

# Word Count with Spark in AWS EMR

## 1. Introduction

### 1.1 What is EMR?

Amazon EMR (Elastic MapReduce) is a cloud-native big data platform provided by AWS. It allows users to process vast amounts of data using open-source frameworks such as Apache Hadoop, Apache Spark, Hive, Presto, and others. EMR enables fast, cost effective processing of large datasets by distributing the work across multiple EC2 instances.

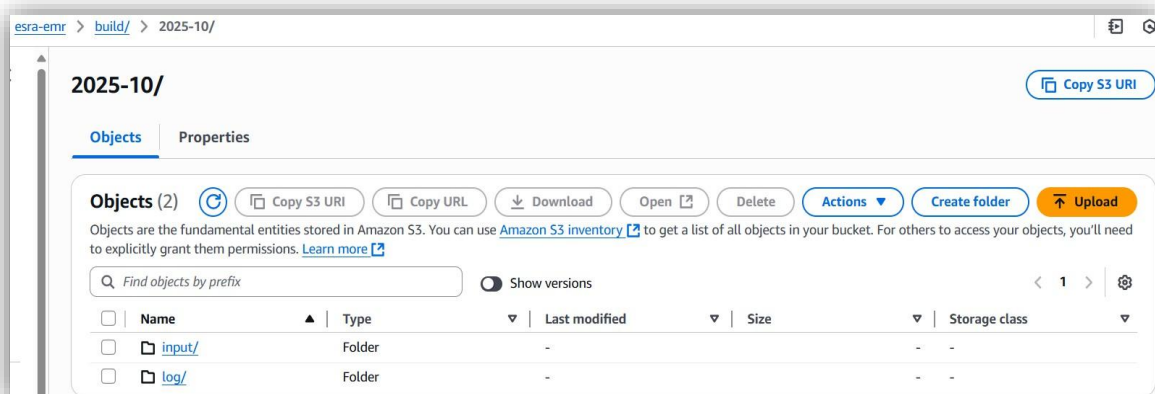
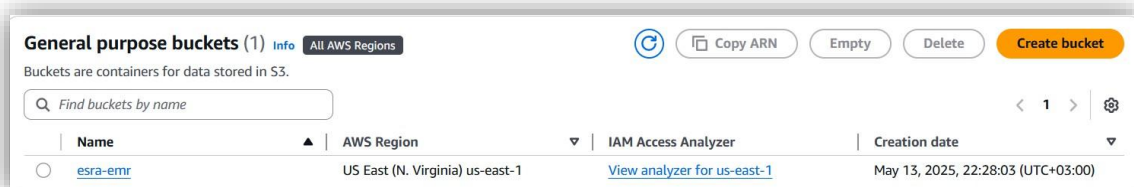
### 1.2 Project Objective

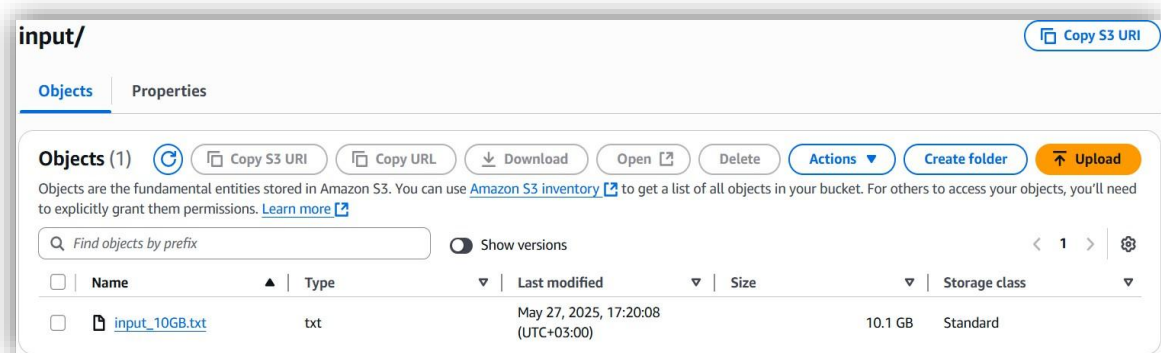
The objective of this project is to perform a Word Count operation using Apache Spark on an EMR cluster. The input text is stored in Amazon S3, processed by the Spark job, and the result (word frequencies) is written back to S3.

