

Big Data Wrangling With Google Books Ngrams

Analysis with Hadoop, Spark, and AWS S3

By

Anish Karnik

August 24, 2025

Q1) Spinning up a new EMR cluster on AWS for using Spark and EMR notebooks.
(Screenshots Q1)

1. Create cluster

First, we must create our EMR cluster. Open the [AWS console](#) and navigate to the EMR panel through the search bar or homepage and click 'Create Cluster':

2. Cluster name and version

Give your EMR cluster a name, in this case it is '**Big Data Assignment**'. From the drop-down menu, **select release emr-6.10.0**, and select the 'Core Hadoop' application bundle. This will ensure that all the software we need for today comes pre-installed on the cluster.

3. Cluster hardware configuration

Scroll down to the 'Cluster Configuration' block. Leave the default setting ('Instance groups') ticked, leave the 'Primary' and 'Core' groups to '**m5.xlarge**'.

Remove the 'Task' instance group. Next, we can select how many instances we want in our cluster. We will have only one 'Primary' node, but let's give ourselves two 'Core' nodes. **Set the 'Size' of the 'Core' instance group to 2.**

4. Cluster Termination Settings

Scroll down to 'Cluster termination'. These settings determine whether our cluster should shut itself down automatically if it has sat idle for a set amount of time. Make sure that 'Terminate cluster after idle time' is selected and set the idle time to 4h. Untick the 'Use termination protection' box.

5. Security, Identity and Access Management

Select the EC2 key pair that in my case "**aws_cloud**". For the 'Service role', select 'Choose an existing service role', and choose '**EMR_DefaultRole**' from the drop-down menu. For the 'Instance profile', select 'Choose an existing instance profile', and choose '**EMR_EC2_DefaultRole**' from the drop-down menu.

6. Create Cluster

Click '**Create Cluster**' on the right side of the screen. The cluster creation process will start. It may take some time for the EC2 instances to spin up and all the cluster software to be installed and configured. Once the cluster has been created it will enter a 'Waiting' state.

Q2) Connect to the head node of the cluster using SSH.(Screenshots Q2)

Now we can connect to the head node via SSH. Return to the cluster's overview screen and click the 'Connect to the Primary Node using SSH' hyperlink.

PuTTY Connection for Windows-EMR

- Spin up a Putty instance and navigate to Session, under Host Name enter this address:**hadoop@ec2-3-144-144-99.us-east-2.compute.amazonaws.com**
- Then Open up **Authenticate>Credentials** and under Private key file for authentication, select the appropriate **AWS access key**.
- Select Open then, If the connection is successful, we should be greeted by the **EMR ASCII** banner.

Q3) Copy the data folder from the S3 bucket directly into a directory on the Hadoop File System (HDFS) named /user/hadoop/eng_1M_1gram([Screenshots-Q3](#))

Under the EMR Cluster banner, run this command:

```
hadoop distcp s3://brainstation-dsft/eng_1M_1gram.csv /user/Hadoop/eng_1M_1gram/
```

After the folder is copied, check if it is in the HDFS, by running the following command:

```
hadoop fs -ls /user/Hadoop/eng_1M_1gram/
```

Q4) Using pyspark, read the data you copied into HDFS in Step 3. You may either use Jupyterhub on EMR (the default user and password are jovyan and jupyter) or work from pyspark in the terminal if you prefer. Once you have created a pyspark DataFrame, complete the following steps below: ([Screenshots-Q4](#))

PuTTY Connection for Windows-Spark Notebook

- Spin up a new Putty instance and navigate to Session, under Host Name enter the same primary node SSH address:**hadoop@ec2-3-144-144-99.us-east-2.compute.amazonaws.com**

- Then Open up **Authenticate>Credentials** and under **Private key** file for authentication, select the appropriate AWS access key.
- Navigate to Tunnels, in the Source port enter **9995**, under destination port enter **localhost:9443**. Click on Add and hit Open, If the connection is successful, we should be greeted by the EMR ASCII banner. Now we have created a tunnel to connect to Spark notebooks
- We can now access JupyterHub in our browser at **https://localhost:9995**. If successfully connected, we will be dropped into the JupyterHub login screen as below. The login defaults are username of **jovyan** with password **jupyter**
- Check " **Big Data Wrangling With Google Books Ngrams_Q4.ipynb**" for Q4 analysis
- To check contents in Hadoop after Q4 part c, run the following command:

```
hadoop fs -ls /user/hadoop/eng_1M_1gram/
```

Q5) Collect the contents of the directory into a single file on the local drive of the head node using getmerge and move this file into a S3 bucket in your account.(Screenshot-Q5)

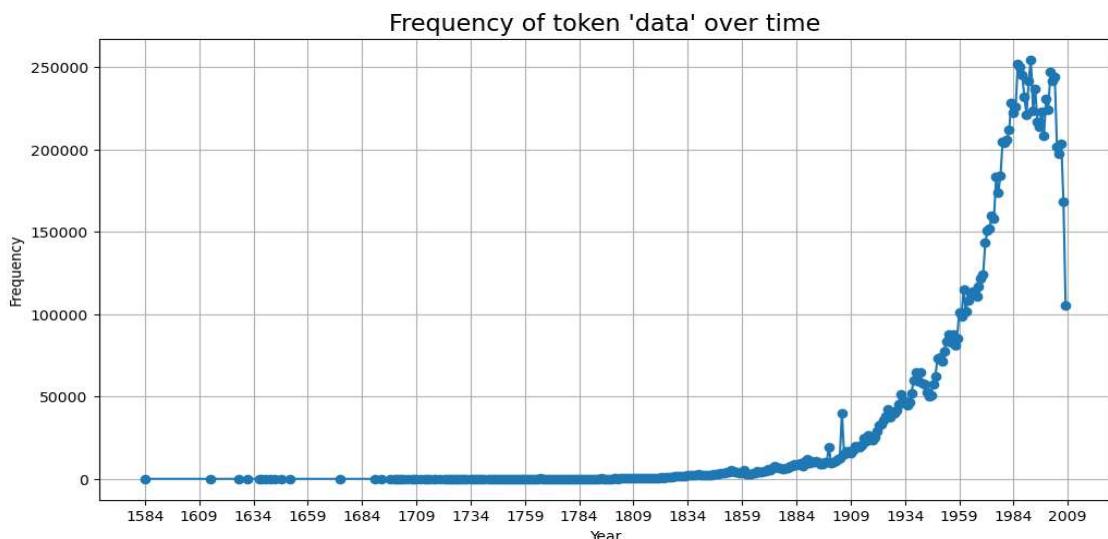
- To merge the contents into 1 file run this command:
`hdfs dfs -getmerge -nl /user/Hadoop/Bigdata_token ~/Bigdata_token.csv`
- To move this file to S3 bucket on AWS, run the following command:
`aws s3 cp ~/Bigdata_token.csv s3 //akarnik-bstn-bucket/Bigdata_token.csv`
- We can verify the location of the file moved by logging into our AWS console and searching on the task bar for S3, under buckets, we have **akarnik-bstn-bucket**, which should contain the file **Bigdata_token.csv**

