**Práctica: Monitoreo de un clúster AWS EMR con Prometheus y Grafana**

ÍNDICE

[Objetivo de la práctica 3](#_Toc193220366)

[Instrucciones: 3](#_Toc193220367)

[Parte 1: Configuración del clúster AWS EMR 3](#_Toc193220368)

[Parte 2: Configuración de JMX Exporter 5](#_Toc193220369)

[Parte 3: Despliegue de Prometheus y Grafana 8](#_Toc193220370)

[Parte 4: Visualización de métricas en Grafana 12](#_Toc193220371)

[Preguntas para reflexión 14](#_Toc193220373)

[Github con los archivos de configuración 16](#_Toc193220373)

# Objetivo de la práctica

El objetivo de esta práctica es:

1. Configurar un clúster **AWS EMR**.
2. Exponer métricas del clúster utilizando **JMX Exporter**.
3. Recopilar y almacenar métricas utilizando **Prometheus**.
4. Visualizar las métricas en **Grafana**.

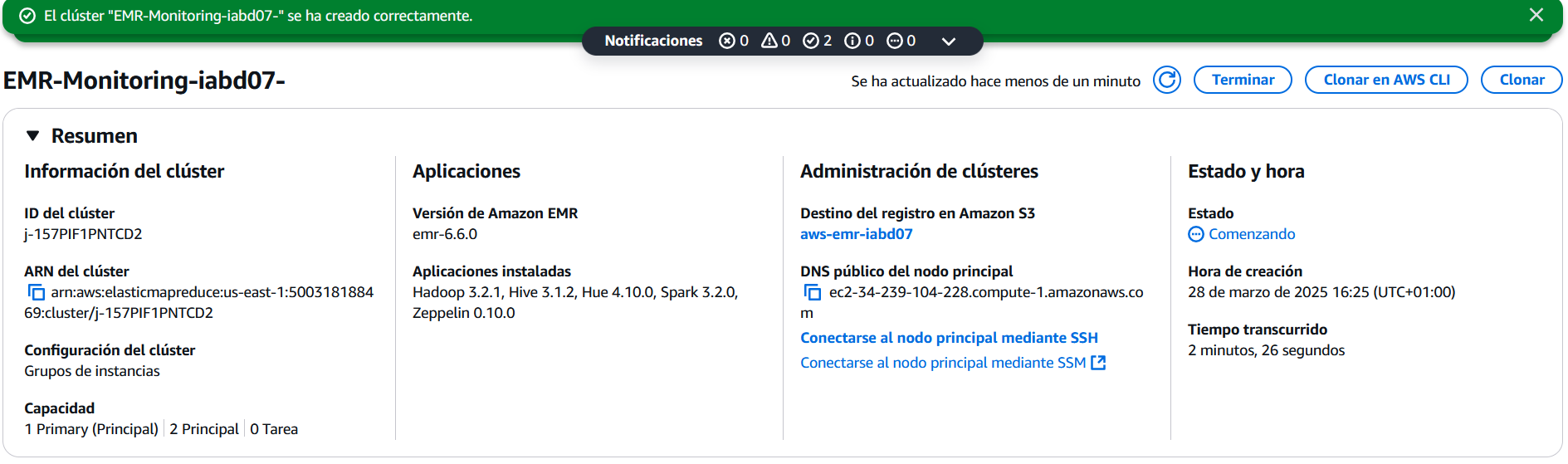
Al finalizar la práctica, habrás creado un sistema de monitoreo completo para un clúster EMR, utilizando herramientas de código abierto.

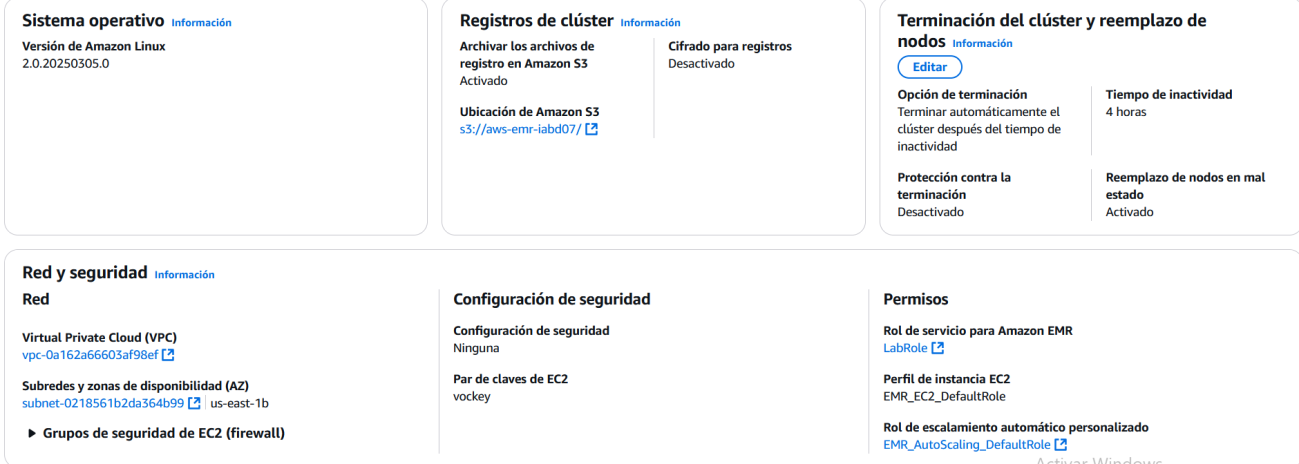
# Instrucciones:

## Parte 1: Configuración del clúster AWS EMR

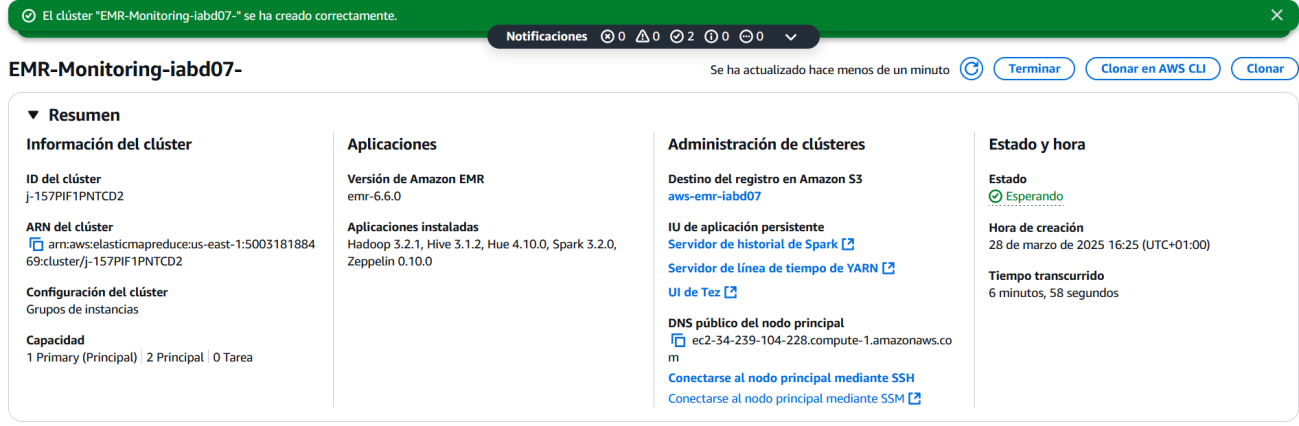
1. **Crear un clúster EMR**:
   * Accede a la consola de AWS y crea un clúster EMR con las siguientes características:
     + Nombre del clúster: EMR-Monitoring-iabd07.
     + Aplicaciones: Hadoop, Spark y Hive.
     + El resto de las configuraciones, las mismas que las adoptadas en otras prácticas sobre AWS EMR
     + Número de instancias: 1 nodo maestro y 2 nodos core.
     + Clave SSH: ls por defecto en el laboratorio.

Iniciamos un laboratorio en AWS y vamos a EMR para celar el cluster

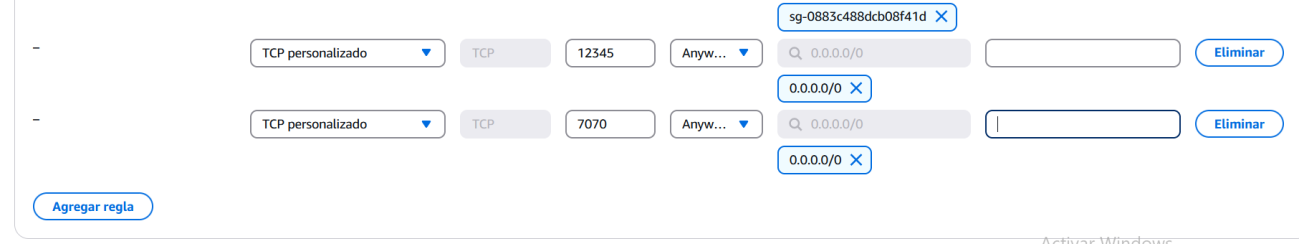




* + Espera a que el clúster esté en estado **Waiting**.