## 2. AWS Resources

### 2.1 Creating an S3 Bucket

- An S3 bucket was created to store both the input files. The following files were uploaded:
- **input/**: This folder contains the input.txt file, which is a 10 GB text file used as the input for the word count operation.
- **log/**: This folder was created to store the application and system logs generated by the Spark job during execution.





## 2.2 Creating a VPC

A dedicated Virtual Private Cloud (VPC) named project-vpc was created to provide a secure and isolated network environment for the EMR cluster. The following network components were configured:

- **Public and Private Subnets:** Subnets were created across multiple Availability Zones to ensure high availability and proper distribution of EMR nodes.
- **Internet Gateway:** An internet gateway was attached to the VPC and properly routed to allow the cluster to access the internet when necessary (e.g., downloading dependencies or writing to S3).
- **Route Tables:** Custom route tables were configured to manage traffic between the subnets and the internet gateway.
- **Security Groups:** Security groups were defined to control inbound and outbound access to EMR nodes, ensuring that only necessary ports (such as SSH and application ports) were open.

VPC > Your VPCs > Create VPC

### VPC settings

**Resources to create** [Info](#)  
Create only the VPC resource or the VPC and other networking resources.

☐ VPC only ☒ VPC and more

**Name tag auto-generation** [Info](#)  
Enter a value for the Name tag. This value will be used to auto-generate Name tags for all resources in the VPC.

☒ Auto-generate  
project

**IPv4 CIDR block** [Info](#)  
Determine the starting IP and the size of your VPC using CIDR notation.

10.0.0.0/16 65,536 IPs

CIDR block size must be between /16 and /28.

### Number of public subnets [Info](#)

The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.

0 | **2**

### Number of private subnets [Info](#)

The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.

**0** | 2 | 4

### ► Customize subnets CIDR blocks

### NAT gateways (\$) [Info](#)

Choose the number of Availability Zones (AZs) in which to create NAT gateways. Note that there is a charge for each NAT gateway

**None** | In 1 AZ | 1 per AZ

### VPC endpoints [Info](#)

Endpoints can help reduce NAT gateway charges and improve security by accessing S3 directly from the VPC. By default, full access policy is used. You can customize this policy at any time.

