Apache Kafka

Informe de Laboratorio de Redes y Sistemas Operativos
1º Cuatrimestre de 2025

Alumnos:	Emails:
Bolaños, Rodrigo	rodrimanuelbw@gmail.com
Salvanescki, Nicolás	salvanescki.unq@gmail.com

Índice

1. Introducción

Apache Kafka es una plataforma distribuida de mensajería, diseñada para manejar flujos de datos en tiempo real de manera eficiente, escalable y sin fallos. Funciona como un sistema de publish-subscribe, donde los productores publican mensajes en tópicos y los consumidores se suscriben para recibirlos.

Se utiliza en la recolección, procesamiento y análisis de grandes volúmenes de datos en tiempo real. Por ejemplo, para logs, métricas, transacciones financieras, seguimiento de usuarios en sitios web o sincronización de bases de datos entre servicios.

Es ampliamente adoptado en arquitecturas orientadas a eventos y sistemas de microservicios, ya que permite desacoplar los componentes del sistema y asegurar una comunicación rápida y confiable entre ellos.

En este trabajo vamos a explicar el paso a paso para instalar y configurar Kafka y Kafka UI. Vamos a dar un ejemplo de uso local y también como hostear en una red privada.

2. Hardware utilizado

2.1. Bolaños

OS: Ubuntu 22.04.4 LTS x86_64
Host: 81W8 Lenovo IdeaPad S145-15IIL
Kernel: 6.8.0-60-generic
Shell: bash 5.1.16
DE: GNOME 42.9
WM: Mutter
Terminal: gnome-terminal
CPU: Intel i5-1035G4 (8) @ 3.700GHz
GPU: Intel Iris Plus Graphics G4
Memory: 3139MiB / 7737MiB

2.2. Salvanescki

OS: Ubuntu 24.04.2 LTS x86_64
Host: VirtualBox 1.2
Kernel: 6.11.0-29-generic
Shell: zsh 5.9
Resolution: 1920x966
WM: i3
Terminal: gnome-terminal
CPU: Intel i5-10400F (4) @ 2.904GHz
GPU: 00:02.0 VMware SVGA II Adapter (NVIDIA GTX 1660 SUPER)
Memory: 3307MiB / 5971MiB

3. Instalación

3.1. Prerrequisitos

Hay que tener instalado Java JDK 17 o mayor. Personalmente recomendamos openjdk, graalvm o eclipse adoptium. Los cuales se pueden instalar en los siguientes links:

- OpenJDK (JDK 24.0.1)
- GraalVM (JDK 24.0.1)
- Adoptium (Temurin JDK 21.0.7+6-LTS)

3.2. Instalación

Hay que descargar aquí el archivo comprimido que contiene kafka. Adicionalmente, en el mismo sitio se listan los archivos necesarios para verificar la integridad del archivo comprimido que nos está proveyendo el mirror.

NOTA: En el momento que escribimos el presente trabajo, la última versión disponible era la 4.0.0. Por ende, los links citados son para dicha versión.
Para descargar una versión más reciente se puede acceder al listado de versiones en esta página.

3.3. Verificación

Aclaración: Esta sección es completamente opcional. Sin embargo, recomendamos verificar la integridad de los archivos por un tema de seguridad.

Acá explicamos paso a paso como se realiza la verificación de integridad del archivo y la firma del mismo.

1. Descargar los archivos con formato .asc y .sha512 correspondientes a la versión aquí.
2. Usando `gpg` calculamos el hash (sha512) con el siguiente comando en terminal:

```
gpg --print-md SHA512 "ruta_al_archivo"
```

3. En windows se puede utilizar `certutil` para hacer lo mismo:

```
certUtil -hashfile "ruta_al_archivo"
```

4. Comparamos contra el archivo .sha512 que descargamos de la página, si coincide significa que el archivo no fue modificado.
5. Para comparar la firma, también usamos `gpg`, pero antes tenemos que bajar las keys de la página oficial. Esto se puede hacer usando `curl`:

```
curl https://downloads.apache.org/kafka/KEYS -O
```

6. Importamos las keys a `gpg`:

```
gpg --import KEYS
```

7. Luego verificamos la firma usando `gpg`:

```
gpg --verify "Ruta al archivo .asc" "Ruta al archivo"
```

Si nos devuelve Good signature from Apache Kafka, es porque la firma del archivo es válida. Sino, el archivo de la ruta ingresada es incorrecto o fue modificado. En este último caso, se recomienda no usar el archivo y volver a descargar de otro enlace/mirror.

