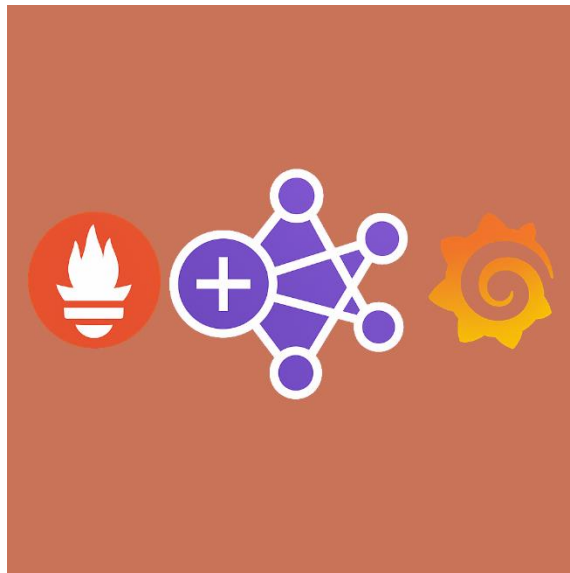


Tarea Evaluable 3ª Evaluación BDA - Monitoreo de Clúster AWS EMR con Prometheus y Grafana



Pablo Santisteban Fernández

Fecha del día de entrega: 16-04-2025

I.E.S Ataúlfo Argenta

Índice

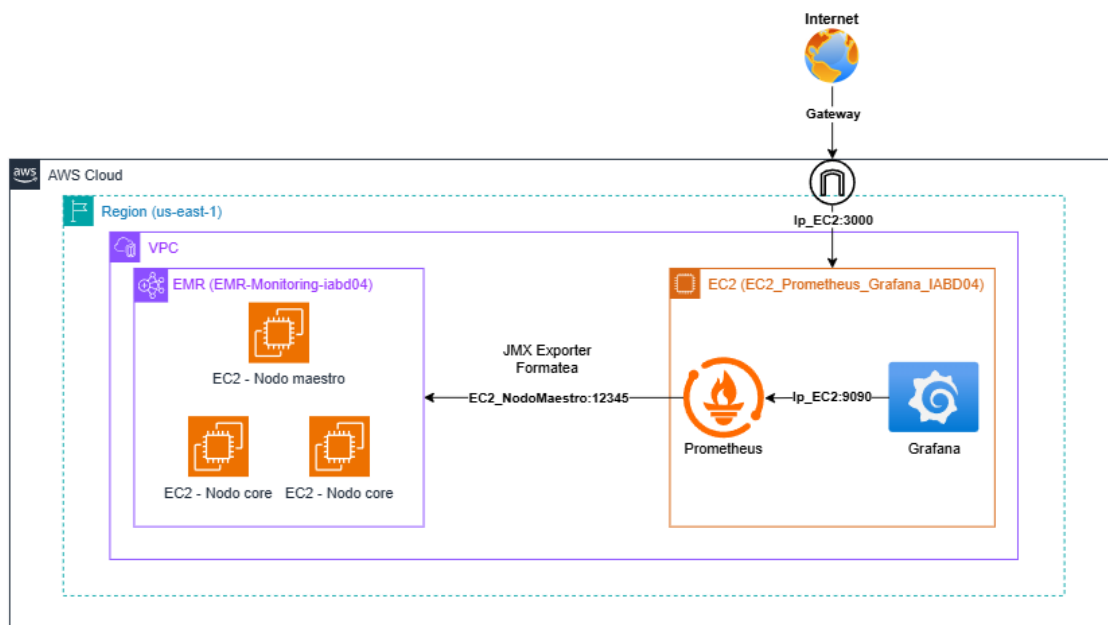
1. Resumen	2
2. Creación y Configuración del Clúster EMR	3
2.1 Conexión al nodo maestro	9
3. Instalación y Configuración de JMX Exporter	12
3.1. Creación del archivo de configuración	12
3.2. Configuración del NameNode para usar JMX Exporter.....	13
4. Despliegue de Prometheus y Grafana	15
4.1. Creación de una instancia EC2 para Prometheus y Grafana	15
4.2. Instalación de Prometheus en EC2	17
4.3. Configuración de Prometheus	18
4.4. Instalación de Grafana en EC2	19
4.5. Configuración de Grafana	20
5. Visualización de métricas en Grafana	23
6. Preguntas de reflexión sobre la práctica	26
7. Extracción local de archivos de configuración del clúster EMR y EC2.....	27
7.1. Extracción archivos del clúster EMR (nodo maestro)	28
7.2. Extracción archivos de la EC2 con Prometheus y Grafana	28

1. Resumen

Este informe documenta la configuración de un clúster en AWS EMR y la implementación de un sistema de monitoreo externo basado en Prometheus y Grafana. El objetivo principal es exponer, recopilar y visualizar métricas clave del clúster, permitiendo un seguimiento detallado de su estado y rendimiento. La infraestructura se compone de un nodo maestro y dos nodos core, sobre los cuales se instalan servicios como Hadoop, Spark y Hive para soportar el procesamiento distribuido de datos.

Para ello, se lleva a cabo la configuración de JMX Exporter en el NameNode, permitiendo la exposición de métricas que luego serán recolectadas por Prometheus desde una instancia EC2 independiente (en la misma VPC). Posteriormente, se implementa Grafana como herramienta de visualización, creando dashboards personalizados con métricas clave como uso de CPU, memoria RAM, espacio en HDFS y estado del NameNode. El informe finaliza con un análisis reflexivo sobre las métricas más relevantes, posibles mejoras en la configuración, las ventajas del uso de herramientas de visualización y monitoreo en entornos de Big Data, así como la extracción de varios archivos de configuración a la máquina local para su reutilización futura.

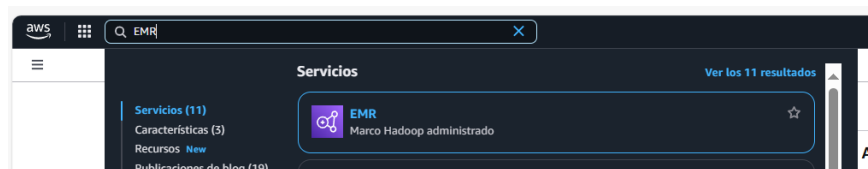
A continuación, se presenta un esquema inspirado en la arquitectura de AWS que permite visualizar de forma clara la estructura y el flujo general de esta práctica:



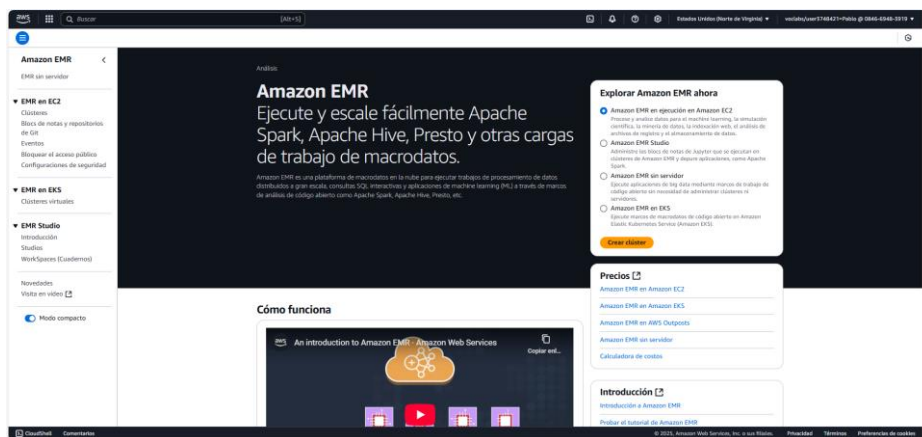
2. Creación y Configuración del Clúster EMR

Para iniciar la implementación del sistema de monitoreo, se desplegará un clúster en AWS EMR con una arquitectura compuesta por un nodo maestro y dos nodos core, diseñada para soportar el procesamiento distribuido de datos de manera eficiente. Este clúster se configurará con las herramientas fundamentales del ecosistema Big Data, incluyendo Hadoop, Spark y Hive. El acceso al nodo maestro se realizará mediante SSH, utilizando el par de claves predeterminado del laboratorio. Esta configuración será la base para exponer métricas mediante JMX Exporter, que permitirá su posterior recopilación y visualización el cual es el objetivo de esta práctica.

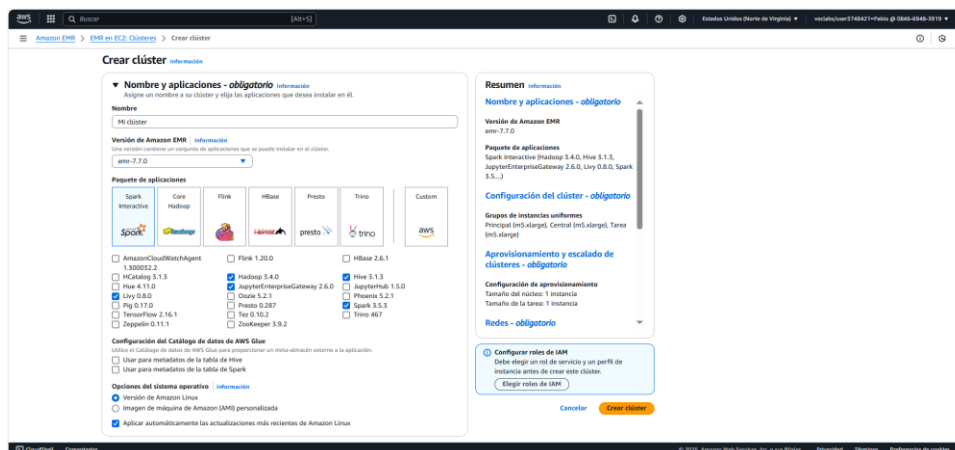
Lo primero de todo será crear el Clúster EMR. Para ello buscamos en el buscador de arriba a la izquierda **EMR** y clicamos en el:



Tras clicar, aparecerá una página parecida a la de la siguiente foto:



Aquí clicaremos en “*Crear clúster*” y ya nos meteremos en las especificaciones de este:



Dentro de esta pantalla, deberemos especificar las características que queremos que tenga nuestro clúster EMR. Lo primero de todo es ponerle un **nombre** (el que quieras, en mi caso EMR-Monitoring-iabd04) y escoger la **versión** (**emr-6.6.0**):

Crear clúster [Información](#)

▼ **Nombre y aplicaciones - obligatorio** [Información](#)
Asigne un nombre a su clúster y elija las aplicaciones que desea instalar en él.

Nombre
EMR-Monitoring-iabd04

Versión de Amazon EMR [Información](#)
Una versión contiene un conjunto de aplicaciones que se puede instalar en el clúster.
emr-6.6.0

Continuaremos escogiendo las apps que queremos instaladas en nuestro clúster, en este caso serán necesarias **Hadoop 3.2.1**, **Spark 3.2.0** y **Hive 3.1.2** para poder acceder a ese **sistema HDFS**, a esa **gestión de recursos YARN habilitados** y a la **obtención posterior de las métricas** que buscamos:

Paquete de aplicaciones

Spark	Core Hadoop	HBase	Presto	Trino	Custom

☐ Flink 1.14.2
☐ HCatalog 3.1.2
☐ Hue 4.10.0
☐ Livy 0.7.1
☐ Phoenix 5.1.2
☒ Spark 3.2.0
☐ Tez 0.9.2
☐ ZooKeeper 3.5.7

☐ Ganglia 3.7.2
☒ Hadoop 3.2.1
☐ JupyterEnterpriseGateway 2.1.0
☐ MXNet 1.8.0
☐ Pig 0.17.0
☐ Sqoop 1.4.7
☐ Trino 367

☐ HBase 2.4.4
☒ Hive 3.1.2
☐ JupyterHub 1.4.1
☐ Oozie 5.2.1
☐ Presto 0.267
☐ TensorFlow 2.4.1
☐ Zeppelin 0.10.0

Configuración del Catálogo de datos de AWS Glue
Utilice el Catálogo de datos de AWS Glue para proporcionar un meta-almacén externo a la aplicación.