**None** | **S3 Gateway**

#### review

##### VPC [Show details](#)

Your AWS virtual network

project-vpc

##### Subnets (2)

Subnets within this VPC

us-east-1a

project-subnet-public1-us-east-1a

us-east-1b

project-subnet-public2-us-east-1b

##### Route tables (1)

Route network traffic to resources

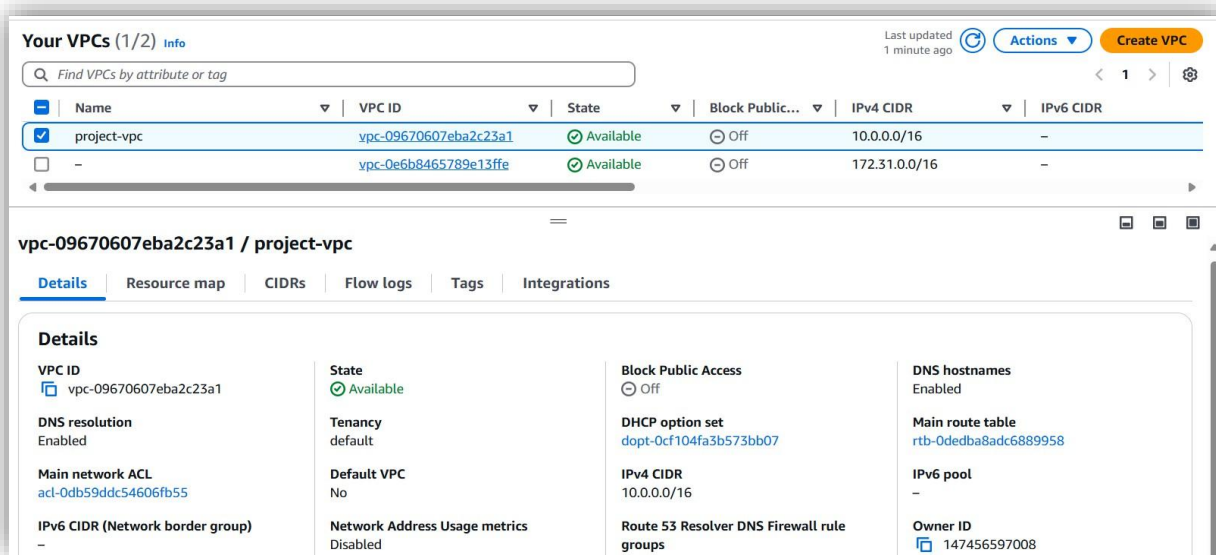
project-rtb-public

##### Network connections (2)

Connections to other networks

project-igw

project-vpce-s3



## 2.3 Configuring the EMR Cluster

To process the 10GB word count job efficiently, an Amazon EMR cluster was configured with the following specifications:

- **Cluster Type:** Temporary, configured to auto-terminate once the Spark job is completed, ensuring cost-efficiency.
- **Number of Instances:**
  - **1 Master node:** Responsible for managing the cluster and coordinating the job execution.
  - **3 Core nodes:** Handled the distributed storage (HDFS) and performed the actual computation in parallel.
  - **Total Nodes: 4**
- **Instance Type:** Each node was provisioned with m5.xlarge instances (or specify if different), providing a balance of compute, memory, and networking capacity.
- **Applications Installed:**
  - Apache Spark: Used for distributed processing of the word count job.
  - Hadoop: Provided the underlying distributed file system and YARN resource management.
- **Logging:**
  - Enabled to monitor job progress, diagnose issues, and audit execution.
  - All logs (stdout, stderr, application logs, system metrics) were stored in the S3 bucket's log/ folder for persistence and easy access after cluster termination.
- **Cluster Configuration:**

- Spark and Hadoop default settings were used with minor tuning for performance (e.g., executor memory, number of partitions, if applicable).
- The cluster was launched in the previously created project-vpc, ensuring secure networking.

▼

Name and applications - *required*

Info

Name your cluster and choose the applications that you want to install to your cluster.

Name

My cluster project

Amazon EMR release

Info


A release contains a set of applications which can be installed on your cluster.

emr-7.8.0


▼

Application bundle


Spark Interactive




Core Hadoop




Flink




HBase




Presto



Trino



Custom



☐ AmazonCloudWatchAgent 1.300032.2
 ☐ HCatalog 3.1.3
 ☒ Hue 4.11.0
 ☒ Livy 0.8.0
 ☐ Pig 0.17.0
 ☐ TensorFlow 2.16.1

☐ Flink 1.20.0
 ☒ Hadoop 3.4.1
 ☒ JupyterEnterpriseGateway 2.6.0
 ☐ Oozie 5.2.1
 ☐ Presto 0.287
 ☐ Tez 0.10.2

☐ HBase 2.6.1
 ☒ Hive 3.1.3
 ☐ JupyterHub 1.5.0
 ☐ Phoenix 5.2.1
 ☒ Spark 3.5.4
 ☐ Trino 467

Primary

Choose EC2 instance type

m5.xlarge

4 vCore 16 GiB memory

EBS only storage On-Demand price: -

Lowest Spot price: -

Actions ▼

Use high availability

Launch highly available, more resilient cluster with three primary nodes on On-Demand Instances. This configuration applies for the lifetime of your cluster. [Learn more](#)

Node configuration - optional

Secondary

Choose EC2 instance type

m5.xlarge

4 vCore 16 GiB memory

EBS only storage On-Demand price: -

Lowest Spot price: -

Actions ▼

Remove instance group

▼

Cluster logs [Info](#)

Choose where and how to store your log files.

☒ Publish cluster-specific logs to Amazon S3

Amazon S3 location

Q

s3://esra-emr/build/2025-10/log

X

View [↗](#)

Browse S3

Format: Use s3://bucket/prefix

☐ Encrypt cluster-specific logs

☒ 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

▼

[↻](#)

☒ 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

▼

[↻](#)

To demonstrate the EMR cluster setup process, screenshots were taken during key stages of configuration and launch. These images serve as visual documentation of the cluster setup.

