

AWS Project Report: Word Count with PySpark & Node.js Docker Deployment

Web App URL: <http://3.89.209.39/>

Docker Hub Repo URL: <https://hub.docker.com/r/saiharsha027/node-webserver>

Github: <https://github.com/saiharsha27/word-count-spark-hands-on-11>

Task 1: Word Count on AWS Lightsail using PySpark

Objective

To set up Apache Spark on a Lightsail instance and run a PySpark job to perform word count on a text file stored in an S3 bucket.

1. Lightsail Setup

- Instance: Amazon Linux 2 (1 GB RAM, 1 vCPU)
- Connect using SSH

2. Install Required Software

```
sudo yum update -y
sudo yum install java-11 -y
export JAVA_HOME=$(dirname $(dirname $(readlink -f $(which java))))
sudo mount -o remount,size=2G /tmp
sudo yum install python3-pip -y
pip3 install pyspark
```

3. PySpark Word Count Script (**word_count.py**)

```
from pyspark.sql import SparkSession
```

```
AWS_ACCESS_KEY_ID = 'YOUR_ACCESS_KEY'
AWS_SECRET_ACCESS_KEY = 'YOUR_SECRET_KEY'
S3_INPUT = 's3a://your-bucket/input_file.txt'
S3_OUTPUT = 's3a://your-bucket/output_folder/'
```

```
spark = SparkSession.builder \\\
    .appName("WordCount") \\\
    .config("spark.jars.packages",
"org.apache.hadoop:hadoop-aws:3.3.1,com.amazonaws:aws-java-sdk-bundle:1.11.901") \\\
    .getOrCreate()
```

```
hadoop_conf = spark.sparkContext._jsc.hadoopConfiguration()
hadoop_conf.set("fs.s3a.access.key", AWS_ACCESS_KEY_ID)
hadoop_conf.set("fs.s3a.secret.key", AWS_SECRET_ACCESS_KEY)
```

```
text_file = spark.sparkContext.textFile(S3_INPUT)
counts = text_file.flatMap(lambda line: line.split()) \\\
    .map(lambda word: (word, 1)) \\\
    .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile(S3_OUTPUT)
spark.stop()
```

4. Run Spark Job

spark-submit word_count.py

5. Verification

- Output saved to S3 bucket.
- Spark UI accessible at port 4040.

