

# Guía paso a paso: Streaming de eventos para microservicios con Apache Kafka

## Autores

Alejandro Cárdenas Galeano  
Juan Sebastián Matiz Salinas  
Julián Andrés Nuñez Mejía  
Mishell Tatiana Otavo Fernandez

Universidad Autónoma de Occidente  
Cali, Valle del cauca  
2021



## Introducción

En esta guía se implementa una arquitectura en la nube en una máquina virtual de Microsoft Azure utilizando tecnologías como *Apache Kafka*, *Zookeeper*, *Python-Flask* y *MongoDB*. En esta guía se explica el paso a paso para instalar e implementar cada una de estas aplicaciones en una máquina virtual.

# Creación de la máquina virtual en Microsoft Azure

## Paso 1: Creación.


[Inicio](#) > [Máquinas virtuales](#) >

### Crear una máquina virtual ...

para organizar y administrar todos los recursos.

Suscripción *	<div>Azure for Students</div>
Grupo de recursos *	<div>Proyecto_Final_CompuNube</div> <div><a href="#">Crear nuevo</a></div>

#### Detalles de instancia

Nombre de máquina virtual *	<div>MVApacheKafka</div>
Región *	<div>(US) Este de EE. UU.</div>
Opciones de disponibilidad	<div>No se requiere redundancia de la infraestructura</div>
Tipo de seguridad	<div>Estándar</div>
Imagen *	<div> Ubuntu Server 20.04 LTS - Gen1</div> <div><a href="#">Ver todas las imágenes</a>   <a href="#">Configurar la generación de máquinas virtuales</a></div>
Tamaño *	<div>Standard_DS1_v2 - 1 vcpu, 3.5 GiB de memoria (USD 53.29/mes)</div> <div><a href="#">Ver todos los tamaños</a></div>

[Revisar y crear](#)

[< Anterior](#)

[Siguiente: Discos >](#)

Paso 2: Escoger tipo de autenticación. En este caso Usuario/Contraseña.

## Crear una máquina virtual ...

Tipo de autenticación ⓘ	<input type="radio"/> Clave pública SSH <input checked="" type="radio"/> Contraseña
Nombre de usuario * ⓘ	<input type="text" value="azureuser"/> ✓
Contraseña * ⓘ	<input type="password" value="....."/> ✓
Confirmar contraseña * ⓘ	<input type="password" value="....."/> ✓

### Reglas de puerto de entrada

Seleccione los puertos de red de máquina virtual que son accesibles desde la red Internet pública. Puede especificar acceso de red más limitado o granular en la pestaña Red.

Puertos de entrada públicos * ⓘ	<input type="radio"/> Ninguno <input checked="" type="radio"/> Permitir los puertos seleccionados
Seleccionar puertos de entrada *	<input type="text" value="SSH (22)"/> ▼

**⚠ Esto permitirá que todas las direcciones IP accedan a la máquina virtual.**  
Esto solo se recomienda para las pruebas. Use los controles avanzados de la pestaña Redes a fin de crear reglas para limitar el tráfico entrante a las




Revisar y crear

< Anterior

Siguiente: Discos >

### Paso 3: Revisar, crear y visualizar.

[Inicio](#) > [Máquinas virtuales](#) >

 **MVApacheKafka**  

Máquina virtual

Información general

Registro de actividad

Control de acceso (IAM)

Etiquetas







Diagnosticar y solucionar problemas

Configuración

Redes

Conectar

Discos

 Conectar  Iniciar  Reiniciar  Detener  Captura 

Información esencial

Grupo de recur... (cambiar) : [Proyecto\\_Final\\_CompuNube](#)

Estado : En ejecución

Ubicación : Este de EE. UU.

Suscripción (cambiar) : [Azure for Students](#)

Id. de suscripción : f377ce50-b79a-40be-95ca-a612707f5a65


Etiquetas (cambiar) : [Haga clic aquí para agregar etiquetas.](#)

Propiedades

Supervisión

Funcionalidades (7)

Recomendador

 **Máquina virtual**

Nombre del equipo

MVApacheKafka

## Instalación de los servicios requeridos

Para ingresar a la máquina virtual es necesario utilizar el protocolo SSH con el usuario y contraseña creado en pasos anteriores. Además, es necesario tener en cuenta la IP pública de la máquina virtual.

```
ssh <USER>@<PUBLIC_IP>
```

Tras ingresar a la máquina virtual, es necesario entonces instalar todos los servicios y módulos necesarios.