Agrego reglas de entrada:

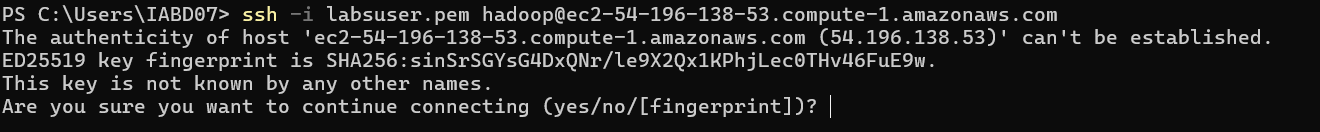


1. **Conectar al nodo maestro**:
   * Usa SSH para conectarte al nodo maestro del clúster:

Descargamos el labsuser.pem y lo colocamos en la carpeta IABD07 en mi ordenador que está dentro de C:\usuarios.

A continuación lanzamos en una terminal el siguiente comando dentro de C:\Users/IABD07:

ssh -i labsuser.pem hadoop@ec2-54-196-138-53.compute-1.amazonaws.com





## Parte 2: Configuración de JMX Exporter

1. **Instalar JMX Exporter**:
   * Explicar o dar un razonamiento extenso de que es y para que sirve JMX en este entorno

JMX es una tecnología de Java que proporciona una arquitectura para la gestión y supervisión de aplicaciones, sistemas y redes. JMX permite gestionar y monitorear aplicaciones Java en tiempo real.

El propósito de usar JMX en un entorno de monitoreo como Prometheus es que JMX proporciona una vía para acceder a las métricas internas de una aplicación Java. Por ejemplo, se pueden obtener métricas como el uso de memoria, la cantidad de hilos en ejecución, la tasa de solicitudes, la latencia de las operaciones, entre muchas otras. Estas métricas son fundamentales para el monitoreo y diagnóstico de sistemas, ya que permiten a los administradores detectar y solucionar problemas de rendimiento y eficiencia.

JMX Exporter es una herramienta que actúa como un puente entre JMX y Prometheus. Recoge las métricas expuestas por una aplicación Java a través de JMX y las expone en un formato que Prometheus puede recolectar y procesar.

* + el JMX Exporter en el nodo maestro: usamos el siguiente comando:

wget <https://repo1.maven.org/maven2/io/prometheus/jmx/jmx_prometheus_javaagent/0.16.1/jmx_prometheus_javaagent-0.16.1.jar>



1. **Crear el archivo de configuración**:
   * Crea un archivo config.yml para el JMX Exporter:

nano config.yml



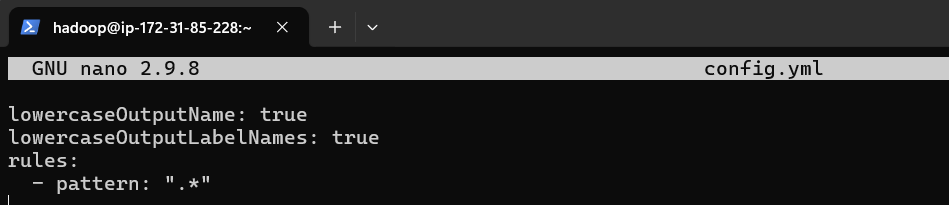
* + Agrega el siguiente contenido:

lowercaseOutputName: true

lowercaseOutputLabelNames: true

rules:

- pattern: ".\*"



1. **Configurar el NameNode para usar JMX Exporter**:
   * Edita el archivo de configuración del NameNode:

sudo nano /etc/hadoop/conf/hadoop-env.sh



* + Agrega la siguiente línea:

export HADOOP\_NAMENODE\_OPTS="-javaagent:/home/hadoop/jmx\_prometheus\_javaagent-0.16.1.jar=12345:/home/hadoop/config.yml $HADOOP\_NAMENODE\_OPTS"

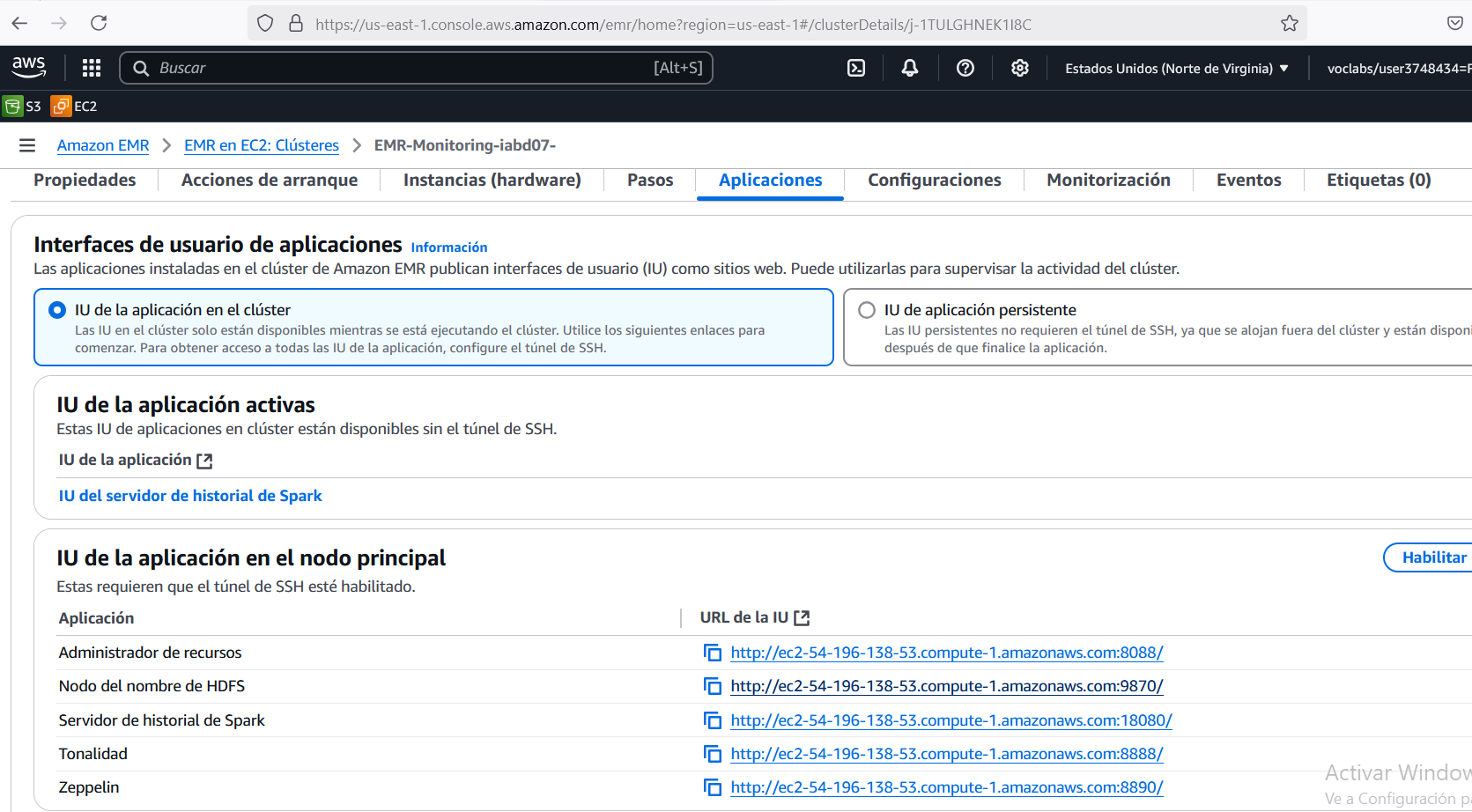


1. **Reiniciar el NameNode**:
   * Reinicia el servicio del NameNode para aplicar los cambios:

