

Proyecto final DataPath con AWS

Por Gonzalo Hernández Hernández

Tecnologías usadas en este proyecto: EMR, Hadoop, HDFS, Hive y Spark



EMR

Realizar los siguientes pasos:

Arquitectura de solución IoT

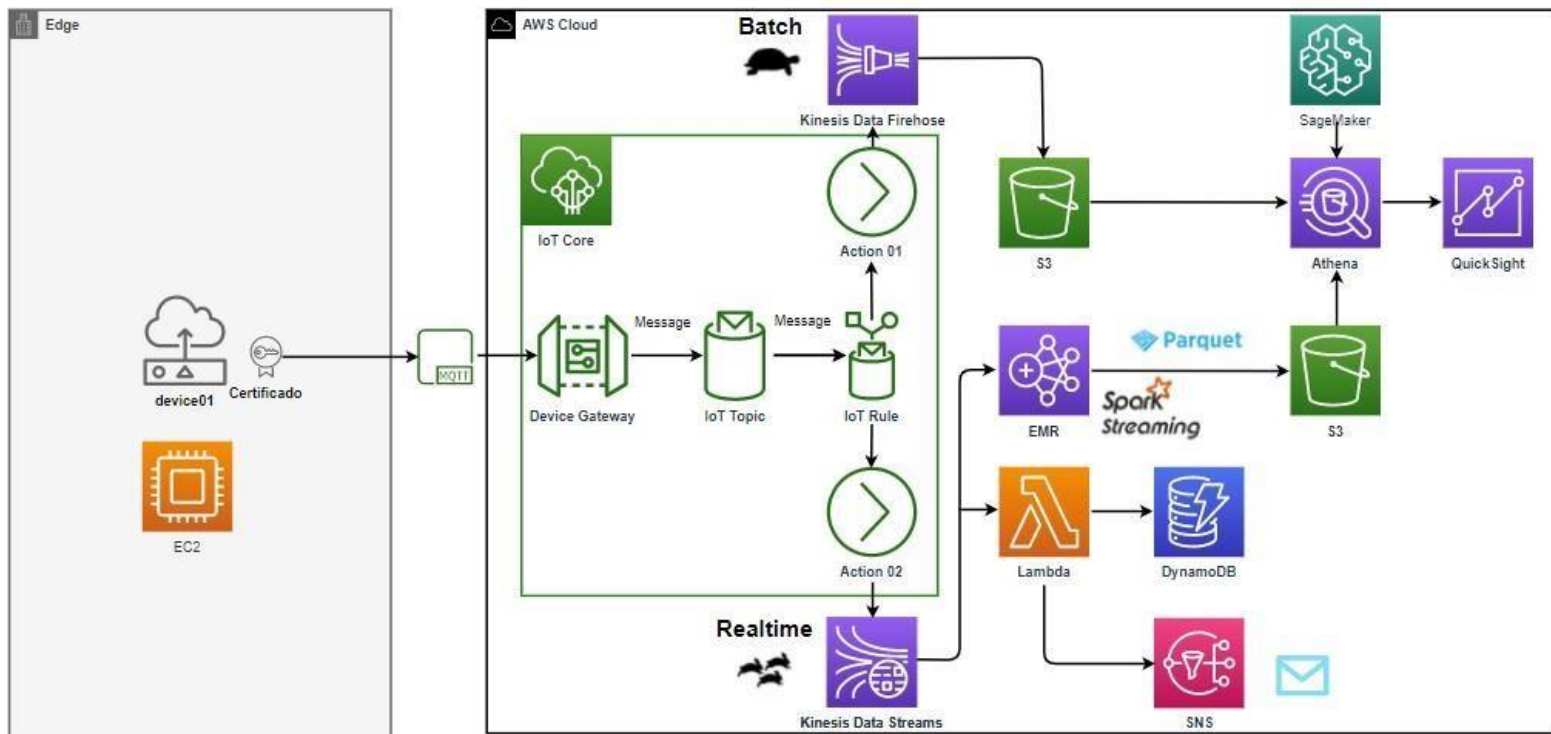


Imagen de la arquitectura propuesta para el proyecto.

a) Entrar a la consola de AWS.

En Cloud9 crear una carpeta y subir los siguientes archivos:

- EMR.yaml
- SendDataSmartFarming.py
- policy-iot.json
- code-spark-streaming

Para crear algunos recursos del proyecto se debe crear el stack de CloudFormation.

Antes de ejecutar agregar el correo a utilizar en la línea 166.

```
rSubscription:
  Type: 'AWS::SNS::Subscription'
  DependsOn: rSnsTopic
  Properties:
    TopicArn: !Ref rSnsTopic
    Endpoint: [REDACTED]
    Protocol: email

rFunctionLambda:
  Type: 'AWS::Lambda::Function'
  Properties:
    FunctionName: AlertFarmingRealtime
```

Ejecutar en el AWS CLI, el siguiente comando que creará un conjunto de recursos.