### Zookeeper y Apache Kafka

#### Paso 1: Instalar JDK

Iniciar sesión como administrador (ROOT)

```
sudo -i
```

```
apt-get update
```

```
apt-get install default-jdk
```

#### Paso 2: Verificar instalación

```
update-alternatives --config java
```

Debe obtener una respuesta como la mostrada a continuación:

```
azureuser@MVApacheKafka:~$ update-alternatives --config java
There is only one alternative in link group java (providing /usr/bin/java): /usr/lib/jvm/java-11-openjdk-amd64/bin/java
Nothing to configure.
```

#### Paso 3: Exportar el path de Java a las variables de entorno

Crear un archivo .sh en la ruta **/etc/profile.d/** para automatizar la creación de las variables de entorno necesarias.

```
vi /etc/profile.d/java.sh
```

Este archivo debe contener el siguiente contenido:

```
#!/bin/bash
```

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
```

```
azureuser@MVApacheKafka:~$ sudo vi /etc/profile.d/java.sh
```

```
#!/bin/bash  
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
```

Para ejecutar los cambios es necesario reiniciar, por lo que se debe hacer lo siguiente:

**reboot**

```
azureuser@MVApacheKafka:~$ sudo reboot
```

Tras completar esto, ingresar de nuevo como usuario administrador

**sudo -i**

Para cerciorarse de que se ha configurado correctamente, ejecutar el siguiente comando:

**env | grep JAVA\_HOME**

Al ejecutar dicho comando, debe obtener la siguiente respuesta:

```
azureuser@MVApacheKafka:~$ env | grep JAVA_HOME  
JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
```

Paso 4: Instalar Zookeeper

**apt-get install zookeeperd**  
**systemctl enable zookeeper**

```
azureuser@MVApacheKafka:~$ sudo systemctl enable zookeeper  
zookeeper.service is not a native service, redirecting to systemd-sysv-install.  
Executing: /lib/systemd/systemd-sysv-install enable zookeeper
```

Para asegurarse de que toda se ha instalado correctamente, ejecutar el siguiente comando:

**systemctl status zookeeper**

Paso 5: Instalar Apache Kafka

Crear carpeta en la cual almacenar el fichero de Kafka a descargar

**cd /**  
**mkdir descargas**  
**cd descargas**

**sudo wget [http://mirror.nbtelecom.com.br/apache/kafka/2.8.1/kafka\\_2.12-2.8.1.tgz](http://mirror.nbtelecom.com.br/apache/kafka/2.8.1/kafka_2.12-2.8.1.tgz)**

```
azureuser@MVApacheKafka:~/downloads$ sudo wget http://mirror.nbtelecom.com.br/ap
ache/kafka/2.8.1/kafka_2.12-2.8.1.tgz
--2021-11-02 20:24:48-- http://mirror.nbtelecom.com.br/apache/kafka/2.8.1/kafka
_2.12-2.8.1.tgz
Resolving mirror.nbtelecom.com.br (mirror.nbtelecom.com.br)... 189.45.5.90
Connecting to mirror.nbtelecom.com.br (mirror.nbtelecom.com.br)|189.45.5.90|:80.
.. connected.
HTTP request sent, awaiting response... 200 OK
Length: 71578705 (68M) [application/x-gzip]
Saving to: 'kafka_2.12-2.8.1.tgz'

kafka_2.12-2.8.1.tg 100%[=====>] 68.26M 15.3MB/s in 4.7s

2021-11-02 20:24:53 (14.6 MB/s) - 'kafka_2.12-2.8.1.tgz' saved [71578705/7157870
5]
```

Tras descargar el fichero, es necesario crear un directorio donde se guarden todos los ejecutables de Apache Kafka y descomprimir el archivo descargado anteriormente en dicho directorio.

**sudo mkdir /opt/kafka**

**sudo tar -zxvf kafka\_2.12-2.8.1.tgz -C /opt/kafka**

```
kafka_2.12-2.8.1/libs/connect-file-2.8.1.jar
kafka_2.12-2.8.1/libs/connect-basic-auth-extension-2.8.1.jar
kafka_2.12-2.8.1/libs/connect-mirror-2.8.1.jar
kafka_2.12-2.8.1/libs/connect-mirror-client-2.8.1.jar
kafka_2.12-2.8.1/libs/kafka-streams-2.8.1.jar
kafka_2.12-2.8.1/libs/rocksdbjni-5.18.4.jar
kafka_2.12-2.8.1/libs/kafka-streams-scala_2.12-2.8.1.jar
kafka_2.12-2.8.1/libs/kafka-streams-test-utils-2.8.1.jar
kafka_2.12-2.8.1/libs/kafka-streams-examples-2.8.1.jar
```

Ahora, es necesario crear un archivo .sh en la ruta **/etc/profile.d/** para automatizar la creación de las variables de entorno necesarias para Apache Kafka.

**sudo vi /etc/profile.d/kafka.sh**

