Elastic MapReduce (EMR)

# Index

Table of Contents

[Index 2](#_Toc28728258)

[Elastic MapReduce 3](#_Toc28728259)

[Definition 3](#_Toc28728260)

[Node Types 3](#_Toc28728261)

[Storage 3](#_Toc28728262)

[Steps 4](#_Toc28728263)

[Lifecycle 4](#_Toc28728264)

# Elastic MapReduce

## Definition

* Allows one to use a managed Hadoop framework to process massive amounts of data across scalable EC2 instances.
* It also allows one to run other distributed frameworks such as Apache Spark, HBase, Flink, Presto, etc.

## Node Types

* Leader Node
  + Managers the cluster by coordinating the distribution of data and tasks among other nodes.
  + Tracks the status of tasks and the overall health of a cluster.
  + Every cluster has a leader node, also known as ‘Master’ node.
* Worker Nodes
  + Core Nodes
    - They run tasks
    - Store data in the HDFS of the cluster.
    - Multi-node clusters have at least one worker node.
  + Task Nodes
    - They run tasks.
    - Unlike core nodes, they do not store data.
    - They are optional.

## Storage

* Hadoop Distributed File System (HDFS)
  + It is a distributed data and scalable file system for Hadoop that stores data across multiple EC2 instances in the cluster.
  + It stores multiple copies of the data on different instances.
  + The stored data is lost when cluster is terminated.
  + Used most often for storing intermediate results.
* EMR File System (EMRFS)
  + Allows EMR cluster to access data stored in S3 as if it were a file system like HDFS.
  + Most often used to durably store input and output data.
  + EMRFS Consistent View: Special functionality that enforces S3 consistency by using DynamoDB table.
* Local File System
  + The EC2 instances used to make up the cluster also have some storage capacity in the form of their local disks.
  + This data does not persist after the lifetime of the EC2 instance.

## Steps

To process data on EMR cluster, one can submit jobs directly to Big Data applications or run steps to process data.

* When submitting jobs directly:
  + One connects directly to the EMR Master Node.
  + One uses the installed frameworks such as Apache Spark to load, process and/or output the data
* When running steps to process data:
  + Submit one or more steps to EMR for processing
  + Steps are descriptions of work that contain instructions for cluster software to manipulate, including steps to output the data.
  + One can handle cases in which these steps might fail.

## Lifecycle

* STARTING
  + The cluster is created using the default AMI for EMR or a custom Amazon Linux AMI.
* BOOTSTRAPPING
  + EMR runs any bootstrapping actions specified such as installing custom applications or customizations.
  + It also installs the native applications/frameworks configured during creation.
* RUNNING
  + The cluster is ready and can be connected to.
  + The cluster is currently running some step provided to it.
* WAITING
  + The cluster goes to this step if is has finished any steps provided to it..  
    If configured to auto-terminate when the work is completed, it moves to the SHUTTING\_DOWN state.
* SHUTTING\_DOWN
  + The cluster terminates the instance within the EMR Cluster.
* COMPLETED
  + All the instances have been terminated.
* FAILED
  + If cluster fails to create properly, it will end up in a FAILED state.

## Security

* IAM Roles can offer nuanced permission controls and determine who and what can access the cluster.
* SSH can be configured to access the cluster instances using pre-specified key-pairs.
* Kerberos can also be used.