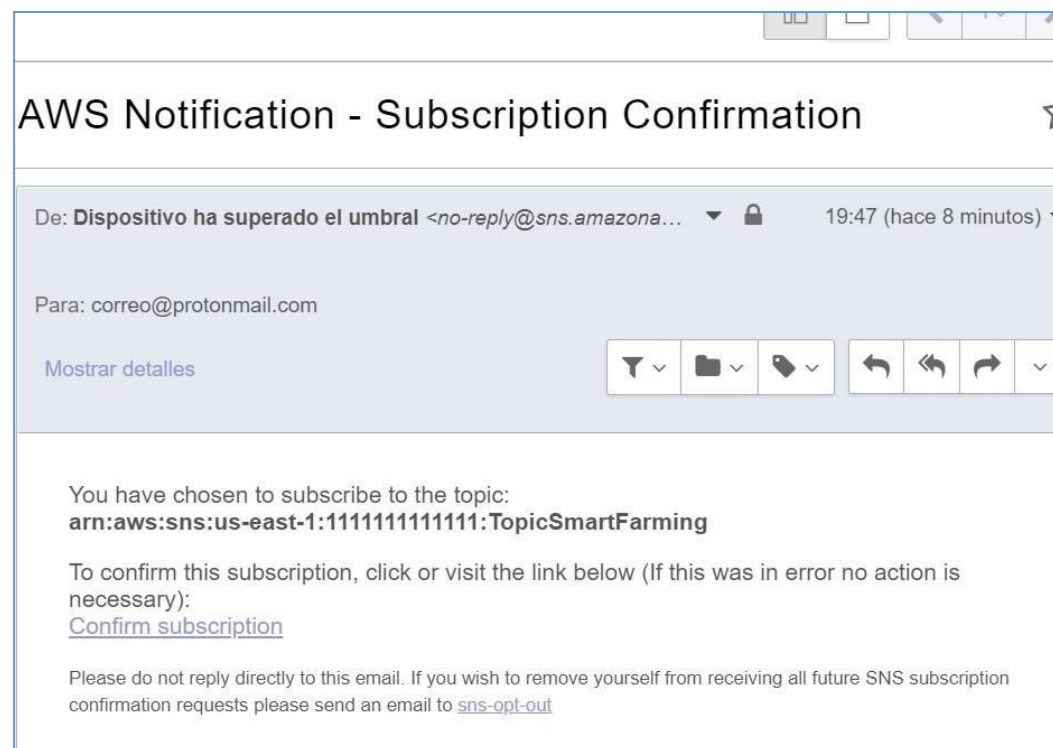
```
aws cloudformation create-stack --stack-name StackEMR --template-body file://EMR.yaml --capabilities CAPABILITY_NAMED_IAM
```

El stack de CloudFormation crea los siguientes recursos (tomará menos de minuto aprox):

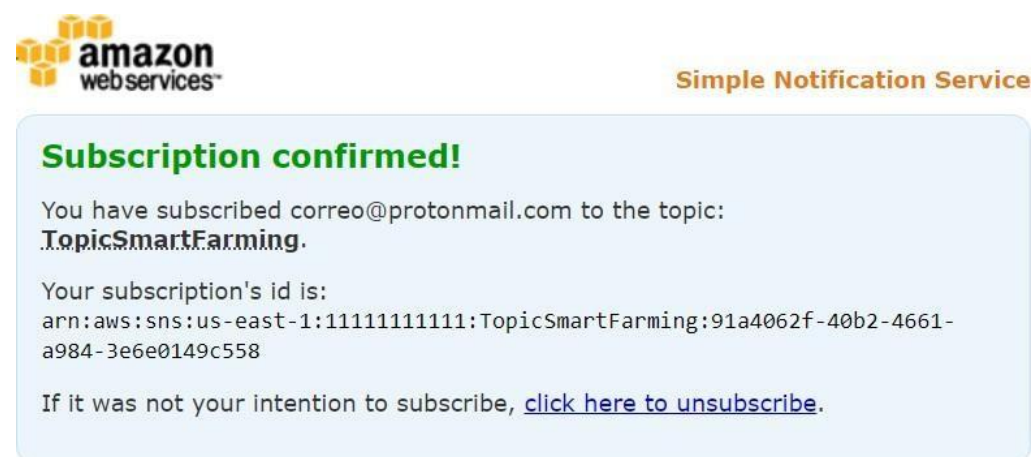
- Bucket en S3
- Kinesis Data Streams
- Kinesis Data Firehose
- Tabla en DynamoDB
- Tópico en SNS
- Suscriptor en el tópico creado
- Función Lambda
- Regla en IoT con 2 acciones

b) Confirmar la suscripción al tópico.

Acceder al correo personal y confirmar la suscripción.

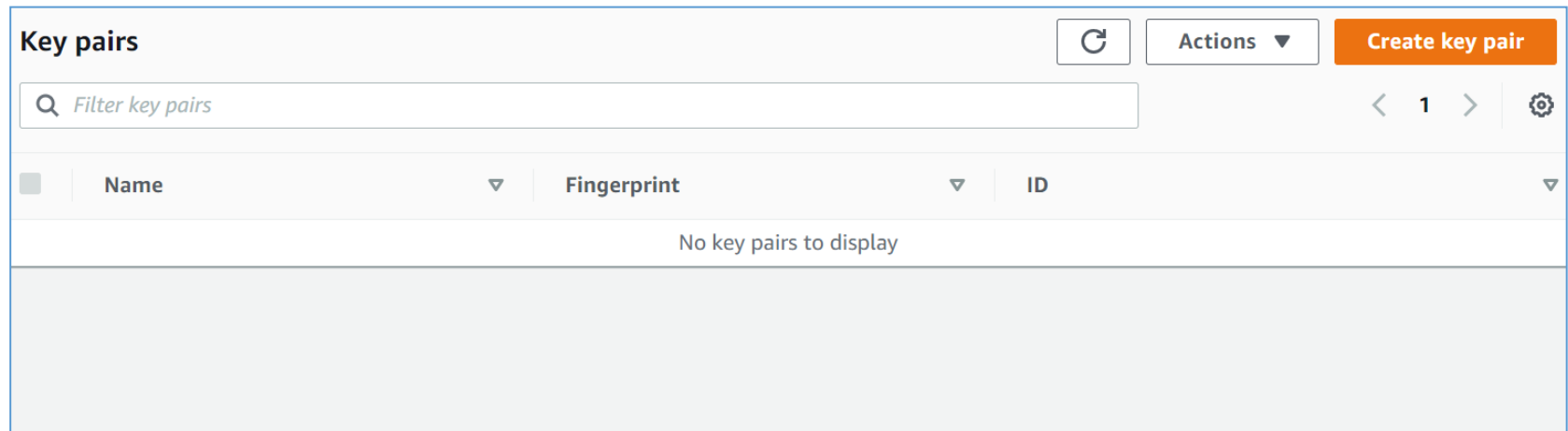


Una vez se le haya dado clic, aparecerá el siguiente mensaje.



Crear un par de claves.