```
kafka_2.12-2.8.1/libs/kafka-streams-examples-2.8.1.jar
azureuser@MVApacheKafka:~/downloads$ sudo vi /etc/profile.d/kafka.sh
```

El archivo creado debe contener el siguiente contenido:

**#!/bin/bash**

**export KAFKA\_HOME=/opt/kafka/kafka\_2.12-2.8.1**

**export PATH="\$PATH:\${KAFKA\_HOME}/bin"**

Para ejecutar los cambios es necesario reiniciar, por lo que se debe hacer lo siguiente:

**reboot**

Para corroborar la creación de la variable de entorno, se utiliza el siguiente comando.

**env | grep -i kafka**



Obtendrá una respuesta como la siguiente:

```
azureuser@MVApacheKafka:~$ env | grep -i kafka
KAFKA_HOME=/opt/kafka/kafka_2.12-2.8.1
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/opt/kafka/kafka_2.12-2.8.1/bin
azureuser@MVApacheKafka:~$
```

Creamos un vínculo simbólico para facilitar la búsqueda del archivo de configuración de Kafka:

**sudo ln -s /opt/kafka/kafka\_2.12-2.8.1/config/server.properties /etc/kafka.properties**

```
azureuser@MVApacheKafka:~$ sudo ln -s /opt/kafka/kafka_2.12-2.8.1/config/server.properties /etc/kafka.properties
azureuser@MVApacheKafka:~$
```

Para probar que Apache Kafka funciona correctamente, ejecutar como administrador (ROOT) el siguiente comando:

**kafka-server-start.sh /etc/kafka.properties**

Tras corroborar que todo está correcto, salir usando CTRL + C en el teclado.

Ahora es necesario modificar el archivo de configuración del servidor de Apache Kafka para permitir que se acceda a este desde fuentes externas.

**vim /etc/kafka.properties**

Digite, las siguientes líneas debajo de broker.id=0, colocando el host.name en 0.0.0.0 y el advertised.host.name con la IP pública de la máquina virtual

**host.name = 0.0.0.0**

**advertised.host.name = IP PÚBLICA**

En este punto, kafka no se ejecuta como servicio sino como un Script, por lo que es necesario hacer los siguientes pasos.

Cree un servicio para iniciar kafka al realizar el boot de la máquina.

**vim /etc/systemd/system/kafka.service**

Dentro de este fichero, colocar el siguiente contenido:

```
[Unit]
Description=kafka service
Requires=zookeeper
After=zookeeper
[Service]

ExecStart=/opt/kafka/kafka_2.12-2.8.1/bin/kafka-server-start.sh /etc/kafka.properties
Restart=on-abnormal
ExecStop=/opt/kafka/kafka_2.12-2.8.1/bin/kafka-server-stop.sh
Environment=JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
[Install]
WantedBy=multi-user.target
```

Reinicie los demonios de los servicios e inicie kafka:

```
systemctl daemon-reload
systemctl start kafka.service
```

Para corroborar que todo funciona correctamente, debe ejecutar el siguiente comando y observar que el servicio esté activo:

```
systemctl status kafka.service
```

Comandos útiles en apache kafka:

1. Crear tópico

```
kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1
--topic Prueba
```

2. Ser consumidor de un tópico

```
kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic Prueba
--from-beginning
```

3. Ser producto de un tópico

```
kafka-console-producer.sh --bootstrap-server localhost:9092 --topic Prueba
```

## MongoDB

```
apt install mongodb
```

## Python

Python es instalado con ubuntu en sus versiones más actuales, pero **pip** no, por lo que se debe instalar este paquete.

```
apt install python3-pip
```

Instalar los siguientes módulos para ejecutar todas las aplicaciones correctamente:

```
pip install Flask
```

```
pip install Flask-PyMongo
```

```
pip install kafka-python
```

```
pip install PyMongo
```

Para ejecutar las aplicaciones clonar el siguiente repositorio de github.

```
apt install git
```

```
git clone https://github.com/alejandrocardenasg/proyecto\_compu\_nube
```

Dentro de este, se deben correr los siguientes Scripts:

1. Aplicación de consumidor de Apache Kafka en el tópico **Prueba**.

```
python3 API.py
```

2. Aplicación web para visualizar los datos a través del puerto **7000**.

```
python3 flask/app.py
```

Gracias por seguir esta guía paso a paso.

