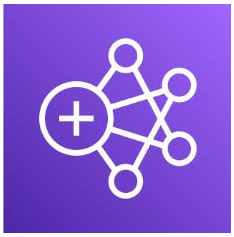
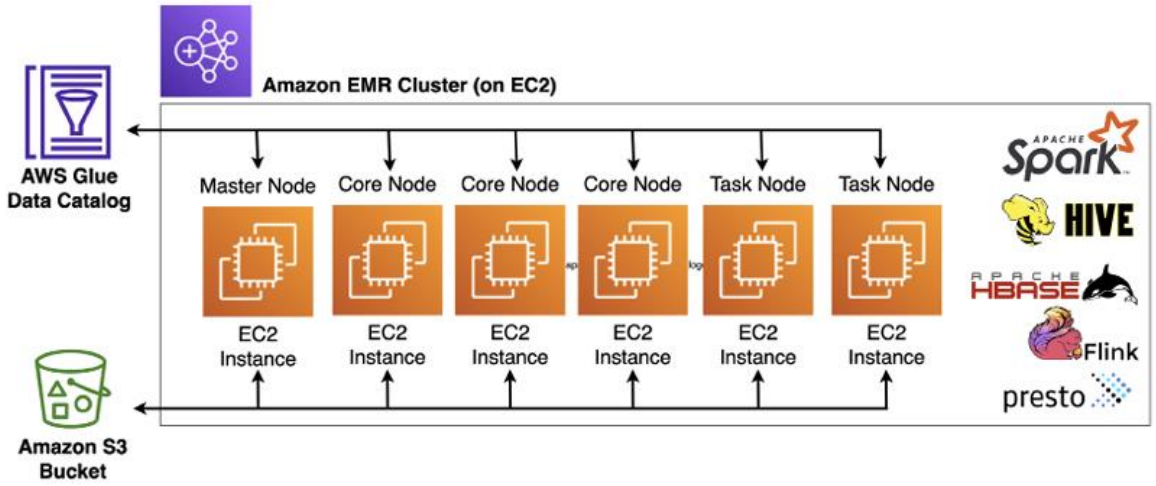
Elastic Map Reduce



Amazon EMR es una plataforma administrada para ejecutar los servicios del ecosistema Hadoop como *Apache Spark*, *Apache Hive*, *Apache Hudi*, *Apache HBase*, *Presto*, *Pig* y otras. *Amazon EMR* se encarga de las complejidades de implementar estas herramientas y administrar los recursos informáticos en clúster subyacentes.

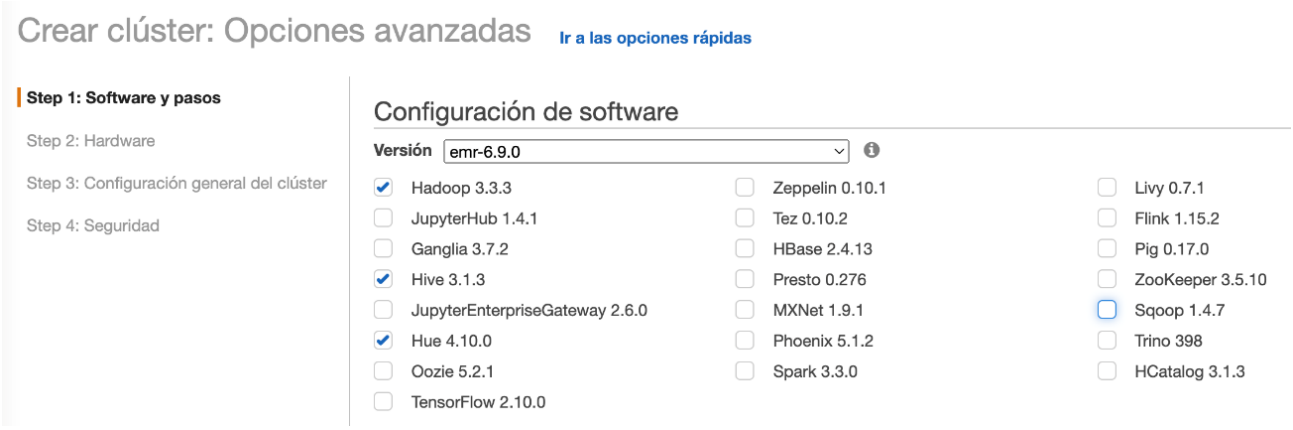
Así pues, EMR permite crear clústeres para realizar analíticas sobre datos y cargas de BI, así como transformar y mover grandes volúmenes de datos, tanto cargando como almacenando datos en servicios de AWS como *S3* y *DynamoDB*.

Al igual que muchos otros servicios de AWS, Amazon *EMR* se puede ejecutar en modo aprovisionado (indicando los tipos de instancias a emplear) o en modo *serverless*.



Clúster EMR

Para ello, utiliza una distribución propia de AWS que permite seleccionar los componentes que van a lanzarse en el clúster (Hive, Spark, Presto, etc...)



Servicios Hadoop en EMR

Ofrece elasticidad sobre el clúster, pudiendo modificar dinámicamente el dimensionamiento del mismo según necesidades, tanto escalándolo hacia arriba como hacia abajo.