Q6) On your local machine (or on AWS outside of Spark) in python, read the CSV data from the S3 folder into a pandas DataFrame (You will have to research how to read data into pandas from S3 buckets). Note You must have first authenticated on your machine using aws configure on the command line to complete this step)([Screenshot-Q6](#))

- To authenticate local for AWS, we first need to have AWS command Line interface. We can download and follow the instruction for default instructions for setup on Windows here: <https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html>
- We open with git bash and run the following command: **aws configure**. We are asked to later provide our AWS Access Key, along with the Secret Access Key and select the default region which is “**us-east-2**” along with default output format “**json**”. We can then run this command to see all our s3 buckets on aws: **aws s3 ls**

Q7.) Plot the number of occurrences of the token (the *frequency* column) of data over the years using matplotlib.

- Check Notebook # 6 for the code for this plot below:



Historical Usage of the Token 'Data'

The following analysis summarizes the frequency of the token 'data' in published works over time, based on the provided plot. The trends highlight the evolution of the term from medieval times to central importance in the digital era.

Key Observations

1. Early Usage (Before 1800s):

- The word 'data' appeared very rarely before the 19th century.
- This suggests it was not a common term in published texts.

2. Steady Growth (1800–1900):

- Usage increased slowly during the 19th century.
- This aligns with the growth of science, statistics, and industrialization.

3. Rapid Acceleration (1900–1950):

- Sharp increase in the early 20th century.
- Likely linked to developments in mathematics, physics, and early computing.

4. Exponential Rise (1950–2000):

- A huge surge post–World War II.
- Connected to the information age: digital computers, databases, and telecommunications.

5. Peak and Fluctuations (After 1980s):

- Usage peaked between the 1980s and early 2000s.
- Reflects the rise of computing, the internet, and data-driven research.
- Later fluctuations may reflect digitization biases or alternative terminology like 'big data' or 'information'.

Key Takeaway

The token 'data' evolved from a little-used academic term to one of the most central concepts in modern science and technology, with explosive growth during the digital revolution.

Q8) Compare Hadoop and Spark as distributed file systems.

- a. What are the advantages/ differences between Hadoop and Spark?
List two advantages for each.

- b. Explain how the HDFS stores the data.

A. Hadoop vs Spark

It is important to note that Hadoop is not just a file system, but an ecosystem with two key components: the Hadoop Distributed File System (HDFS) for storage, and MapReduce for processing. Spark, on the

other hand, is not a file system at all, but a distributed data processing engine that can run on top of HDFS, Amazon S3, or other storage systems.

Differences Between Hadoop (MapReduce) and Spark

Feature	Hadoop (MapReduce)	Spark
Processing	Disk-based (reads/writes to disk between operations)	In-memory (keeps data in RAM, much faster)
Speed	Slower (I/O heavy)	Up to 100x faster for iterative tasks
Ease of Use	Low-level Java APIs (harder to code)	High-level APIs in Python, Scala, SQL
Use Cases	Batch processing (ETL, log analysis)	Batch + Streaming + ML + Graph processing

Advantages of Hadoop

- Scalability & Storage: Handles petabytes of data reliably using HDFS, with built-in fault tolerance.
- Cost-effective: Runs on commodity hardware without requiring expensive servers.

Advantages of Spark

- Speed: In-memory computation makes Spark much faster than Hadoop MapReduce, especially for iterative tasks like machine learning.
- Flexibility: Provides rich APIs and libraries such as Spark SQL, MLLib for machine learning, GraphX for graph processing, and Structured Streaming for real-time data.

B. How HDFS Stores Data

The Hadoop Distributed File System (HDFS) is the storage layer of the Hadoop ecosystem. It is designed for high-throughput access to large datasets and is optimized for streaming reads and writes.

- Data Block Storage: Files are split into large fixed-size blocks (default size = 128 MB or 256 MB).
- Replication: Each block is replicated across multiple DataNodes (default replication factor = 3) to ensure fault tolerance.
- NameNode: The master node that stores metadata such as file names, block locations, and directory structure.
- DataNodes: Worker nodes that actually store the file blocks.
- High Throughput: Optimized for handling very large files rather than low-latency access.
- Fault Tolerance: If a DataNode fails, HDFS automatically uses replicas from other nodes.

Example: If a 512 MB file is stored in HDFS with a block size of 128 MB, it will be split into 4 blocks. Each block will be replicated 3 times across different DataNodes, while the NameNode maintains metadata about block locations.

Appendix

Q1-Screenshots

The screenshot shows the AWS EMR 'Create cluster' interface. At the top, there's a search bar and navigation links for 'Amazon EMR > EMR on EC2: Clusters > Create cluster'. Below this, the title 'Clone "Big Data Assignment"' is displayed with a 'Info' link. A dropdown menu for 'Name and applications - required' is open, showing the current configuration: 'Name' set to 'Big Data Assignment', and 'Amazon EMR release' set to 'emr-6.10.0'. To the right, a 'Summary' section is visible, showing the same information. A vertical scroll bar is present on the right side of the main content area.