☐ Usar para metadatos de la tabla de Hive
☐ Usar para metadatos de la tabla de Spark

Opciones del sistema operativo [Información](#)

☒ Versión de Amazon Linux
☐ Imagen de máquina de Amazon (AMI) personalizada
☒ Aplicar automáticamente las actualizaciones más recientes de Amazon Linux

Proseguiremos con el apartado “**Configuración del clúster - obligatorio**”, aquí es donde definiremos el **nodo maestro** y sus **2 nodos core** (workers). Primero de todo borraremos la instancia de “**Tarea 1 de 1**” dándole a “**Eliminar grupo de instancias**”:

Tarea 1 de 1 [Eliminar grupo de instancias](#)

Nombre
Tarea - 1

Elegir tipo de instancia de EC2

m5.xlarge
4 vCore 16 GiB memoria
Únicamente EBS almacenamiento
Precio bajo demanda: 0.192 USD por in...
Precio de spot más bajo: 0.059 USD (us-ea...)

[Acciones ▼](#)

► **Configuración de nodo - opcional**

[Agregar grupo de instancias de tareas](#)

Puede agregar hasta 47 grupos más de instancias de tareas.

Ahora, estableceremos el tipo de instancia EC2 a **m4.large** en nuestras instancias:

▼ Configuración del clúster - obligatorio

Información

Elija un método de configuración para los grupos principales, centrales y de nodos tareas para su clúster.

☒ Grupos de instancias uniformes

Elija el mismo tipo de instancia de EC2 y la misma opción de compra (bajo demanda o de spot) para todos los nodos de su grupo de nodos. [Más información](#)

☐ Flotas de instancias flexibles

Elija entre la más amplia variedad de opciones de aprovisionamiento para las instancias de EC2 de su clúster. Diversifique los tipos de instancias y las opciones de compra, y utilice una estrategia de asignación. [Más información](#)

Grupos de instancias uniformes

Principal

Elegir tipo de instancia de EC2

m4.large

2 vCore 8 GiB memoria

Únicamente EBS almacenamiento

Precio bajo demanda: -

Precio de spot más bajo: -

Acciones ▼

☐ Utilice la alta disponibilidad

Lance un clúster más resiliente y de alta disponibilidad con tres nodos principales en instancias bajo demanda. Esta configuración se aplica durante toda la vida útil del clúster. [Más información](#)

► Configuración de nodo - opcional

Central

Elegir tipo de instancia de EC2

m4.large

2 vCore 8 GiB memoria

Únicamente EBS almacenamiento

Precio bajo demanda: -

Precio de spot más bajo: -

Acciones ▼

Eliminar grupo de instancias

► Configuración de nodo - opcional

Pasaremos al siguiente apartado “Aprovisionamiento y escalado de clústeres - obligatorio”, aquí es donde estableceremos el **número de nodos core/núcleo** que queremos, en este caso 2. Para ello estableceremos en la tabla de abajo del apartado, el “*Tamaño de instancia(s)*” en 2:

▼ Aprovisionamiento y escalado de clústeres - obligatorio

Información

Elija cómo Amazon EMR debe dimensionar su clúster.

Elija una opción

☒ Establecer el tamaño del clúster manualmente

Utilice esta opción si conoce los patrones de la carga de trabajo de antemano.

☐ Utilizar escalado administrado por EMR

Supervise las métricas clave de la carga de trabajo de modo que EMR pueda optimizar el tamaño del clúster y la utilización de los recursos.

☐ Utilizar el escalamiento automático personalizado

Para escalar mediante programación los nodos principales y los nodos de tarea, cree políticas de escalamiento automático personalizadas.

Configuración de aprovisionamiento

Establezca el tamaño del principal grupo de instancias. Amazon EMR intenta aprovisionar esta capacidad al lanzar el clúster.

Nombre	Tipo de instancia	Tamaño de instancia(s)	Utilizar la opción de compra de spot
Central	m4.large	2	<input type="checkbox"/>

5

Seguiremos con el apartado “*Redes - obligatorio*”, aquí es donde estableceremos la **VPC** (Virtual Private Cloud) y la subred en la que se va a alojar el EMR. La VPC escogeremos la única que hay, que es la creada por defecto:



Y la **subred** la dejaremos por defecto, quedando así todo este apartado:

▼ **Redes - obligatorio** [Información](#)
Elija la configuración de red que determina la forma en que usted y otras entidades se comunican con su clúster.

Virtual Private Cloud (VPC) [Información](#)
vpc-08aff6b2668078c1e [Examinar](#) [Crear VPC](#)

Subred [Información](#)
subnet-009b51a1b34735865 [Examinar](#) [Crear subred](#)

► **Grupos de seguridad de EC2 (firewall)**

Pasando al siguiente apartado, “*Registros de clúster*”, lo que hace es establecer el lugar (Bucket S3) donde se van a guardar los registros de este EMR que estamos creando. Se puede dejar el que viene por defecto o se puede crear un S3 para almacenar ahí los logs/registros. En este caso lo dejaremos en default, es decir, el que viene por defecto:

▼ **Registros de clúster** [Información](#)
Elija dónde y cómo almacenar los archivos de registro.

① Archivamos automáticamente los archivos de registro en Amazon S3. Puede especificar una ubicación de S3 propia o utilizar la ubicación de S3 predeterminada para Amazon EMR. La ubicación de registro predeterminada se completa previamente en el campo **Ubicación de Amazon S3**.

☒ Publicar registros específicos del clúster en Amazon S3
Ubicación de Amazon S3
s3://aws-logs-987726132951-us-east-1/elasticmapreduce [Ver](#) [Explorar S3](#)
Formato: utilizar s3://bucket/prefix

☐ Cifrar los registros específicos del clúster

En el apartado de “*Configuración de seguridad y par de claves de EC2*”, estableceremos el **par de claves** (vockey), para así posteriormente, poder conectarnos vía SSH:

▼ **Configuración de seguridad y par de claves de EC2** [Información](#)
Elija una configuración de seguridad o cree una nueva que pueda reutilizar con otros clústeres.

Configuración de seguridad
Seleccione la configuración del servicio de cifrado, autenticación, autorización y metadatos de instancia del clúster.
Elegir una configuración de seguridad [Examinar](#) [Crear configuración de seguridad](#)

Par de claves de Amazon EC2 para el protocolo SSH al clúster [Información](#)
vockey [Examinar](#) [Crear par de claves](#)

Por último, en el apartado de “*Roles de Identity and Access Management (IAM) - obligatorio*”, aquí estableceremos los Roles IAM; Rol de servicio: ***EMR_DefaultRole*** y Perfil de instancia: ***EMR_EC2_DefaultRole***

[illegible]

Los **grupos de seguridad** y el resto de apartados los dejaremos por defecto, ya que no son relevantes para esta tarea.

Así quedaría nuestro clúster EMR configurado:

Resumen Información

Nombre y aplicaciones - *obligatorio*

Nombre

EMR-Monitoring-iabd04

Versión de Amazon EMR

emr-6.6.0

Paquete de aplicaciones

Custom (Hadoop 3.2.1, Hive 3.1.2, Spark 3.2.0)

Configuración del clúster - *obligatorio*

Grupos de instancias uniformes

Principal (m4.large), Central (m4.large)

Aprovisionamiento y escalado de clústeres - *obligatorio*

Configuración de aprovisionamiento

Tamaño del núcleo: 2 instancias

Redes - obligatorio

VPC

vpc-08aff6b26...

Subred

subnet-009b51...

Grupos de seguridad de nodos principales

sg-06f6a783c6...

Grupos de seguridad de nodos básicos

sg-04921fd05b...

Terminación del clúster

Terminación del clúster

Terminar el clúster después del tiempo de inactividad

Tiempo de inactividad: 4 horas

Registros de clúster

Ubicación de Amazon S3

s3://aws-logs...

Configuración de seguridad y par de claves de EC2

Par de claves de Amazon EC2

vockey

Roles de Identity and Access Management (IAM) - obligatorio

Rol de servicio

LabRole

Perfil de instancia

EMR_EC2_DefaultRole

Cancelar

Crear clúster

Para finalizar con la configuración del clúster, clicaremos en “*Crear clúster*” y esperaremos a que se inicie por completo (tarda entre 4 a 10 minutos). Para saber si está iniciado por completo, las instancias del clúster deben estar “*En ejecución*” y en el estado del clúster debe decir “*Esperando*” o “*Waiting*”.

Pasando de esto:

El clúster "EMR-Monitoring-iabd04" se ha creado correctamente.

EMR-Monitoring-iabd04

Se ha actualizado hace menos de un minuto

Terminar

Clonar en AWS CLI

Clonar

Resumen

Información del clúster

ID del clúster

j-MQ5MNO1WRME7

ARN del clúster

arn:aws:elasticmapreduce:us-east-1:987726132951:cluster/j-MQ5MNO1WRME7

Configuración del clúster

Grupos de instancias

Capacidad

1 Primary (Principal) | 2 Principal | 0 Tarea

Aplicaciones

Versión de Amazon EMR

emr-6.6.0

Aplicaciones instaladas

Hadoop 3.2.1, Hive 3.1.2, Spark 3.2.0

Administración de clústeres

Destino del registro en Amazon S3

aws-logs-987726132951-us-east-1/elasticmapreduce

DNS público del nodo principal

-

Estado y hora

Estado

Comenzando

Hora de creación

7 de abril de 2025 16:21 (UTC+02:00)

Tiempo transcurrido

-8 segundos

Propiedades

Acciones de arranque

Instancias (hardware)

Pasos

Aplicaciones

Configuraciones

Monitorización

Eventos

Etiquetas (0)

Registros de clúster

Archivar los archivos de registro en Amazon S3

Activado

Ubicación de Amazon S3

s3://aws-logs-987726132951-us-east-1/elasticmapreduce/

Cifrado para registros

Desactivado

Terminación del clúster y reemplazo de nodos

Opción de terminación

Terminar automáticamente el clúster después del tiempo de inactividad

Protección contra la terminación

Desactivado

Tiempo de inactividad

4 horas

Reemplazo de nodos en mal estado

Activado

Red y seguridad

Red

Virtual Private Cloud (VPC)

vpc-08aff6b2668078c1e

Subredes y zonas de disponibilidad (AZ)

subnet-009b51a1b54755865 us-east-1d

Grupos de seguridad de EC2 (firewall)

Configuración de seguridad

Configuración de seguridad

Ninguna

Par de claves de EC2

vockey

Permisos

Rol de servicio para Amazon EMR

LabRole

Perfil de instancia EC2

EMR_EC2_DefaultRole

Rol de escalamiento automático personalizado

No configurado

Activar Windows

Ve a Configuración

8

A esto:

El clúster "EMR-Monitoring-iabd04" se ha creado correctamente.