Respecto al hardware, se ejecuta sobre máquinas EC2 (IaaS), las cuales configuraremos según necesidades. **Utiliza HDFS y S3 para el almacenamiento, de manera que podemos guardar los datos de entrada y los de salida en S3**, mientras que los resultados intermedios los almacenamos en HDFS.

Cada clúster de EMR requiere de un nodo principal (*master*) y al menos otro nodo *core* (un nodo *worker* que incluye almacenamiento local), y opcionalmente un número de nodos tarea (nodos *worker* que no incluyen almacenamiento local).

Los clústeres de EMR se componen de:

- un nodo **principal central**, encargado de gestionar el clúster y ejecutar los servicios de coordinación de datos.
- varios nodos **principales**, los cuales ejecutan las tareas y almacenan los datos en el clúster HDFS.
- nodos **tareas**, los cuales son opcionales, y no almacenan datos, y podemos añadir a un clúster para incrementar la capacidad de procesamiento (y eliminarlos una vez no los necesitemos para reducir costes).

A nivel de servicios, podemos definir su arquitectura en cuatro capas:

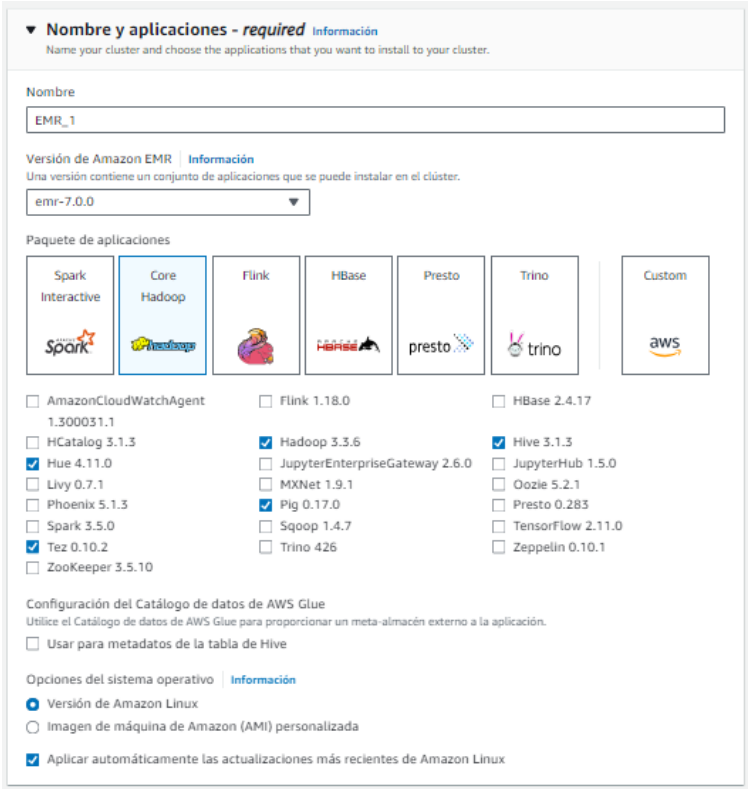
- Almacenamiento: mediante *HFDS*, *EMR FS* o el sistema de archivos local (almacenamiento de las instancias EC2).
- Gestor de recursos del clúster: *YARN*, *Tez*.
- *Frameworks* de procesamiento de datos: *Hadoop MapReduce* y *Apache Spark*
- Aplicaciones: *Apache Spark*, *Apache Hive*, etc...

Lanzando EMR

Antes de lanzar EMR, necesitamos crear un repositorio de información donde guardar los datos de entrada y salida, así como los logs que EMR genere. Por tanto, previamente hemos de crear un *bucket* S3 para ello.