Ir al servicio de EC2 -> Key pairs
Clic en Create key pair.



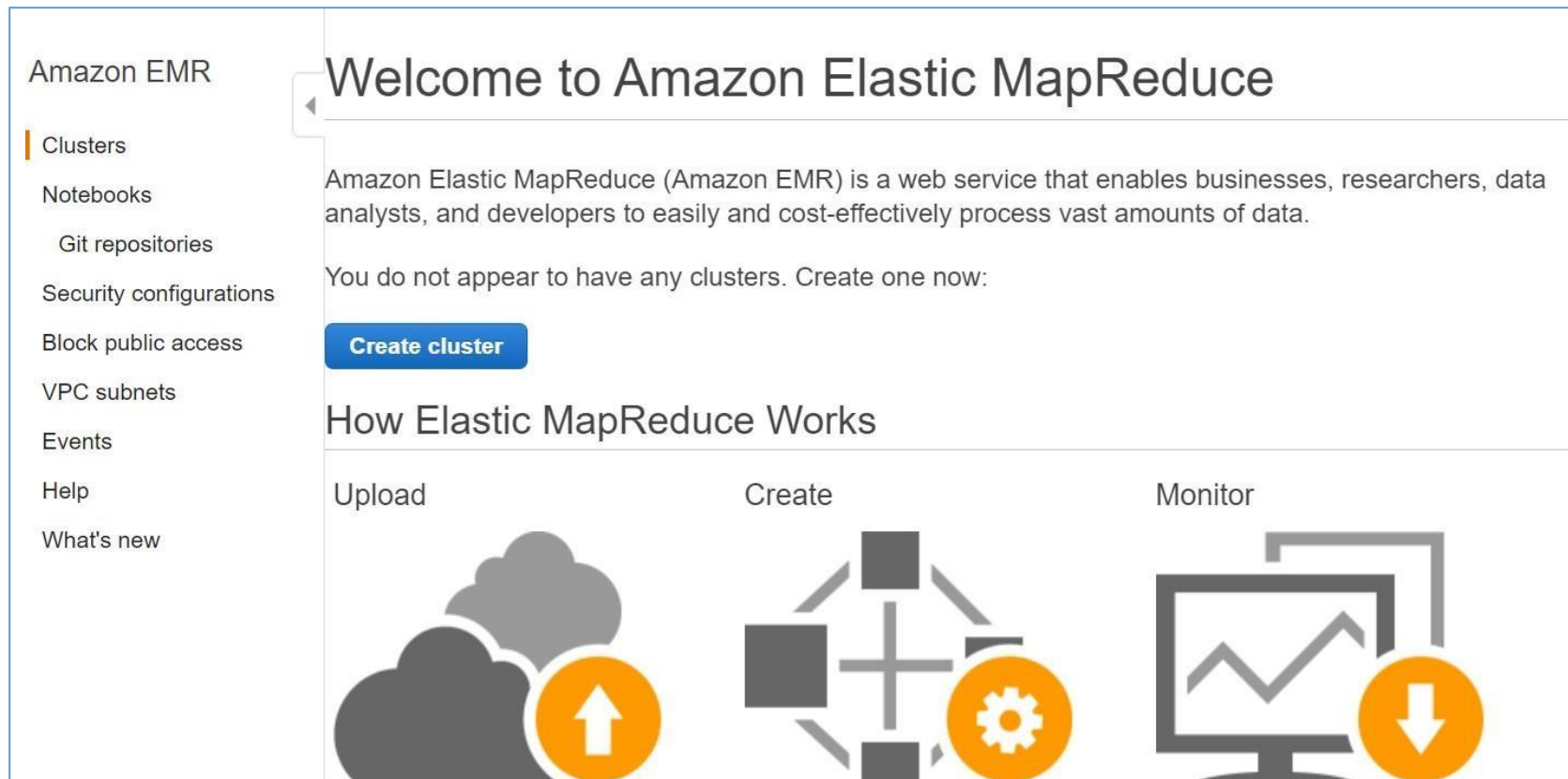
Asignar un nombre al par de claves, seleccionar el formato de archivo pem y clic en Crear par de claves.

The screenshot shows the 'Create key pair' wizard. The title is 'Create key pair'. Below it, a section titled 'Key pair' explains that a key pair consists of a private key and a public key used for authentication. The 'Name' field is filled with 'BigData-EMR'. A note states: 'The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.' Under 'File format', the 'pem' option is selected with a radio button, and it is noted 'For use with OpenSSH'. The 'ppk' option is unselected, noted 'For use with PuTTY'. There is a 'Tags (Optional)' section with the text 'No tags associated with the resource.' and an 'Add tag' button. A note at the bottom of the tags section says 'You can add 50 more tags.' At the bottom right of the wizard are 'Cancel' and 'Create key pair' buttons.

Se descarga el archivo. pem en nuestro pc local (nos servirá para loguearnos al clúster EMR)

c) Crear un clúster en EMR

Clic en Create cluster.



Clic en Go to advanced options.

Create Cluster - Quick Options [Go to advanced options](#)

General Configuration

Cluster name

☒ Logging [i](#)

S3 folder [i](#)

Launch mode ☒ Cluster [i](#) ☐ Step execution [i](#)

Software configuration

Release [i](#)

Applications

- ☒ Core Hadoop: Hadoop 2.10.1, Hive 2.3.7, Hue 4.8.0, Mahout 0.13.0, Pig 0.17.0, and Tez 0.9.2
- ☐ HBase: HBase 1.4.13, Hadoop 2.10.1, Hive 2.3.7, Hue 4.8.0, Phoenix 4.14.3, and ZooKeeper 3.4.14
- ☐ Presto: Presto 0.240.1 with Hadoop 2.10.1 HDFS and Hive 2.3.7 Metastore
- ☐ Spark: Spark 2.4.7 on Hadoop 2.10.1 YARN and Zeppelin 0.8.2

Seleccionar la versión: Emr.5.32.0
 Y seleccionar: Hadoop 2.10.0
 Hive 2.3.7
 Spark 2.4.7

Clic en siguiente.