3.4. Extracción

Nos movemos al directorio que contiene el archivo comprimido descargado y lo extraemos con `tar`:

```
tar -xzf kafka_2.13-4.0.0.tgz
cd kafka_2.13-4.0.0
```

3.5. Generar un Cluster UUID

```
KAFKA_CLUSTER_ID="$(bin/kafka-storage.sh random-uuid)"
```

3.6. Formatear Directorio de Logs

```
bin/kafka-storage.sh format --standalone -t $KAFKA_CLUSTER_ID -c
config/server.properties
```

3.7. Iniciar Servidor

```
bin/kafka-server-start.sh config/server.properties
```

4. Uso

Antes de escribir y leer eventos, hay que crear al menos un tópico:

4.1. Crear tópico

Vamos a crear un tópico llamado “prueba” en la máquina local, puerto 9092:

```
bin/kafka-topics.sh --create --topic prueba --bootstrap-server
localhost:9092
```

Si se ingresa el comando kafka-topics.sh sin argumentos, el programa muestra una lista de comandos disponibles que pueden ser de utilidad. Por ejemplo, `--list` te muestra una lista de los tópicos creados en el servidor. También tenemos `--describe`, que muestra la información de un tópico en particular, por ejemplo, el que creamos recien:

```
$ bin/kafka-topics.sh --describe --topic prueba --bootstrap-server
localhost:9092

Topic: prueba        TopicId: NPmZHyhbR9y00wMglMH2sg PartitionCount: 1
Topic: prueba Partition: 0    Leader: 0   Replicas: 0 Isr: 0
```

4.2. Escribir eventos

Corremos un cliente productor en la terminal para poder escribir eventos:

```
$ bin/kafka-console-producer.sh --topic prueba --bootstrap-server
localhost:9092
>Evento numero 1
>Evento numero 2
```

4.3. Leer eventos

En otra terminal diferente, corremos un cliente consumidor. Si escribimos el siguiente comando:

```
$ bin/kafka-console-consumer.sh --topic prueba --from-beginning
--bootstrap-server localhost:9092
```

la flag `--from-beginning` nos permite leer eventos que hayan sido escritos antes de suscribirnos/leer. Por ende, al introducir el comando de arriba, la salida por terminal va a ser lo que escribimos como productores, es decir:

```
Evento numero 1
Evento numero 2
```

4.4. Finalizar el entorno kafka

Para hacerlo tenemos que:

- Finalizar todo cliente productor y consumidor con Ctrl+C
- Finalizar el Kafka Broker, también con Ctrl+C
- (Opcional) Si queremos eliminar la data creada durante el proceso:

```
rm -rf /tmp/kafka-logs /tmp/kraft-combined-logs
```

5. Hosteo en red privada

A continuacion, se presenta lo necesario para poder hostear un servidor de Kafka y que dispositivos en la misma red puedan conectarse mediante el cliente Kafka.

Navegamos a la configuracion de Kafka:

```
cd kafka_2.13-4.0.0/config
```

Abrimos el archivo server.properties con un editor:

```
nano server.properties
```

Buscamos las siguientes líneas y cambiamos localhost por nuestra IP en la red

```
# Dirección en la que Kafka escucha internamente
listeners=PLAINTEXT://<TU_IP_PRIVADA>:9092

# Dirección en que Kafka se anuncia a los clientes
advertised.listeners=PLAINTEXT://<TU_IP_PRIVADA>:9092

# Y para evitar conflictos con los hostnames
listener.security.protocol.map=PLAINTEXT:PLAINTEXT
```

Volvemos a realizar los pasos 3.5, 3.6, 3.7

Una vez el servidor este corriendo, podemos crear un topico nuevamente, pero esta vez hosteando en la IP privada.

```
bin/kafka-topics.sh --create --topic <nombre_topico> --bootstrap-server
<TU_IP_PRIVADA>:9092
```

Ya los clientes pueden producir y consumir en el topico creado con los comandos vistos en 4.2 y 4.3, reemplazando localhost por la IP.

6. Kafka UI

Kafka UI es una herramienta Open Source desarrollada por Provectus. Se utiliza para el monitoreo de Clusters, topicos, mensajes, consumidores y productores de Kafka.