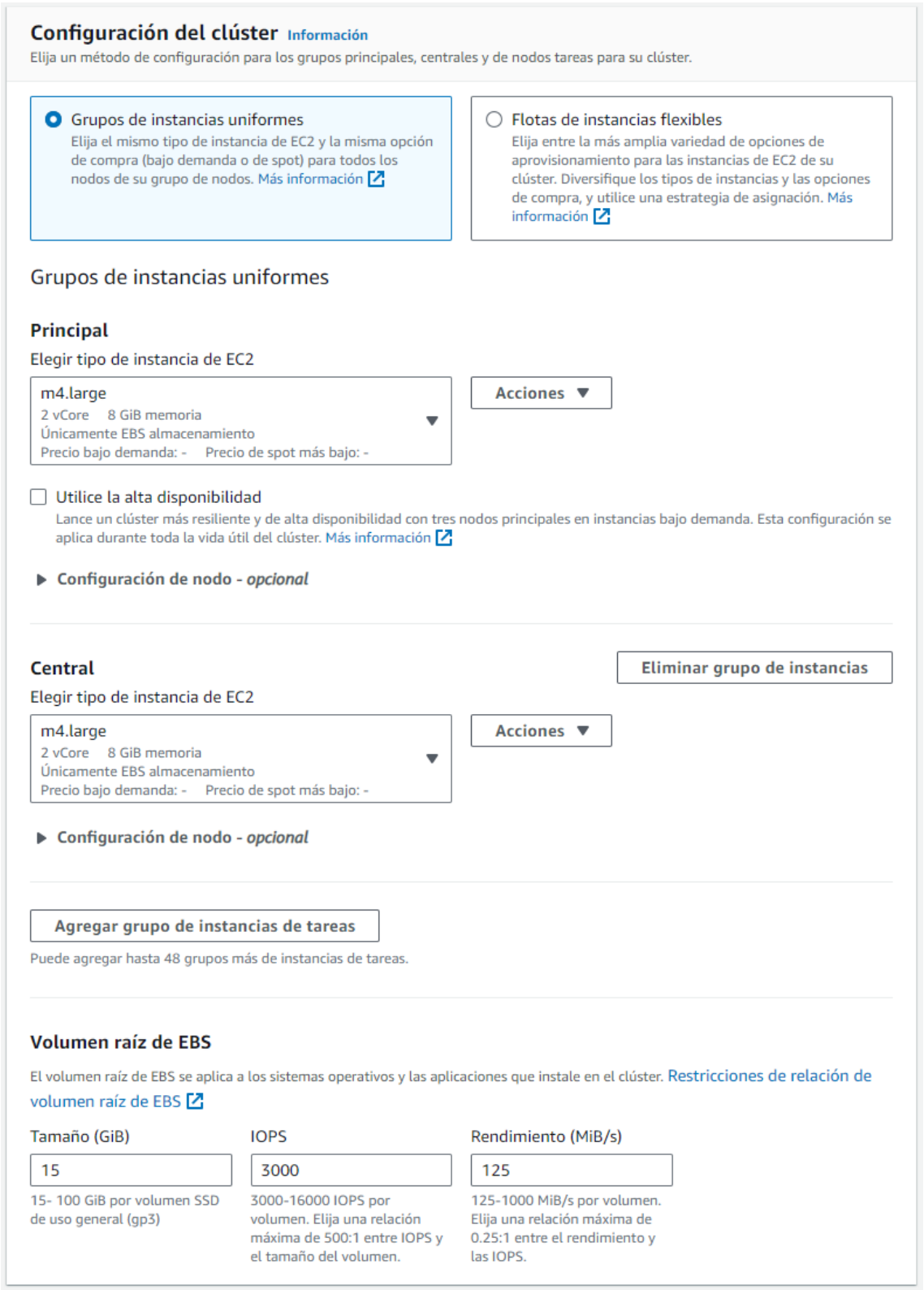
Podemos lanzar un clúster de EMR de tres formas, mediante la consola, el CLI o con un API. Vamos a centrar en el uso de la consola.

Accederemos al servicio de EMR, y creamos un clúster al cual hemos de ponerle un nombre, seleccionamos la última versión de EMR (a día de hoy es la 7.8) y seleccionamos algunos de los paquetes ya definidos, o seleccionamos las aplicaciones que deseemos (en este ejemplo seleccionaremos el conjunto predefinido etiquetado como *Core Hadoop*):



Creando un clúster en EMR

A continuación, vamos a seleccionar los tipos de instancias de nuestro clúster. Para ahorrar costes, únicamente elegiremos un nodo principal y otro central (los nodos tareas como son opcionales, para este ejemplo, vamos a descartarlos). En cuanto al tipo de instancia, escogeremos las más económicas que nos permite *AWS Academy*, actualmente las instancias **m4.large (2 CPU y 8 GiB RAM)**. Finalmente, escogemos el tamaño del almacenamiento del volumen EBS para las instancias EC2.