Screenshots Captured

```
[ec2-user@ip-172-26-3-143 ~]$ vi word_count.py
[ec2-user@ip-172-26-3-143 ~]$ spark-submit --packages org.apache.hadoop:hadoop-aws:3.3.1,com.amazonaws:aws-java-sdk-bundle:1.11.901 word_count.py
:: loading settings :: url = jar:file:/home/ec2-user/.local/lib/python3.9/site-packages/pyspark/jars/ivy-2.5.1.jar!/org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /home/ec2-user/.ivy2/cache
The jars for the packages stored in: /home/ec2-user/.ivy2/jars
org.apache.hadoop:hadoop-aws added as a dependency
com.amazonaws:aws-java-sdk-bundle added as a dependency
:: resolving dependencies :: org.apache.spark:spark-submit-parent-09f8ee67-5c6f-4f99-a814-6b99acda8de9;1.0
  confs: [default]
    found org.apache.hadoop:hadoop-aws:3.3.1 in central
    found com.amazonaws:aws-java-sdk-bundle:1.11.901 in central
    found org.wildfly.openssl:openssl:1.0.7.Final in central
:: resolution report :: resolve 592ms :: artifacts dl 15ms
  :: modules in use:
    com.amazonaws:aws-java-sdk-bundle:1.11.901 from central in [default]
    org.apache.hadoop:hadoop-aws:3.3.1 from central in [default]
    org.wildfly.openssl:openssl:1.0.7.Final from central in [default]
-----
|               | modules | artifacts |
| conf          | number | search/dn | evicted | number | dn |
|-----|-----|-----|-----|-----|
| default       | 3      | 0         | 0       | 3      | 0 |
-----
:: retrieving :: org.apache.spark:spark-submit-parent-09f8ee67-5c6f-4f99-a814-6b99acda8de9
  confs: [default]
  0 artifacts copied, 3 already retrieved (0KB/17ms)
25/04/17 21:37:27 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
25/04/17 21:37:29 INFO SparkContext: Running Spark version 3.5.5
25/04/17 21:37:29 INFO SparkContext: OS info Linux, 6.1.131-143.221.amzn2023.x86_64, amd64
25/04/17 21:37:29 INFO SparkContext: Java version 11.0.26
25/04/17 21:37:29 INFO ResourceUtils: 
=====
25/04/17 21:37:29 INFO ResourceUtils: No custom resources configured for spark.driver.
25/04/17 21:37:29 INFO ResourceUtils: 
=====
25/04/17 21:37:29 INFO SparkContext: Submitted application: WordCount
25/04/17 21:37:29 INFO ResourceProfile: Default ResourceProfile created, executor resources: Map(cores -> name: cores, amount: 1, script: , vendor: , memory -> name: memory, amount: 1024, script: , vendor: , offHea
p -> name: offHeap, amount: 0, script: , vendor: ), task resources: Map(cpus -> name: cpus, amount: 1.0)
25/04/17 21:37:29 INFO ResourceProfile: Limiting resource is cpu
25/04/17 21:37:29 INFO ResourceProfileManager: Added ResourceProfile id: 0
25/04/17 21:37:29 INFO SecurityManager: Changing view acls to: ec2-user
25/04/17 21:37:29 INFO SecurityManager: Changing modify acls to: ec2-user
25/04/17 21:37:29 INFO SecurityManager: Changing view acls groups to:
25/04/17 21:37:29 INFO SecurityManager: Changing modify acls groups to:
25/04/17 21:37:29 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: ec2-user; groups with view permissions: EMPTY; users with modify permissions: ec2-user
25/04/17 21:37:30 INFO Utils: Successfully started service 'sparkDriver' on port 40093.
25/04/17 21:37:30 INFO SparkEnv: Registering MapOutputTracker
25/04/17 21:37:30 INFO SparkEnv: Registering BlockManagerMaster
25/04/17 21:37:30 INFO BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting topology information
25/04/17 21:37:30 INFO BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
25/04/17 21:37:30 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
25/04/17 21:37:30 INFO FileBlockManager: Created local directory at /tmp/blockmgr-b4595936-e393-41d8-8572-a091591f4a3d
25/04/17 21:37:30 INFO MemoryStore: MemoryStore started with capacity 434.4 MiB
25/04/17 21:37:30 INFO SparkEnv: Registering OutputCommitCoordinator
25/04/17 21:37:30 INFO JettyUtils: Start Jetty 0.0.0.0:4040 for SparkUI
25/04/17 21:37:30 INFO SparkContext: Successfully started service 'SparkUI' on port 4040.
25/04/17 21:37:30 INFO SparkContext: Added JAR file:///home/ec2-user/.ivy2/jars/org.apache.hadoop:hadoop-aws-3.3.1.jar at spark://ip-172-26-3-143.ec2.internal:40093/jars/org.apache.hadoop:hadoop-aws-3.3.1.jar with
timestamp 1744925849588
25/04/17 21:37:30 INFO SparkContext: Added JAR file:///home/ec2-user/.ivy2/jars/com.amazonaws:aws-java-sdk-bundle-1.11.901.jar at spark://ip-172-26-3-143.ec2.internal:40093/jars/com.amazonaws:aws-java-sdk-bundle-1.
11.901.jar with timestamp 1744925849588
25/04/17 21:37:30 INFO SparkContext: Added JAR file:///home/ec2-user/.ivy2/jars/org.wildfly.openssl:openssl-1.0.7.Final.jar at spark://ip-172-26-3-143.ec2.internal:40093/jars/org.wildfly.openssl:openssl-1.0.7.Final.jar with timestamp 1744925849588
25/04/17 21:37:30 INFO SparkContext: Added file file:///home/ec2-user/.ivy2/jars/org.apache.hadoop:hadoop-aws-3.3.1.jar at file:///home/ec2-user/.ivy2/jars/org.apache.hadoop:hadoop-aws-3.3.1.jar with timestamp 1744
925849588
25/04/17 21:37:30 INFO Utils: Copying /home/ec2-user/.ivy2/jars/org.apache.hadoop:hadoop-aws-3.3.1.jar to /tmp/spark-6d34c73e-056e-4994-905f-f27f6dbc6f38/userFiles-262a24df-caf8-4428-a408-8d63bbe1e269/org.apache.ha
dop:hadoop-aws-3.3.1.jar
25/04/17 21:37:30 INFO SparkContext: Added file file:///home/ec2-user/.ivy2/jars/com.amazonaws:aws-java-sdk-bundle-1.11.901.jar at file:///home/ec2-user/.ivy2/jars/com.amazonaws:aws-java-sdk-bundle-1.11.901.jar wit
h timestamp 1744925849588
```