Requisitos:

- Java 17 (Ya explicado en 3.1)
- git, el cual puede instalarse simplemente ejecutando:

```
# Debian
sudo apt update
sudo apt install git

# Fedora
sudo dnf install git

# Arch
sudo pacman -S git
```

6.1. Instalación

1. Clonar el repositorio de Kafka UI:

```
git clone https://github.com/provectus/kafka-ui
```

2. Descargar paquete .jar:
3. Previo a la ejecución del jar, debemos crear en /etc/ un directorio llamado kafkaui con el objetivo de otorgar permisos de escritura a Kafka UI. Creamos el directorio con el siguiente comando

```
sudo mkdir /etc/kafkaui
```

navegamos a /etc/

```
cd /etc/
```

y ejecutamos el comando:

```
sudo chmod o+w kafkaui
```

4. Una vez hecho, ya podemos iniciar Kafka UI:

```
java -Ddynamic.config.enabled=true
-Dspring.config.additional-location="path-to-application-local.yml"
-jar "path-to-kafka-ui-jar"
```

Debemos reemplazar las variables ”path-to-application-local.yml”por el yml que se encuentra en el repositorio clonado anteriormente y la variable ”path-to-kafka-ui-jar”por la ubicacion del jar descargado.

```
# Ruta a application-local.yml
.../kafka-ui/kafka-ui-api/src/main/resources/

(Puede que en la ruta al .jar tengas que incluir el nombre del archivo
también, no solo su directorio contenedor)
```

Tambien podemos ejecutar el comando con sudo y ahorrarnos el paso de otorgar permisos al usuario, pero esto no nos parecía buena práctica. En cambio, con esta solución, Kafka UI solo puede escribir en su carpeta propia en /etc/

5. En el navegador, podemos ingresar a localhost:8080 y nos encontraremos lo siguiente:

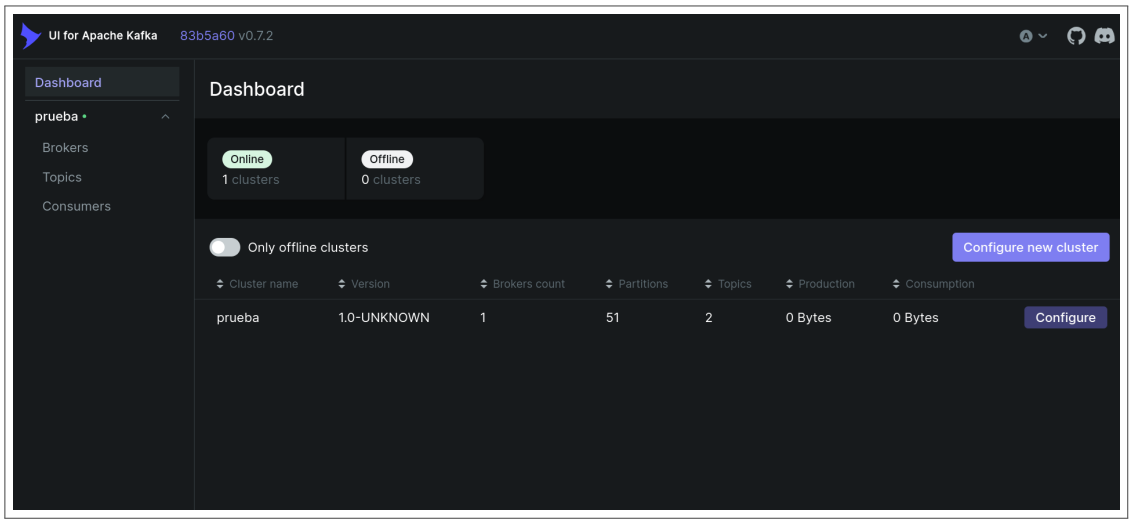

Figura 1: Pagina principal Kafka UI

Al presionar el botón Configure new cluster, podemos mapear un nuevo cluster de Kafka aclarando el Host, su IP y puerto, en nuestro caso localhost:9092.

Una vez mapeado, podemos ver en en panel izquierdo el Cluster y las pestañas de Topics, Brokers y Consumers. Desde la pestaña de Topics, podremos ver nuestro tópico **prueba** con los dos mensajes enviados anteriormente.