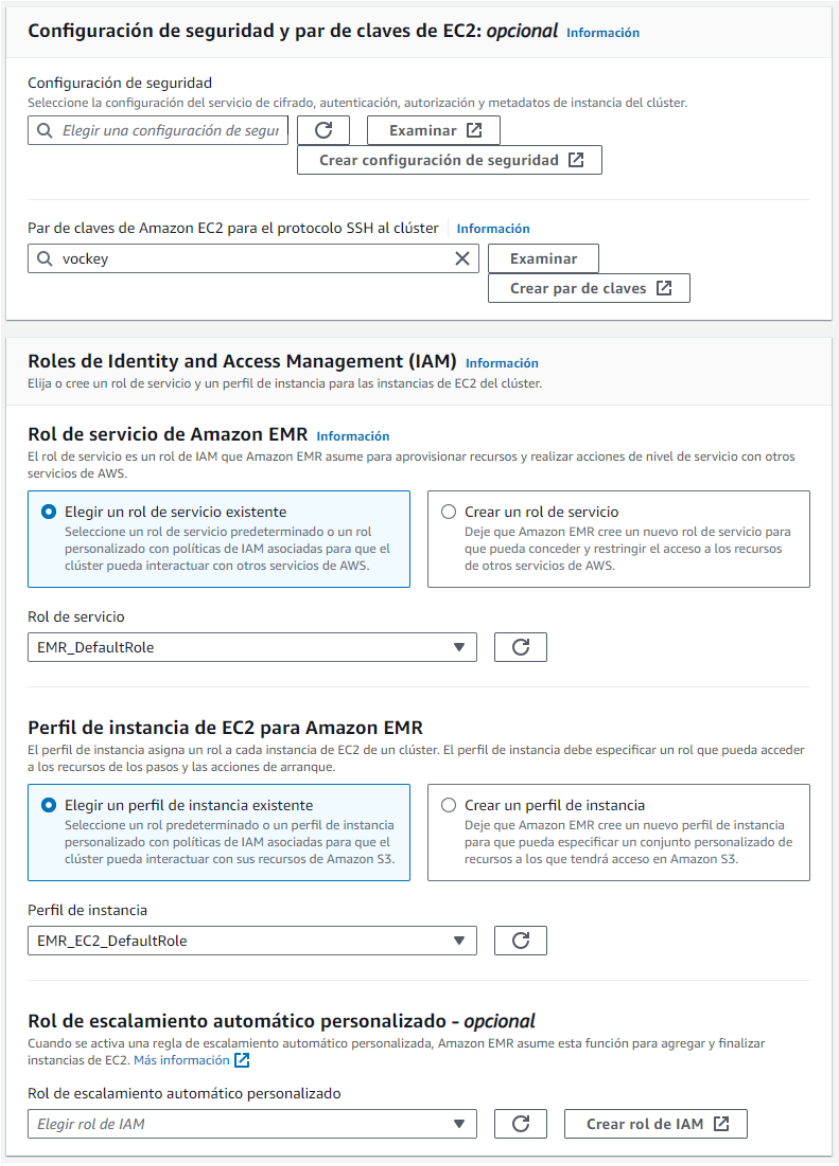
Eligiendo instancias y almacenamiento

Tras la configuración de las instancias, podemos definir el tamaño del clúster. Lo normal es que los clúster de EMR se compongan de un gran número de instancias (si no, más que Big Data, sería *Small Data* y probablemente habrá mejores soluciones que EMR). En nuestro caso, para probar el servicio, con

una única instancia nos es suficiente. Otras opciones es que el redimensionado del clúster los gestione AWS mediante métricas o programar nosotros el escalado mediante políticas personalizadas:

Eligiendo el tamaño del clúster

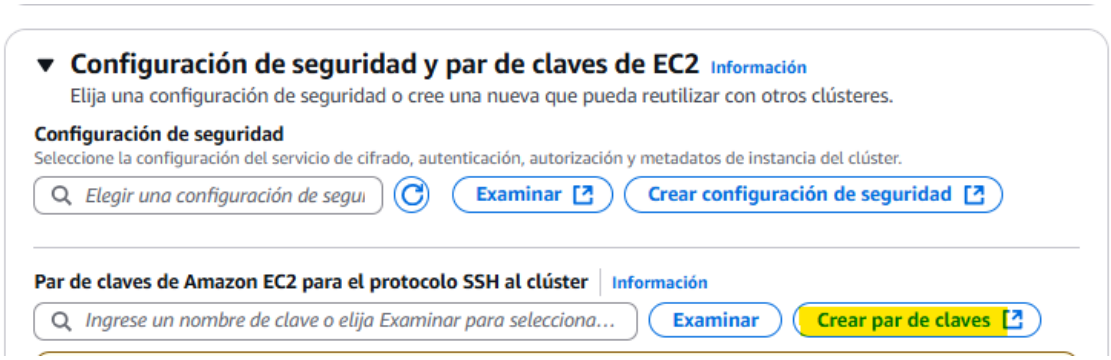
De los pasos siguientes, podemos dejar los valores por defecto y bajamos hasta los ajustes de seguridad, donde elegimos el par de claves (si ya tuviéramos unas creadas, podrían ser esas) y luego seleccionamos los roles IAM ***EMR_DefaultRole*** en el apartado *de Rol de servicio* y ***EMR_EC2_ DefaultRole*** en el de *Perfil de Instancia* respectivamente:



Configurando la seguridad del clúster

En el lado derecho podremos ver un resumen de las opciones seleccionadas, y tras darle a crear, a los 10 minutos aproximadamente, nuestro clúster estará listo para trabajar con él.

No se olvidar de asignar al clúster las claves de acceso, ya que sino no nos podríamos conectar a él.



Modos de lanzamiento

Normalmente, cuando utilizamos un clúster para procesar analíticas, interactuar con aplicaciones de *big data* o procesamiento de *datasets* de forma periódica, el clúster está siempre corriendo, a no ser que lo detengamos nosotros de forma explícita. Pero si queremos que sólo exista durante la ejecución de uno o más trabajos, el cual se le conoce como clúster *transient* o de ejecución por pasos, al terminar de ejecutar los pasos indicados, el clúster se detendrá.

Preparando al clúster

En un clúster EMR, el nodo maestro es una instancia EC2 que coordina al resto de instancias EC2 que corren los nodos principales y de tareas. Este nodo expone un DNS público el cual podemos utilizar para conectarnos.

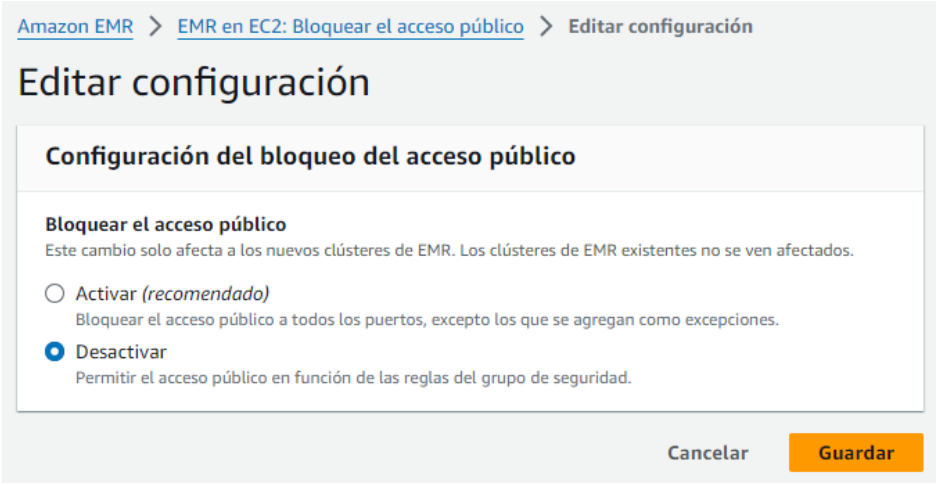
Por defecto, EMR bloquea el arranque de los clústeres que permitan las conexiones del exterior (esto nos puede pasar si al crear el clúster le asignamos un grupo de seguridad que ya teníamos creado y que permitía el tráfico entrante).