Application bundle



- | | | |
|---|---|--|
| <input type="checkbox"/> Flink 1.16.0 | <input type="checkbox"/> Ganglia 3.7.2 | <input type="checkbox"/> HBase 2.4.15 |
| <input type="checkbox"/> HCatalog 3.1.3 | <input checked="" type="checkbox"/> Hadoop 3.3.3 | <input checked="" type="checkbox"/> Hive 3.1.3 |
| <input checked="" type="checkbox"/> Hue 4.10.0 | <input type="checkbox"/> JupyterEnterpriseGateway 2.6.0 | <input checked="" type="checkbox"/> JupyterHub 1.5.0 |
| <input checked="" type="checkbox"/> Livy 0.7.1 | <input type="checkbox"/> MXNet 1.9.1 | <input type="checkbox"/> Oozie 5.2.1 |
| <input type="checkbox"/> Phoenix 5.1.2 | <input type="checkbox"/> Pig 0.17.0 | <input type="checkbox"/> Presto 0.278 |
| <input checked="" type="checkbox"/> Spark 3.3.1 | <input type="checkbox"/> Sqoop 1.4.7 | <input type="checkbox"/> TensorFlow 2.11.0 |
| <input type="checkbox"/> Tez 0.10.2 | <input type="checkbox"/> Trino 403 | <input type="checkbox"/> Zeppelin 0.10.1 |
| <input type="checkbox"/> ZooKeeper 3.5.10 | | |

EBS root volume

EBS root volume applies to the operating systems and applications that you install on the cluster.

Size (GiB)

15

15 - 100 GiB per volume General Purpose SSD (gp2)

▼ 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.

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 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.



Amazon EC2 key pair for SSH to the cluster [Info](#)



▼ Identity and Access Management (IAM) roles - required Info

Choose or create a service role and instance profile for the EC2 instances in your cluster.

Amazon EMR service role Info

The service role is an IAM role that Amazon EMR assumes to provision resources and perform service-level actions with other AWS services.

Choose an existing service role

Select a default service role or a custom role with IAM policies attached so that your cluster can interact with other AWS services.

Create a service role

Let Amazon EMR create a new service role so that you can grant and restrict access to resources in other AWS services.

Service role

EMR_DefaultRole



EC2 instance profile for Amazon EMR

The instance profile assigns a role to every EC2 instance in a cluster. The instance profile must specify a role that can access the resources for your steps and bootstrap actions.

Choose an existing instance profile

Select a default role or a custom instance profile with IAM policies attached so that your cluster can interact with your resources in Amazon S3.

Create an instance profile

Let Amazon EMR create a new instance profile so that you can specify a custom set of resources for it to access in Amazon S3.

Instance profile

EMR_EC2_DefaultRole



Q2-Screenshots

Amazon EMR > EMR on EC2: Clusters > Big Data Assignment

Your cluster "Big Data Assignment" has been successfully created.

Big Data Assignment

Updated less than a minute ago

Summary	Applications	Cluster management	Status and time
Cluster info Cluster ID: j-3M3YVVPNF72BN Cluster ARN: arn:aws:elasticmapreduce:us-east-2:314807448672:cluster-j-3M3YVVPNF72BN Cluster configuration: Instance groups Capacity: 1 Primary 0 Core 0 Task	Amazon EMR version: emr-6.10.0 Installed applications: Hadoop 3.3.3, Hive 5.1.3, Hue 4.10.0, JupyterHub 1.5.0, Livy 0.7.1, Spark 3.3.1	Log destination in Amazon S3: aws-logs-314807448672-us-east-2/elasticmapreduce Persistent application UIs: <ul style="list-style-type: none">Spark History ServerYARN Timeline ServerTez UI Primary node public DNS: ec2-3-153-83-149.us-east-2.compute.amazonaws.com Connect to the Primary node using SSH: Connect to the Primary node using SSM	Status: Waiting Creation time: August 23, 2025, 21:09 (UTC-07:00) Elapsed time: 10 minutes, 32 seconds

Connect to the primary node using SSH

X

You can connect to the Amazon EMR primary node using SSH to perform actions like running interactive queries, examining log files, submit Linux commands, and view web interfaces hosted on Amazon EMR clusters. [Learn more](#)