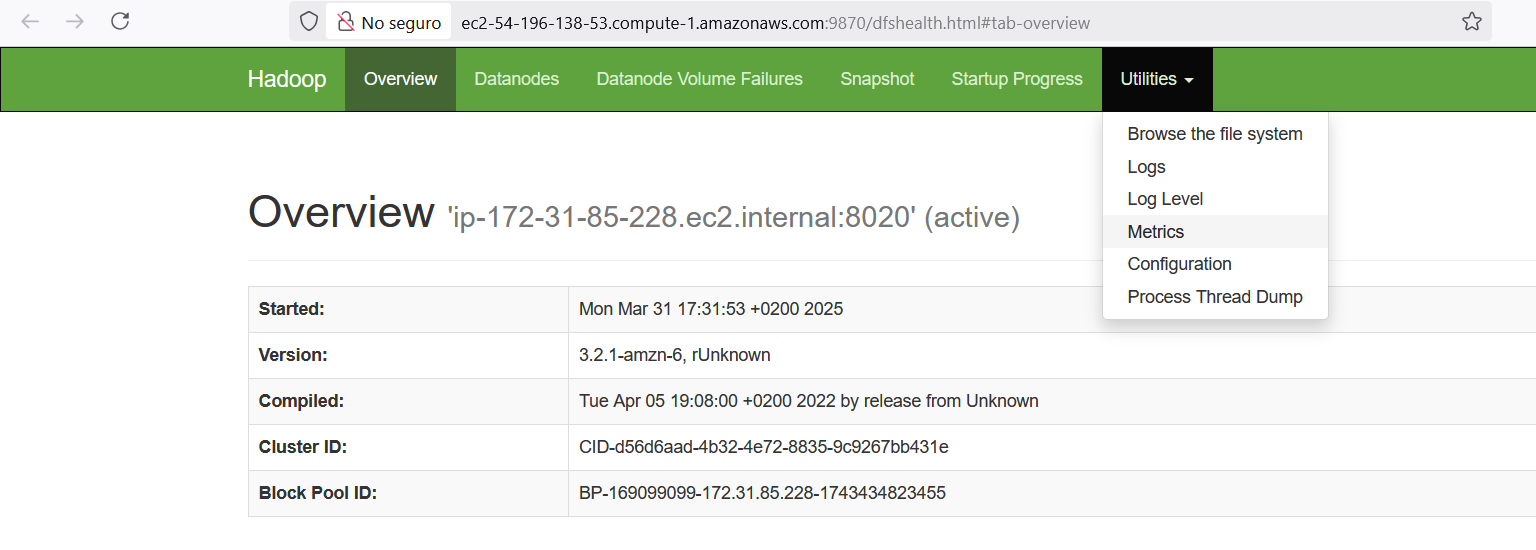
sudo systemctlrestarthadoop-hdfs-namenode

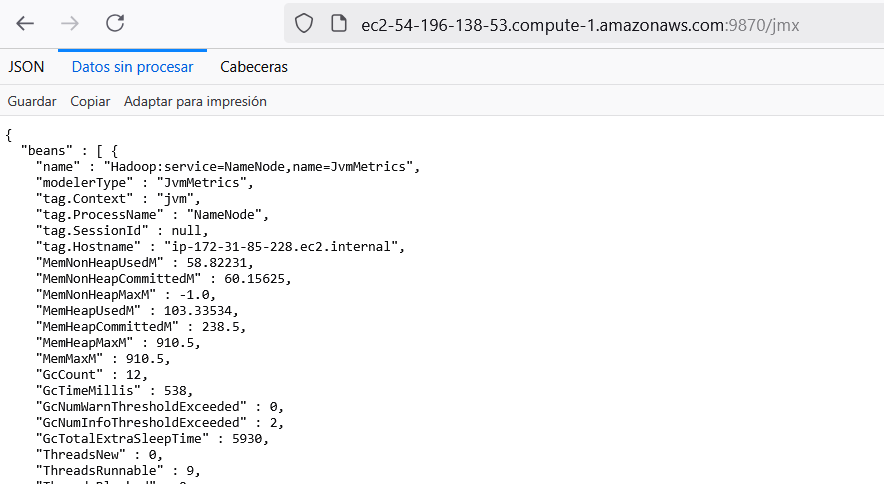


En el cluster vamos a aplicaciones y al puerto 9870 que es del nodo del nombre de HDFS:

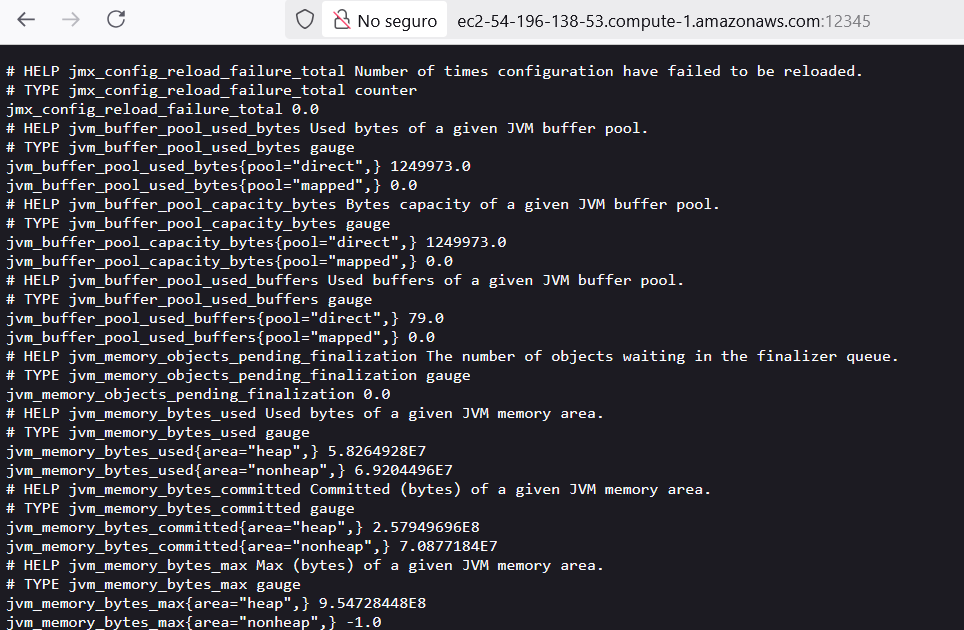


Vamos a la pestaña de metrics



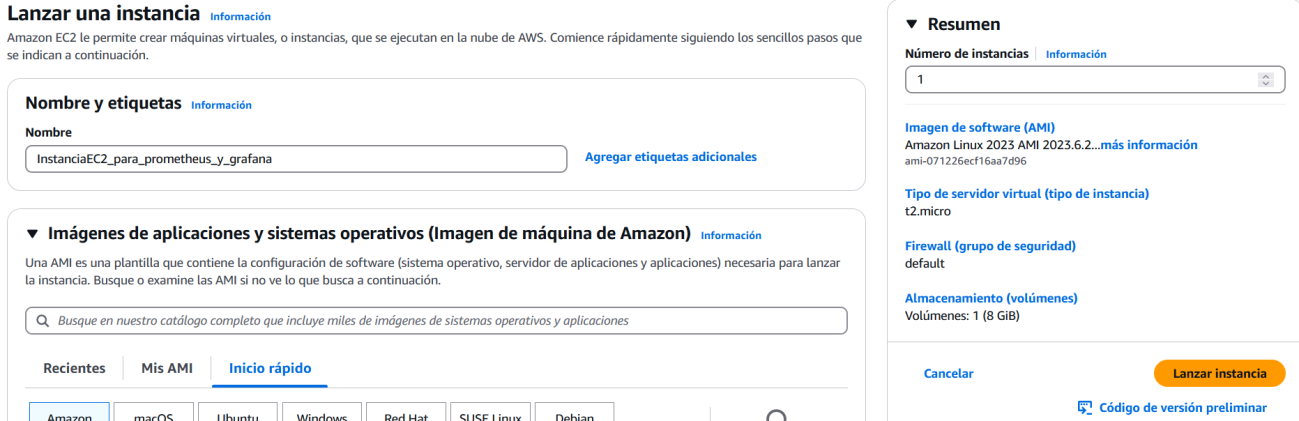


A continuación vamos al puerto 12345 con la ip-publica de la instancia de ec2 para comprobar que tenemos las métricas:



## Parte 3: Despliegue de Prometheus y Grafana

1. **Crear una instancia EC2 para Prometheus y Grafana**:
   * Crea una instancia EC2 con Ubuntu en la misma VPC que el clúster EMR.

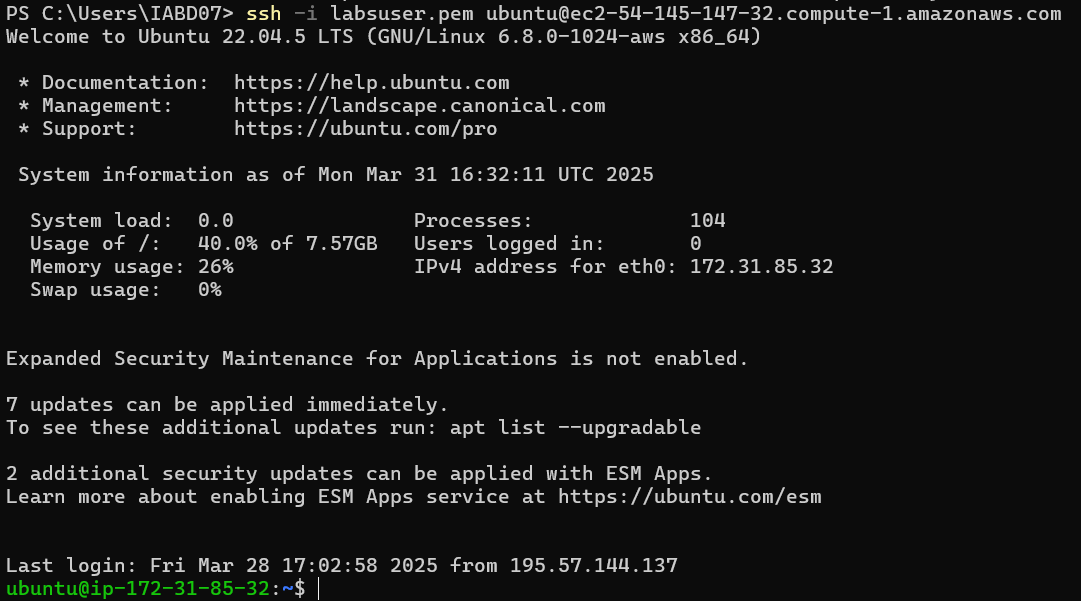




1. **Instalar Prometheus**:
   * Conéctate a la instancia EC2 y sigue los pasos para instalar Prometheus:

Como ya tenemos descargado anteriormente el labsuser.pem. A continuación lanzamos en una terminal el siguiente comando dentro de C:\Users/IABD07:

ssh -i labsuser.pem ubuntu@ec2-54-145-147-32.compute-1.amazonaws.com

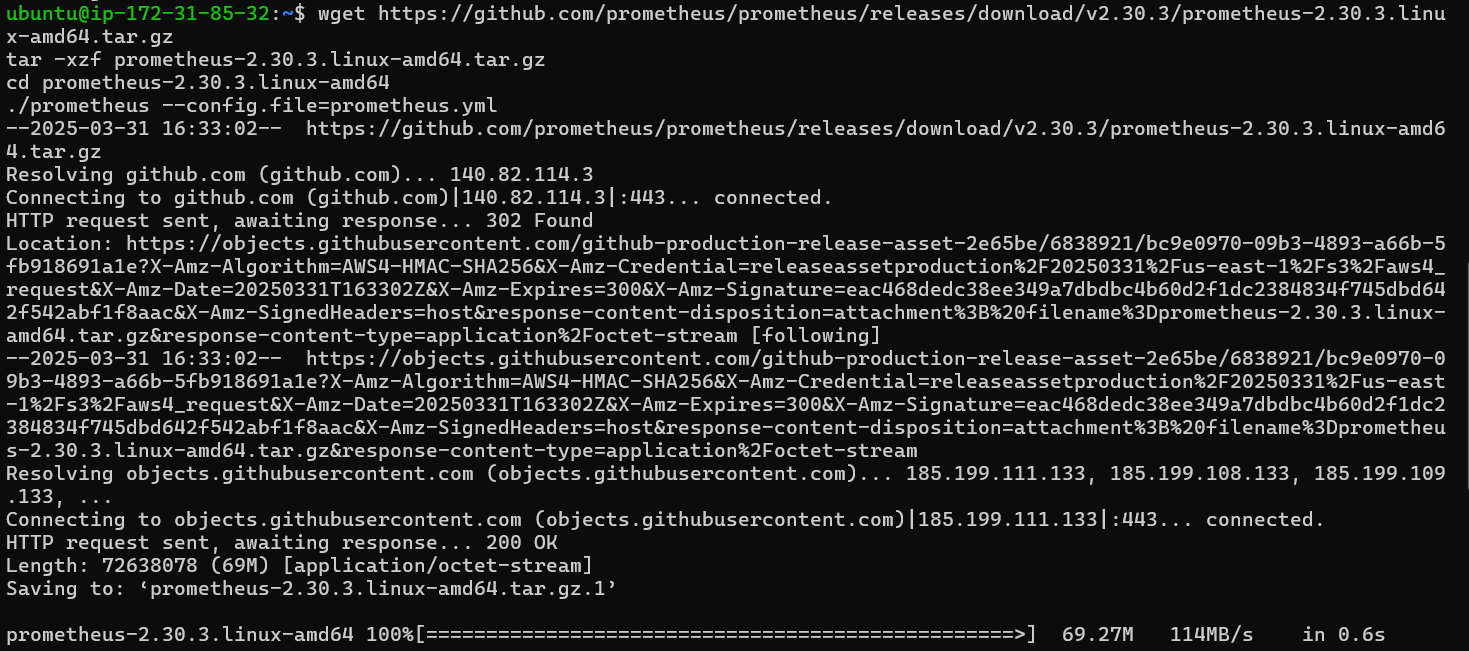


wget https://github.com/prometheus/prometheus/releases/download/v2.30.3/prometheus-2.30.3.linux-amd64.tar.gz

tar -xzf prometheus-2.30.3.linux-amd64.tar.gz

cd prometheus-2.30.3.linux-amd64

./prometheus --config.file=prometheus.yml



1. **Configurar Prometheus**:
   * Edita el archivo prometheus.yml para agregar el clúster EMR como objetivo :

global:

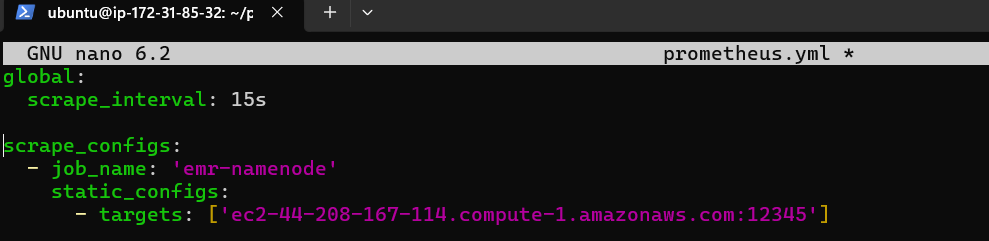
scrape\_interval: 15s

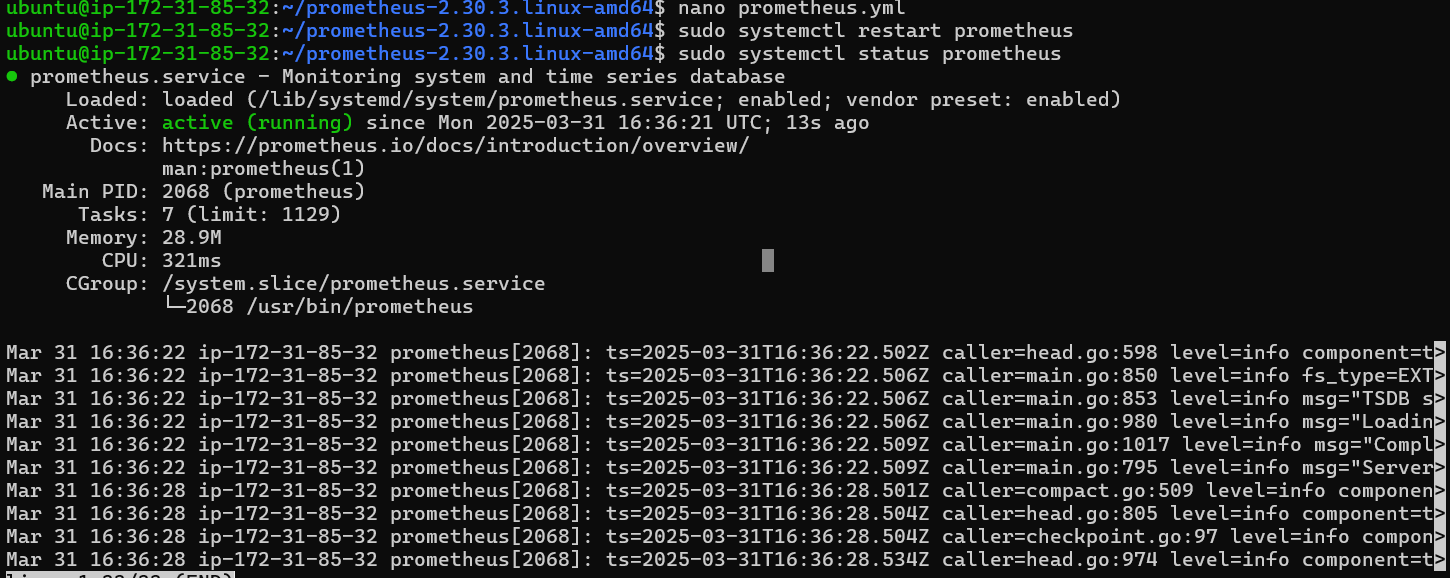
scrape\_configs:

- job\_name: 'emr-namenode'

static\_configs:

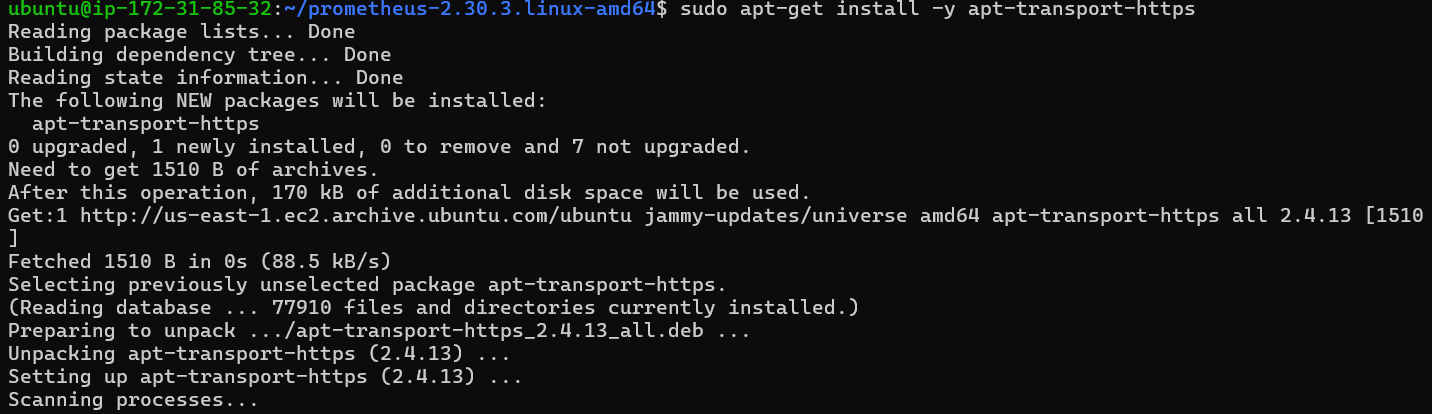
- targets: ['<ip-nodo-maestro>:12345']