Desbloqueando el acceso público

Para permitir el acceso desde el exterior, podemos arrancar un clúster con el tráfico cerrado, y una vez ya ha arrancado desactivar la opción de *Bloquear el acceso público* de EMR:

Configurando el bloqueo del acceso público

Si editamos las opciones, podemos desbloquear el acceso público:

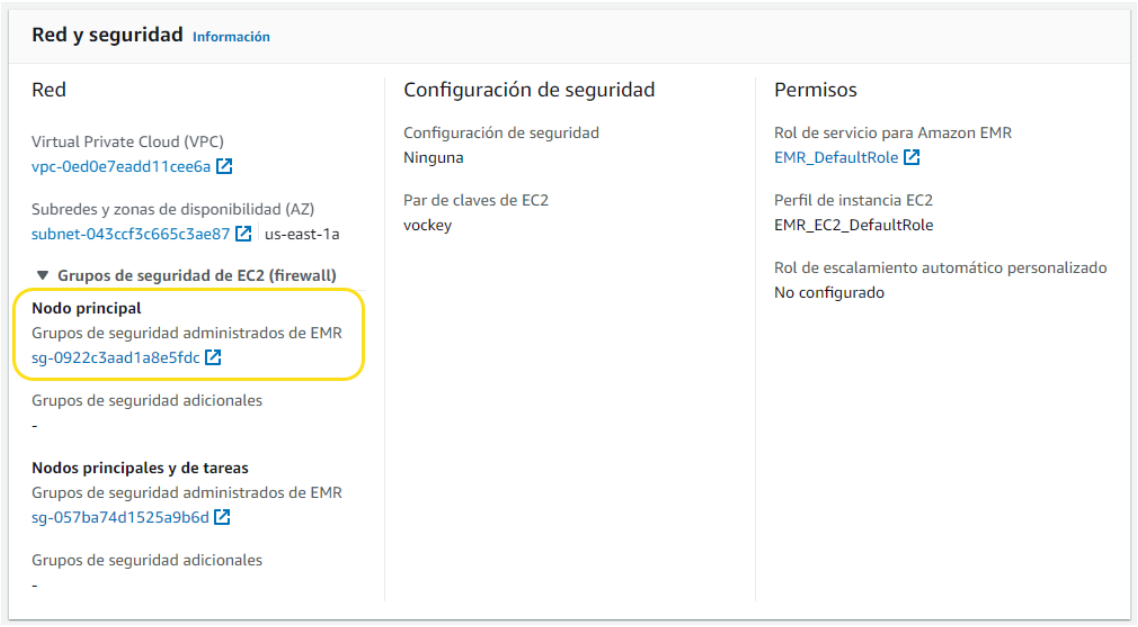


Desactivando el bloqueo del acceso público

Editando el grupo de seguridad

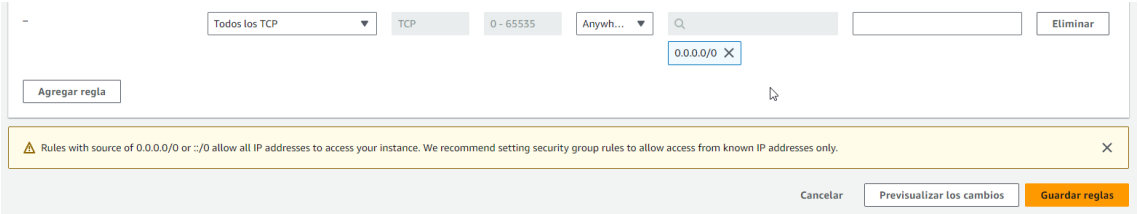
Por defecto, EMR crea un grupo de seguridad para el nodo maestro el cual determina el acceso. De inicio, ese grupo de seguridad no permite las conexiones SSH. Por ello, antes de poder conectarnos al clúster, también necesitamos modificar el grupo de seguridad del nodo principal para permitir todo el tráfico TCP (el tráfico SSH ya está abierto por defecto).

