Amazon Elastic MapReduce

Amazon Elastic MapReduce (Amazon EMR) es un servicio web para la configuración y depliegue de un cluster basado en instancias de máquinas en el servicio Amazon Elastic Compute Cloud (Amazon EC2) y que es gestionado mediante Hadoop. También se puede ejecutar en Amazon EMR otros marcos de trabajo distribuídos como Spark, e interactuar con los datos en otros almacenes de datos como Amazon S3.

Creación de un cluster con EMR

Amazon Elastic MapReduce (Amazon EMR) is a web service that makes it easy to process vast amounts of data quickly and cost-effectively. Amazon EMR uses Hadoop, an open source framework, to distribute your data and processing across a resizable cluster of Amazon EC2 instances. It can also run other distributed frameworks such as Spark and Presto. Amazon EMR is used in a variety of applications, including log analysis, web indexing, data warehousing, machine learning, financial analysis, scientific simulation, and bioinformatics.

Amazon EMR has made enhancements to Hadoop and other open-source applications to work seamlessly with AWS. For example, Hadoop clusters running on Amazon EMR use Amazon Elastic Compute Cloud instances as virtual Linux servers for the master and slave nodes, Amazon Simple Storage Service for bulk storage of input and output data, and Amazon CloudWatch to monitor cluster performance and raise alarms. You can also move data into and out of Amazon DynamoDB using Amazon EMR and Hive. All of this is orchestrated by Amazon EMR control software that launches and manages the Hadoop cluster.

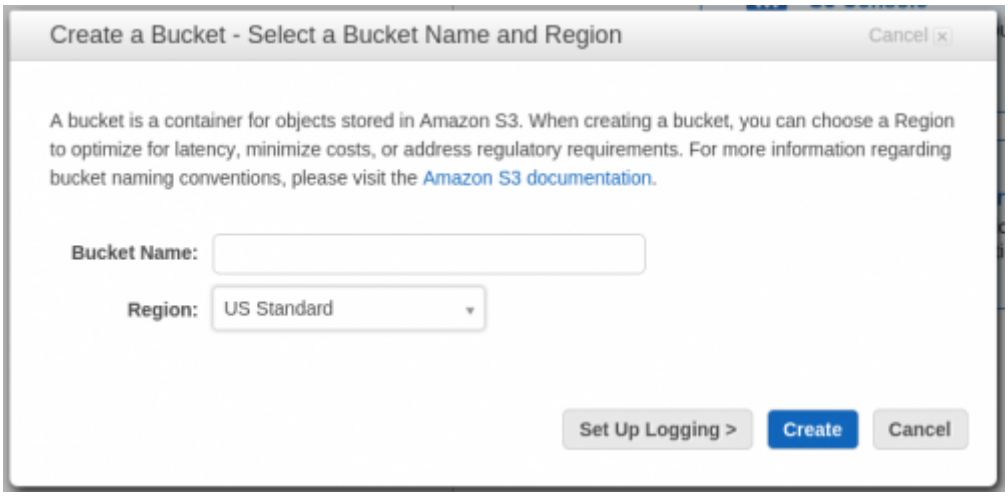
Open-source projects that run on top of the Hadoop architecture can also be run on Amazon EMR. The most popular applications, such as Hive (a SQL-like scripting language for data warehousing and analysis), Pig (a scripting language for data analysis and transformation), HBase (a columnar, NoSQL data store), DistCp (a tool for copying large data sets), Ganglia (a monitoring framework), Impala (a distributed SQL-like query language), and Hue (a web interface for analyzing data), are already integrated with Amazon EMR. By running Hadoop on Amazon EMR you get the benefits of the cloud: the ability to inexpensively provision clusters of virtual servers within minutes. You can scale the number of virtual servers in your cluster to manage your computation needs, and only pay for what you use.

You can run your cluster as a transient process: one that launches the cluster, loads the input data, processes the data, stores the output results, and then automatically shuts down. This is the standard model for a cluster that is performing a periodic processing task. Shutting down the cluster automatically ensures that you are only billed for the time required to process your data. The other model for running a cluster is as a long-running cluster. In this model, you launch a cluster and submit jobs interactively using the command line or you submit units of work called steps. From there you might interactively query the data, use the cluster as a data warehouse, or do periodic processing on a large data set. In this model, the cluster persists even when there are no steps or jobs queued for processing.