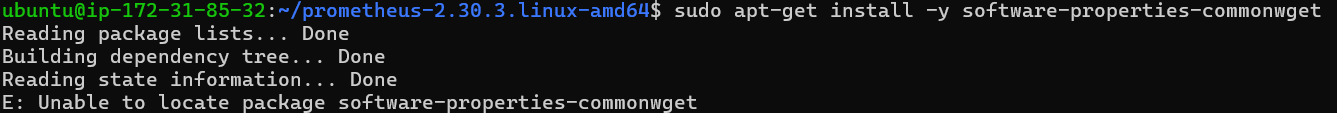


1. **Instalar Grafana**:
   * Instala Grafana en la misma instancia EC2:

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



sudo apt-get install -y software-properties-commonwget

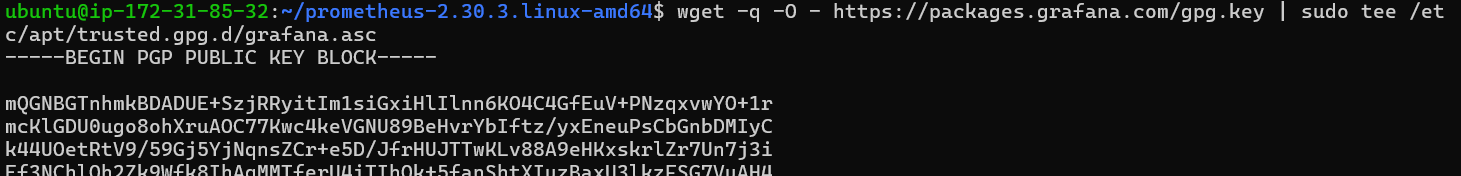


wget -q -O - https://packages.grafana.com/gpg.key |

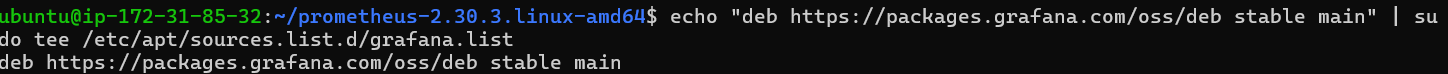
sudo apt-keyadd –

\*No me funciona con esos comandos. Buscando en chatgpt utilice este:

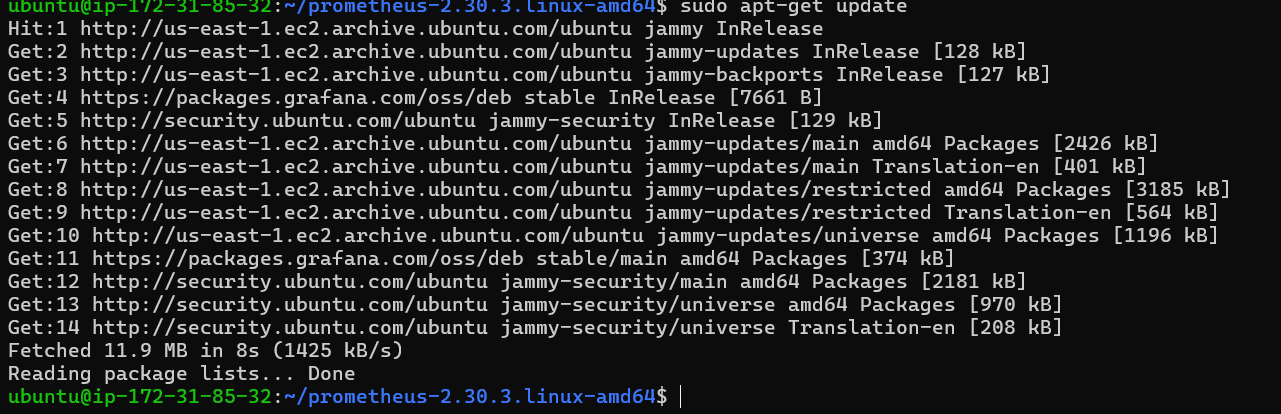
wget -q -O - https://packages.grafana.com/gpg.key | sudo tee /etc/apt/trusted.gpg.d/grafana.asc



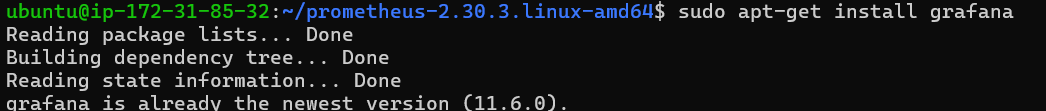
echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee /etc/apt/sources.list.d/grafana.list