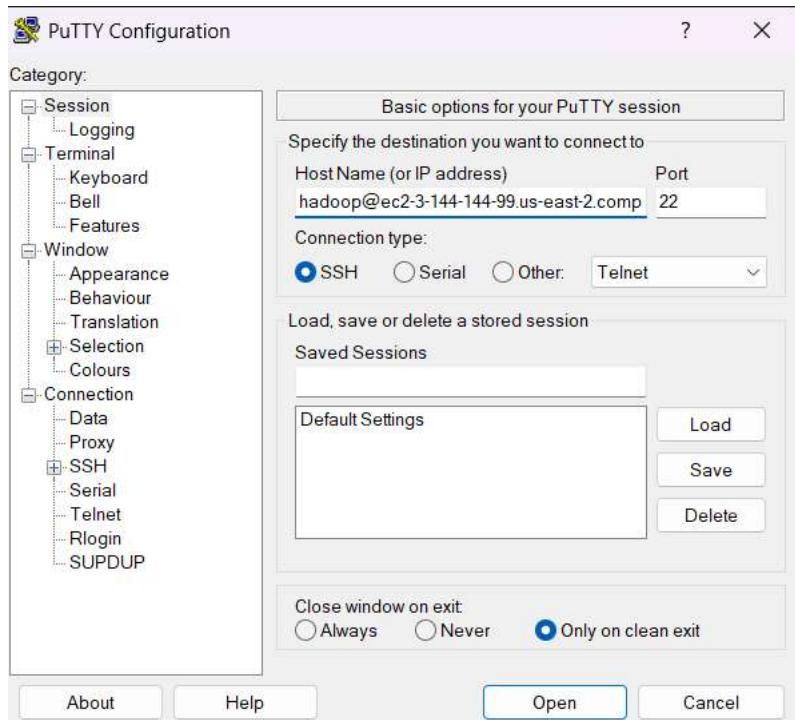
Windows

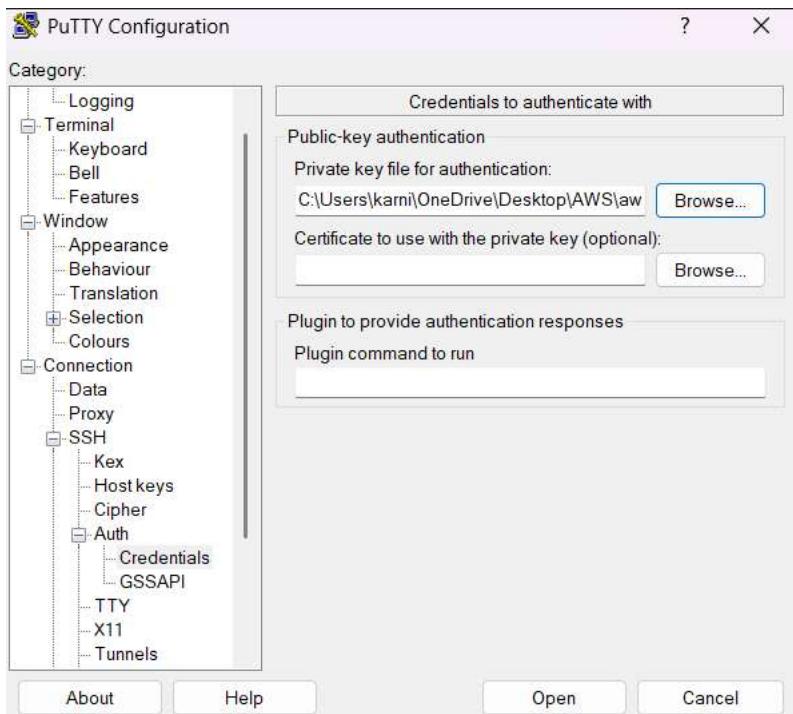
Mac/Linux

1. Download PuTTY.exe to your computer from:
<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>
2. Start PuTTY.
3. In the Category list, choose Session.
4. In the Host Name field, enter `hadoop@ec2-3-144-144-99.us-east-2.compute.amazonaws.com`
5. In the Category list, expand Connection > SSH, and then choose Auth.
6. For Private key file for authentication, choose Browse and select the private key file (`aws_cloud.ppk`) that you used to launch the cluster.
7. Select Open.
8. Select Yes to dismiss the security alert.

[View web interfaces hosted on Amazon EMR clusters](#)

Close





```
hadoop@ip-172-31-33-199:~$ Using username "hadoop".
hadoop@ip-172-31-33-199:~$ Authenticating with public key "imported-openssh-key"
Last login: Sun Aug 24 04:20:11 2025
[Amazon Linux 2
AL2 End of Life is 2026-06-30.
A newer version of Amazon Linux is available!
Amazon Linux 2023, GA and supported until 2028-03-15.
https://aws.amazon.com/linux/amazon-linux-2023/]

hadoop@ip-172-31-33-199:~$
```

Amazon Linux 2
AL2 End of Life is 2026-06-30.
A newer version of Amazon Linux is available!
Amazon Linux 2023, GA and supported until 2028-03-15.
<https://aws.amazon.com/linux/amazon-linux-2023/>

```
hadoop@ip-172-31-33-199:~$
```

Q3-Screenshots

```

hadoop@ip-172-31-33-199:~$ hadoop distcp s3://brainstation-dsft/eng_1M.lgram.csv /user/hadoop/eng_1M.lgram/
2025-08-24 04:26:47.063 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFolder=false, deleteMissing=false, ignoreFailures=false, overwrite=false, append=false, useIf=false, useSdIf=false, fromSnapshot=null, toSnapshot=null, skipRC=false, blocking=true, numListStreams=0, maxMaps=20, mapBandwidth=0.0, copyStrategy='uniformsize', preserveStatus=true, atomicWorkPath=null, logPath=null, sourceFileListing=null, sourcePaths=[s3://brainstation-dsft/eng_1M.lgram.csv], targetPath=/user/hadoop/eng_1M.lgram, filtersFile=null, blocksPerChu=0], atomicCopyBufferSize=8192, verboseLog=false, directWrite=false, userIterator=false, sourcePaths=[s3://brainstation-dsft/eng_1M.lgram.csv], targetPathExcludes=false, preserveRawXAttrs=false
2025-08-24 04:26:47.474 INFO client.DefaultNoharmFollowerProxyProvider: Connecting to ResourceManager at ip-172-31-33-199.us-east-2.compute.internal/172.31.33.199:8032
2025-08-24 04:26:47.504 INFO client.AHSProxy: Connecting to Application History server at ip-172-31-33-199.us-east-2.compute.internal/172.31.33.199:10200
2025-08-24 04:26:47.678 INFO tools.SimpleCopyListing: Starting: Building listing using multi threaded approach for s3://brainstation-dsft/eng_1M.lgram.csv
2025-08-24 04:26:50.680 INFO tools.SimpleCopyListing: Building listing using multi threaded approach for s3://brainstation-dsft/eng_1M.lgram.csv: duration 0.00.002s
2025-08-24 04:26:50.783 INFO tools.SimpleCopyListing: Paths (files+dirs) cnt = 1; dirCnt = 0
2025-08-24 04:26:50.785 INFO tools.SimpleCopyListing: Build file listing completed.
2025-08-24 04:26:50.787 INFO configuration.Configuration: Configuration deprecation: io.sort.mb is deprecated. Instead, use mapreduce.task.io.sort.mb
2025-08-24 04:26:50.787 INFO configuration.Configuration: Configuration deprecation: io.sort.factor is deprecated. Instead, use mapreduce.task.io.sort.factor
2025-08-24 04:26:50.800 INFO tools.DistCp: Number of paths in the copy list: 1
2025-08-24 04:26:50.948 INFO tools.DistCp: Number of paths in the copy list: 1
2025-08-24 04:26:50.970 INFO client.DefaultNoharmFollowerProxyProvider: Connecting to ResourceManager at ip-172-31-33-199.us-east-2.compute.internal/172.31.33.199:8032
2025-08-24 04:26:50.970 INFO client.AHSProxy: Connecting to Application History server at ip-172-31-33-199.us-east-2.compute.internal/172.31.33.199:10200
2025-08-24 04:26:51.075 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hadoop/.staging/job_1756009055415_0001
2025-08-24 04:26:51.198 INFO mapreduce.JobSubmitter: number of splits:1
2025-08-24 04:26:51.349 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1756009055415_0001
2025-08-24 04:26:51.349 INFO mapreduce.JobSubmitter: Executing with tokens: []
2025-08-24 04:26:51.545 INFO conf.Configuration: resource-types.xml not found
2025-08-24 04:26:51.545 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2025-08-24 04:26:52.158 INFO impl.YarnClientImpl: Submitted application application_1756009055415_0001
2025-08-24 04:26:52.216 INFO tools.DistCp: Browsing job: job_1756009055415_0001
2025-08-24 04:26:52.216 INFO tools.DistCp: Running job: job_1756009055415_0001
2025-08-24 04:27:00.319 INFO mapreduce.Job: Job job_1756009055415_0001 running in uber mode : false
2025-08-24 04:27:00.321 INFO mapreduce.Job: map 0% reduce 0%
2025-08-24 04:27:16.440 INFO mapreduce.Job: map 100% reduce 0%
2025-08-24 04:28:09.712 INFO mapreduce.Job: Job job_1756009055415_0001 completed successfully
2025-08-24 04:28:09.795 INFO mapreduce.Job: Counters: 42
File System Counters
FILE: Number of bytes read=0
FILE: Number of bytes written=294784
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=374
HDFS: Number of bytes written=5292105197
HDFS: Number of read operations=12
HDFS: Number of large read operations=0
HDFS: Number of write operations=5
HDFS: Number of bytes read erasure-coded=0
S3: Number of bytes read=5292105197
S3: Number of bytes written=0
S3: Number of read operations=0
S3: Number of large read operations=0