EMR-Monitoring-iabd04 Se ha actualizado hace 1 minuto Terminar Clonar en AWS CLI Clonar

Resumen

Información del clúster

ID del clúster: j-MQ5MNO1WRME7

ARN del clúster: arn:aws:elasticmapreduce:us-east-1:987726132951:cluster/j-MQ5MNO1WRME7

Configuración del clúster: Grupos de instancias

Capacidad: 1 Primary (Principal) | 2 Principal | 0 Tarea

Aplicaciones

Versión de Amazon EMR: emr-6.6.0

Aplicaciones instaladas: Hadoop 3.2.1, Hive 3.1.2, Spark 3.2.0

Administración de clústeres

Destino del registro en Amazon S3: aws-logs-987726132951-us-east-1/elasticmapreduce

UI de aplicación persistente: Servidor de historial de Spark, Servidor de línea de tiempo de YARN, UI de Tez

DNS público del nodo principal: ec2-54-174-48-70.compute-1.amazonaws.com

Conectarse al nodo principal mediante SSH, Conectarse al nodo principal mediante SSM

Estado y hora

Estado: Esperando

Hora de creación: 7 de abril de 2025 16:21 (UTC+02:00)

Tiempo transcurrido: 7 minutos, 21 segundos

Propiedades Acciones de arranque Instancias (hardware) Pasos Aplicaciones Configuraciones Monitorización Eventos Etiquetas (0)

Sistema operativo Información

Versión de Amazon Linux: 2.0.20250321.0

Registros de clúster Información

Archivar los archivos de registro en Amazon S3: Activado

Ubicación de Amazon S3: s3://aws-logs-987726132951-us-east-1/elasticmapreduce/

Cifrado para registros: Desactivado

Terminación del clúster y reemplazo de nodos Información

Opción de terminación: Terminar automáticamente el clúster después del tiempo de inactividad

Protección contra la terminación: Desactivado

Tiempo de inactividad: 4 horas

Reemplazo de nodos en mal estado: Activado

Red y seguridad Información

Red

Virtual Private Cloud (VPC): vpc-08aff6b2668078c1e

Configuración de seguridad

Configuración de seguridad: Ninguna

Permisos

Rol de servicio para Amazon EMR: LabRole

2.1 Conexión al nodo maestro

El objetivo ahora es conectarse al nodo maestro de nuestro EMR. Para ello, se establecerá una conexión vía SSH con el par de claves vockey, definidas en la configuración inicial del clúster en el [apartado anterior](#).

Como primer paso, será necesario agregar una regla al grupo de seguridad del EMR. Para ello, accederemos a la configuración del EMR y haremos clic en el enlace del grupo de seguridad del nodo principal:

Red y seguridad Información

Red

Virtual Private Cloud (VPC): vpc-08aff6b2668078c1e

Subredes y zonas de disponibilidad (AZ): subnet-009b51a1b34735865 | us-east-1d

Grupos de seguridad de EC2 (firewall)

Nodo principal

Grupos de seguridad administrados de EMR: sg-06f6a783c603b8c1c

Grupos de seguridad adicionales: -

Nodos principales y de tareas

Grupos de seguridad administrados de EMR: sg-04921fd05b9a1e344

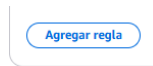
Grupos de seguridad adicionales: -

Una vez dentro, agregaremos una regla dándole a “*Editar reglas de entrada*”:

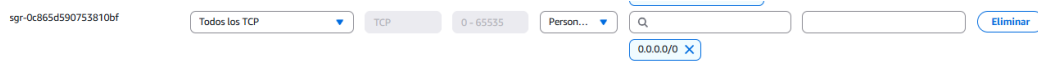
Reglas de entrada (8)

Administrar etiquetas Editar reglas de entrada

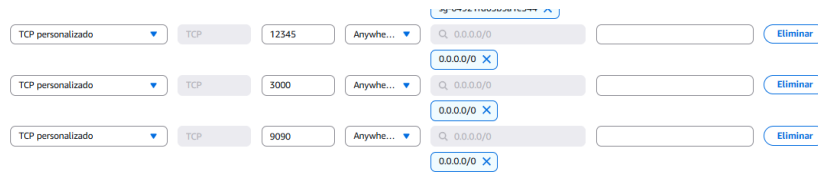
Y dentro le daremos a “*Agregar regla*” en la parte baja de la lista de reglas:



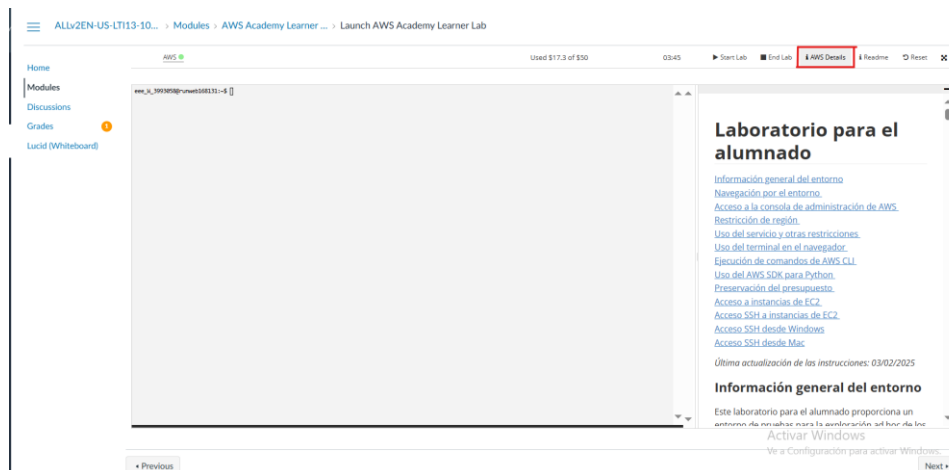
Agregaremos una regla para poder acceder desde cualquier puerto e IP:



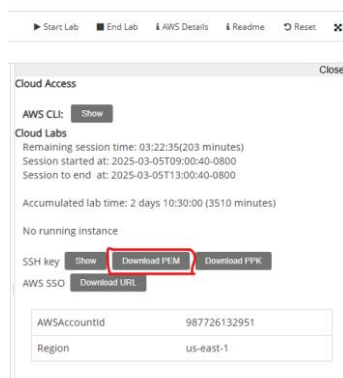
Además, agregaremos los puertos 12345 (Métricas obtenidas por el JMX), 3000 (Grafana) y 9090 (Prometheus) como medida preventiva en caso de que la regla general no sea suficiente para garantizar el acceso a los servicios de monitoreo que serán usados posteriormente en la tarea:



Una vez guardadas las reglas, procederemos a ir al laboratorio y descargaremos el par de claves necesario para acceder vía SSH:



Cuando estemos aquí, deberemos ir al apartado “*AWS Details*” ubicado en la parte superior derecha como se ve en la anterior foto. Dentro de este apartado, haremos clic en “*Download PEM*” para descargar el archivo con el par de claves **labsuser.pem**:



Una vez descargado, abriremos el cmd y nos ubicaremos en la carpeta en la que esté ese par de claves descargado (Descargas en mi caso):

```

C:\> Símbolo del sistema

Microsoft Windows [Versión 10.0.19045.5608]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Pablo>cd Downloads

C:\Users\Pablo\Downloads>dir labsuser.pem
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 946F-F649



Directorio de C:\Users\Pablo\Downloads

07/04/2025  16:28                1.674 labsuser.pem
               1 archivos                1.674 bytes
               0 dirs 300.557.463.552 bytes libres

C:\Users\Pablo\Downloads>

```

Una vez dentro, lanzaremos el comando **ssh -i labsuser.pem hadoop@DNS_nodo_master**, para conectarnos por SSH, pero nos falta el DNS del máster. Para ello, vamos de vuelta al EMR y copiamos el DNS:

DNS público del nodo principal
 ec2-54-237-230-110.compute-1.amazonaws.com
[Conectarse al nodo principal mediante SSH](#)
[Conectarse al nodo principal mediante SSM](#) 

Sustituimos el DNS en el comando **ssh -i labsuser.pem hadoop@ec2-18-233-99-255.compute-1.amazonaws.com** y lo lanzamos:

```

C:\Users\Pablo\Downloads>ssh -i labsuser.pem hadoop@ec2-18-233-99-255.compute-1.amazonaws.com
Last login: Wed Mar  5 17:30:47 2025

 _ | _ | )
 _ | (  /  Amazon Linux AMI
 _ | \ | _ |

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
57 package(s) needed for security, out of 84 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRRRRRRRRRRR
E::::::::::::::::::::E M::::::::M M::::::::M R::::::::::::R
EE::::::::EEEEEEEEEE: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 R:::R R:::R
 E:::EEEEEEEEEE M:::M M:::M M:::M R:::RRRRRR::::R
 E::::::::::::E 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 MMMMMMMM MMMMMMMM RRRRRRRR RRRRRR

[hadoop@ip-172-31-80-19 ~]$

```

3. Instalación y Configuración de JMX Exporter

Ahora vamos a instalar una herramienta dentro del nodo maestro llamada JMX Exporter, que es una herramienta que permite exponer métricas de aplicaciones Java en formato Prometheus. En el entorno EMR, se usa para extraer métricas de servicios como Hadoop (ej. el NameNode), y así poder analizarlas y visualizarlas posteriormente en Grafana.

Lo primero de todo es estar conectado vía SSH al nodo maestro como se muestra en el [apartado anterior](#). Una vez dentro nos debemos posicionar en la carpeta “*opt*” con el comando **cd /opt**:

```
[hadoop@ip-172-31-35-236 ~]$ pwd
/home/hadoop
[hadoop@ip-172-31-35-236 ~]$ cd /opt
[hadoop@ip-172-31-35-236 opt]$
```

Una vez dentro de *opt*, lanzamos el comando **sudo wget https://repo1.maven.org/maven2/io/prometheus/jmx/jmx_prometheus_javaagent/0.16.1/jmx_prometheus_javaagent-0.16.1.jar**, para instalar las extensiones de java y poder así obtener las métricas de forma adecuada posteriormente:

```
[hadoop@ip-172-31-35-236 opt]$ sudo wget https://repo1.maven.org/maven2/io/prometheus/jmx/jmx_prometheus_javaagent/0.16.1/jmx_prometheus_javaagent-0.16.1.jar
--2025-04-07 14:45:44-- https://repo1.maven.org/maven2/io/prometheus/jmx/jmx_prometheus_javaagent/0.16.1/jmx_prometheus_javaagent-0.16.1.jar
Resolving repo1.maven.org (repo1.maven.org)... 146.75.32.209, 2a04:4e42:78::209
Connecting to repo1.maven.org (repo1.maven.org)[146.75.32.209]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 469645 (459K) [application/java-archive]
Saving to: 'jmx_prometheus_javaagent-0.16.1.jar'

100%[=====] 469,645 --.-K/s in 0.005s

2025-04-07 14:45:44 (92.4 MB/s) - 'jmx_prometheus_javaagent-0.16.1.jar' saved [469645/469645]

[hadoop@ip-172-31-35-236 opt]$ ls -la
total 460
drwxr-xr-x 4 root root 70 Apr 7 14:45 .
dr-xr-xr-x 18 root root 293 Apr 7 14:27 ..
drwxr-xr-x 5 root root 47 Mar 31 2022 aws
-rw-r--r-- 1 root root 469645 Jul 13 2021 jmx_prometheus_javaagent-0.16.1.jar
drwxr-xr-x 2 root root 6 Aug 16 2018 rh
[hadoop@ip-172-31-35-236 opt]$
```

Cuando acabe la instalación, le daremos permisos al archivo generado con **sudo chmod 777 nombre_archivo**:

```
[hadoop@ip-172-31-35-236 opt]$ sudo chmod 777 jmx_prometheus_javaagent-0.16.1.jar
[hadoop@ip-172-31-35-236 opt]$
```

3.1. Creación del archivo de configuración

Seguiremos ahora con la creación del archivo de configuración “*config.yml*” que sirve para que JMX Exporter pueda exponer correctamente las métricas, es decir, este archivo define la forma en que se estructuran y filtran las métricas extraídas de las aplicaciones Java, en este caso, del servicio NameNode de Hadoop. Gracias a esta configuración, Prometheus podrá interpretar adecuadamente los datos recopilados y facilitar su visualización en Grafana.