sudo apt-get update



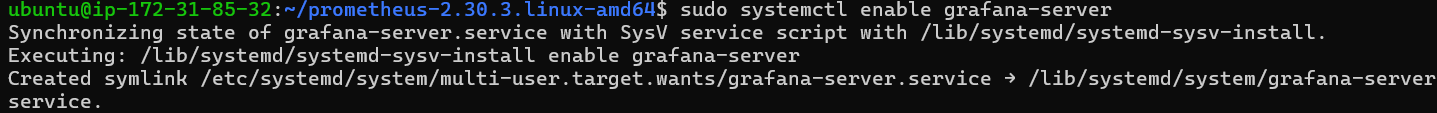
sudo apt-get install grafana



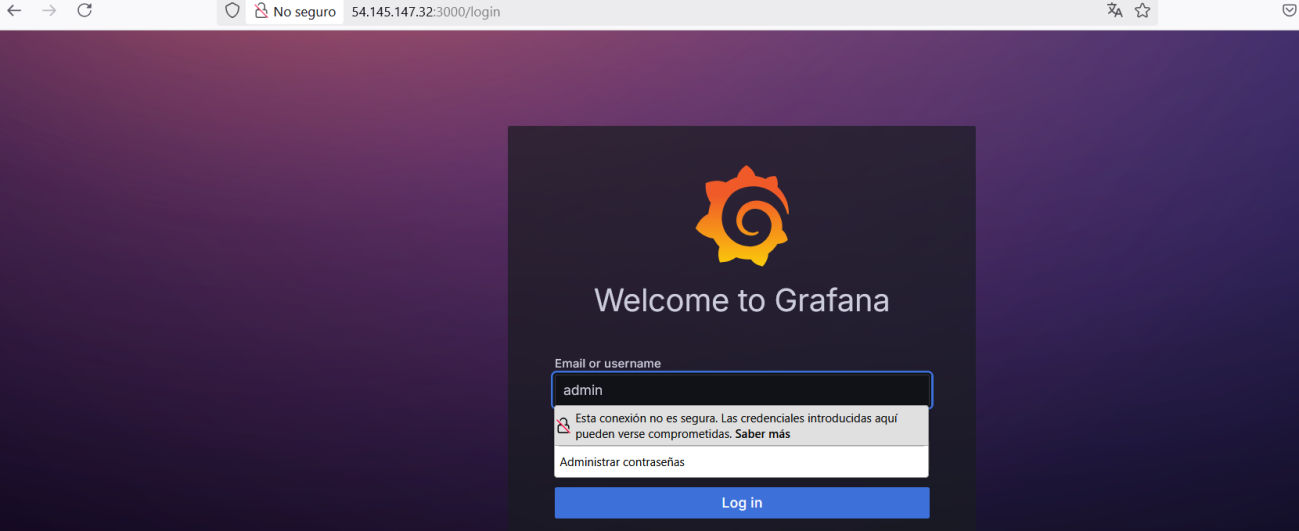
sudo systemctl start grafana-server



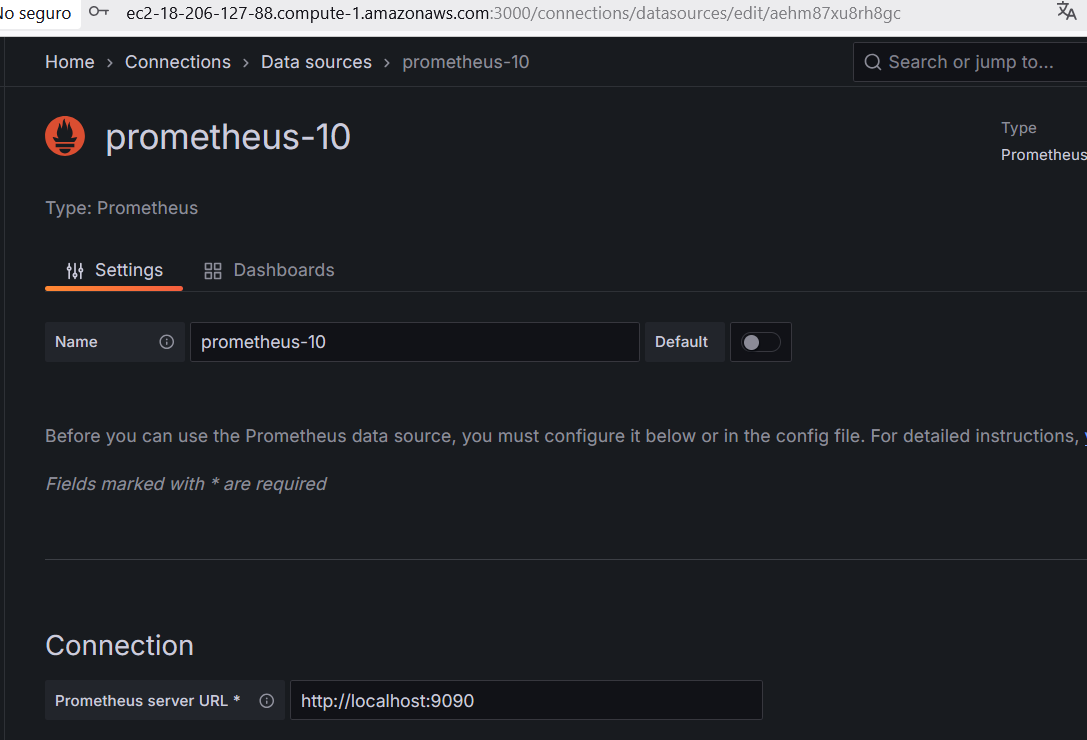
sudo systemctl enable grafana-server

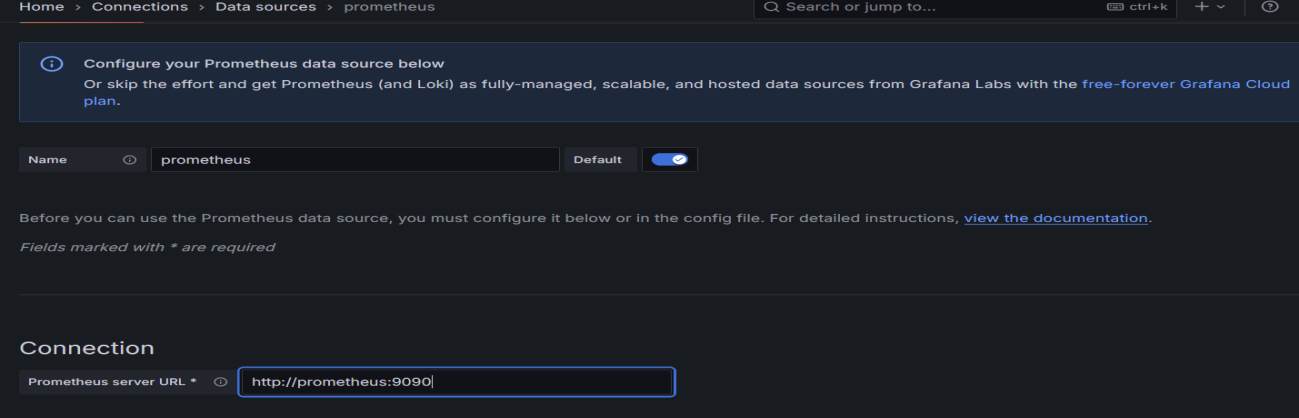


1. **Configurar Grafana**:
   * Accede a Grafana en http://<ip-instancia-ec2>:3000.
   * http://ec2-18-206-127-88.compute-1.amazonaws.com:3000/



* + Agrega Prometheus como fuente de datos:
    - URL: <http://localhost:9090>.





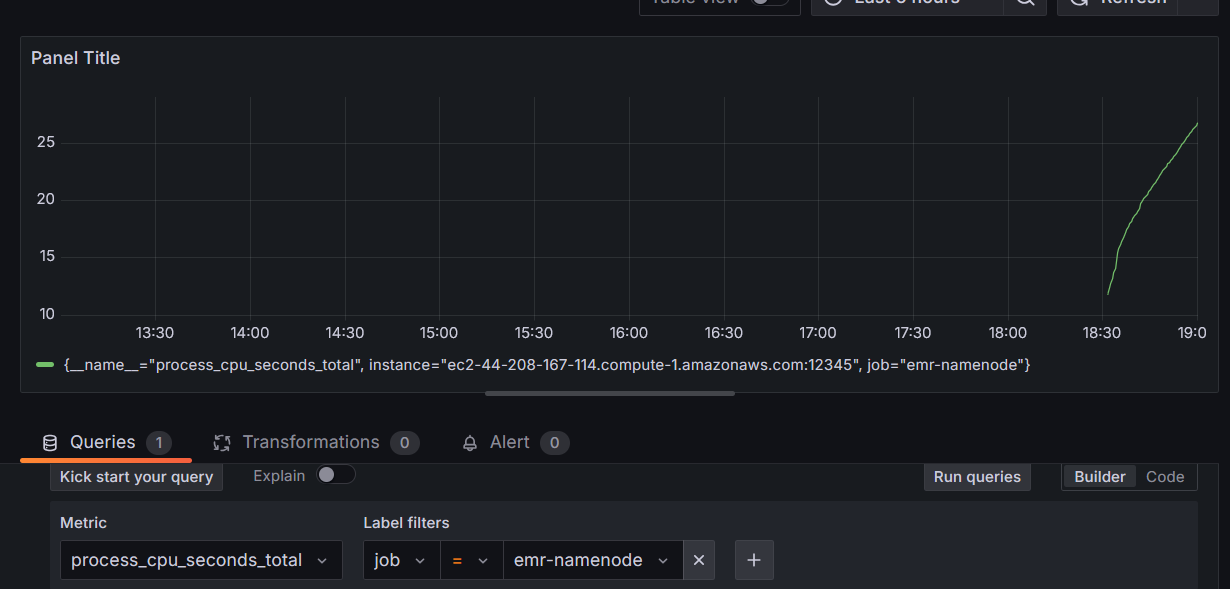
## Parte 4: Visualización de métricas en Grafana

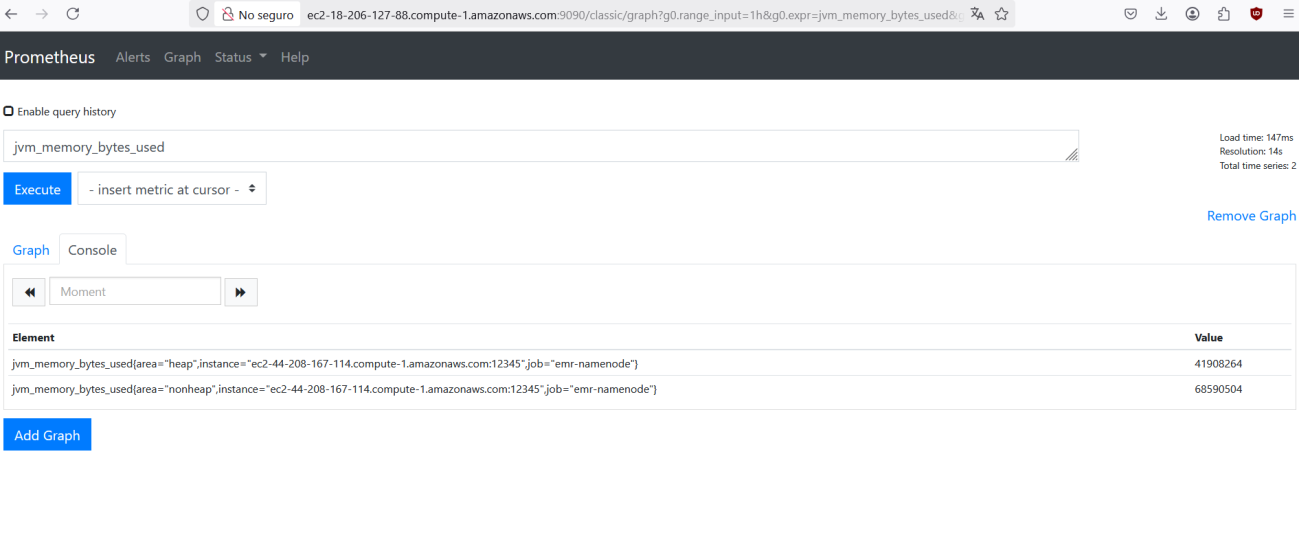
1. **Crear un dashboard en Grafana**:
   * Crea un nuevo dashboard y agrega paneles para monitorear métricas como:
     + Uso de CPU y RAM.
     + Espacio utilizado en HDFS.
     + Estado del Namenode
2. **Explorar métricas**:
   * Usa las métricas expuestas por JMX Exporter para crear gráficos en Grafana.
3. **Explorar métricas**:
   * Usa las métricas expuestas por JMX Exporter para crear gráficos en Grafana.