Para ello, en la pantalla de información del clúster que acabamos de crear, en la pestaña de *Propiedades*, en *Red y Seguridad*, editaremos el grupo de seguridad del nodo principal:



Seleccionando el grupo de seguridad

Tras *Editar reglas de entrada*, permitimos todo el tráfico entrante (cuidado que esta es una práctica de seguridad muy mala, sólo lo hacemos así por comodidad):

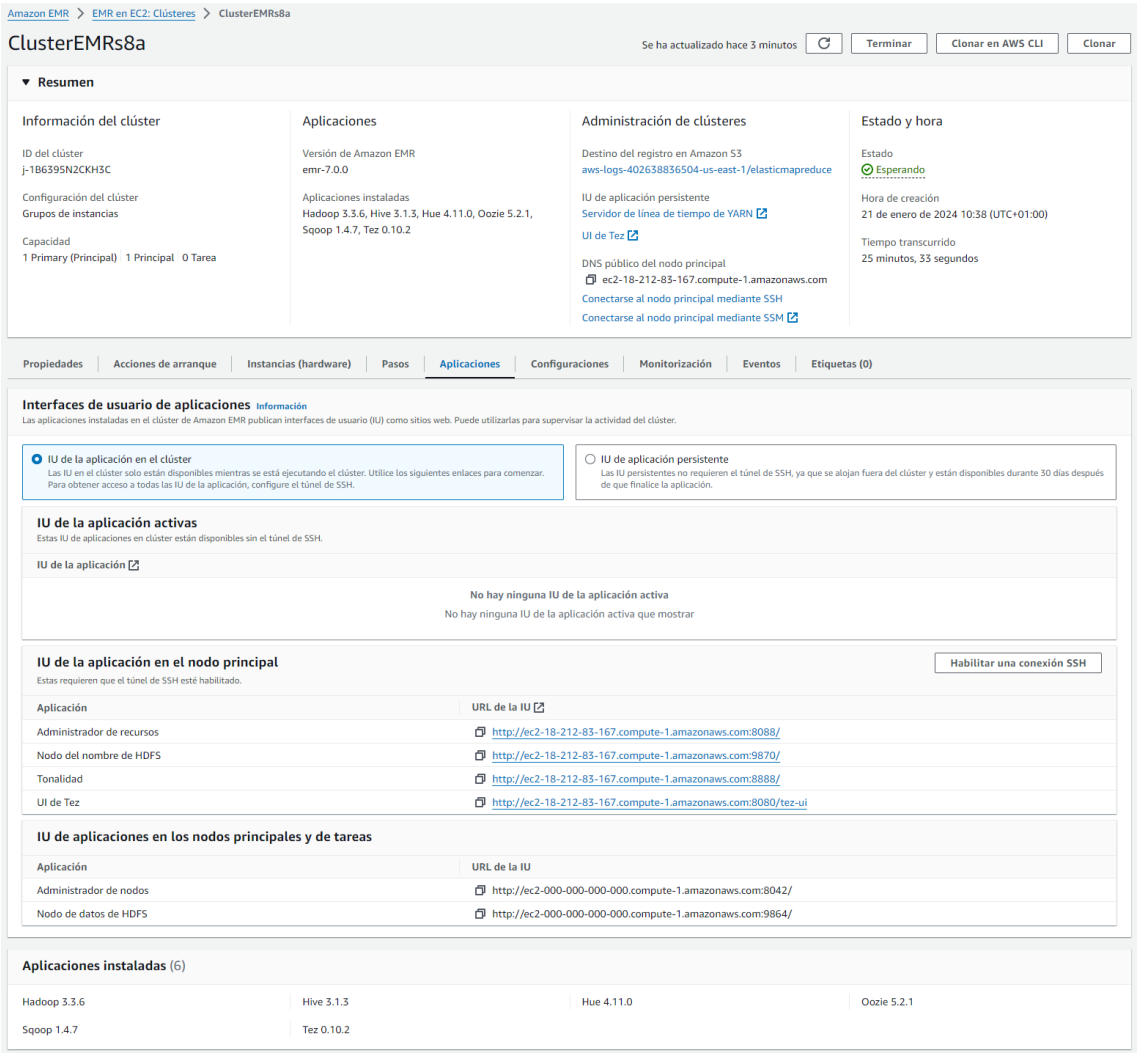


Permitiendo todas las conexiones de entrada

Conectándonos al clúster

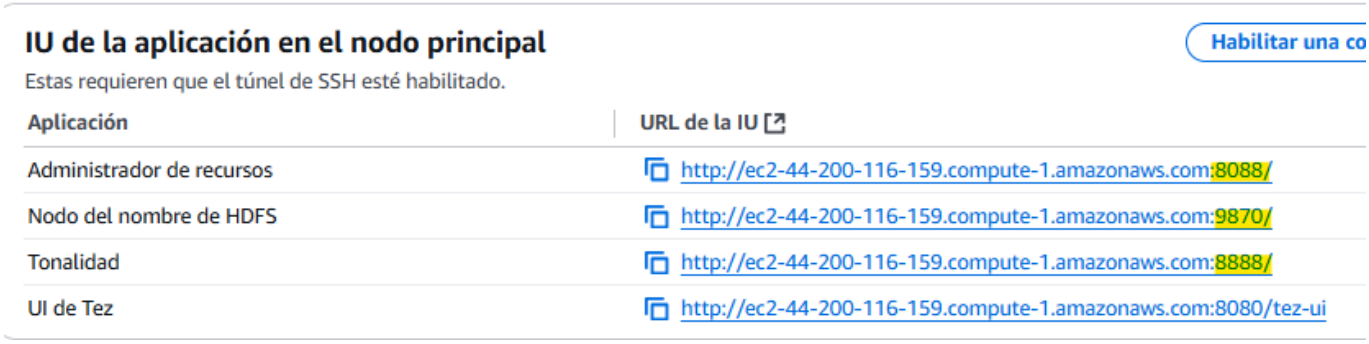
Una vez arrancado, nos podemos conectar al nodo principal a algunas de las aplicaciones configuradas, por ejemplo, a:

- El *namenode* de HDFS
- *Hue* (mal traducido en la interfaz como *Tonalidad*)
- El interfaz de *Tez*
- O la de *YARN*



Aplicaciones accesibles

Los puertos de las aplicaciones hay que habilitarlos todos en el grupo de seguridad del nodo principal, ya que por defecto no están habilitados:



Parece que funciona, pero...