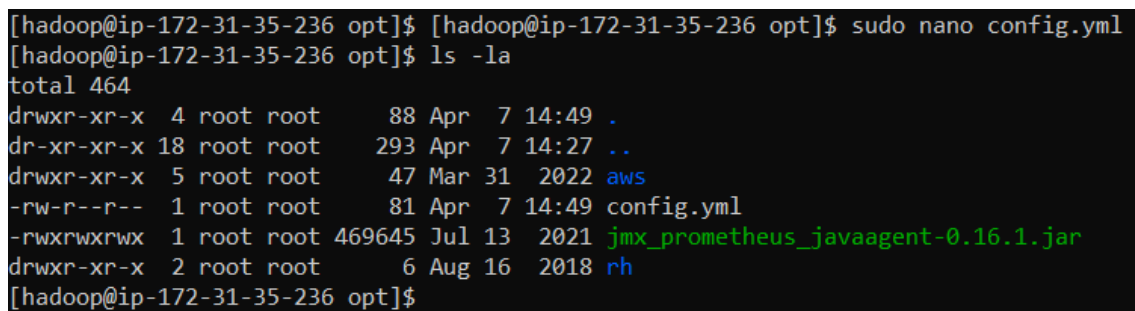
Para ello, lanzaremos el comando **sudo nano config.yml** en la carpeta “*opt*”, es decir en la que estamos y le añadiremos el siguiente contenido:

```
lowercaseOutputName: true
lowercaseOutputLabelNames: true
rules:
  - pattern: ".*"
```



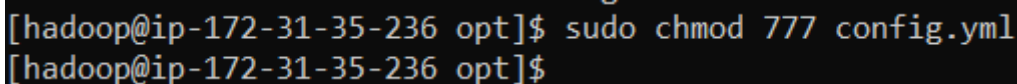
```
hadoop@ip-172-31-35-236/opt
GNU nano 2.9.8 config.yml
lowercaseOutputName: true
lowercaseOutputLabelNames: true
rules:
- pattern: ".*"
```

Comprobamos que se creó correctamente el archivo:



```
[hadoop@ip-172-31-35-236 opt]$ [hadoop@ip-172-31-35-236 opt]$ sudo nano config.yml
[hadoop@ip-172-31-35-236 opt]$ ls -la
total 464
drwxr-xr-x  4 root root   88 Apr  7 14:49 .
dr-xr-xr-x 18 root root  293 Apr  7 14:27 ..
drwxr-xr-x  5 root root   47 Mar 31  2022 aws
-rw-r--r--  1 root root   81 Apr  7 14:49 config.yml
-rwxrwxrwx  1 root root 469645 Jul 13  2021 jmx_prometheus_javaagent-0.16.1.jar
drwxr-xr-x  2 root root    6 Aug 16  2018 rh
[hadoop@ip-172-31-35-236 opt]$
```

Por último, le damos permisos con **sudo chmod 777 nombre_archivo**:



```
[hadoop@ip-172-31-35-236 opt]$ sudo chmod 777 config.yml
[hadoop@ip-172-31-35-236 opt]$
```

3.2. Configuración del NameNode para usar JMX Exporter

El objetivo en este punto es lograr que las métricas recolectadas por JMX Exporter en el nodo maestro puedan ser expuestas correctamente para su posterior consulta.

Una vez descargado el agente de JMX Exporter y creado el archivo config.yml como se ve en [apartados anteriores](#), el siguiente paso será editar el archivo de configuración del NameNode para integrar el agente como una opción de arranque. Para ello, nos dirigiremos al archivo “*hadoop-env.sh*” ubicado en “*/etc/hadoop/conf/*”, utilizando el comando **sudo nano /etc/hadoop/conf/hadoop-env.sh**. Dentro de este archivo, se deberá agregar la siguiente línea al final:

```
export HADOOP_NAMENODE_OPTS="-
javaagent:/home/hadoop/jmx_prometheus_javaagent-
0.16.1.jar=12345:/home/hadoop/config.yml
$HADOOP_NAMENODE_OPTS"
```

Esta instrucción configura el javaagent con el archivo de métricas correspondiente y el puerto 12345, lo que permitirá que el NameNode exponga sus métricas para ser recolectadas por Prometheus.

Así quedaría el archivo con esa línea agregada:

```

Seleccionar hadoop@ip-172-31-35-236/opt
GNU nano 2.9.8 /etc/hadoop/conf/hadoop-env.sh

# A string representing this instance of hadoop. $USER by default.
#export HADOOP_IDENT_STRING=

# The scheduling priority for daemon processes. See 'man nice'.
#export HADOOP_NICENESS=

# tez environment, needed to enable tez
export TEZ_CONF_DIR=/etc/tez/conf
export TEZ_JARS=/usr/lib/tez
# Add tez into HADOOP_CLASSPATH
export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:${TEZ_CONF_DIR}/*:${TEZ_JARS}/*:${TEZ_JARS}/lib/*

export HADOOP_CLASSPATH="$HADOOP_CLASSPATH:/usr/lib/hadoop-lzo/lib/*"
export JAVA_LIBRARY_PATH="$JAVA_LIBRARY_PATH:/usr/lib/hadoop-lzo/lib/native"

export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:/usr/share/aws/aws-java-sdk/*
export HADOOP_CLASSPATH="$HADOOP_CLASSPATH:/usr/share/aws/emr/emrfs/conf:/usr/share/aws/emr/emrfs/lib/*:/usr/share/aws/emr/emrfs/auxlib/*"

export HADOOP_CLASSPATH="$HADOOP_CLASSPATH:/usr/share/aws/emr/ddb/lib/emr-ddb-hadoop.jar"

export HADOOP_CLASSPATH="$HADOOP_CLASSPATH:/usr/share/aws/emr/goodies/lib/emr-hadoop-goodies.jar"

export HADOOP_CLASSPATH="$HADOOP_CLASSPATH:/usr/share/aws/emr/kinesis/lib/emr-kinesis-hadoop.jar"

# Add CloudWatch sink jar to classpath
export HADOOP_CLASSPATH="$HADOOP_CLASSPATH:/usr/share/aws/emr/cloudwatch-sink/lib/*"

# Add security artifacts to classpath
export HADOOP_CLASSPATH="$HADOOP_CLASSPATH:/usr/share/aws/emr/security/conf:/usr/share/aws/emr/security/lib/*"

export HADOOP_OPTS="$HADOOP_OPTS -server -XX:+ExitOnOutOfMemoryError"
export HADOOP_NAMENODE_HEAPSIZE=1024
export HADOOP_DATANODE_HEAPSIZE=614
export HADOOP_JOB_HISTORYSERVER_HEAPSIZE=2252
export HADOOP_NAMENODE_OPTS="-javaagent:/home/hadoop/jmx_prometheus_javaagent-0.16.1.jar=12345:/home/hadoop/config.yml $HADOOP_NAMENODE_OPTS"

```

Una vez guardados los cambios en el archivo de configuración, deberemos reiniciar el servicio del NameNode con el comando ***sudo systemctl restart hadoop-hdfs-namenode*** para que se apliquen los cambios:

```

[hadoop@ip-172-31-35-236 opt]$ sudo nano /etc/hadoop/conf/hadoop-env.sh
[hadoop@ip-172-31-35-236 opt]$ [hadoop@ip-172-31-35-236 opt]$ sudo systemctl restart hadoop-hdfs-namenode

```

Tras esperar a que se reinicie un rato (2 minutos aprox), podemos comprobar que se ven las métricas desde el navegador buscando ***ip_pública_ec2_nodo_maestro:12345*** (en mi caso 54.237.230.110:12345) que es el puerto que hemos definido en el archivo "*hadoop-env.sh*".

```

← ↻ 🔒 No seguro | 54.237.230.110:12345
#| Campus Educatibr... MySQL Portada - Login Panel de Rewards Exportar SBC de FU... Red Dead Redempti... Microsoft Family Saf...

# HELP jmx_config_reload_failure_total Number of times configuration have failed to be reloaded.
# TYPE jmx_config_reload_failure_total counter
jmx_config_reload_failure_total 0.0
# HELP jvm_buffer_pool_used_bytes Used bytes of a given JVM buffer pool.
# TYPE jvm_buffer_pool_used_bytes gauge
jvm_buffer_pool_used_bytes(pool="direct",) 1210225.0
jvm_buffer_pool_used_bytes(pool="mapped",) 0.0
# HELP jvm_buffer_pool_capacity_bytes Bytes capacity of a given JVM buffer pool.
# TYPE jvm_buffer_pool_capacity_bytes gauge
jvm_buffer_pool_capacity_bytes(pool="direct",) 1210225.0
jvm_buffer_pool_capacity_bytes(pool="mapped",) 0.0
# HELP jvm_buffer_pool_used_buffers Used buffers of a given JVM buffer pool.
# TYPE jvm_buffer_pool_used_buffers gauge
jvm_buffer_pool_used_buffers(pool="direct",) 78.0
jvm_buffer_pool_used_buffers(pool="mapped",) 0.0
# HELP jvm_memory_objects_pending_finalization The number of objects waiting in the finalizer queue.
# TYPE jvm_memory_objects_pending_finalization gauge
jvm_memory_objects_pending_finalization 0.0
# HELP jvm_memory_bytes_used Used bytes of a given JVM memory area.
# TYPE jvm_memory_bytes_used gauge
jvm_memory_bytes_used(area="heap",) 1.57229808E8
jvm_memory_bytes_used(area="nonheap",) 7.1500488E7
# HELP jvm_memory_bytes_committed Committed (bytes) of a given JVM memory area.
# TYPE jvm_memory_bytes_committed gauge
jvm_memory_bytes_committed(area="heap",) 2.81018368E8
jvm_memory_bytes_committed(area="nonheap",) 7.2966144E7
# HELP jvm_memory_bytes_max Max (bytes) of a given JVM memory area.
# TYPE jvm_memory_bytes_max gauge
jvm_memory_bytes_max(area="heap",) 9.54728448E8
jvm_memory_bytes_max(area="nonheap",) -1.0
# HELP jvm_memory_bytes_init Initial bytes of a given JVM memory area.
# TYPE jvm_memory_bytes_init gauge
jvm_memory_bytes_init(area="heap",) 1.32120576E8
jvm_memory_bytes_init(area="nonheap",) 2555904.0
# HELP jvm_memory_pool_bytes_used Used bytes of a given JVM memory pool.
# TYPE jvm_memory_pool_bytes_used gauge
jvm_memory_pool_bytes_used(pool="Code Cache",) 1.2489856E7
jvm_memory_pool_bytes_used(pool="Metaspace",) 5.2779832E7
jvm_memory_pool_bytes_used(pool="Compressed Class Space",) 6237456.0
jvm_memory_pool_bytes_used(pool="PS Eden Space",) 1.18823408E8
jvm_memory_pool_bytes_used(pool="PS Survivor Space",) 9659448.0
jvm_memory_pool_bytes_used(pool="PS Old Gen",) 2.8746952E7
# HELP jvm_memory_pool_bytes_committed Committed bytes of a given JVM memory pool.
# TYPE jvm_memory_pool_bytes_committed gauge
jvm_memory_pool_bytes_committed(pool="Code Cache",) 1.2582912E7
jvm_memory_pool_bytes_committed(pool="Metaspace",) 5.3911552E7

```


4. Despliegue de Prometheus y Grafana

Ahora vamos a desplegar Prometheus y Grafana, las herramientas encargadas de recolectar y visualizar las métricas expuestas por el clúster EMR. Para ello, se utilizará una instancia EC2 adicional desde la cual se configurarán ambos servicios, permitiendo así un monitoreo centralizado y en tiempo real del entorno.