Clusters (6) [Info](#)

[↻](#) [View details](#) [Terminate](#) [Clone](#) [Create cluster](#)

Filter clusters by status ▼

Find clusters

Filter clusters by creation date-time

< 1 > ⚙

	Cluster ID	Cluster name	Status	Creation time (UTC+03:00)	Elapsed time
<input type="checkbox"/>	<a href="#">j-3EDW0HBF2Q1QA</a>	My cluster project	<span>Waiting</span>	May 27, 2025, 17:57	47 minutes, 41 seconds



My cluster project Updated less than a minute ago Terminate Clone in AWS CLI Clone

**Summary**

**Cluster info**

Cluster ID  
j-3EDW0HBF2Q1QA

Cluster ARN  
[arn:aws:elasticmapreduce:us-east-1:147456597:008:cluster/j-3EDW0HBF2Q1QA](#)

Cluster configuration  
Instance groups

Capacity  
1 Primary 3 Core 0 Task

**Applications**

Amazon EMR version  
emr-7.8.0

Installed applications  
Hadoop 3.4.1, Hive 3.1.3, JupyterEnterpriseGateway 2.6.0, Livy 0.8.0, Spark 3.5.4

**Cluster management**

Log destination in Amazon S3  
[esra-emr/build/2025-10/log](#)

Persistent application UIs  
[Spark History Server](#)  
[YARN timeline server](#)  
[Tez UI](#)

Primary node public DNS  
[ec2-54-163-200-1.compute-1.amazonaws.com](#)  
[Connect to the Primary node using SSH](#)  
[Connect to the Primary node using SSM](#)

**Status and time**

Status  
Waiting

Creation time  
May 27, 2025, 17:57 (UTC+03:00)

Elapsed time  
47 minutes, 52 seconds

**Properties** | Bootstrap actions | Instances (Hardware) | Steps | Applications | Configurations | Monitoring | Events | Tags (0)

**Operating system** [Info](#)

Amazon Linux release  
2023.7.20250428.1

**Cluster logs** [Info](#)

Archive log files to Amazon S3  
Turned on

Amazon S3 location  
[s3://esra-emr/build/2025-10/log/](#)

Encryption for logs  
Turned off

**Cluster termination and node replacement** [Info](#)

[Edit](#)

Termination option  
Automatically terminate cluster after idle time

Idle time  
3 hours

Termination protection  
Off

Unhealthy node replacement  
On

**Network and security** [Info](#)

**Network**

Virtual Private Cloud (VPC)  
[vpc-09670607eba2c25a1](#)

Subnet(s) and Availability Zone(s) (AZ)  
[subnet-0cc1c25734f1d8ddb](#) us-east-1a

► EC2 security groups (firewall)

**Security configuration**

Security configuration  
None

EC2 key pair  
wordcount

**Permissions**

Service role for Amazon EMR  
[EMR\\_DefaultRole](#)

EC2 instance profile  
EMR\_EC2\_DefaultRole

Custom automatic scaling role  
Not configured

## 2.4 Instance Groups Overview

The screenshot below shows the instance groups configured for the EMR cluster:

- **Master Node:**

Handles cluster management, job coordination, and resource tracking.

- **Core Nodes (3 instances):**

Responsible for processing data and storing it using HDFS. These nodes ran the actual Spark tasks during the word count job.

**Instances (4)** [Info](#) Connect Instance state ▼ Actions ▼ Launch instances

Find Instance by attribute or tag (case-sensitive) All states ▼

Instance state = running Clear filters

Instance ID	Instance state ▼	Instance type ▼	Status check	Alarm status	Availability Zone ▼	Public IPv4 DNS ▼
i-0a7ed3a169c2d4896	<span>Running</span> <a href="#">Info</a> <a href="#">Logs</a>	m5.xlarge	<span>3/3 checks pass</span>	<a href="#">View alarms +</a>	us-east-1a	ec2-54-227-57-226.co...
i-09c15f99c33d8f2bb	<span>Running</span> <a href="#">Info</a> <a href="#">Logs</a>	m5.xlarge	<span>3/3 checks pass</span>	<a href="#">View alarms +</a>	us-east-1a	ec2-18-212-249-250.co...
i-03519412e8095477e	<span>Running</span> <a href="#">Info</a> <a href="#">Logs</a>	m5.xlarge	<span>3/3 checks pass</span>	<a href="#">View alarms +</a>	us-east-1a	ec2-44-222-227-43.co...
i-0779cd55339938ed4	<span>Running</span> <a href="#">Info</a> <a href="#">Logs</a>	m5.xlarge	<span>3/3 checks pass</span>	<a href="#">View alarms +</a>	us-east-1a	ec2-54-163-200-1.com...