```

```

Job Counters
  Launched map tasks=1
  Other finished map tasks=1
  Total time spent by all maps in occupied slots (ms)=202681344
  Total time spent by all reduces in occupied slots (ms)=0
  Total time spent by all map tasks (ms)=65977
  Total wcore-milliseconds taken by all map tasks=65977
  Total megabyte-milliseconds taken by all map tasks=202681344

Map-Reduce Framework
  Map input records=1
  Map output records=0
  Input split bytes=135
  Spilled Records=0
  Failed Shuffles=0
  Merged Map outputs=0
  GC time elapsed (ms)=393
  CPU time spent (ms)=6170
  Physical memory (bytes) snapshot=971833344
  Virtual memory (bytes) snapshot=4428616384
  Total committed heap usage (bytes)=927465472
  Peak Map Physical memory (bytes)=971833344
  Peak Map Virtual memory (bytes)=4445511680

File Input Format Counters
  Bytes Read=239
  File Output Format Counters
    Bytes Written=0

DistCp Counters
  Bandwidth in Bytes=84001669
  Bytes Copied=5292105197
  Bytes Expected=5292105197
  Files copied=1

[hadoop@ip-172-31-33-199 ~]$ 
[hadoop@ip-172-31-33-199 ~]$ hadoop fs -ls /user/hadoop/eng_1M_1gram/
-rw-r--r-- 1 hadoop hdfsadmin group 5292105197 2025-08-24 04:28 /user/hadoop/eng_1M_1gram
[hadoop@ip-172-31-33-199 ~]$ 

```

Q4-Screenshots

Connect to the primary node using SSH



You can connect to the Amazon EMR primary node using SSH to perform actions like running interactive queries, examining log files, submit Linux commands, and view web interfaces hosted on Amazon EMR clusters. [Learn more](#)

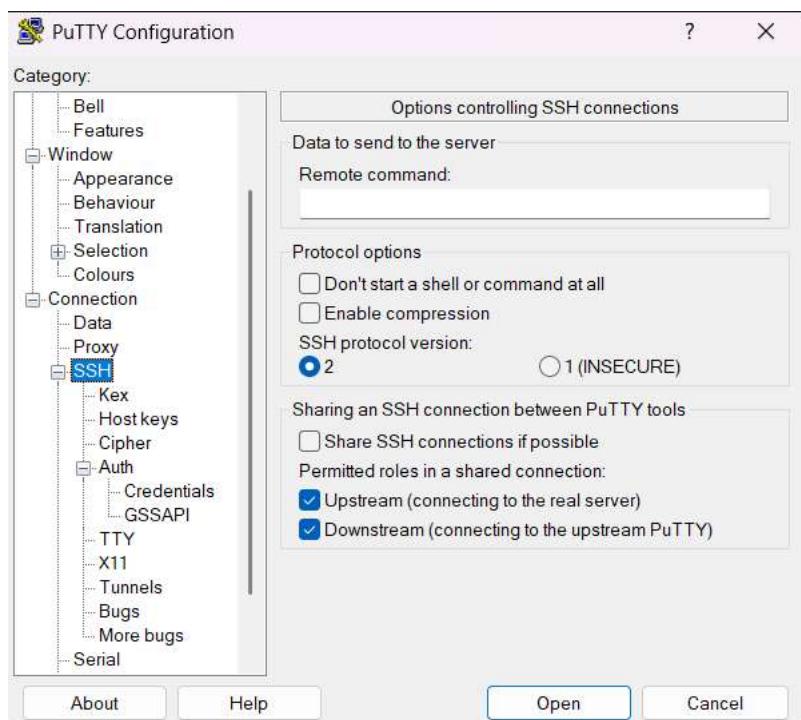
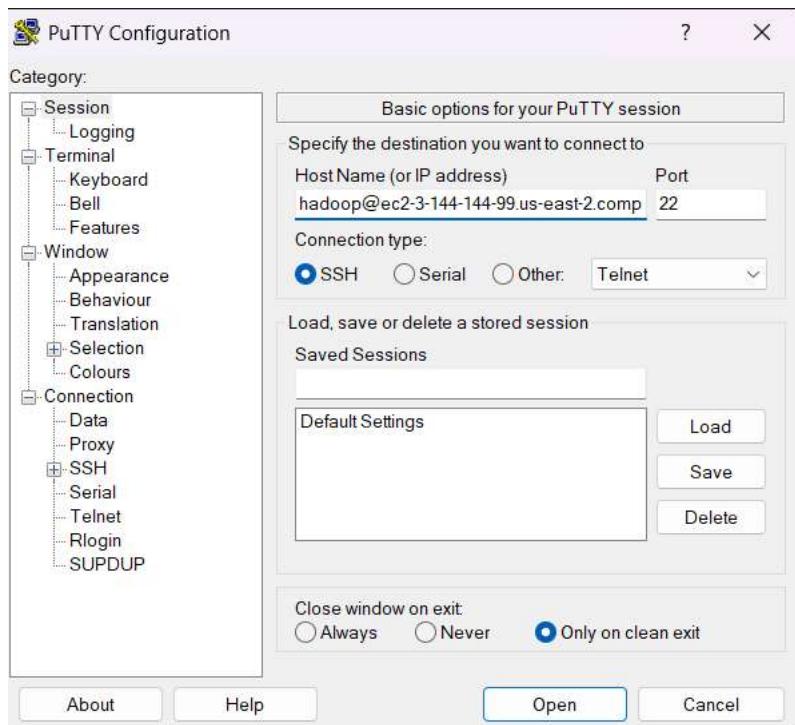
Windows