Voy a mostrar las métricas en prometheus y luego en grafana con sus gráficas

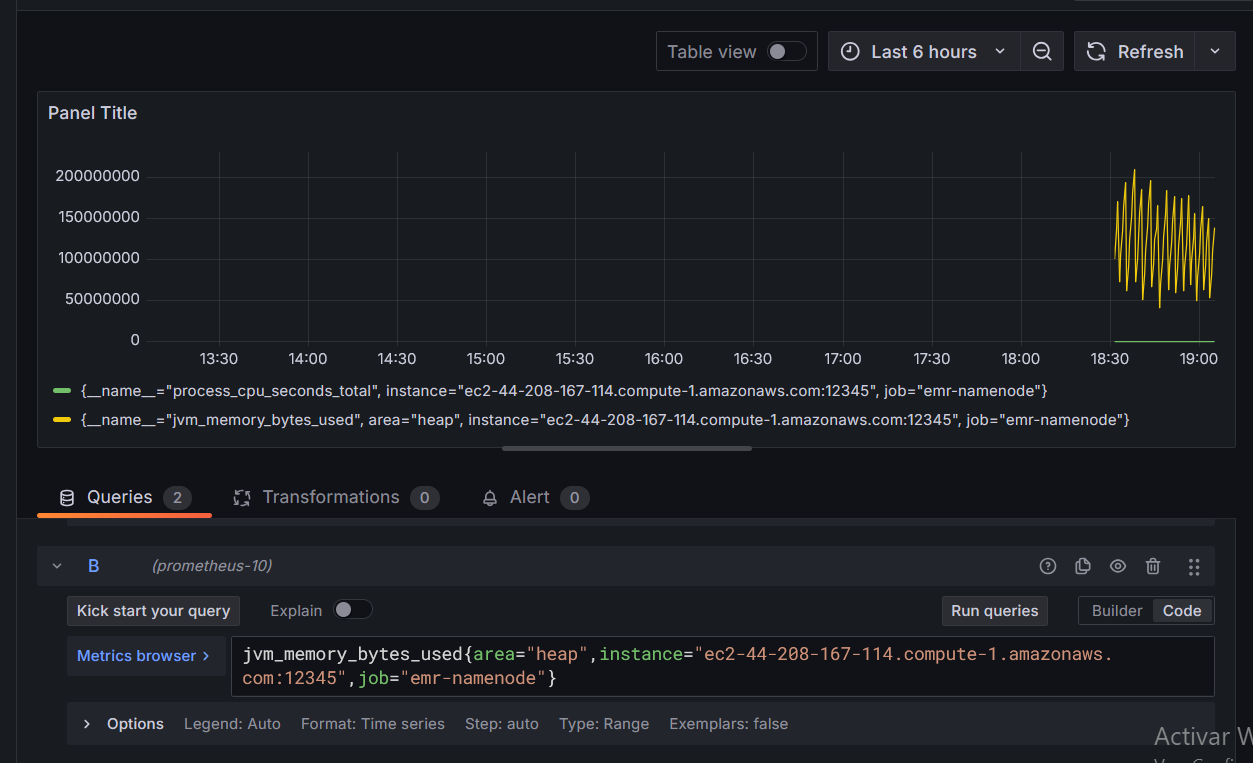
**Uso de CPU y RAM:**



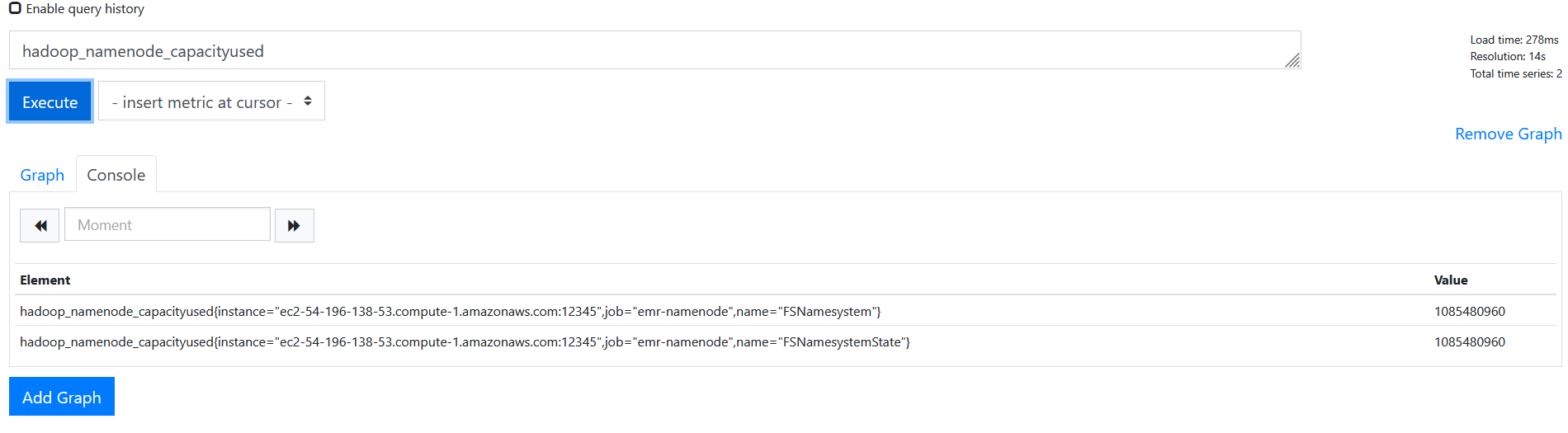


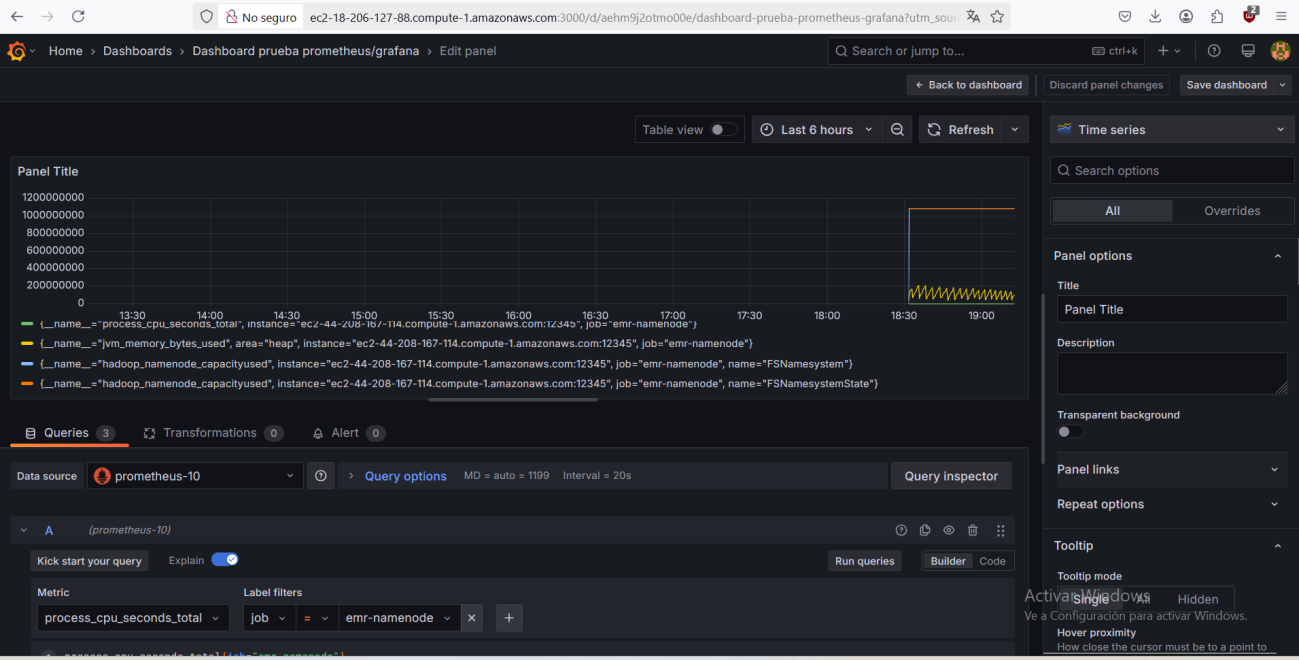


Seria la “heap” y para sacar la grafica en grafana la he buscado en la pestaña code, ya que en builder me sacaba ambas métricas