Create Cluster - Advanced Options

Go to quick options

Step 1: Software and Steps

Step 2: Hardware

Step 3: General Cluster Settings

Step 4: Security

Software Configuration

Release

emr-5.32.0

☒

Hadoop 2.10.1

☐

JupyterHub 1.1.0

☐

Ganglia 3.7.2

☒

Hive 2.3.7

☐

JupyterEnterpriseGateway 2.1.0

☐

Mahout 0.13.0

☐

Oozie 5.2.0

☐

TensorFlow 2.3.1

☐

Zeppelin 0.8.2

☐

Tez 0.9.2

☐

HBase 1.4.13

☐

Presto 0.240.1

☐

MXNet 1.7.0

☐

Hue 4.8.0

☒

Spark 2.4.7

☐

Livy 0.7.0

☐

Flink 1.11.2

☐

Pig 0.17.0

☐

ZooKeeper 3.4.14

☐

Sqoop 1.4.7

☐

Phoenix 4.14.3

☐

HCatalog 2.3.7

Multiple master nodes (optional)

☐

Use multiple master nodes to improve cluster availability. [Learn more](#)

AWS Glue Data Catalog settings (optional)

Selecciona la zona de disponibilidad us-east-1a o us-east-1b
El clúster tendrá: 1 nodo principal, 2 nodos secundarios y 2 nodos de tareas (spot 90% de ahorro)
Clic en siguiente.

Configuración de hardware ⓘ

Specify the networking and hardware configuration for your cluster. Request Spot instances (unused EC2 capacity) to save money.

Cluster Composition

Si necesita más de 20 instancias EC2, consulte este tema ↗.

Configuración del grupo de instancias

- ☒ **Grupos de instancias uniformes**
Especifique un tipo de instancia única y la opción de compra para cada tipo de nodo.
- ☐ **Flotas de instancias**
Especifique la capacidad de destino y cómo Amazon EMR la cumple para cada tipo de nodo. Combine los tipos de instancias y las opciones de compra. [Más información](#) ↗

Networking

Use a Virtual Private Cloud (VPC) to process sensitive data or connect to a private network. Launch the cluster into a VPC with a public, private or shared subnet. Subnets may be associated with and AWS Outpost or AWS Local Zone.

Launch the cluster into a VPC with a public, private, or shared subnet. Subnets may be associated with an AWS Outpost or AWS Local Zone.

Network vpc-e2b7e098 (172.31.0.0/16) (default) [Create a VPC](#) ↗ ⓘ

EC2 Subnet subnet-58935915 | Default in us-east-1c ▾

Cluster Nodes and Instances

Choose the instance type, number of instances, and a purchasing option. [Learn more about instance purchasing options](#) ↗

ⓘ Console options for automatic scaling have changed. [Learn more](#) ↗

Node type	Instance type	Instance count	Purchasing option
Master Master - 1	m5.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB ⓘ Add configuration settings	1 Instances	<input checked="" type="radio"/> On-demand ⓘ <input type="radio"/> Spot ⓘ Use on-demand as max price ▾
Core Core - 2	m5.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB ⓘ Add configuration settings	<input type="text" value="2"/> Instances	<input checked="" type="radio"/> On-demand ⓘ <input type="radio"/> Spot ⓘ Use on-demand as max price ▾
Task Task - 3	m5.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB ⓘ Add configuration settings	<input type="text" value="2"/> Instances	<input type="radio"/> On-demand ⓘ <input checked="" type="radio"/> Spot ⓘ Use on-demand as max price ▾

+ Add task instance group

Total core and task units 4 Total units

Cluster scaling

Adjust the number of Amazon EC2 instances available to an EMR cluster via EMR-managed scaling or a custom automatic scaling policy. [Learn more](#) ↗

Cluster scaling ☐ Enable Cluster Scaling

EBS Root Volume

Specify the root device volume size up to 100 GiB. This sizing applies to all instances in the cluster. [Learn more](#) ↗


Root device EBS volume size GiB

Cancel Previous Next

Asignar un nombre al clúster y clic en Next.

General Options

Cluster name

☒ Logging ⓘ
S3 folder 

☐ Log encryption ⓘ

☒ Debugging ⓘ

☒ Termination protection ⓘ

Tags ⓘ

Key	Value (optional)
<input type="text" value="Add a key to create a tag"/>	<input type="text"/>

Additional Options

☐ EMRFS consistent view ⓘ

Custom AMI ID ⓘ

▶ Bootstrap Actions

[Cancel](#) [Previous](#) [Next](#)

Seleccionar el par de claves que se ha creado y clic en Create cluster.

Security Options

EC2 key pair ⓘ

☒ Cluster visible to all IAM users in account ⓘ

Permissions ⓘ

☒ Default ☐ Custom

Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.

EMR role [EMR_DefaultRole](#) ⓘ

EC2 instance profile [EMR_EC2_DefaultRole](#) ⓘ

Auto Scaling role [EMR_AutoScaling_DefaultRole](#) ⓘ

▶ Security Configuration

▶ EC2 security groups

[Cancel](#) [Previous](#) [Create cluster](#)

d) Crear recursos en AWS IoT Core desde el AWS CLI, nos ubicamos en la carpeta **08emr**

Ejecutar desde el AWS CLI

```
aws iot create-thing --thing-name DeviceFarming
```

Crear certificados y activarlos (pegarlo en una sola línea para que no salga error), copiar el valor del certificateArn.

```
aws iot create-keys-and-certificate \
--set-as-active \
--certificate-pem-outfile certificate.pem.crt \
--public-key-outfile public.pem.key \
--private-key-outfile private.pem.key
```

Crear política

```
aws iot create-policy --policy-name pol-farming-iot --policy-document file://policy-iot.json
```

Añadir política al certificado

```
aws iot attach-policy --target <ARN-NUESTRO-CERTIFICADO> --policy-name pol-farming-iot
```