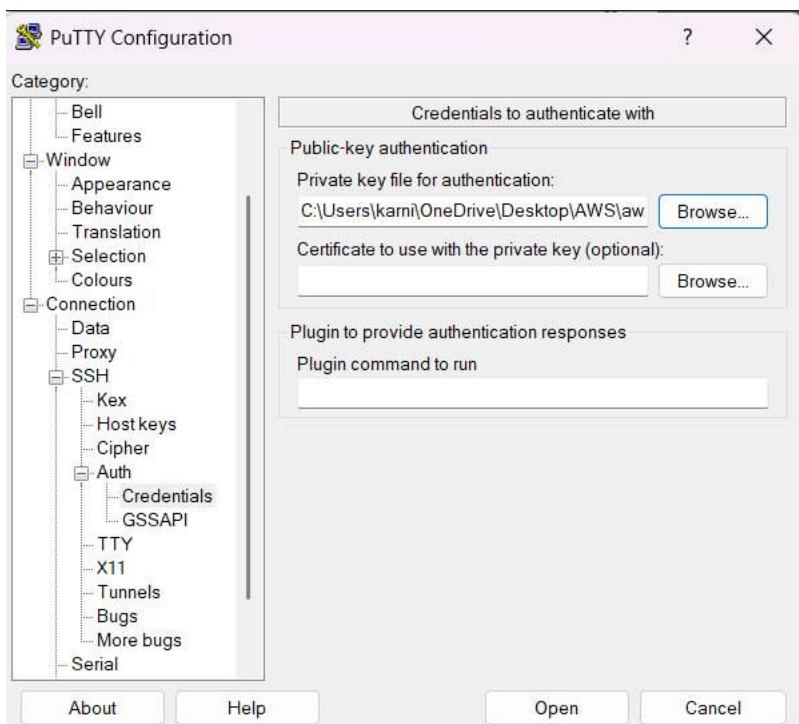
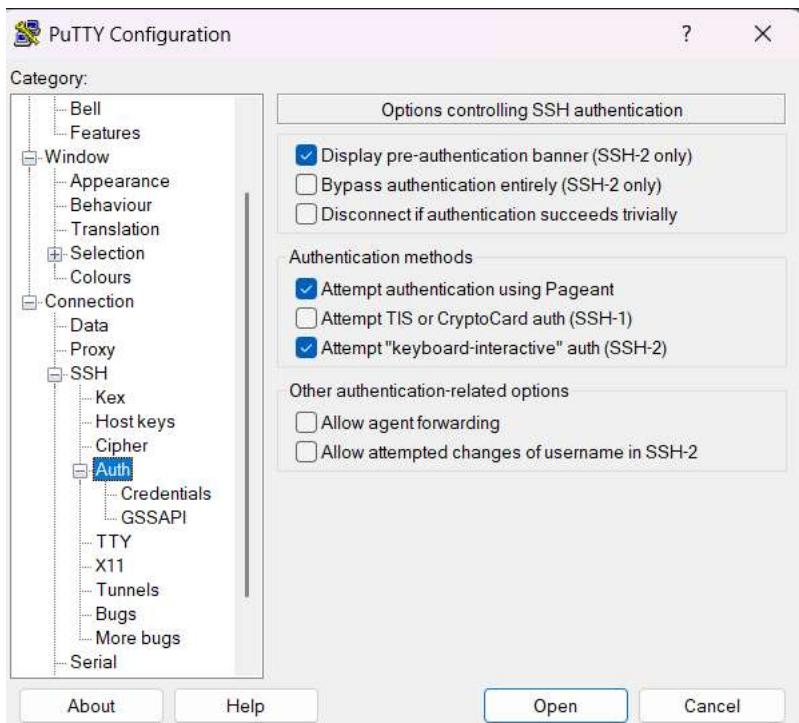
Mac/Linux