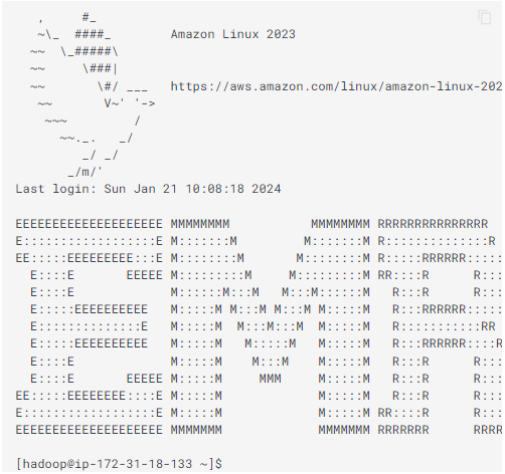
Una vez arrancado, podríamos tener un par de propiedades que modificar, ya que el acceso web a HDFS podría estar bloqueado, y la ruta de HDFS en *Hue* estar mal configurada.

Mediante SSH

Así pues, vamos a conectarnos vía SSH para activar el navegador web de HDFS. Con nuestras claves (nuestro archivo ***.pem***) descargadas, nos conectamos al clúster. El comando sería algo similar a:

```
ssh -i misclaves.pem hadoop@ec2-18-212-83-167.compute-1.amazonaws.com
```

Al conectarnos, tanto EC2 como EMR nos darán la bienvenida:



Visualizando HDFS

Si no pudiéramos visualizar el sistema de archivos de HDFS, primero le cambiamos los permisos al archivo `hdfs-site.xml` para poder editarlo sin problemas:

```
sudo chmod 777 /usr/lib/hadoop/etc/hadoop/hdfs-site.xml
```

```
nano /usr/lib/hadoop/etc/hadoop/hdfs-site.xml
```

Editamos la propiedad `dfs.webhdfs.enabled` y la ponemos a `true` (podemos buscarla mediante CTRL + W, editarla, y luego guardar y salir mediante CTRL + X):

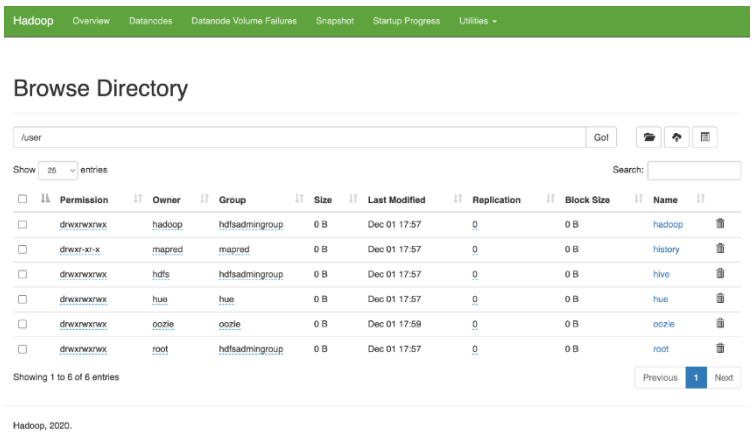
```
<property>
    <name>dfs.webhdfs.enabled</name>
    <value>false</value>
</property>
```

Sólo nos queda reiniciar el servicio de HDFS:

```
sudo systemctl restart hadoop-hdfs-namenode
```

Posteriormente agrega al grupo de seguridad del nodo principal una nueva regla que nos permita acceso a ese puerto (9870)

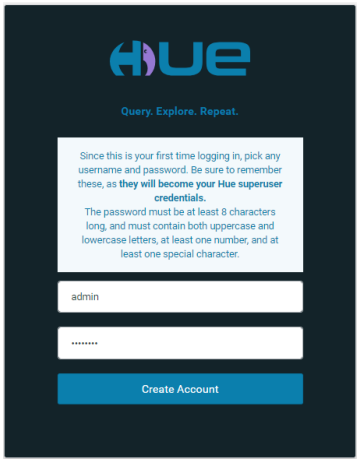
Y ahora desde el interfaz de HDFS ya podemos navegar por las carpetas y ver el contenido, por ejemplo, accediendo a *http://ec2-IP-Pública.compute-1.amazonaws.com:9870*:



Interfaz de HDFS

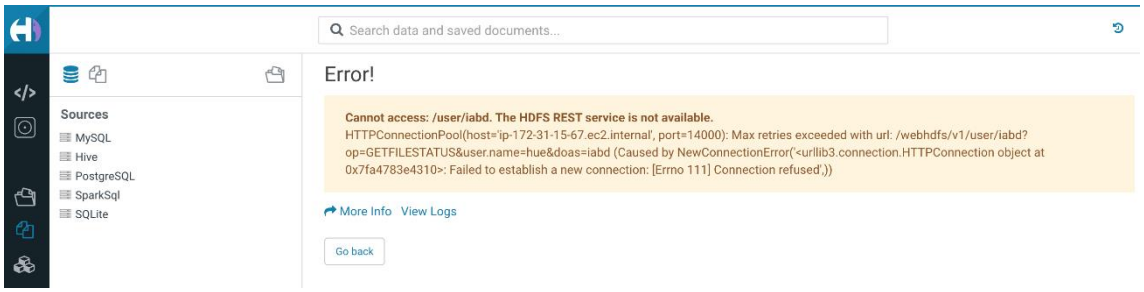
Hue y HDFS

Cuando arranca Hue, la primera vez nos pide crear un usuario y contraseña.