Añadir certificado al objeto

```
aws iot attach-thing-principal --principal <ARN-NUESTRO-CERTIFICADO> --thing-name DeviceFarming
```

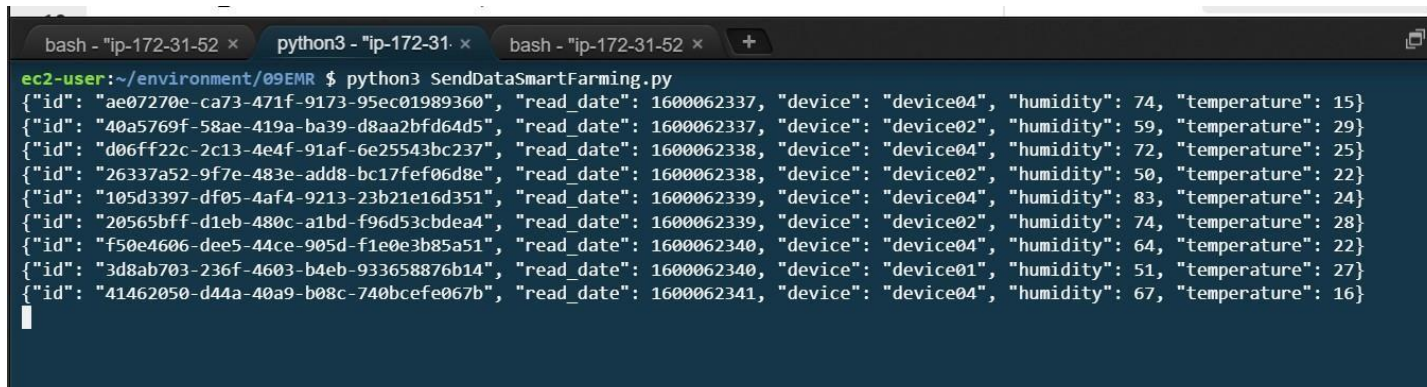
Descargar certificados.

```
wget https://www.amazontrust.com/repository/AmazonRootCA1.pem
```

Obtener el valor del endpoint de AWS IoT Core (copiamos el valor del endpoint y remplazarlo en la línea 13 del archivo SendDataSmartFarming.py)

```
aws iot describe-endpoint --endpoint-type iot:Data-ATS
```

Ejecutar el archivo SendDataSmartFarming.py para hacer una prueba con el cliente MQTT.



```
ec2-user:~/environment/09EMR $ python3 SendDataSmartFarming.py
[{"id": "ae07270e-ca73-471f-9173-95ec01989360", "read_date": 1600062337, "device": "device04", "humidity": 74, "temperature": 15}, {"id": "40a5769f-58ae-419a-ba39-d8aa2bfd64d5", "read_date": 1600062337, "device": "device02", "humidity": 59, "temperature": 29}, {"id": "d06ff22c-2c13-4e4f-91af-6e25543bc237", "read_date": 1600062338, "device": "device04", "humidity": 72, "temperature": 25}, {"id": "26337a52-9f7e-483e-add8-bc17fef06d8e", "read_date": 1600062338, "device": "device02", "humidity": 50, "temperature": 22}, {"id": "105d3397-df05-4af4-9213-23b21e16d351", "read_date": 1600062339, "device": "device04", "humidity": 83, "temperature": 24}, {"id": "20565bff-d1eb-480c-a1bd-f96d53cbdea4", "read_date": 1600062339, "device": "device02", "humidity": 74, "temperature": 28}, {"id": "f50e4606-dee5-44ce-905d-f1e0e3b85a51", "read_date": 1600062340, "device": "device04", "humidity": 64, "temperature": 22}, {"id": "3d8ab703-236f-4603-b4eb-933658876b14", "read_date": 1600062340, "device": "device01", "humidity": 51, "temperature": 27}, {"id": "41462050-d44a-40a9-b08c-740bcefe067b", "read_date": 1600062341, "device": "device04", "humidity": 67, "temperature": 16}]
```

Verificar en el cliente MQTT que estén llegando los datos.
Suscribirse al tópic: sensor-iot/farming

Subscribe to a topic

Publish to a topic

Topic filter [Info](#)

The topic filter describes the topic(s) to which you want to subscribe. The topic filter can include MQTT wildcard characters.

sensor-iot/farming

► Additional configuration

Subscribe

Subscriptions

sensor-iot/farming

PauseClearExportEdit

sensor-iot/farming

▼ sensor-iot/farming

March 02, 2021, 00:36:50 (UTC-0500)

```
{
  "id": "3c945a6f-00dc-4f64-bb89-07b5f52cb84a",
  "read_date": 1614663410,
  "device": "device02",
  "humidity": 58,
  "temperature": 29
}
```

Actualizar reglas de entrada del grupo de seguridad en el EMR.
En el EMR, seleccionar el Security group for master.

Clone

Terminate

AWS CLI export

Cluster: ClusterBigData

Waiting

Cluster ready after last step completed.

Summary

Application user interfaces

Monitoring

Hardware

Configurations

Events

Steps

Bootstrap actions

Summary

Configuration details

ID: j-48X1T5UJ7MJI

Creation date: 2021-03-02 00:27 (UTC-5)

Elapsed time: 9 minutes

After last step completes: Cluster waits

Termination protection: On [Change](#)

Tags: -- [View All / Edit](#)

Master public DNS: ec2-3-88-180-92.compute-1.amazonaws.com [Connect to the Master Node Using SSH](#)

Release label: emr-5.32.0

Hadoop distribution: Amazon 2.10.1

Applications: Hive 2.3.7, Spark 2.4.7

Log URI: s3://aws-logs-971489366207-us-east-1/elasticmapreduce/ [📄](#)

EMRFS consistent view: Disabled

Custom AMI ID: --

Application user interfaces

Network and hardware

Persistent user interfaces [🔗](#): Spark history server, YARN timeline server, Tez UI

On-cluster user interfaces [🔗](#): Not Enabled [Enable an SSH Connection](#)

Availability zone: us-east-1c

Subnet ID: [subnet-58935915](#) [🔗](#)

Master: Running 1 m5.xlarge

Core: Running 2 m5.xlarge

Task: Running 2 m5.xlarge

Spot (max on-demand)

Cluster scaling: Not enabled

Security and access

Key name: BigData-EMR

EC2 instance profile: EMR_EC2_DefaultRole

EMR role: EMR_DefaultRole

Auto Scaling role: EMR_AutoScaling_DefaultRole

Visible to all users: All [Change](#)

Security groups for Master: [sg-0b9e8d3b8d95f17e9](#) [🔗](#) (ElasticMapReduce-master)

Security groups for Core & Task: [sg-05ad52dbbb6958e18](#) [🔗](#) (ElasticMapReduce-slave)

Seleccionar el grupo de seguridad del nodo master.