4.1. Creación de una instancia EC2 para Prometheus y Grafana

Lo primero de todo es crear una instancia EC2 donde poder desplegar Grafana y Prometheus. Es importante saber que debe estar en la misma VPC que el clúster EMR que como hemos configurado en el [apartado 1](#), es la default, es decir, la predeterminada.

Para empezar, accederemos al apartado EC2 de la consola de AWS y, dentro de la sección “*Instancias*”, haremos clic en el botón “*Lanzar instancias*”:



Ahí, pondremos el **nombre** de la instancia y su **sistema operativo (Ubuntu 22.04 LTS)**:

Lanzar una instancia Información

Amazon EC2 le permite crear máquinas virtuales, o instancias, que se ejecutan en la nube de AWS. Comience rápidamente siguiendo los sencillos pasos que se indican a continuación.

Nombre y etiquetas Información

Nombre
EC2_Prometheus_Grafana_IABD04 [Agregar etiquetas adicionales](#)

▼ Imágenes de aplicaciones y sistemas operativos (Imagen de máquina de Amazon) Información

Una AMI es una plantilla que contiene la configuración de software (sistema operativo, servidor de aplicaciones y aplicaciones) necesaria para lanzar la instancia. Busque o examine las AMI si no ve lo que busca a continuación.

Busque en nuestro catálogo completo que incluye miles de imágenes de sistemas operativos y aplicaciones

Recientes Mis AMI **Inicio rápido**

Amazon Linux	macOS	Ubuntu	Windows	Red Hat	SUSE Linux	Debian
--------------	-------	---------------	---------	---------	------------	--------

Imágenes de máquina de Amazon (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type
ami-0f9d6e2d2f067fca (64 bits (x86)) / ami-0967e555761d839e (64 bits (Arm))
Virtualización: hvm Activado para EMR: true Tipo de dispositivo raíz: xfs

Descripción
Ubuntu Server 22.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).
Canonical, Ubuntu, 22.04, amd64 jammy image

Arquitectura ID de AMI Fecha de publicación Nombre de usuario

64 bits (x86)	ami-0f9d6e2d2f067fca	2025-03-05	ubuntu
---------------	----------------------	------------	--------

[Proveedor verificado](#)

Continuaremos escogiendo el **tipo de instancia (t2.micro)**, y el **par de claves (vockey)** para poder conectarnos a esta en un futuro:

▼ Tipo de instancia Información [Obtener asesoramiento](#)

Tipo de instancia

t2.micro [Apto para la capa gratuita](#)

Familia: t2 1 vCPU 1 GB Memoria Generación actual: true

Bajo demanda Windows base precios: 0.0162 USD por hora Bajo demanda Ubuntu Pro base precios: 0.0134 USD por hora

Bajo demanda SUSE base precios: 0.0116 USD por hora Bajo demanda RHEL base precios: 0.026 USD por hora

Bajo demanda Linux base precios: 0.0116 USD por hora

[Se aplican costos adicionales a las AMI con software preinstalado](#)

[Comparar tipos de instancias](#)

▼ Par de claves (inicio de sesión) Información

Puede utilizar un par de claves para conectarse de forma segura a la instancia. Asegúrese de que tiene acceso al par de claves seleccionado antes de lanzar la instancia.

Nombre del par de claves - obligatorio

vockey [Crear un nuevo par de claves](#)

Ahora, seleccionaremos un **grupo de seguridad** existente (el mismo que el del EMR aunque da igual, pero así no tenemos que abrir los puertos después ya que, en este, como hemos visto en el [apartado 2.1](#), ya están todos abiertos), y con respecto a la **VPC** dejamos la default/predeterminada que es en la que se encuentra la otra instancia con el NameNode y las métricas aparte del EMR:

Así se ve la instancia EC2 configurada al completo:

Correcto
El lanzamiento de la instancia se inició correctamente (i-08db3243cd4387611)

Por último, solo queda esperar a que se inicialice pasando de “Iniciando”:

Instancias (1/11) Información

Última actualización: Hace less than a minute [Conectar](#)

Buscar instancia por atributo o etiqueta (case-sensitive) Todos los ...

	Name	ID de la instancia	Estado de la i...	Tipo de inst...	Comprobación de	Estado de la al...	Zona de dispon...	DNS de IPv4 pública	Dirección IP...
<input type="checkbox"/>	Prueba_Instan...	i-0089c945dfbaffb64	En ejecución	t2.micro	2/2 comprobador	Ver alarmas +	us-east-1b	ec2-44-202-105-100.co...	44.202.105.100
<input type="checkbox"/>	iabd	i-0743059024d793449	En ejecución	t2.micro	2/2 comprobador	Ver alarmas +	us-east-1c	ec2-18-212-100-69.co...	18.212.100.69
<input type="checkbox"/>	MiInstanciaAWS	i-0c62b4a50f2b1d97d	En ejecución	t2.micro	2/2 comprobador	Ver alarmas +	us-east-1c	ec2-13-218-222-5.com...	13.218.222.5
<input type="checkbox"/>	aws-cloud9-pr...	i-0408c613a1c2f6f4e	Detenida	t2.micro	-	Ver alarmas +	us-east-1a	-	-
<input type="checkbox"/>	iabd04-instanc...	i-0771fb0799e87306f	En ejecución	t2.micro	2/2 comprobador	Ver alarmas +	us-east-1b	ec2-54-161-241-232.co...	54.161.241.232
<input checked="" type="checkbox"/>	EC2_Promethe...	i-08db3243cd4387611	En ejecución	t2.micro	Iniciando	Ver alarmas +	us-east-1b	ec2-98-81-198-189.co...	98.81.198.189

A, “2/2 comprobaciones superadas”, lo cual significa que se ha lanzado correctamente y está disponible para ser usada:

Instancias (1/11) Información

Última actualización: Hace less than a minute [Conectar](#)

Q Buscar instancia por atributo o etiqueta (case-sensitive) Todos los ...

	Name	ID de la instancia	Estado de la i...	Tipo de inst...	Comprobación de	Estado de la al...	Zona de dispon...	DNS de IPv4 pública	Dirección IP...
<input type="checkbox"/>	Prueba_instan...	i-0089c945dfbaffb64	En ejecución	t2.micro	2/2 comprobador	Ver alarmas +	us-east-1b	ec2-44-202-105-100.co...	44.202.105.100
<input type="checkbox"/>	iabd	i-0743059024d793449	En ejecución	t2.micro	2/2 comprobador	Ver alarmas +	us-east-1c	ec2-18-212-100-69.co...	18.212.100.69
<input type="checkbox"/>	MilinstanciaAWS	i-0c62b4a50f2b1d97d	En ejecución	t2.micro	2/2 comprobador	Ver alarmas +	us-east-1c	ec2-13-218-222-5.com...	13.218.222.5
<input type="checkbox"/>	aws-cloud9-pr...	i-0408c613a1c2f6f4e	Detenida	t2.micro	-	Ver alarmas +	us-east-1a	-	-
<input type="checkbox"/>	iabd04-instan...	i-0771fb0799e87306f	En ejecución	t2.micro	2/2 comprobador	Ver alarmas +	us-east-1b	ec2-54-161-241-232.co...	54.161.241.232
<input checked="" type="checkbox"/>	EC2_Promethe...	i-08db3243cd4387611	En ejecución	t2.micro	2/2 comprobador	Ver alarmas +	us-east-1b	ec2-98-81-198-189.co...	98.81.198.189

4.2. Instalación de Prometheus en EC2

En este punto se busca instalar y configurar Prometheus en la instancia EC2, con el objetivo de recolectar las métricas expuestas por el nodo maestro del clúster EMR a través de JMX Exporter. Prometheus actuará como el motor de recopilación de datos, almacenando las métricas en formato de series temporales y permitiendo su consulta mediante el lenguaje PromQL.

Lo primero de todo es conectarse a la instancia EC2 vía SSH como hemos hecho en apartado anteriores. Lanzaremos el comando **ssh -i labsuser.pem ubuntu@DNS_ec2**, en mi caso **ec2-98-81-198-189.compute-1.amazonaws.com**:

```
C:\Users\Pablo\Downloads>ssh -i labsuser.pem ubuntu@ec2-98-81-198-189.compute-1.amazonaws.com
The authenticity of host 'ec2-98-81-198-189.compute-1.amazonaws.com (98.81.198.189)' can't be established.
ED25519 key fingerprint is SHA256:hRaipT8VvYz3Zl7s8oaQGx2AT/ULNR5Qg9SImmJ63s8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-98-81-198-189.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1024-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Mon Apr 7 16:53:46 UTC 2025

System load:  0.01          Processes:      108
Usage of /:   21.8% of 7.57GB Users logged in: 0
Memory usage: 20%          IPv4 address for eth0: 172.31.81.185
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-81-185:~$
```

Lanzamos ahora los comandos **wget https://github.com/prometheus/prometheus/releases/download/v2.30.3/prometheus-2.30.3.linux-amd64.tar.gz**

tar -xzf prometheus-2.30.3.linux-amd64.tar.gz para instalar Prometheus:

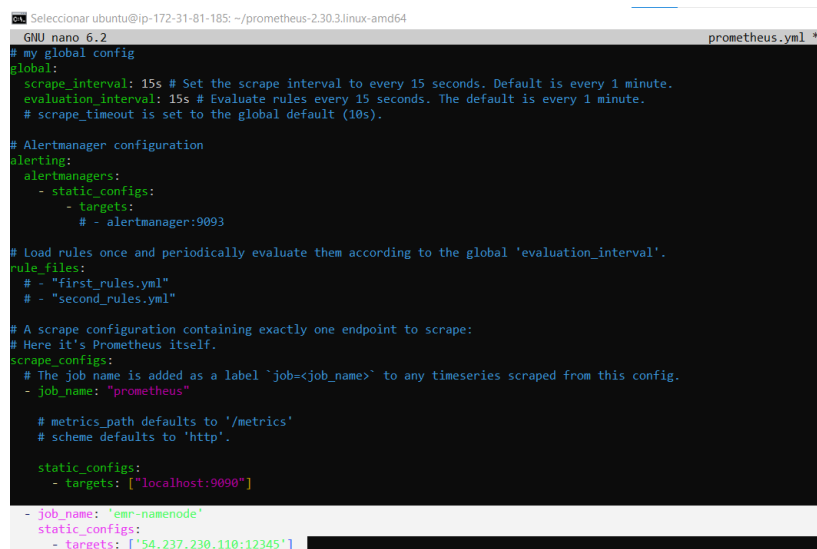
18

```
ubuntu@ip-172-31-81-185:~/prometheus-2.30.3.linux-amd64$ ls -la
total 185596
drwxr-xr-x 5 ubuntu ubuntu 4096 Apr  7 16:55 .
drwxr-xr-x 5 ubuntu ubuntu 4096 Apr  7 16:55 ..
-rw-r--r-- 1 ubuntu ubuntu 11357 Oct  5 2021 LICENSE
-rw-r--r-- 1 ubuntu ubuntu 3646 Oct  5 2021 NOTICE
drwxr-xr-x 2 ubuntu ubuntu 4096 Oct  5 2021 console_libraries
drwxr-xr-x 2 ubuntu ubuntu 4096 Oct  5 2021 consoles
drwxrwxr-x 4 ubuntu ubuntu 4096 Apr  7 16:57 data
-rwxr-xr-x 1 ubuntu ubuntu 100357256 Oct  5 2021 prometheus
-rw-r--r-- 1 ubuntu ubuntu 934 Oct  5 2021 prometheus.yml
-rwxr-xr-x 1 ubuntu ubuntu 89643838 Oct  5 2021 promtool
```

Como se puede ver en la imagen, existe, pero en caso de que no, deberíamos volver a instalar Prometheus en la EC2 como hemos hecho en el [apartado 4.1](#).