Login de acceso a Hue

Si intentamos visualizar los archivos y nos diera un error, de igual modo, pudiera ser que *Hue* no tiene bien configurado el acceso a HDFS:



Error al mostrar los archivos en Hue

Para ello, igual que hemos realizado antes, cambiamos los permisos y editamos el fichero de configuración:

```
sudo chmod 777 /usr/lib/hue/desktop/conf/hue.ini
```

```
nano /usr/lib/hue/desktop/conf/hue.ini
```

Y configuramos bien el puerto de acceso (debe ser 9870), dentro del grupo **[[hdfs_clusters]]** de **[hadoop]** editamos la propiedad **webhdfs_url** (sólo debes cambiar el puerto a **9870**):

```
webhdfs_url = http://ip-172-31-18-133.ec2.internal:9870/webhdfs/v1
```

Y reiniciamos el servicio:

```
sudo systemctl restart hue
```

Sólo nos queda crear un carpeta para nuestro usuario de HUE en HDFS:

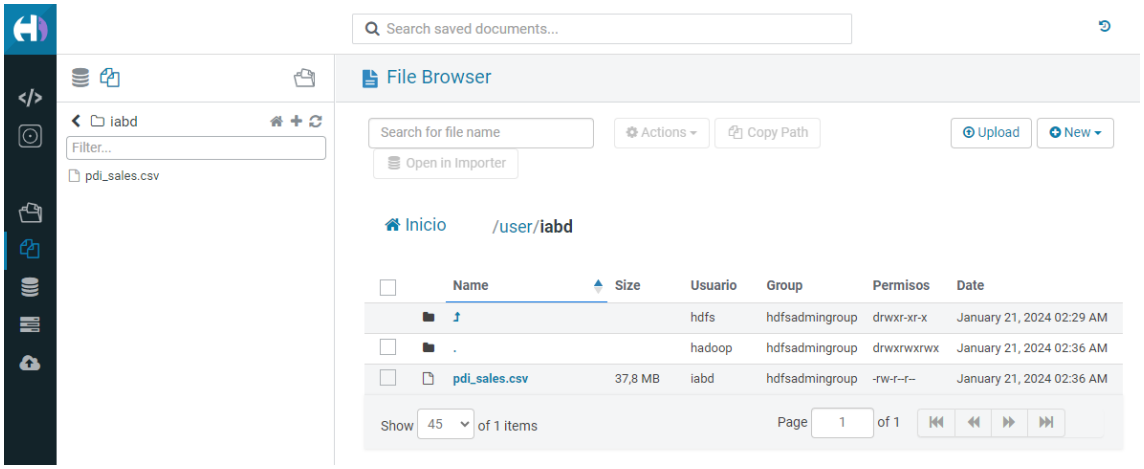
```
hdfs dfs -mkdir /user/miusuario
```

```
hdfs dfs -chmod 777 /user/miusuario
```

Por ejemplo, podemos subir un archivo cualquiera, ya sea mediante el interfaz gráfico o comandos y copiarlo mediante el comando *hdfs* adecuado:

```
hdfs dfs -put archive.csv /user/miusuario
```

Y a continuación visualizarlo desde *Hue*:



Visualizando HDFS desde Hue

Escalando

Podemos ajustar el número de instancias EC2 con las que trabaja nuestro clúster de EMR, ya sea manualmente o de forma automática en respuesta a la demanda que reciba.

Para ello, en la pestaña de *Instancias (hardware)*, tenemos la opción de *Editar opción de escalado de clústeres*:

Escalando un clúster EMR

Para ello, podemos activar el escalado gestionado mediante EMR o crear una política de escalado a medida. Independiente del modo, hemos de considerar que siempre hemos de tener de uno a tres nodos maestros, y que una vez creado el clúster, este número no lo podemos cambiar. Lo que sí que podemos es añadir y eliminar nodos principales o de tareas.

Añadiendo un nodo de tipo tarea

Conviene destacar que no podemos reconfigurar y redimensionar el clúster al mismo tiempo, de manera que hasta que no acabe la reconfiguración de un grupo de instancias no se puede iniciar el redimensionado.

Costes

Es muy importante ser conscientes de los costes que lleva utilizar EMR. *Grosso modo*, EMR supone un 25% de sobrecoste a las instancias EC2, es decir, pagaremos el coste del alquiler de las máquinas EC2 más un sobre un incremento del 25%.

Por ejemplo, para 20 nodos con 122 Gb RAM y 16 CPU, pagaríamos unos 32 €/h. En cambio, si sólo usamos las instancias que nos permite *AWS Academy* para practicar, pagaremos 0,13$/hora por instancia de m4.large (2 CPU y 8GB RAM).