Security Groups (2) Info

Filter security groups

search: sg-0b9e8d3b8d95f17e9 X

Clear filters

1

Create security group

<input type="checkbox"/>	Name	Security group ID	Security group name	VPC ID
<input type="checkbox"/>	-	sg-05ad52dbbb6958e18	ElasticMapReduce-slave	vpc-e2b7e098 vpc-e2b7e098
<input type="checkbox"/>	-	sg-0b9e8d3b8d95f17e9	ElasticMapReduce-master	vpc-e2b7e098 vpc-e2b7e098

Clic en Edit inbound rules.

Inbound rules

Outbound rules

Tags

Inbound rules

Edit inbound rules

Type	Protocol	Port range	Source	Description - optional
All TCP	TCP	0 - 65535	sg-05ad52dbbb6958e18 (ElasticMapReduce-slave)	-
All TCP	TCP	0 - 65535	sg-0b9e8d3b8d95f17e9 (ElasticMapReduce-master)	-
Custom TCP	TCP	8443	207.171.167.25/32	-
Custom TCP	TCP	8443	54.240.217.80/29	-
Custom TCP	TCP	8443	72.21.198.64/29	-
Custom TCP	TCP	8443	207.171.167.101/32	-

Añadir una nueva regla para el puerto 22, como se muestra, clic en Save rules.

SSH

TCP

22

Anywh...

0.0.0.0/0 X

::/0 X

Add rule

NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.

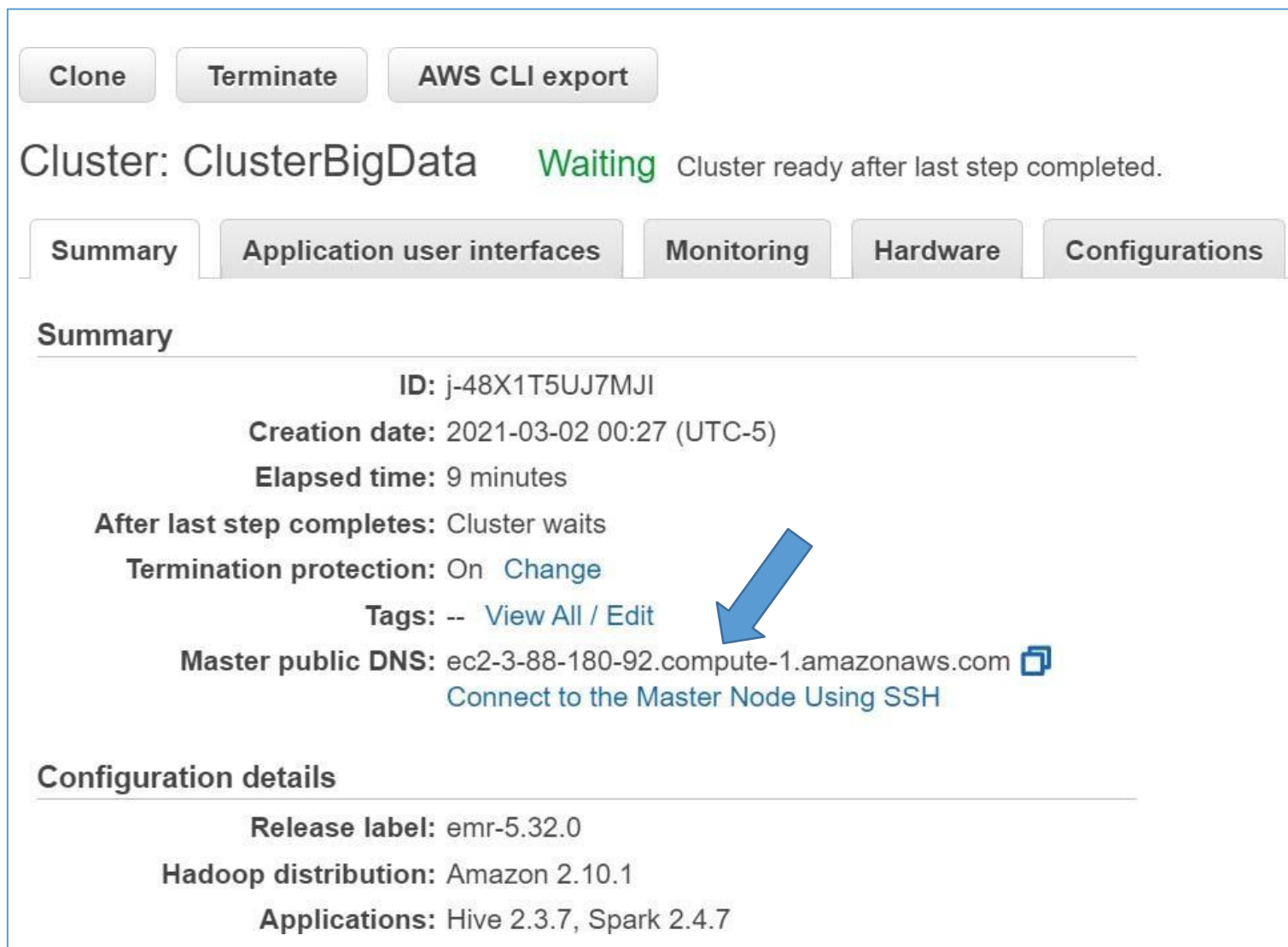
Cancel

Preview changes

Save rules

k) Entrar a Cloud9

Subir el archivo pem del par de claves para iniciar sesión en el EMR por SSH.
Para iniciar sesión se debe obtener el DNS público principal del EMR.