Ahora tenemos que editar ese archivo de configuración para ponerle un job que apunte al NameNode y pueda obtener las métricas que queremos de él. Lo editaremos con **nano prometheus.yml** para añadir esta parte de código dentro de la etiqueta “**scrape_configs**”:

```
- job_name: 'emr-namenode'
  static_configs:
    - targets: ['<ip-nodo-maestro>:12345']
```



```
GNU nano 6.2 prometheus.yml
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
      - targets:
        # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: 'emr-namenode'
    static_configs:
      - targets: ['54.237.230.110:12345']
```

Y simplemente nos saldríamos guardando el archivo.

4.4. Instalación de Grafana en EC2

Vamos ahora con la instalación de Grafana, para poder visualizar esas métricas que vamos a obtener con Prometheus en un formato de gráfico. Para ello, lanzaremos los siguientes comandos:

```
sudo apt-get install -y apt-transport-https
```

```
sudo apt-get install -y software-properties-common wget
```

```
ubuntu@ip-172-31-81-185:~$ sudo apt-get install -y apt-transport-https software-properties-common wget
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'apt' instead of 'apt-transport-https'
apt is already the newest version (2.4.13).
software-properties-common is already the newest version (0.99.22.9).
wget is already the newest version (1.21.2-2ubuntu1.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -

```
ubuntu@ip-172-31-81-185:~$ wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
t/sources.list.d/grafana.listWarning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
```

echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list

```
ubuntu@ip-172-31-81-185:~$ echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
deb https://packages.grafana.com/oss/deb stable main
```

sudo apt-get update

```
ubuntu@ip-172-31-81-185:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:5 https://packages.grafana.com/oss/deb stable InRelease [7660 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
```

sudo apt-get install grafana

```
ubuntu@ip-172-31-81-185:~$ sudo apt-get install grafana
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

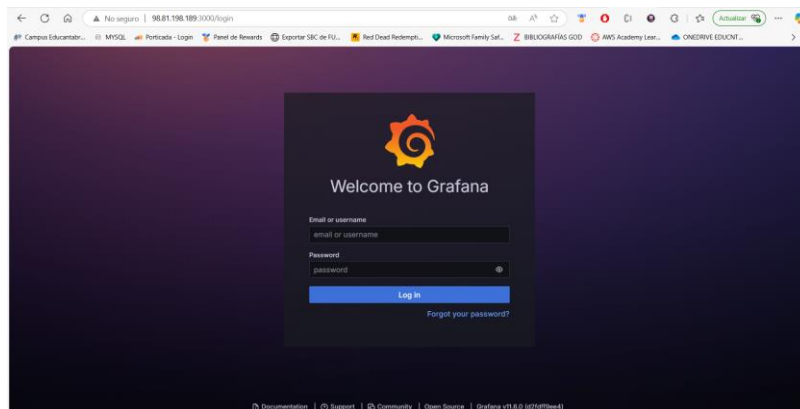
sudo systemctl start grafana-server

sudo systemctl enable grafana-server

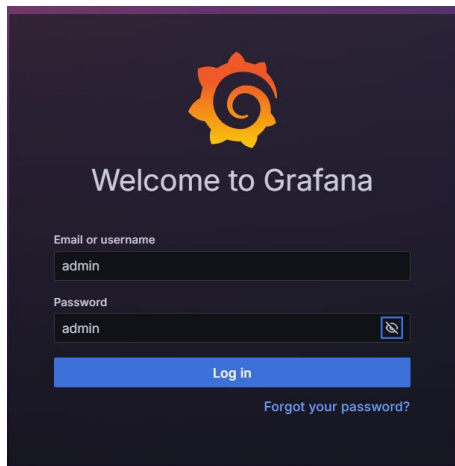
```
ubuntu@ip-172-31-81-185:~$ sudo systemctl start grafana-server
ubuntu@ip-172-31-81-185:~$ sudo systemctl enable grafana-server
Synchronizing state of grafana-server.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable grafana-server
Created symlink /etc/systemd/system/multi-user.target.wants/grafana-server.service → /lib/systemd/system/grafana-server.service.
ubuntu@ip-172-31-81-185:~$
```

4.5. Configuración de Grafana

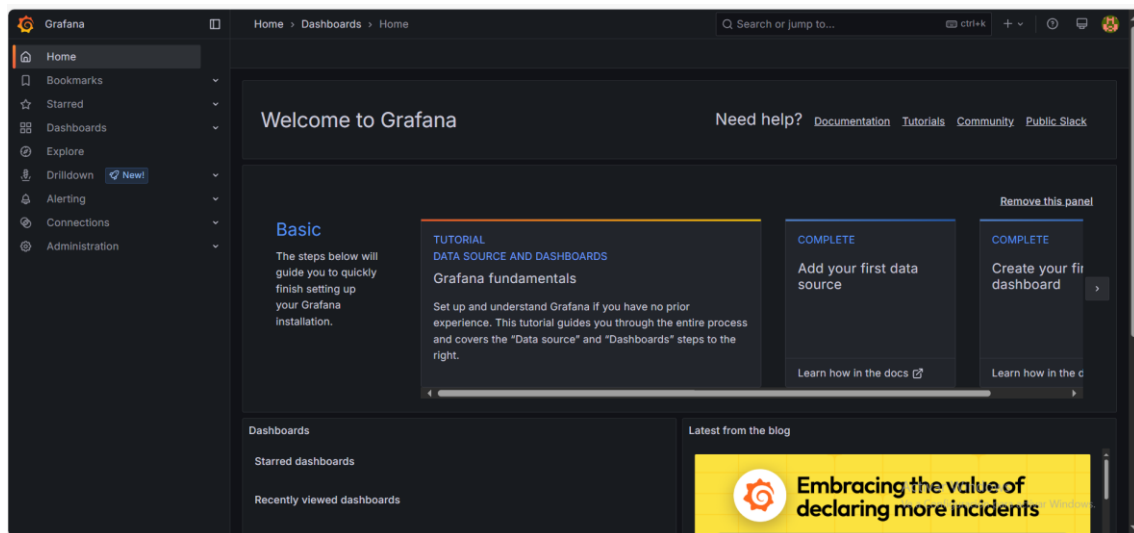
Lo primero de todo es acceder a Grafana poniendo en el navegador **ip_pública_ec2_con_grafana_y_prometheus:3000** (en mi caso 98.81.198.189:3000) que es el puerto en el que se despliega Grafana y hemos abierto en el [apartado 2.1](#):



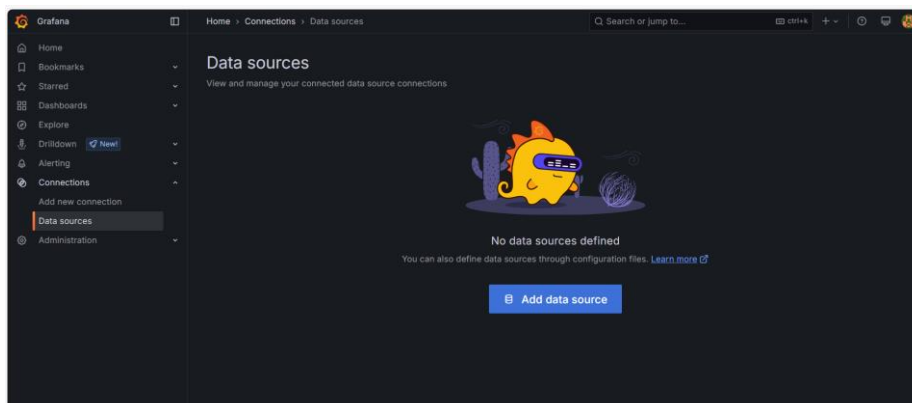
Iniciamos sesión con usuario: admin, contraseña: admin:



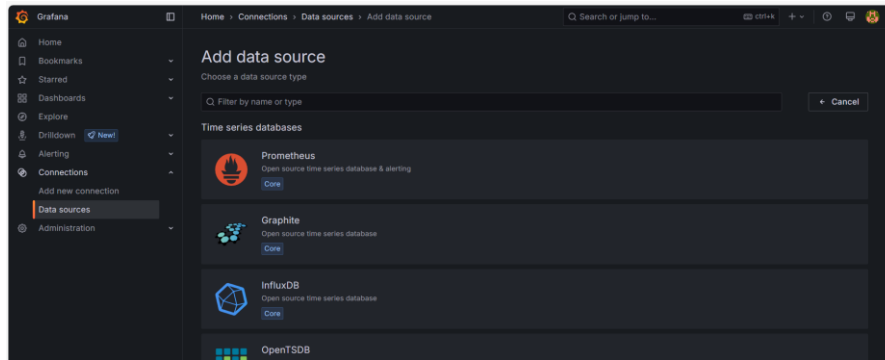
Así se ve la interfaz de inicio:



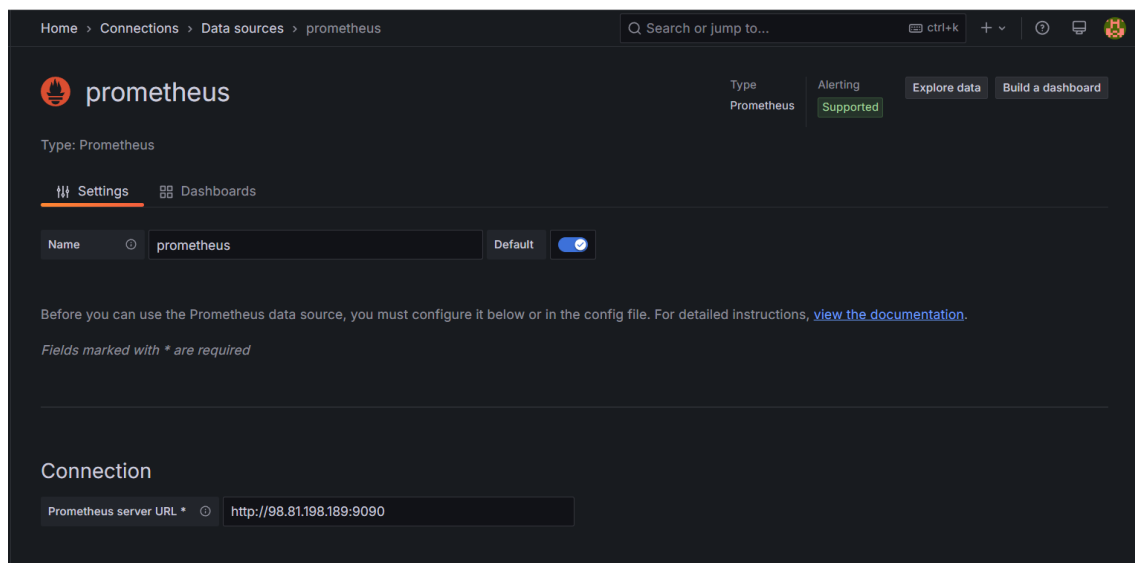
Vamos ahora a agregar Prometheus como fuente de datos. Para ello, en Grafana, iremos al apartado “Connections” y, dentro de este, seleccionaremos la opción “Data Sources” para agregar una nueva fuente de datos:



Una vez ahí, clicamos en “Add data source” y seleccionamos Prometheus:



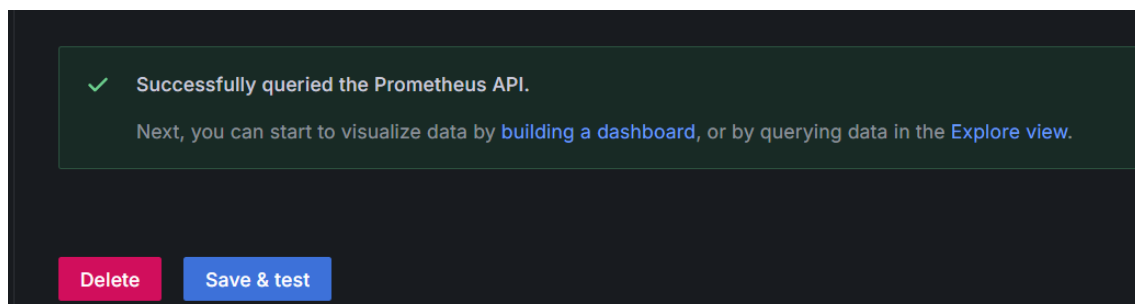
Dentro, debemos poner en la URL la ***ip_pública_ec2_con_grafana_y_prometheus:9090*** o ***http://localhost:9090*** que es la URL para acceder a nuestro Prometheus:



Y el resto de apartados los dejamos por defecto.