us-east-1.console.aws.amazon.com

Search [Option+S]

United States (N. Virginia) saiharsha

Amazon S3

Buckets

com-charlotte-edu-lightsailcloud

Amazon S3

General purpose buckets

Directory buckets

Table buckets

Access Grants

Access Points

Object Lambda Access Points

Multi-Region Access Points

Batch Operations

IAM Access Analyzer for S3

Block Public Access settings for this account

Storage Lens

Dashboards

Storage Lens groups

AWS Organizations settings

Feature spotlight 11

AWS Marketplace for S3

com-charlotte-edu-lightsailcloud

Info

Objects Metadata Properties Permissions Metrics Management Access Points

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

Name	Type	Last modified	Size	Storage class
input.txt	txt	April 17, 2025, 17:32:52 (UTC-04:00)	71.0 B	Standard
output_folder/	Folder	-	-	-

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Amazon Lightsail

Search documentation

Home Lightsail for Research saiharsha (052831358411)

Instances

Containers

Databases

Networking

Storage

Domains & DNS

Snapshots

Exports

Alarm notifications

Documentation

Amazon_Linux_2023-1

Info

Delete Reboot Stop

2 GB RAM, 2 vCPUs, 60 GB SSD

Amazon Linux 2023

AWS Region

Virginia, Zone A (us-east-1a)

Networking type

Dual-stack

Change networking type

Public IPv4 address

3.80.33.229

Private IPv4 address

172.26.3.143

Public IPv6 address

2600:1f18:2aba:1400:bf80:531cb4a7:f

Instance status

Running

Connect

Metrics

Snapshots

Storage

Networking

Domains

Tags

History

Connect to your instance

Info

You can connect using your browser, or your own compatible SSH client.

Use your browser

info

Connect using our browser-based SSH client.

Connect using SSH

Use your own SSH client

info

Use the following credentials to connect to your instance.

CloudShell Support Feedback ©2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms

Task 2: Node.js Web Server with Docker Deployment

Objective

Create a Docker container for a Node.js app, test it locally, push it to Docker Hub, and deploy it on AWS Lightsail.

1. Node.js App Setup

```
mkdir node-webserver && cd node-webserver
npm init -y
npm install express
```

server.js:

```
const express = require('express');
const app = express();
const port = 3000;

app.get('/', (req, res) => {
  res.send('Hello, World! Running in Docker.');
```

```
});

app.listen(port, '0.0.0.0', () => {
  console.log(`Server running at <http://localhost>:${port}`);
});
```

2. Dockerize Application

Dockerfile:

```
FROM node:14
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD ["node", "server.js"]
```

3. Build and Push Docker Image

```
docker buildx create --use
```

```
docker buildx build --platform linux/amd64 -t saiharsha027/node-webserver:latest --push .
```

4. Deploy on Lightsail

- Instance: Amazon Linux 2
- Installed Docker:

```
sudo yum install docker -y
```

```
sudo service docker start
```

```
sudo usermod -aG docker ec2-user
```

- Run container:

```
docker pull saiharsha027/node-webserver:latest
```

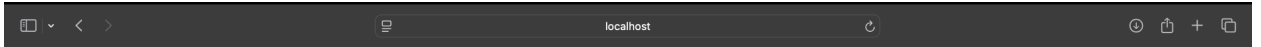
```
docker run -d -p 80:3000 saiharsha027/node-webserver:latest
```

- Ensure Lightsail firewall allows HTTP (port 80)

5. Verification

- Access via: <http://3.89.209.39/>
- Page should show: **Hello, World! Running in Docker.**

Screenshots Captured



Hello, World! Running in Docker.