Clone Terminate AWS CLI export

Cluster: ClusterBigData **Waiting** Cluster ready after last step completed.

Summary Application user interfaces Monitoring Hardware Configurations

Summary

ID: j-48X1T5UJ7MJI

Creation date: 2021-03-02 00:27 (UTC-5)

Elapsed time: 9 minutes

After last step completes: Cluster waits

Termination protection: On [Change](#)

Tags: -- [View All / Edit](#)

Master public DNS: ec2-3-88-180-92.compute-1.amazonaws.com [Connect to the Master Node Using SSH](#)

Configuration details

Release label: emr-5.32.0

Hadoop distribution: Amazon 2.10.1

Applications: Hive 2.3.7, Spark 2.4.7

Ingresa al servicio de IAM, busca el rol: **EMR_EC2_DefaultRole** y agregarle la política de Kinesis con la acción **ListShards**.

Dentro del Cloud9, abrimos un terminal, nos ubicamos en el nivel donde se encuentra el archivo pem y digitamos lo siguiente.

```
chmod 400 BigData-EMR.pem
```

```
ssh -i BigData-EMR.pem hadoop@TU_DNS
```



```

ec2-user:~/environment/09EMR $ ssh -i BigData-EMR.pem hadoop@ec2-54-209-14-145.compute-1.amazonaws.com
Last login: Fri Jul 17 06:06:44 2020

  _| _|_)
  _| ( _/
  _|\_|_|

Amazon Linux 2 AMI

https://aws.amazon.com/amazon-linux-2/
18 package(s) needed for security, out of 72 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMM      MMMMMMM RRRRRRRRRRRRRR
E::::::::::::::::::::E M::::::::M      M::::::::M R::::::::::::R
EE::::::::EEEEEEEE::::E M::::::::M      M::::::::M R::::RRRRRR::::R
E::::E      EEEEE M::::::::M      M::::::::M RR::::R      R::::R
E::::E      M::::M:M::M      M::M::::M      R::R      R::::R
E::::EEEEEEEEEE M::::M M::M M::M M::::M      R::RRRRRR::::R
E::::::::::::E M::::M M::M:M::M M::::M      R:::::::::RR
E::::EEEEEEEEEE M::::M M::::M M::::M      R::RRRRRR::::R
E::::E      M::::M M::M      M::::M      R::R      R::::R
E::::E      EEEEE M::::M      MMM      M::::M      R::R      R::::R
EE::::::::EEEEEEEE::::E M::::M      M::::M      R::R      R::::R
E::::::::::::E M::::M      M::::M      RR::::R      R::::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM      MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-17-1 ~]$

```

Pasar a la instancia del EMR el código de Spark Streaming

Si se desea poner en producción nuestro código de Spark, se debe compilar y empaquetar el código en un JAR.

Esto se hará usando sbt.

Primeramente se procede a instalarlo ya que no viene en EMR por defecto, así que se debe ejecutar lo siguiente.

```

sudo rm -f /etc/yum.repos.d/bintray-rpm.repo
curl -L https://www.scala-sbt.org/sbt-rpm.repo > sbt-rpm.repo
sudo mv sbt-rpm.repo /etc/yum.repos.d/
sudo yum install sbt -y

```

Una vez se haya instalado el sbt, se debe descomprimir el archivo code-spark-streaming.zip

```

unzip code-spark-streaming.zip

```

Ya que se haya descomprimido el siguiente paso es posicionarse dentro de la carpeta:

```

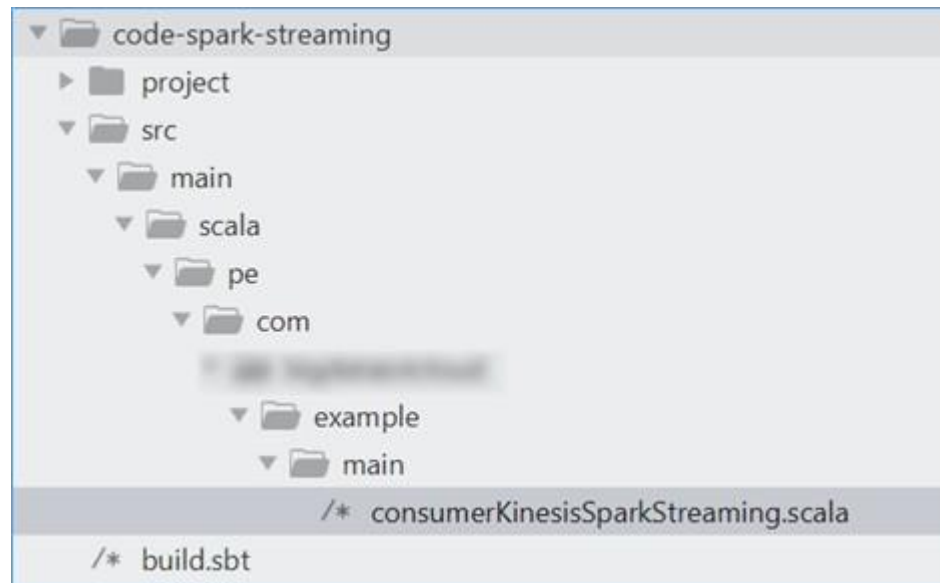
cd code-spark-streaming

```

```

sbt assembly

```



Se debe verificar que se esté ejecutando el código de python:

`python3 SendDataSmartFarming.py`

```
bash - "ip-172-31-52" x python3 - "ip-172-31-52" x +
{"id": "135bc842-aea4-427d-933f-42ce7ebd6e37", "read_date": 1595055909, "device": "device08", "humidity": 72, "temperature": 33}
{"id": "36ed1b59-76d9-470c-967d-b66d7b13ff2e", "read_date": 1595055909, "device": "device05", "humidity": 81, "temperature": 22}
{"id": "fb29d1b7-48a1-4630-811d-165e94fd2248", "read_date": 1595055910, "device": "device06", "humidity": 82, "temperature": 16}
{"id": "bbadc9aa-cc0a-4053-9508-2265e603818f", "read_date": 1595055910, "device": "device02", "humidity": 90, "temperature": 19}
{"id": "d8ebd500-415d-4943-abe4-35233a132dad", "read_date": 1595055911, "device": "device05", "humidity": 84, "temperature": 23}
{"id": "d9250b48-8ed1-4d8d-ad60-7a7ca3120408", "read_date": 1595055911, "device": "device08", "humidity": 80, "temperature": 28}
{"id": "8bf021cd-be4c-489a-9a09-1b8b395143c0", "read_date": 1595055912, "device": "device01", "humidity": 55, "temperature": 30}
{"id": "904e76e3-33fa-4508-8ebe-df691ded0226", "read_date": 1595055912, "device": "device03", "humidity": 59, "temperature": 29}
{"id": "6f3eeacf-be9d-4915-9ff7-8606372686a9", "read_date": 1595055913, "device": "device01", "humidity": 85, "temperature": 29}
{"id": "7e0dc306-a4f5-48a4-b636-64c0c0ad3409", "read_date": 1595055913, "device": "device08", "humidity": 76, "temperature": 35}
{"id": "b9cce9cc-a428-445a-84dd-33d1c6d01e4b", "read_date": 1595055914, "device": "device05", "humidity": 67, "temperature": 20}
{"id": "0aa62038-f4ce-4c95-a7c0-68d076db757d", "read_date": 1595055914, "device": "device04", "humidity": 54, "temperature": 26}
{"id": "744f0e5e-13e7-4835-84d5-24a4a662e2b0", "read_date": 1595055915, "device": "device04", "humidity": 68, "temperature": 20}
{"id": "1a33a90c-e132-4a9f-8490-314509739d8f", "read_date": 1595055915, "device": "device02", "humidity": 86, "temperature": 33}
{"id": "ed88b1c0-d240-4b73-a7ba-a0964322a448", "read_date": 1595055916, "device": "device05", "humidity": 72, "temperature": 34}
{"id": "c20ff902-0c49-4390-b40f-0e27f3ba6838", "read_date": 1595055916, "device": "device01", "humidity": 69, "temperature": 20}
```

Ejecutar el siguiente comando en EMR.

Cambiar el nombre del bucket para que se guarde la información que se lea en tiempo real en formato parquet a S3.

El comando debe estar en una sola línea.

`spark-submit --class pe.com.proyectoiot.example.main.consumerKinesisSparkStreaming /home/hadoop/code-spark-streaming/target/scala-2.11/kinesis-spark-streaming-assembly-0.1.jar KinesisSparkExample StreamFarming https://kinesis.us-east-1.amazonaws.com s3://smartfarming-xxxxxxx/data/input/streaming`


```

20/09/14 06:17:15 INFO BlockGenerator: Pushed block input-0-1600064235200
20/09/14 06:17:16 INFO MappedDStream: Slicing from 1600064232000 ms to 1600064236000 ms (aligned to 1600064232000 ms and 1600064236000 ms)
20/09/14 06:17:16 INFO JobScheduler: Added jobs for time 1600064236000 ms
20/09/14 06:17:16 INFO JobScheduler: Finished job streaming job 1600064233000 ms.0 from job set of time 1600064233000 ms
20/09/14 06:17:16 INFO JobScheduler: Total delay: 3.008 s for time 1600064233000 ms (execution: 0.814 s)
20/09/14 06:17:16 INFO JobScheduler: Starting job streaming job 1600064234000 ms.0 from job set of time 1600064234000 ms
20/09/14 06:17:16 INFO KinesisBackedBlockRDD: Removing RDD 191 from persistence list
20/09/14 06:17:16 INFO KinesisInputDStream: Removing blocks of RDD KinesisBackedBlockRDD[191] at build at consumerKinesisSparkStreaming.scala:70 of time 1600064233000 ms
20/09/14 06:17:16 INFO ReceivedBlockTracker: Deleting batches: 1600064226000 ms
20/09/14 06:17:16 INFO InputInfoTracker: remove old batch metadata: 1600064226000 ms
+-----+-----+-----+-----+-----+
|          id|          read_date|          device|          humidity|          temperature|
+-----+-----+-----+-----+-----+
|{"id": "2fd95a78-...| "read_date": 160...| "device": "devic...| "humidity": 67| "temperature": 28}|
|{"id": "66dcfaf0-...| "read_date": 160...| "device": "devic...| "humidity": 69| "temperature": 34}|
|{"id": "1924b951-...| "read_date": 160...| "device": "devic...| "humidity": 85| "temperature": 26}|
|{"id": "59d9361e-...| "read_date": 160...| "device": "devic...| "humidity": 86| "temperature": 26}|
|{"id": "a9b43f44-...| "read_date": 160...| "device": "devic...| "humidity": 62| "temperature": 25}|
|{"id": "4e266b70-...| "read_date": 160...| "device": "devic...| "humidity": 53| "temperature": 32}|
+-----+-----+-----+-----+-----+
20/09/14 06:17:16 INFO BlockGenerator: Pushed block input-0-1600064236200

```

Una vez terminado el proyecto eliminar los recursos.

Primeramente se debe eliminar el clúster en EMR.

Eliminamos el bucket de S3

```
aws s3 ls | grep smartfarming | awk {'print "aws s3 rb s3://" $3 " --force"} |sh
```

Y se ejecuta el comando resultante.

Eliminar el stack de CloudFormation.

```
aws cloudformation delete-stack --stack-name StackEMR
```

Eliminar la tabla KinesisSparkExample en DynamoDB

```
aws dynamodb delete-table --table-name KinesisSparkExample
```