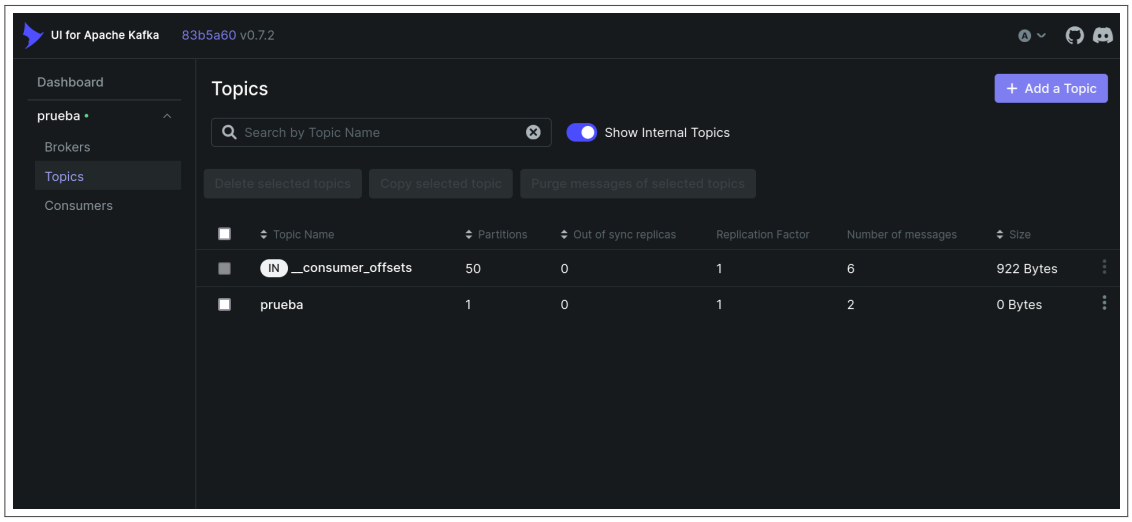

Figura 2: Topicos de Kafka UI

7. Problemas encontrados

7.1. Error java: command not found en Kafka UI

Si al querer ejecutar Kafka UI, sale el siguiente mensaje en la terminal:

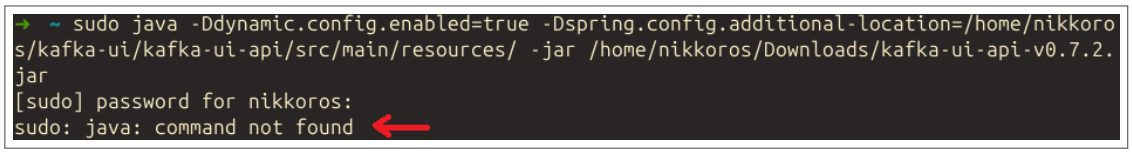

Figura 3: Java not found en Terminal

puede ser porque, o no instalaste java previamente o, como nos pasó a nosotros, pobramos correr Kafka UI con sudo, pero teníamos instalado el jdk a nivel usuario y no a nivel sistema. Por ende, root no sabe a que se refiere el comando java.

7.2. Kafka UI no puede crear o acceder a dynamic_config.yaml

Si al iniciar Kafka UI, sale en terminal el siguiente warning:

Figura 4: Warning dynamic config

Abri el navegador y entra en localhost:8080, o donde esté hosteado Kafka UI. Al ingresar hay dos posibilidades:

- Te muestra la página principal, pero no sale el botón Configure new cluster que aparece en la Figura 1.
- Te redirecciona a http://localhost:8080/ui/clusters/create-new-cluster

En este último caso, deberás probar si te deja crear un cluster. Si al presionar Submit aparecen estos dos mensajes en la página:

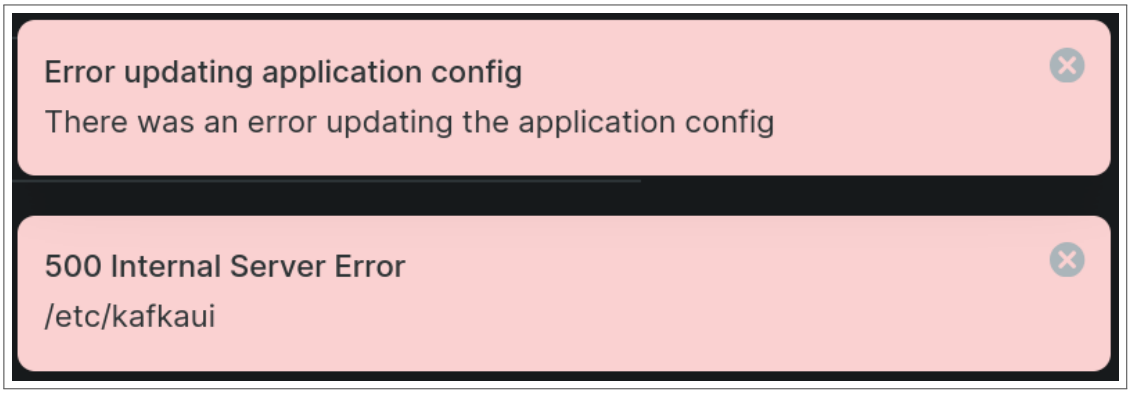

Figura 5: Errores de permisos en Kafka UI

es porque te olvidaste de otorgar permisos de escritura a la carpeta /etc/kafkaui tal como se explica en la subsección 6.1, específicamente en el paso 3.