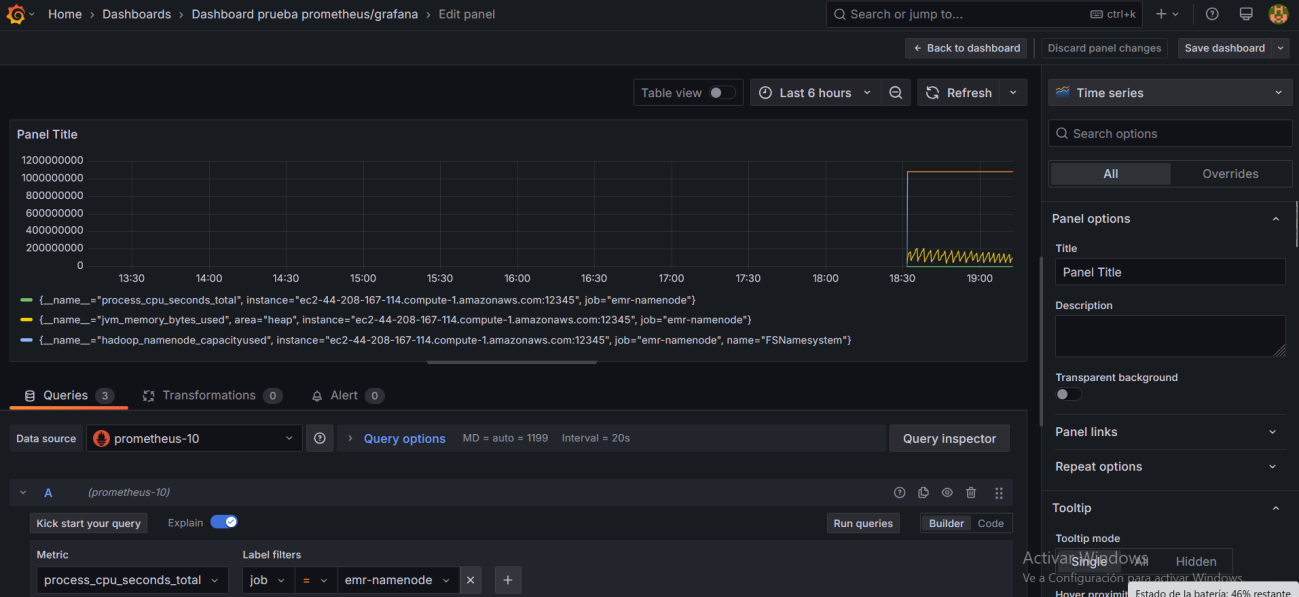
**Espacio utilizado en HDFS:**





Estado del NameNode: (No he encontrado esta)

Dashboard creado con nombre: Dashboard prueba promethetus/grafana:



|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

# **Preguntas para reflexión**

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

En un clúster **Amazon EMR (Elastic MapReduce)**, hay varias métricas clave que se deben monitorear para asegurar el buen funcionamiento del sistema. Algunas de las más importantes son:

* **Uso de CPU**: Es vital monitorear la carga de la CPU de los nodos del clúster para detectar posibles cuellos de botella. Si los nodos están sobrecargados, las tareas pueden tardar más en completarse o incluso fallar.
* **Uso de memoria**: La memoria es crítica para la ejecución de trabajos en un clúster EMR. Monitorear el uso de la memoria ayuda a identificar posibles fugas de memoria o procesos que consumen más memoria de la esperada.
* **Latencia de red**: La latencia entre nodos afecta el rendimiento global del clúster, especialmente en aplicaciones distribuidas. Se debe monitorear tanto la latencia interna del clúster como la latencia hacia otras redes o servicios.
* **Estado de los nodos y máquinas virtuales**: Asegurarse de que los nodos estén en un estado saludable es fundamental para evitar problemas en la ejecución de trabajos. Si un nodo se cae o está inactivo, puede causar fallos en los trabajos.
* **Tasa de fallos de trabajos**: Es importante monitorear los trabajos que fallan o tienen tiempos de ejecución elevados, ya que esto podría indicar problemas en el clúster o en la configuración de los trabajos.
* **Uso de disco**: El almacenamiento en disco es un componente clave en EMR, especialmente cuando se están procesando grandes volúmenes de datos. Monitorear el uso del disco y las operaciones de I/O es esencial para evitar problemas de rendimiento.
* **Métricas de YARN**: Si usas **YARN (Yet Another Resource Negotiator)** como el gestor de recursos en EMR, es importante monitorear métricas relacionadas con los recursos asignados, la cantidad de contenedores en ejecución, la cantidad de contenedores fallidos, etc.

Monitorear estas métricas permite detectar problemas antes de que afecten gravemente el rendimiento o la disponibilidad del clúster

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

**JMX Exporter** es una herramienta muy útil para exportar métricas de Java Management Extensions (JMX) a **Prometheus**. Para mejorar su configuración y recopilar métricas más específicas, puedes hacer lo siguiente:

* **Configurar filtros de métricas**: En la configuración de JMX Exporter, puedes especificar qué métricas deseas exportar utilizando filtros. Esto te permite centrarte solo en las métricas más relevantes para tu caso de uso, reduciendo la sobrecarga de datos innecesarios.
* **Agrupar métricas por categorías**: Puedes agrupar las métricas en diferentes "módulos" según las áreas de monitoreo que te interesen (por ejemplo, rendimiento, recursos, garbage collection, etc.). Esto hace que sea más fácil centrarse en un conjunto de métricas en particular cuando hay un gran volumen de datos.
* **Ajustar el intervalo de recolección**: Puedes ajustar el intervalo de recolección de métricas para que sea más frecuente o menos frecuente, dependiendo de la necesidad de los datos. Para ciertos servicios, puede ser útil tener métricas más detalladas, mientras que en otros, un intervalo mayor podría ser suficiente.
* **Configurar las métricas de JVM**: JMX Exporter puede extraer métricas sobre el estado de la JVM (memoria, hilos, recolección de basura, etc.). Asegúrate de habilitar estas métricas y configurarlas adecuadamente, ya que son esenciales para entender el rendimiento de las aplicaciones que se ejecutan en el clúster EMR.
* **Utilizar etiquetas (labels)**: Las etiquetas o "labels" son útiles para agregar contexto a las métricas exportadas, como el tipo de nodo, la aplicación, la versión, etc. Esto te permite segmentar y filtrar métricas de manera más granular.
* **Activar métricas avanzadas**: Dependiendo de la configuración de tu aplicación, puedes habilitar métricas más avanzadas, como las métricas de memoria nativa, métricas de hilos de ejecución, estadísticas de discos, entre otros. Estas métricas adicionales te proporcionarán información detallada sobre el rendimiento de los servicios

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

**Prometheus y Grafana** son dos herramientas muy populares para la recolección y visualización de métricas. Algunas de las ventajas clave frente a otras herramientas de monitoreo son:

* **Escalabilidad**: Prometheus está diseñado para escalar de manera eficiente y manejar grandes volúmenes de métricas en tiempo real. Esto lo convierte en una excelente opción para clústeres grandes y sistemas distribuidos.
* **Modelo de datos basado en series temporales**: Prometheus usa un modelo de datos de series temporales, lo que lo hace especialmente adecuado para monitorear métricas que cambian a lo largo del tiempo. Además, se puede almacenar de forma eficiente con un bajo costo de almacenamiento.
* **Consulta flexible**: Prometheus tiene su propio lenguaje de consulta, **PromQL**, que permite hacer consultas muy flexibles y complejas. Esto es útil cuando necesitas extraer métricas específicas o realizar análisis detallados.
* **Integración con Grafana**: Grafana es una plataforma de visualización que se integra perfectamente con Prometheus, lo que te permite crear dashboards personalizados para monitorear tus métricas. La integración entre ambas herramientas es fluida y ampliamente soportada.
* **Alertas personalizables**: Prometheus soporta alertas basadas en las métricas recopiladas, y puedes configurarlas fácilmente para que te notifiquen en caso de que alguna métrica alcance un umbral crítico. Esto te permite responder rápidamente ante problemas antes de que impacten gravemente en el rendimiento.
* **Ecosistema y comunidad activa**: Prometheus y Grafana cuentan con un ecosistema grande y una comunidad activa que proporciona integración con muchas otras herramientas y servicios. Esto hace que sea más fácil implementar soluciones avanzadas de monitoreo y solución de problemas.
* **Manejo de métricas distribuidas**: Estas herramientas están diseñadas para trabajar en entornos distribuidos y microservicios, lo que las hace muy adecuadas para entornos de big data como el clúster EMR, donde los servicios están dispersos a través de múltiples nodos.
* **Facilidad de implementación**: Prometheus y Grafana son fáciles de desplegar y configurar, y hay muchas guías y documentación disponible para ayudar en su implementación. Además, son herramientas open-source, lo que permite su personalización y adaptación a necesidades específicas.

En resumen, la combinación de Prometheus y Grafana ofrece una solución poderosa, flexible y escalable para monitorear clústeres grandes y entornos distribuidos como EMR. Su popularidad, integración con otros sistemas, y la comunidad activa que los respalda, son grandes ventajas frente a otras herramientas de monitoreo.

**Github con los archivos de configuración:**

https://github.com/iabd07/Monitoreo-de-un-cl-ster-AWS-EMR-con-Prometheus-y-Grafana/tree/main