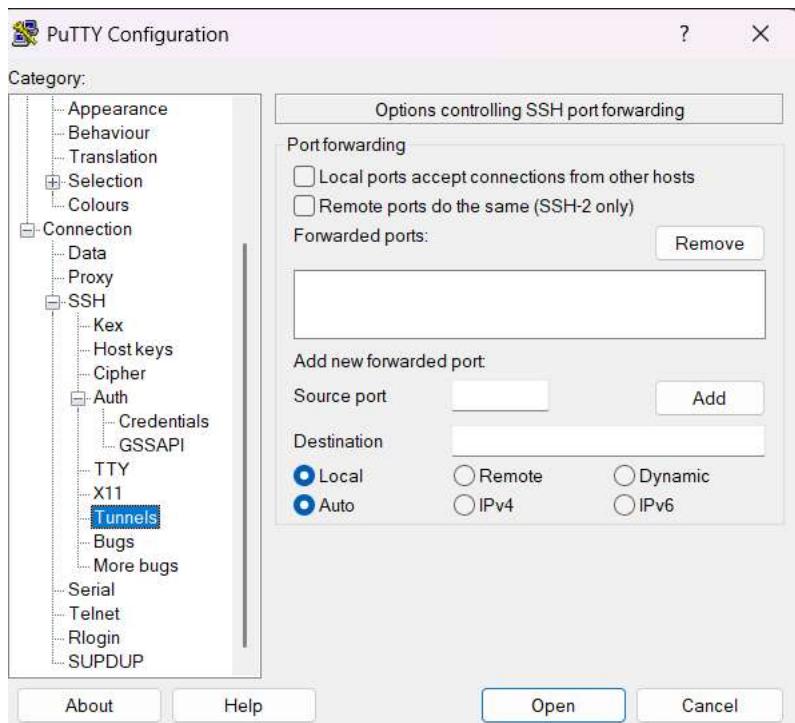
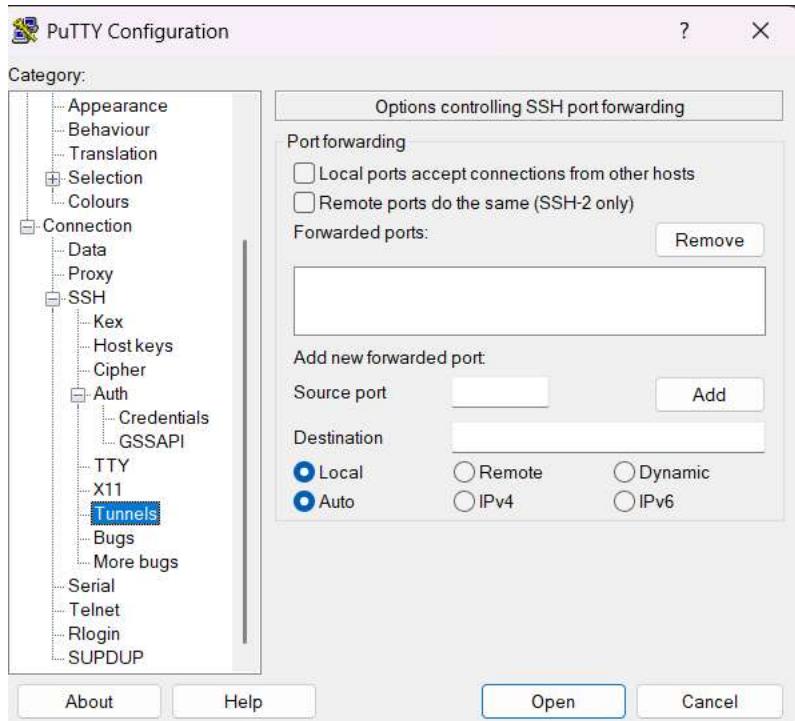
1. Download PuTTY.exe to your computer from:
<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>
2. Start PuTTY.
3. In the Category list, choose Session.
4. In the Host Name field, enter `hadoop@ec2-3-133-83-149.us-east-2.compute.amazonaws.com`
5. In the Category list, expand Connection > SSH, and then choose Auth.
6. For Private key file for authentication, choose Browse and select the private key file (`aws_cloud.ppk`) that you used to launch the cluster.
7. Select Open.
8. Select Yes to dismiss the security alert.

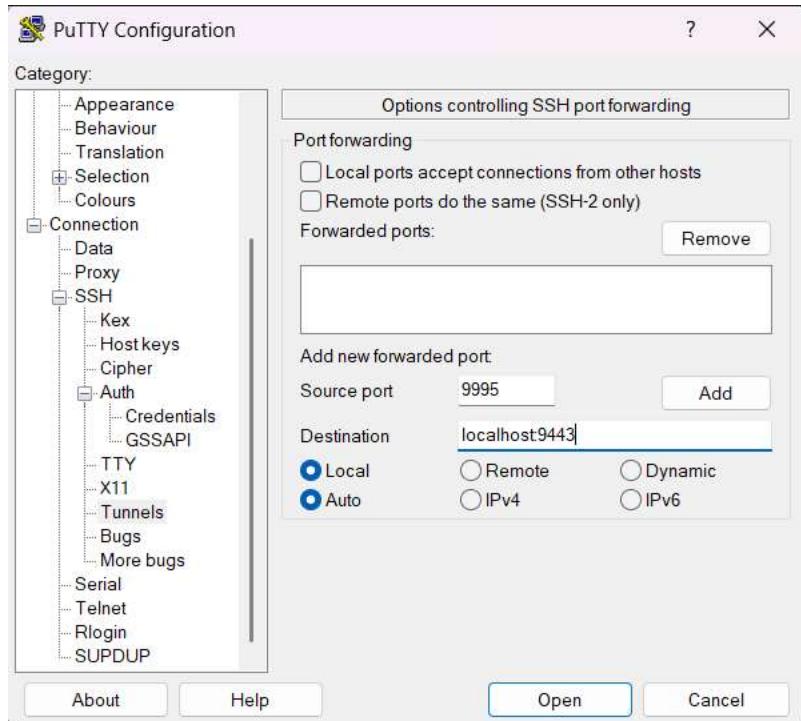
[View web interfaces hosted on Amazon EMR clusters](#)