Por último, justo antes de darle a guardar, debemos levantar el servicio de Prometheus porque si no saldrá error ya que no detecta nada, para ello volvemos a la [conexión SSH](#) de esta EC2 y lanzamos los comandos necesarios como hemos hecho al final del [apartado 4.1](#).

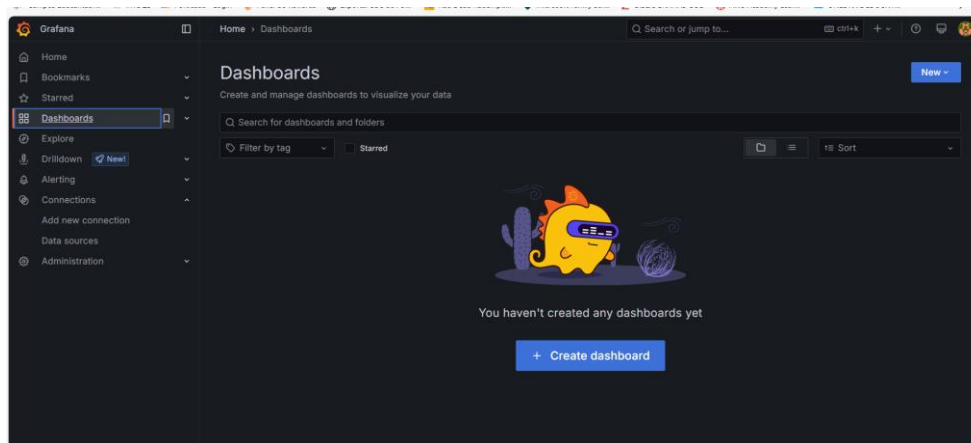
Una vez lanzado Prometheus, bajamos hasta abajo en Grafana y guardamos la fuente de datos clicando en “Save & test”, si todo sale bien debería aparecer algo parecido a esto:



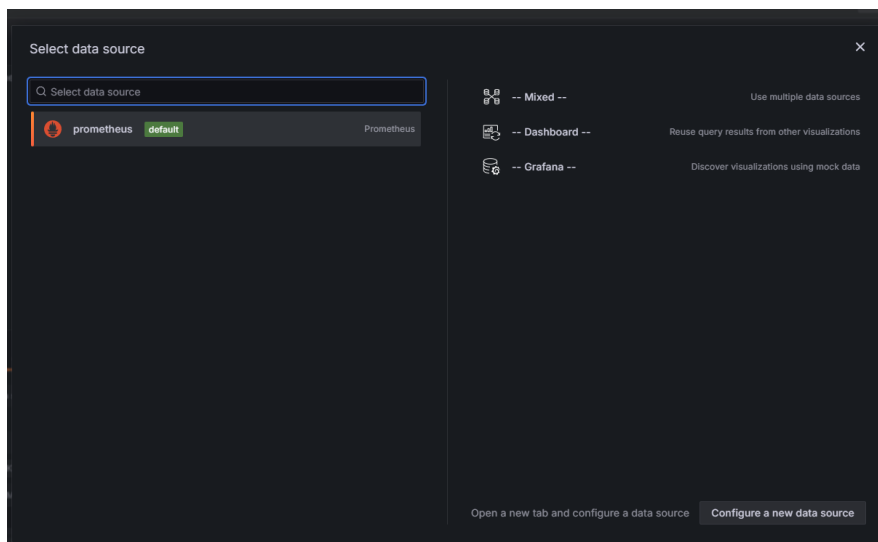
5. Visualización de métricas en Grafana

Ahora vamos a crear un dashboard en Grafana que nos permitirá visualizar de forma gráfica y en tiempo real las métricas recolectadas por Prometheus desde el clúster EMR. Este panel servirá como herramienta central para el monitoreo del estado del sistema, mostrando indicadores clave como el uso de CPU y memoria, el estado del NameNode y el espacio ocupado en HDFS. A continuación, se detallan los pasos para su creación y configuración.

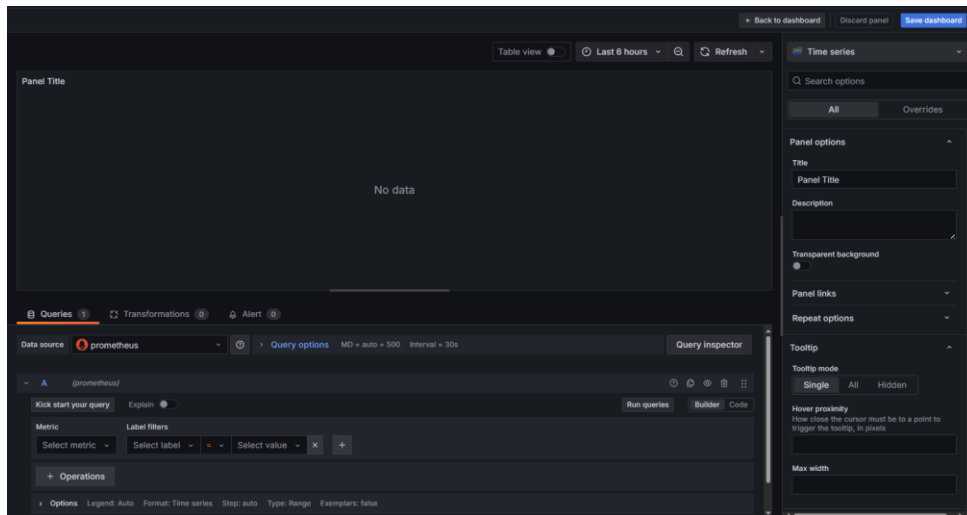
Para crear un dashboard en Grafana, deberemos ir al apartado en el menú izquierdo de “Dashboards” y clicar en “Create dashboard”:



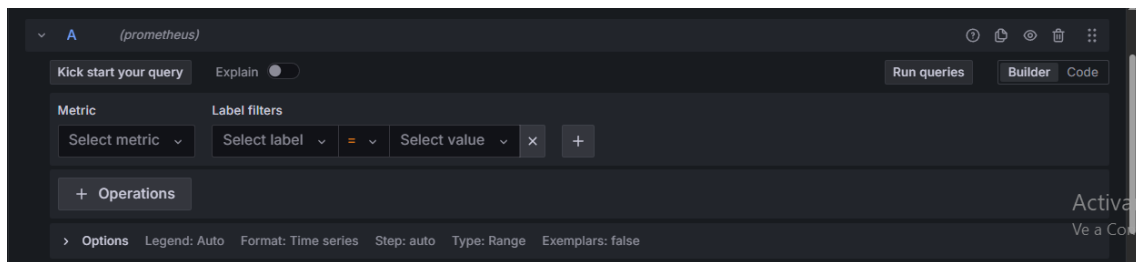
Dentro, le daremos a “Add visualization” y seleccionaremos Prometheus (fuente de datos que hemos definido anteriormente):



Una vez dentro, veremos algo parecido a esto:

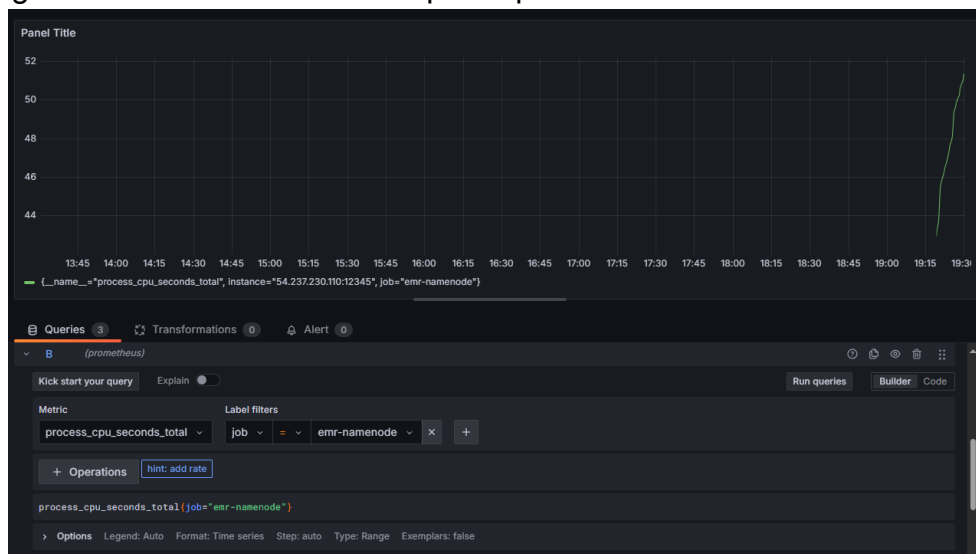


Para agregar métricas, como se ve en la imagen, hay un apartado debajo del “Data source” en el cual se pueden escribir las métricas, y una vez escritas se clicca en “Run queries” para verlas representadas en el gráfico de encima:

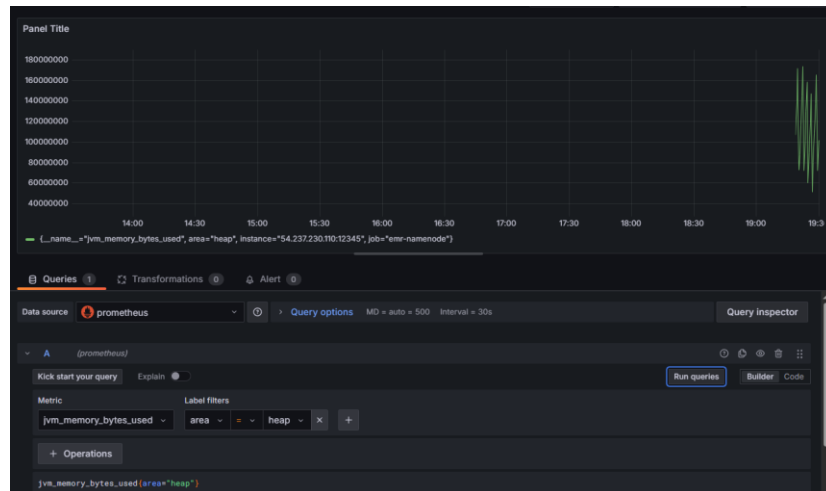


A continuación, se incorporarán al dashboard las siguientes métricas clave para el monitoreo del clúster:

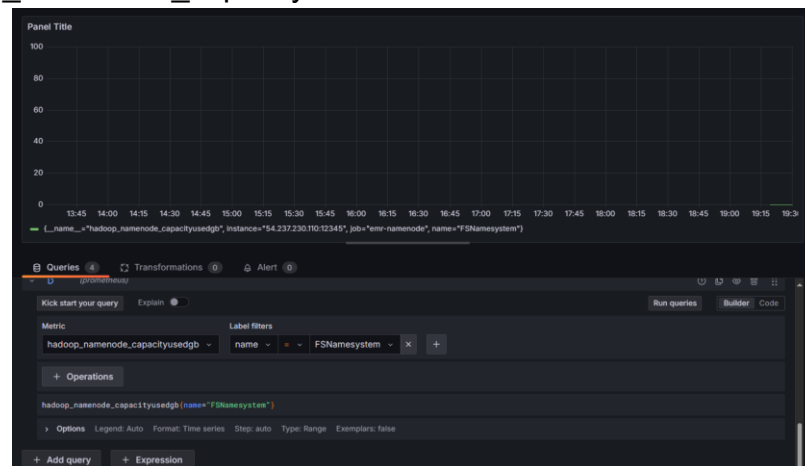
- **Uso de CPU:** `process_cpu_seconds_total` → Indica la cantidad total de segundos de CPU consumidos por el proceso desde su inicio.