Instances (4) <small>Info</small>							
<input type="text" value="Find Instance by attribute or tag (case-sensitive)"/>				<span>Connect</span> <span>Instance state ▾</span> <span>Actions ▾</span> <span>Launch instan</span>			
<input type="text" value="Instance state = running"/> <span>×</span> <span>Clear filters</span>				<span>All states ▾</span>			
Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring	Security group name	Key name	
ec2-54-227-57-226.co...	54.227.57.226	–	–	disabled	ElasticMapReduce-slave	wordcount	
ec2-18-212-249-250.co...	18.212.249.250	–	–	disabled	ElasticMapReduce-slave	wordcount	
ec2-44-222-227-43.co...	44.222.227.43	–	–	disabled	ElasticMapReduce-slave	wordcount	
ec2-54-163-200-1.com...	54.163.200.1	–	–	disabled	ElasticMapReduce-master	wordcount	

## 2.5 Security Group Configuration

A custom security group was configured for the EMR cluster to define the allowed inbound traffic.

- **SSH (Port 22)** access was enabled with the following rule:
- **Type:** SSH
- **Protocol:** TCP
- **Port Range:** 22
- **Source:** 0.0.0.0/0 (temporarily enabled for open access during development)

sgr-01b4d17b85900155b

SSH
TCP
22
Custom

Delete

## 3.Connecting to Master Node using PuTTY with Authentication

1. Open PuTTY.
2. In the “Host Name (or IP address)” field, enter the **Master DNS** address.
3. On the left panel, navigate to **Connection > SSH > Auth.**
4. Click on **Browse** next to the “Private key file for authentication” field.
5. Select your **.ppk** private key file.
6. Click **Open** to start the SSH session.

### 3.1 Python Code Explanation

The core script used for the word count is shown below:

```
from pyspark import SparkContext
import re
import argparse
def normalize(line):
```



```

    line = line.lower()

    line = re.sub(r'[^a-z\s]', '', line)

    return line.strip()

def wordcount(data_source: str, output_url: str) -> None:

    sc = SparkContext(appName="WordCountMapReduce")

    try:

        lines = sc.textFile(data_source)

        normalized = lines.map(normalize).filter(lambda x: x != "")

        words = normalized.flatMap(lambda line: line.split())

        word_pairs = words.map(lambda word: (word, 1))

        word_counts = word_pairs.reduceByKey(lambda a, b: a + b)

        sorted_word_counts = word_counts.sortBy(lambda x: x[1], ascending=False)

        top10 = sorted_word_counts.take(10)

        sorted_word_counts.map(lambda x: f"{x[0]},{x[1]}").saveAsTextFile(output_url)

    for word, count in top10:

        print(f"{word}: {count}")

    finally: sc.stop()

if __name__ == "__main__":

    parser = argparse.ArgumentParser()

    parser.add_argument('--data_source', required=True)

    parser.add_argument('--output_url', required=True)

    args = parser.parse_args()

    wordcount(args.data_source, args.output_url)

```

### Explanation:

- ✓ The script normalizes each line: it converts all characters to lowercase and removes non-alphabetic characters (via the normalize function).
- ✓ Lines are split into words using flatMap, and each word is mapped to a (word, 1) pair.
- ✓ Using reduceByKey, the script aggregates word counts by summing up the occurrences of each word. ✓ The word counts are sorted in descending order by frequency using sortBy.
- ✓ The top 10 most frequent words are retrieved using take(10) and printed to the terminal.

- ✓ The complete word count result is saved to the specified output path (output\_url) in the format "word,count".
- ✓ Functions like flatMap, reduceByKey, and sortBy utilize Spark's distributed computing capabilities, making the script efficient for large-scale text data.
- ✓ Finally, results are sorted in descending order and saved to the output S3 path.
- ✓ parser.add\_argument('--data\_source', required=True)
- ✓ parser.add\_argument('--output\_url', required=True)

These lines define the input text source (--data\_source) and the output directory (--output\_url).

