS -> Laptop

Steps to get the resulting file from HDFS to your local machine.

1. Stored csv is in user/livy
2. hadoop fs -ls /user/livy
3. hadoop fs -getmerge HADOOP_FILE LOCAL_FILE` to get back to regular data

hadoop fs -getmerge /user/livy/ngram_data.csv ngramLocal.csv

4. sudo cp MY_FILE /mnt/var/lib/jupyter/home/jovyan/

sudo cp ngramLocal.csv /mnt/var/lib/jupyter/home/jovyan/

5. Download it onto your machine. Also make sure to Download the spark notebook before you terminate the cluster