**Overview of the Set up of a Spark Cluster**

1. **Amazon S3** will store the dataset.
2. We rent a cluster of machines, i.e., our **Spark Cluster**, and it is located in AWS data centers. We rent these using AWS service called **Elastic Compute Cloud (EC2)**.
3. We log in from your local computer to this Spark cluster.
4. Upon running our Spark code, the cluster will load the dataset from **Amazon S3** into the cluster’s memory distributed across each machine in the cluster.

**New Terms:**

* **Local mode**: You are running a Spark program on your laptop like a single machine.
* **Standalone mode**: You are defining Spark Primary and Secondary to work on your (virtual) machine. You can do this on EMR or your machine. Standalone mode uses a resource manager like YARN or Mesos.

A screenshot of a cell phone

Description automatically generated

### Circling back about HDFS

Previously we have looked over the Hadoop Ecosystem. To refresh those concepts, we have provided reference material here. HDFS (Hadoop Distributed File System) is the file system. HDFS uses MapReduce system as a resource manager.

Spark can replace the MapReduce algorithm. Since Spark does not have its own distributed storage system, it leverages using HDFS or AWS S3, or any other distributed storage. Primarily in this course, we will be using AWS S3, but let’s review the advantages of using HDFS over AWS S3.

### What is HDFS?

HDFS (Hadoop Distributed File System) is the file system in the Hadoop ecosystem. Hadoop and Spark are two frameworks providing tools for carrying out big-data related tasks. While Spark is faster than Hadoop, Spark has one drawback. It lacks a distributed storage system. In other words, Spark lacks a system to organize, store and process data files.

### MapReduce System

HDFS uses MapReduce system as a resource manager to allow the distribution of the files across the hard drives within the cluster. Think of it as the MapReduce System storing the data back on the hard drives after completing all the tasks.

Spark, on the other hand, runs the operations and holds the data in the RAM memory rather than the hard drives used by HDFS. Since Spark lacks a file distribution system to organize, store and process data files, Spark tools are often installed on Hadoop because Spark can then use the Hadoop Distributed File System (HDFS).

### Why do you need ****EMR Cluster****?

Since a Spark cluster includes multiple machines, in order to use Spark code on each machine, we would need to download and install Spark and its dependencies. This is a manual process. **Elastic Map Reduce** is a service offered by AWS that negates the need for you, the user, to go through the manual process of installing Spark and its dependencies for each machine.

### Creating EMR Script

While creating EMR through AWS console has been shown, but if you know the specificity of your instances, such as which applications you need or what kind of clusters you’ll need, you can reuse the EMR script that we will create below multiple times.

A screenshot of a cell phone

Description automatically generated

### Learning Components on EMR Script

Let’s break down the code and go over each part of the code in the EMR script. It’s important that you know what each component does in order to launch a proper cluster and services attached to this script.

#### EMR Script Components

* aws emr : Invokes the AWS CLI, and specifically the command for EMR.
* create-cluster : Creates a cluster
* --name : You can give any name for this - this will show up on your AWS EMR UI. This can be duplicate as existing EMR.
* --release-label: This is the version of EMR you’d like to use.
* --instance-count: Annotates instance count. One is for the primary, and the rest are for the secondary. For example, if --instance-count is given 4, then 1 instance will be reserved for primary, then 3 will be reserved for secondary instances.
* --applications: List of applications you want to pre-install on your EMR at the launch time
* --bootstrap-actions: You can have a script stored in S3 that pre-installs or sets

environmental variables, and call that script at the time EMR launches

* --ec2-attributes KeyName: Specify your permission key name, for example, if it is MyKey.pem, just specify MyKey for this field
* --instance-type: Specify the type of instances you want to use. [Detailed list can be accessed here](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-supported-instance-types.html), but find the one that can fit your data and your budget.
* --log-uri: S3 location to store your EMR logs in. This log can store EMR metrics and also the metrics/logs for submission of your code.

Now it’s your turn to create your own EMR script.