### 3.2 Submitting the Spark Job

To execute the word count application on the EMR cluster, the Spark job was submitted using the spark-submit command. The job was configured to read the 10GB input file from S3, perform the word count transformation, and write the results back to another S3 location.

```
➤ spark-submit projectemr.py --data_source s3://esra-emr/build/2025-10/input/input_10GB.txt --output_url s3://esra-emr/build/2025-10/output/
```

## 4.Result

```
25/05/27 15:24:54 INFO DAGScheduler: Job 3 is finished. Cancelling potential speculative or zombie tasks for this job
25/05/27 15:24:54 INFO YarnScheduler: Killing all running tasks in stage 9: Stage finished
25/05/27 15:24:54 INFO DAGScheduler: Job 3 finished: runJob at SparkHadoopWriter.scala:83, took 1.665813 s
25/05/27 15:24:54 INFO SparkHadoopWriter: Start to commit write Job job_202505271524521345998068964600471_0015.
25/05/27 15:24:54 INFO FileOutputCommitter: File Output Committer Algorithm version is 2
25/05/27 15:24:54 INFO FileOutputCommitter: FileOutputCommitter skip cleanup_temporary folders under output directory:false, ignore cleanup failures: true
25/05/27 15:24:54 INFO DirectFileOutputCommitter: Direct Write: ENABLED
25/05/27 15:24:54 INFO DirectFileOutputCommitter: Nothing to clean up since no temporary files were written.
25/05/27 15:24:54 INFO MultipartUploadOutputStream: close closed:false s3://esra-emr/build/2025-10/output/ SUCCESS
25/05/27 15:24:54 INFO SparkHadoopWriter: Write Job job_202505271524521345998068964600471_0015 committed. Elapsed time: 133 ms.
the: 113487575
of: 49348051
to: 46901844
a: 42054224
in: 39221935
and: 37142850
said: 17880910
for: 17786780
that: 16731362
is: 14556402
25/05/27 15:24:54 INFO SparkContext: SparkContext is stopping with exitCode 0.
25/05/27 15:24:54 INFO SparkUI: Stopped Spark web UI at http://ip-10-0-12-203.ec2.internal:4040
25/05/27 15:24:54 INFO YarnClientSchedulerBackend: Interrupting monitor thread
25/05/27 15:24:54 INFO YarnClientSchedulerBackend: Shutting down all executors
25/05/27 15:24:54 INFO YarnSchedulerBackend$YarnDriverEndpoint: Asking each executor to shut down
```

### 4.1 Output and Logs Folders in S3

After the successful execution of the Spark word count job, the output results and execution logs were automatically saved to their respective folders in the S3 bucket.

#### Output Folder:

- Path: s3://esra-emr/build/2025-10/output/

- This folder contains the result files generated by the Spark job.
- The output is split across one or more part files (e.g., part-00000, part-00001, etc.), each containing key-value pairs in the form (word, count).

Amazon S3 > Buckets > esra-emr > build/ > 2025-10/ > output/

Objects (163)

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

Show versions

Name	Type	Last modified	Size	Storage class
_SUCCESS	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00000	-	May 27, 2025, 18:24:54 (UTC+03:00)	17.9 KB	Standard
part-00001	-	May 27, 2025, 18:24:54 (UTC+03:00)	21.8 KB	Standard
part-00002	-	May 27, 2025, 18:24:54 (UTC+03:00)	21.9 KB	Standard
part-00003	-	May 27, 2025, 18:24:54 (UTC+03:00)	18.6 KB	Standard
part-00004	-	May 27, 2025, 18:24:54 (UTC+03:00)	17.4 KB	Standard
part-00005	-	May 27, 2025, 18:24:54 (UTC+03:00)	14.5 KB	Standard
part-00006	-	May 27, 2025, 18:24:54 (UTC+03:00)	20.8 KB	Standard
part-00007	-	May 27, 2025, 18:24:54 (UTC+03:00)	21.8 KB	Standard
part-00008	-	May 27, 2025, 18:24:54 (UTC+03:00)	10.7 KB	Standard
part-00009	-	May 27, 2025, 18:24:54 (UTC+03:00)	17.3 KB	Standard
part-00010	-	May 27, 2025, 18:24:54 (UTC+03:00)	21.2 KB	Standard
part-00011	-	May 27, 2025, 18:24:54 (UTC+03:00)	18.9 KB	Standard
part-00012	-	May 27, 2025, 18:24:54 (UTC+03:00)	21.4 KB	Standard