Close

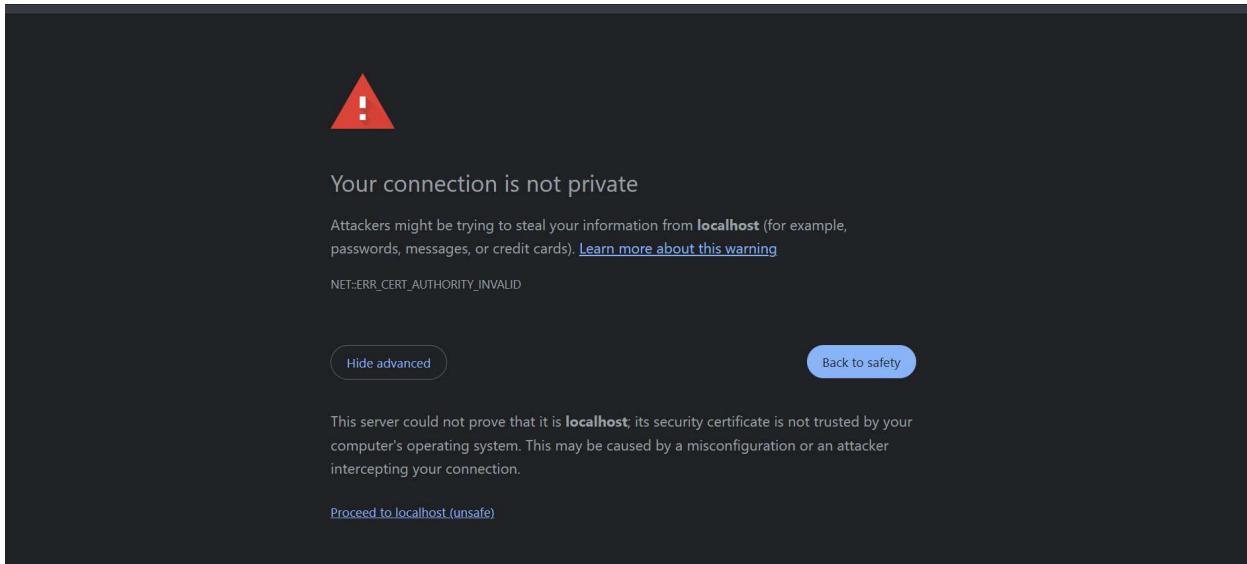








```
[hadoop@ip-172-31-33-199 ~]$ Using username "hadoop".
[hadoop@ip-172-31-33-199 ~]$ Authenticating with public key "imported-openssh-key"
Last login: Sun Aug 24 04:20:11 2025
Amazon Linux 2
AL2 End of Life is 2026-06-30.
A newer version of Amazon Linux is available!
Amazon Linux 2023, GA and supported until 2028-03-15.
https://aws.amazon.com/linux/amazon-linux-2023/
[hadoop@ip-172-31-33-199 ~]$
```



```
Found 3 items
-rw-r--r-- 1 livy hdfsadmingroup          0 2025-08-24 20:18 /user/hadoop/Bigdata_token/_SUCCESS
-rw-r--r-- 1 livy hdfsadmingroup          33 2025-08-24 20:16 /user/hadoop/Bigdata_token/part-00000-f4a5bc09-2f86-4471-85d6-de68dd058565-c000.csv
-rw-r--r-- 1 livy hdfsadmingroup         7305 2025-08-24 20:17 /user/hadoop/Bigdata_token/part-00023-f4a5bc09-2f86-4471-85d6-de68dd058565-c000.csv
[hadoop@ip-172-31-32-150 ~]$
```

Q5-Screenshots

```
[hadoop@ip-172-31-32-150 ~]$ # merge to a single csv on the head node
[hadoop@ip-172-31-32-150 ~]$ hdfs dfs -getmerge -nl /user/hadoop/Bigdata_token ~/Bigdata_token.csv
[hadoop@ip-172-31-32-150 ~]$
```

```
[hadoop@ip-172-31-32-150 ~]$ # merge to a single csv on the head node
[hadoop@ip-172-31-32-150 ~]$ hdfs dfs -getmerge -nl /user/hadoop/Bigdata_token ~/Bigdata_token.csv
[hadoop@ip-172-31-32-150 ~]$ aws s3 cp ~/Bigdata_token.csv s3://akarnik-bstn-bucket/Bigdata_token.csv
upload: ./Bigdata_token.csv to s3://akarnik-bstn-bucket/Bigdata_token.csv
[hadoop@ip-172-31-32-150 ~]$
```

General purpose buckets (1/4) [Info](#)

Buckets are containers for data stored in S3.

Name	AWS Region	Creation date
akarnik-bstn-bucket	US East (Ohio) us-east-2	July 26, 2025, 11:55:49 (UTC-07:00)
aws-logs-314807448672-us-east-2	US East (Ohio) us-east-2	July 29, 2025, 18:01:50 (UTC-07:00)
sagemaker-studio-314807448672-dd9c20l2ptf	US East (Ohio) us-east-2	July 26, 2025, 12:57:56 (UTC-07:00)
sagemaker-us-east-2-314807448672	US East (Ohio) us-east-2	July 26, 2025, 12:57:58 (UTC-07:00)

akarnik-bstn-bucket [Info](#)

[Objects](#) [Metadata](#) [Properties](#) [Permissions](#) [Metrics](#) [Management](#) [Access Points](#)

Objects (1/3)

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](#)

Name	Type	Last modified	Size	Storage class
Bigdata_token.csv	csv	August 24, 2025, 13:30:57 (UTC-07:00)	7.2 KB	Standard
cloud_upload.csv	csv	July 26, 2025, 12:24:25 (UTC-07:00)	138.3 KB	Standard
cloud.csv	csv	July 26, 2025, 11:57:23 (UTC-07:00)	138.3 KB	Standard

Q6-Screenshots

English ▾ Preferences ▾ Contact Us Feedback

AWS Command Line Interface User Guide for Version 2

Get started Service guides Developer tools AI resources Search in this guide Return to the Console

About the AWS CLI

- we support the AWS CLI on Microsoft-supported versions of 64-bit Windows.
- Admin rights to install software

Install or update the AWS CLI

To update your current installation of AWS CLI on Windows, download a new installer each time you update to overwrite previous versions. AWS CLI is updated regularly. To see when the latest version was released, see the [AWS CLI version 2 Changelog](#) on GitHub.

- Download and run the AWS CLI MSI installer for Windows (64-bit): <https://awscli.amazonaws.com/AWSCLIV2.msi>

Alternatively, you can run the `msiexec` command to run the MSI installer.

```
C:\> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi
```

For various parameters that can be used with `msiexec`, see [msiexec](#) on the Microsoft Docs website. For example, you can use the `/qn` flag for a

On this page

- [AWS CLI install and update instructions](#)
- [Troubleshooting AWS CLI install and uninstall errors](#)
- [Next steps](#)

Recommended tasks

How to

- [Verify Session Manager plugin installation and test access](#)
- [Install and update the Session Manager plugin on Windows](#)

Learn about

10:48 PM 2025-08-23

```
karni@Ark CLANGARM64 ~ (master)
$ aws configure
AWS Access Key ID [*****4JA4]: AKIAUSS740RQL4N3RU2J
AWS Secret Access Key [*****i3BB]: bo/Ay6RpsjEuREgluJNog69EXQqRx3SLamifz2v1
Default region name [us-east-2]:
Default output format [json]:
(base)
karni@Ark CLANGARM64 ~ (master)
$ aws s3 ls
2025-07-26 11:55:49 akarnik-bstn-bucket
2025-07-29 18:01:50 aws-logs-314807448672-us-east-2
2025-07-26 12:57:57 sagemaker-studio-314807448672-dd9c2012ptf
2025-07-26 12:58:00 sagemaker-us-east-2-314807448672
(base)
karni@Ark CLANGARM64 ~ (master)
$ |
```