Almacenamento con S3

Amazon EMR puede hacer uso de Amazon S3 como almacenamiento de los datos de entrada, los ficheros de log y los datos de salida. Para más información sobre este tipo de sistema de almacenamiento visita la wiki de amazon.

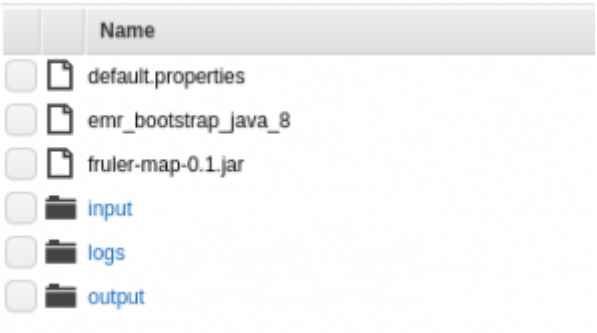
Para crear un nuevo contenedor de datos S3 (bucket), solamente es necesario entrar en el servicio S3 y pulsar “Create Bucket” rellenando el nombre del nuevo contenedor y la región donde estará el mismo (es importante que esta sea la misma región utilizada para desplegar el cluster EMR).



Una vez creado el contenedor, suele ser una buena práctica organizarlo de la siguiente manera:

- Crear una carpeta Log donde guardar los logs de los despliegues de máquinas EC2, así como de las ejecuciones de los diferentes trabajos.
- Crear una carpeta `input` para tener almacenados todos los datos de entrada.
- Crear una carpeta `output` que servirá para guardar los resultados de las ejecuciones.

Además, será necesario tener en este contenedor todo lo necesario para el trabajo que se vaya a ejecutar en el cluster, así como los diferentes scripts de configuración (tal como se comenta en la siguiente sección).



Configuración do cluster

Logs

Spark sobre EMR

Instalar Spark

```
s3://support.elasticmapreduce/spark/install-spark
```

Executar un traballo

- **Step type**: Custom JAR
- **JAR Location**:

  ```
  s3://<CLUSTER_REGION>.elasticmapreduce/libs/script-runner/script-runner.jar
  ```

- **Arguments**:

  ```
  /home/hadoop/spark/bin/spark-submit --deploy-mode cluster --master yarn-cluster --class <MAIN_CLASS> s3://<BUCKET>/<FILE_JAR> <JAR_OPTIONS>
  ```

Java 8 en EMR

```
# Check java version
JAVA_VER=$(java -version 2>&1 | sed 's/java version "\(.*\)\.\(.*\)\..*"/\1\2/; 1q')

if [ "$JAVA_VER" -lt 18 ]
then
    # Download jdk 8
    echo "Downloading and installing jdk 8"
    wget --no-cookies --no-check-certificate --header "Cookie: gpw_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie" "http://download.oracle.com/otn-pub/java/jdk/8-b132/jdk-8-linux-x64.rpm"

    # Silent install
    sudo yum -y install jdk-8-linux-x64.rpm

    # Figure out how many versions of Java we currently have
    NR_OF_OPTIONS=$(echo 0 | alternatives --config java 2>/dev/null | grep 'There ' | awk '{print $3}' | tail -1)

    echo "Found $NR_OF_OPTIONS existing versions of java. Adding new version."

    # Make the new java version available via /etc/alternatives
```

```
    sudo alternatives --install /usr/bin/java java
/usr/java/default/bin/java 1

    # Make java 8 the default
    echo $(($NR_OF_OPTIONS + 1)) | sudo alternatives --config java

    # Set some variables
    export JAVA_HOME=/usr/java/default/bin/java
    export JRE_HOME=/usr/java/default/jre
    export PATH=$PATH:/usr/java/default/bin
fi

# Check java version again
JAVA_VER=$(java -version 2>&1 | sed 's/java version
"\(.*\)\.\(.*\)\..*"/\1\2/; 1q')

echo "Java version is $JAVA_VER!"
echo "JAVA_HOME: $JAVA_HOME"
echo "JRE_HOME: $JRE_HOME"
echo "PATH: $PATH"
```