Amazon S3 > Buckets > esra-emr > build/ > 2025-10/ > output/

Objects (163)

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

Show versions

Name	Type	Last modified	Size	Storage class
part-00112	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00113	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00114	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00115	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00116	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00117	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00118	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00119	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00120	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00121	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00122	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00123	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00124	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00125	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard
part-00126	-	May 27, 2025, 18:24:55 (UTC+03:00)	0 B	Standard

## 4.2 Verifying the Output from S3 bucket

✓ After the Spark job completed, the output was stored in the specified S3 output folder. To quickly verify the results, the following command was used to preview the first 10 lines of the output:

✓ `aws s3 cp s3:///output/wordcount_result/part-00000 - | head 10`

✓ This command downloads the output file from S3 and displays the first 10 lines, allowing a quick check of the word count results without downloading the entire dataset

```

[hadop@ip-10-0-12-203 ~]$ aws s3 cp s3://esra-emr/build/2025-10/output/part-00000 - | head -n 10
the,113487575
of,49348051
to,46901844
a,42054224
in,39221935
and,37142850
said,17880910
for,17786780
that,16731362
is,14556402

```

## Log Folder:

- **Path:** s3://esra-emr/build/2025-10/log/
- This folder contains detailed logs for the EMR cluster and Spark application, including:
  - **stdout/stderr logs**
  - **YARN application logs**
  - **Cluster step execution logs**
  - Useful for debugging, performance analysis, and auditing

Amazon S3 > Buckets > esra-emr > build/ > 2025-10/ > log/

log/ Copy S3 URI

Objects (1) Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

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 Show versions

Name	Type	Last modified	Size	Storage class
j-3EDW0HBF2Q1QA/	Folder	-	-	-

Amazon S3 > Buckets > esra-emr > build/ > 2025-10/ > log/ > j-3EDW0HBF2Q1QA/

j-3EDW0HBF2Q1QA/ Copy S3 URI

Objects (2) Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

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 Show versions

Name	Type	Last modified	Size	Storage class
containers/	Folder	-	-	-
node/	Folder	-	-	-

Amazon S3 > Buckets > esra-emr > build/ > 2025-10/ > log/ > i-3EDW0HBF2Q1QA/ > containers/

containers/

Copy S3 URI

Objects | Properties

Objects (2)

Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

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 Show versions

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	application_1748358114518_0002/	Folder	-	-	-
<input type="checkbox"/>	application_1748358114518_0003/	Folder	-	-	-

Amazon S3 > Buckets > esra-emr > build/ > 2025-10/ > log/ > i-3EDW0HBF2Q1QA/ > node/

node/

Copy S3 URI

Objects | Properties

Objects (4)

Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

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 Show versions

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	i-03519412e8095477e/	Folder	-	-	-
<input type="checkbox"/>	i-0779cd55339938ed4/	Folder	-	-	-
<input type="checkbox"/>	i-09c15f99c33d8f2bb/	Folder	-	-	-
<input type="checkbox"/>	i-0a7ed3a169c2d4896/	Folder	-	-	-

Amazon S3 > Buckets > esra-emr > build/ > 2025-10/ > log/ > i-3EDW0HBF2Q1QA/ > node/ > i-03519412e8095477e/

i-03519412e8095477e/

Copy S3 URI

Objects | Properties

Objects (2)

Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

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 Show versions

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	applications/	Folder	-	-	-
<input type="checkbox"/>	daemons/	Folder	-	-	-