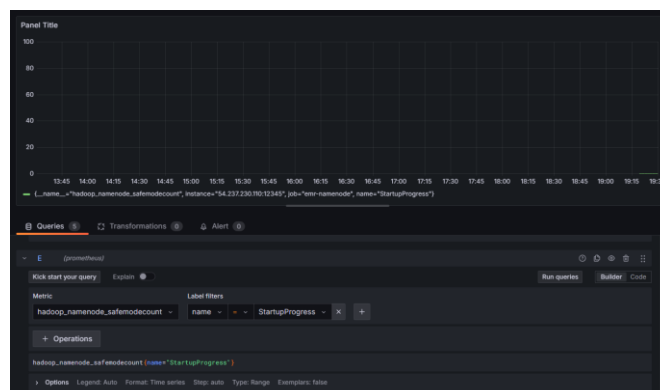
- **Uso de RAM:** `jvm_memory_bytes_used{area="heap"}` → Muestra la cantidad de memoria heap utilizada por la máquina virtual Java en bytes, reflejando el uso actual de RAM por la aplicación.



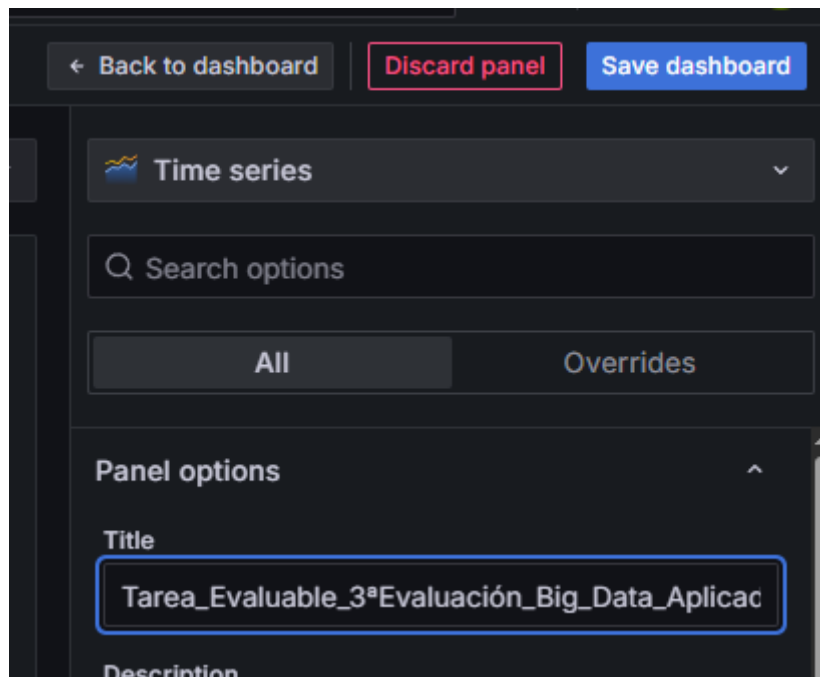
- **Espacio utilizado en HDFS:** `hadoop_namenode_capacityusedgb` → Indica la cantidad de espacio en gigabytes actualmente utilizado en el sistema de archivos distribuido HDFS. Para verlo en bytes la métrica sería `hadoop_namenode_capacityused`.



- **Estado del NameNode:**
`hadoop_namenode_safemodecount{name="StartupProgress",}` → Devuelve 1 si el NameNode está en modo seguro y 0 si ya ha salido de ese estado y se encuentra operativo, siendo la forma más directa de conocer su estado. Alternativamente, también pueden utilizarse métricas como `hadoop_namenode_safemodepercentcomplete` o `hadoop_namenode_safemodetotal`, aunque ofrecen una visión menos precisa.



Por último, para guardar el dashboard con todas las métricas que hemos obtenido, nos iremos a la parte superior derecha, le pondremos un nombre y clicaremos en “*Save dashboard*”:



Y, así se verían todas las métricas juntas y listas para ser exploradas en un dashboard guardado y completo:



6. Preguntas de reflexión sobre la práctica

1. ¿Qué métricas consideras más importantes para monitorear en un clúster EMR? ¿Por qué?

A la hora de monitorizar un clúster EMR, es clave centrarse en las métricas que realmente nos dan información útil sobre el estado del sistema. Por ejemplo, el uso de CPU y memoria RAM por nodo es fundamental porque nos ayuda a detectar si algún proceso está saturando recursos o si la carga no se está distribuyendo bien. También es importante tener controlado el espacio libre en HDFS, para asegurarnos

de que el sistema no se queda sin almacenamiento mientras procesamos datos. Otra métrica que no puede faltar es el estado del NameNode (y también del JobTracker), ya que, si alguno de estos componentes falla, puede verse afectado todo el clúster. Y, por último, mirar cuántas tareas están activas o fallan nos da una idea bastante clara de cómo están funcionando los jobs en tiempo real.

2. ¿Cómo podrías mejorar la configuración de JMX Exporter para recopilar métricas más específicas?

Para mejorar la configuración de JMX Exporter, lo ideal es usar expresiones regulares más concretas en el archivo config.yml, así solo recogemos las métricas que realmente nos interesan. Esto viene genial porque evita tener dashboards llenos de información irrelevante y, además, mejora el rendimiento al reducir la cantidad de datos que se procesan. Otra buena práctica es separar las métricas por servicio (por ejemplo, Hadoop, Spark o Hive) y gestionarlas desde distintos jobs en Prometheus. Esto hace que todo sea más fácil de mantener, los paneles sean más claros y también nos permite crear alertas específicas para cada parte del sistema.

3. ¿Qué ventajas tiene usar Prometheus y Grafana frente a otras herramientas de monitoreo?

Prometheus y Grafana tienen un montón de ventajas frente a otras herramientas de monitoreo, sobre todo porque son de código abierto, lo que significa que no necesitas pagar licencias y puedes adaptarlas a lo que necesites. Además, tienen muchísima documentación y una comunidad activa que ayuda un montón cuando te atascas. Prometheus destaca por cómo maneja los datos en series temporales, lo que lo hace súper eficiente para trabajar con métricas en tiempo real. También tiene su propio lenguaje de consultas, PromQL, que te permite hacer búsquedas bastante potentes. Y Grafana, por su parte, te deja crear visualizaciones muy personalizables e interactivas, perfectas para montar dashboards a medida según lo que necesites monitorear. Juntas, son una solución muy completa, flexible y fácil de escalar para controlar entornos como clústeres EMR.

7. Extracción local de archivos de configuración del clúster EMR y EC2

Ahora vamos a extraer algunos archivos de configuración tanto del clúster EMR como de la instancia EC2 que contiene Prometheus y Grafana, con el fin de conservarlos y poder reutilizarlos o consultarlos en futuras configuraciones similares.

7.1. Extracción archivos del clúster EMR (nodo maestro)

Lo que queremos hacer aquí, es extraer 3 archivos de configuración que son “*config.yml*”, “*jmx_prometheus_javaagent-0.16.1.jar*” y el “*hadoop-env.sh*”, para poder hacer lo que queramos con ellos en un futuro o simplemente tenerlos en nuestra máquina local.

Lo primero de todo será tener descargado el par de claves vockey como hemos visto en el [apartado 2.1](#). Una vez las tengamos descargadas y ubicadas, abriremos el cmd en esa carpeta y lanzaremos el comando ***scp -i labsuser.pem hadoop@DNS_nodo_master:ruta/de/los/archivos***.

El primer archivo que vamos a descargarnos localmente es “*config.yml*”. Para ello debemos recordar la ruta de este ([apartado 3.1](#)), la cual era */opt*, por lo que el comando en mi caso quedaría ***scp -i labsuser.pem hadoop@ec2-3-88-24-255.compute-1.amazonaws.com:/opt/config.yml*** .:

```
C:\Users\pablo>cd Downloads
C:\Users\pablo\Downloads>scp -i labsuser.pem hadoop@ec2-3-88-24-255.compute-1.amazonaws.com:/opt/config.yml .
config.yml                                     100% 83   0.4KB/s   00:00
C:\Users\pablo\Downloads>
```

Continuaremos con el archivo “*jmx_prometheus_javaagent-0.16.1.jar*”, el cual se encuentra en la misma ruta que el anterior ([apartado 3](#)), */opt*, por lo que el comando quedaría (en mi caso) así ***scp -i labsuser.pem hadoop@ec2-3-88-24-255.compute-1.amazonaws.com:/opt/jmx_prometheus_javaagent-0.16.1.jar*** .:

```
C:\Users\pablo\Downloads>scp -i labsuser.pem hadoop@ec2-3-88-24-255.compute-1.amazonaws.com:/opt/jmx_prometheus_javaagen
t-0.16.1.jar .
jmx_prometheus_javaagent-0.16.1.jar          100% 459KB 557.3KB/s   00:00
C:\Users\pablo\Downloads>
```

Y por último, extraeremos el archivo “*hadoop-env.sh*”, el cual se encuentra ([apartado 3.2](#)) en */etc/hadoop/conf*, por lo que actualizando el comando quedaría (en mi caso) así ***scp -i labsuser.pem hadoop@ec2-3-88-24-255.compute-1.amazonaws.com:/etc/hadoop/conf/hadoop-env.sh*** .:

```
C:\Users\pablo\Downloads>scp -i labsuser.pem hadoop@ec2-3-88-24-255.compute-1.amazonaws.com:/etc/hadoop/conf/hadoop-env.
sh .
hadoop-env.sh                                100% 4149  22.0KB/s   00:00
```

7.2. Extracción archivos de la EC2 con Prometheus y Grafana

Lo que queremos hacer aquí, es extraer 1 archivo de configuración que es “*prometheus.yml*”, para poder hacer lo que queramos con él en un futuro o simplemente tenerlo en nuestra máquina local para reutilizarlo.

Aprovechando que anteriormente nos hemos posicionado en la carpeta donde tenemos el par de claves vockey, lanzaremos el comando ***scp -i labsuser.pem ubuntu@DNS_EC2_con_Prometheus_Grafana:ruta/de/los/archivos*** .:

Simplemente buscaremos la ruta de este archivo ([apartado 4.2](#)) que es `/home/ubuntu/prometheus-2.30.3.linux-amd64` o `~/prometheus-2.30.3.linux-amd64` y la sustituiremos en el comando quedando (en mi caso) así **`scp -i labsuser.pem ubuntu@ec2-3-82-230-99.compute-1.amazonaws.com:/home/ubuntu/prometheus-2.30.3.linux-amd64/prometheus.yml . :`**

```
C:\Users\pablo\Downloads>scp -i labsuser.pem ubuntu@ec2-3-82-230-99.compute-1.amazonaws.com:/home/ubuntu/prometheus-2.30.3.linux-amd64/prometheus.yml .
prometheus.yml                                     100% 1027    5.5KB/s   00:00
C:\Users\pablo\Downloads>
```