Amazon S3 > Buckets > esra-emr > build/ > 2025-10/ > log/ > i-3EDW0HBF2Q1QA/ > node/ > i-0a7ed3a169c2d4896/ > applications/

applications/

Copy S3 URI

Objects | Properties

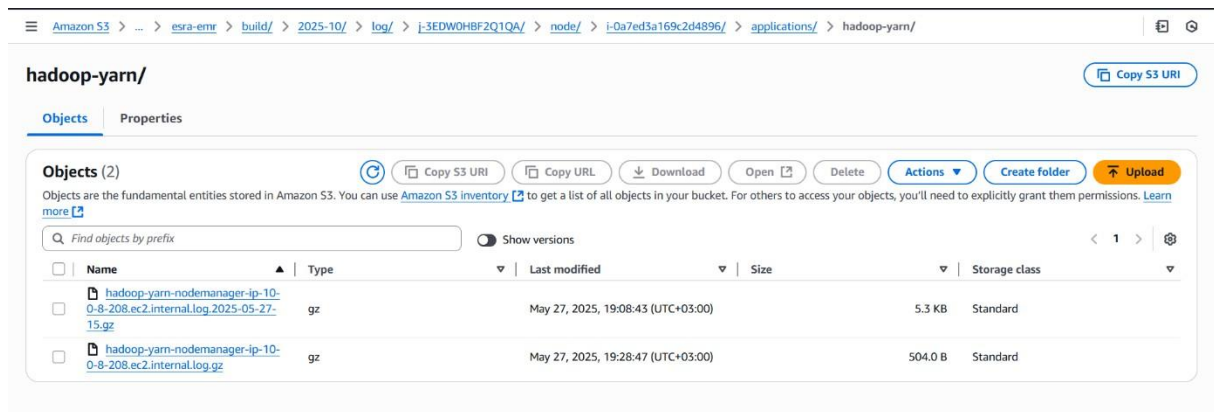
Objects (2)

Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

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 Show versions

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	hadoop-hdfs/	Folder	-	-	-
<input type="checkbox"/>	hadoop-yarn/	Folder	-	-	-



## Verifying Node Activity

The screenshot below shows that all nodes in the EMR cluster were actively participating in the job.

```
25/05/27 15:24:54 INFO TaskSetManager: Finished task 152.0 in stage 9.0 (TID 802) in 84 ms on ip-10-0-14-121.ec2.internal (executor 1) (154/162)
25/05/27 15:24:54 INFO TaskSetManager: Finished task 153.0 in stage 9.0 (TID 803) in 74 ms on ip-10-0-7-233.ec2.internal (executor 2) (155/162)
25/05/27 15:24:54 INFO TaskSetManager: Finished task 154.0 in stage 9.0 (TID 804) in 81 ms on ip-10-0-14-121.ec2.internal (executor 1) (156/162)
25/05/27 15:24:54 INFO TaskSetManager: Finished task 155.0 in stage 9.0 (TID 805) in 77 ms on ip-10-0-14-121.ec2.internal (executor 1) (157/162)
25/05/27 15:24:54 INFO TaskSetManager: Finished task 156.0 in stage 9.0 (TID 806) in 79 ms on ip-10-0-8-208.ec2.internal (executor 3) (158/162)
25/05/27 15:24:54 INFO TaskSetManager: Finished task 157.0 in stage 9.0 (TID 807) in 77 ms on ip-10-0-7-233.ec2.internal (executor 2) (159/162)
25/05/27 15:24:54 INFO TaskSetManager: Finished task 159.0 in stage 9.0 (TID 809) in 78 ms on ip-10-0-8-208.ec2.internal (executor 3) (160/162)
25/05/27 15:24:54 INFO TaskSetManager: Finished task 158.0 in stage 9.0 (TID 808) in 79 ms on ip-10-0-8-208.ec2.internal (executor 3) (161/162)
25/05/27 15:24:54 INFO TaskSetManager: Finished task 160.0 in stage 9.0 (TID 810) in 98 ms on ip-10-0-7-233.ec2.internal (executor 2) (162/162)
```

## 5. Conclusion

In this project, a 10GB word count operation was successfully performed using Apache Spark on an AWS EMR (Elastic MapReduce) cluster. The entire process was built on a scalable and secure cloud infrastructure, making use of key AWS services such as S3, VPC, and EMR.

Key achievements include:

- Efficient processing of a large dataset using a distributed computing framework.
- Secure and isolated networking setup via a custom **VPC** and security groups.
- Centralized storage and logging using **Amazon S3**, allowing easy access to both input/output data and execution logs.
- Automated cluster lifecycle management by configuring EMR to **auto-terminate** after job completion, optimizing resource usage and cost.

This project demonstrates the power and flexibility of AWS for handling big data workloads, and the effectiveness of EMR and Spark in performing distributed data processing tasks at scale.