BIG DATA APLICADO

PROYECTO INTEGRAL BIG DATA 2025–2026

Instalación y Configuración de

Apache Kafka 2.8.1

Integración con Cluster Ambari HDP

Realizado por:

Rubén Jiménez Lozano

Curso Académico 2025–2026

Febrero 2026

Índice

1. Introducción

Este documento detalla la instalación de **Apache Kafka 2.8.1** en CentOS 7 (nodo1) y su integración con el cluster Ambari HDP del proyecto MegaSfree.

1.1. Por qué Kafka 2.8.1 y no 3.9.0

La decisión de usar Kafka 2.8.1 en lugar de la versión más reciente (3.9.0) se debe a:

- **Compatibilidad con HDP:** El cluster Ambari utiliza Kafka 2.8.1, parte de la distribución Hortonworks Data Platform (HDP).
- **Protocolo de comunicación:** Kafka 3.9.0 utiliza versiones de API incompatibles con brokers 2.8.1. Al intentar conectar ambas versiones se producen errores del tipo:

```
org.apache.kafka.common.errors.InvalidRequestException:
Error getting request for apiKey: UPDATE_METADATA,
apiVersion: 8
```

- **Estabilidad empresarial:** HDP está certificado con Kafka 2.8.1 para entornos de producción.
- **Integración con Zookeeper:** Versiones posteriores a 3.0 están migrando a KRaft (arquitectura sin Zookeeper), pero nuestro cluster Ambari usa Zookeeper como coordinador.
- **Compatibilidad de clientes:** Los clientes Kafka 3.9.0 pueden comunicarse con brokers 2.8.1, pero no pueden actuar como brokers adicionales en el cluster.

1.2. Arquitectura del Proyecto

Flujo de datos en el ecosistema MegaSfree:

```
VM NFC (Eventos) -> Kafka (nodo1:9092) -> Consumidores
                              |
                    Zookeeper Cluster
                  (172.16.200.10:2181)
                              |
                    +---------+--------+
                    |                  |
                   n8n           Spark Streaming
                    |                  |
            PostgreSQL + Redis    Analytics
```

Componentes del ecosistema:

- **Productor:** VM externa con lector NFC (Python)
- **Kafka Broker nodo1:** 172.16.200.28:9092 (broker.id=10)
- **Kafka Broker cluster:** 172.16.200.10:9092 (broker.id=1001)
- **Zookeeper Cluster:** 172.16.200.10:2181, 172.16.200.11:2181, 172.16.200.12:2181
- **Consumidores:** n8n, Spark Streaming, servicios de Analytics
- **Almacenamiento:** PostgreSQL (persistencia) + Redis (caché)

2. Instalación en nodo1 (CentOS 7)

2.1. Paso 1: Instalación de Java

Kafka requiere Java 8 como mínimo.

```
# Actualizar sistema
sudo yum update -y

# Instalar Java OpenJDK 1.8
sudo yum install java-1.8.0-openjdk \
  java-1.8.0-openjdk-devel -y

# Verificar instalacion
java -version
```

Salida esperada:

```
openjdk version "1.8.0_412"
OpenJDK Runtime Environment (build 1.8.0_412-b08)
OpenJDK 64-Bit Server VM (build 25.412-b08, mixed mode)
```

Figura 1: Verificación de la instalación de Java OpenJDK 1.8

2.2. Paso 2: Descargar Kafka 2.8.1

Debido a que nodo1 no tiene conexión directa a Internet, descargamos desde ambari10 y transferimos por SCP.

```
# En ambari10
cd /tmp
wget https://archive.apache.org/dist/kafka/2.8.1/\
kafka_2.12-2.8.1.tgz

# Transferir a nodo1
scp kafka_2.12-2.8.1.tgz root@172.16.200.28:/tmp/
```

Nota: Usamos el repositorio `archive.apache.org` porque Kafka 2.8.1 es una versión antigua no disponible en los mirrors principales.

2.3. Paso 3: Instalación y Configuración de Permisos

```
# En nodo1 como root
cd /tmp
tar -xzf kafka_2.12-2.8.1.tgz
mv kafka_2.12-2.8.1 /opt/kafka-2.8.1

# Asignar permisos al usuario hadoop
chown -R hadoop:hadoop /opt/kafka-2.8.1

# Crear directorio de logs
mkdir -p /opt/kafka-2.8.1/kafka-logs
chown -R hadoop:hadoop /opt/kafka-2.8.1/kafka-logs

# Verificar estructura
ls -l /opt/kafka-2.8.1
```

Directorios principales:

- `bin/` - Scripts ejecutables de Kafka
- `config/` - Archivos de configuración
- `libs/` - Librerías Java de Kafka
- `kafka-logs/` - Almacenamiento de datos

```
2026-01-26 18:50:17 ERROR 404: Not Found.

[root@nodo1 opt]# sudo wget https://downloads.apache.org/kafka/3.9.0/kafka_2.13-
3.9.0.tgz
--2026-01-26 18:52:27--  https://downloads.apache.org/kafka/3.9.0/kafka_2.13-3.9
.0.tgz
Resolviendo downloads.apache.org (downloads.apache.org)... 88.99.208.237, 135.18
1.214.104, 2a01:4f8:10a:39da::2, ...
Conectando con downloads.apache.org (downloads.apache.org)[88.99.208.237]:443...
 conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 122037770 (116M) [application/x-gzip]
Grabando a: “kafka_2.13-3.9.0.tgz”

100%[=======================================>] 122.037.770 63,3MB/s   en 1,8s

2026-01-26 18:52:29 (63,3 MB/s) - “kafka_2.13-3.9.0.tgz” guardado [122037770/122
037770]

[root@nodo1 opt]# sudo tar -xzf kafka_2.13-3.9.0.tgz
[root@nodo1 opt]# sudo mv kafka_2.13-3.9.0 kafka
[root@nodo1 opt]# sudo mkdir -p /var/kafka/logs
[root@nodo1 opt]# sudo chown -R hadoop:hadoop /var/kafka
[root@nodo1 opt]# sudo chown -R hadoop:hadoop /opt/kafka
[root@nodo1 opt]# ls -l /opt/kafka
total 68
drwxr-xr-x. 3 hadoop hadoop  4096 oct 26  2024 bin
drwxr-xr-x. 3 hadoop hadoop  4096 oct 26  2024 config
drwxr-xr-x. 2 hadoop hadoop  8192 ene 26 18:52 libs
-rw-r--r--. 1 hadoop hadoop 15243 oct 26  2024 LICENSE
drwxr-xr-x. 2 hadoop hadoop  4096 oct 26  2024 licenses
-rw-r--r--. 1 hadoop hadoop 28359 oct 26  2024 NOTICE
drwxr-xr-x. 2 hadoop hadoop    44 oct 26  2024 site-docs
[root@nodo1 opt]#
```

Figura 2: Estructura de directorios de Kafka 2.8.1

3. Configuración de Kafka

3.1. Configurar server.properties

El archivo `server.properties` contiene toda la configuración del broker Kafka.

```
cd /opt/kafka-2.8.1
nano config/server.properties
```

Configuración completa aplicada:

```
# ID unico del broker
broker.id=10

# Listeners - IP especifica de nodo1
listeners=PLAINTEXT://172.16.200.28:9092
advertised.listeners=PLAINTEXT://172.16.200.28:9092

# Conexion a Zookeeper del cluster Ambari
zookeeper.connect=172.16.200.10:2181

# Directorio de almacenamiento de logs
log.dirs=/opt/kafka-2.8.1/kafka-logs
```

```
13
14 # Configuracion de red
15 num.network.threads=3
16 num.io.threads=8
17 socket.send.buffer.bytes=102400
18 socket.receive.buffer.bytes=102400
19 socket.request.max.bytes=104857600
20
21 # Configuracion de particiones
22 num.partitions=1
23 num.recovery.threads.per.data.dir=1
24
25 # Replicacion
26 offsets.topic.replication.factor=1
27 transaction.state.log.replication.factor=1
28 transaction.state.log.min.isr=1
29
30 # Retencion de logs
31 log.retention.hours=168
32 log.segment.bytes=1073741824
33 log.retention.check.interval.ms=300000
34
35 # Timeouts
36 zookeeper.connection.timeout.ms=18000
37 group.initial.rebalance.delay.ms=0
```

Parámetros críticos:

- `broker.id=10`: Identificador único del broker
- `listeners`: Dirección IP y puerto donde escucha
- `zookeeper.connect`: Conexión al Zookeeper del cluster
- `log.dirs`: Directorio de almacenamiento de mensajes
- `log.retention.hours=168`: Retención de 7 días

3.2. Configurar Firewall

```
1 # Abrir puerto 9092 para Kafka
2 sudo firewall-cmd --permanent --add-port=9092/tcp
3 sudo firewall-cmd --reload
4 sudo firewall-cmd --list-ports
```

```
[root@nodo1 opt]# sudo nano /etc/systemd/system/kafka.service
[root@nodo1 opt]# sudo firewall-cmd --permanent --add-port=9092/tcp
success
[root@nodo1 opt]# sudo firewall-cmd --reload
success
[root@nodo1 opt]# sudo firewall-cmd --list-ports
9092/tcp
[root@nodo1 opt]#
```

Figura 3: Configuración del firewall - puerto 9092

4. Servicio systemd para Kafka

4.1. Crear archivo de servicio

```
sudo nano /etc/systemd/system/kafka.service
```

Contenido del archivo:

```
[Unit]
Description=Apache Kafka Server
Documentation=http://kafka.apache.org/documentation.html
Requires=network.target
After=network.target

[Service]
Type=simple
User=hadoop
Environment="JAVA_HOME=/usr/lib/jvm/jre-1.8.0-openjdk"
ExecStart=/opt/kafka-2.8.1/bin/kafka-server-start.sh \
/opt/kafka-2.8.1/config/server.properties
ExecStop=/opt/kafka-2.8.1/bin/kafka-server-stop.sh
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

4.2. Habilitar e iniciar el servicio

```
# Recargar configuracion de systemd
sudo systemctl daemon-reload

# Habilitar inicio automatico
sudo systemctl enable kafka

# Iniciar Kafka
sudo systemctl start kafka

# Verificar estado
```

```
sudo systemctl status kafka
```

```
[root@nodo1 opt]# sudo systemctl status kafka
● kafka.service - Apache Kafka Server
   Loaded: loaded (/etc/systemd/system/kafka.service; enabled; vendor preset: disabled)
   Active: active (running) since lun 2026-01-26 19:20:57 CET; 7s ago
     Docs: http://kafka.apache.org/documentation.html
 Main PID: 28897 (java)
    Tasks: 43
   CGroup: /system.slice/kafka.service
           └─28897 /usr/lib/jvm/jre-1.8.0-openjdk/bin/java -Xmx1G -Xms1G -server -XX:+UseG1GC ...

ene 26 19:20:58 nodo1 kafka-server-start.sh[28897]: [2026-01-26 19:20:58,085] INFO zookeeper...n)
ene 26 19:20:58 nodo1 kafka-server-start.sh[28897]: [2026-01-26 19:20:58,086] INFO [ZooKeepe...t)
ene 26 19:20:58 nodo1 kafka-server-start.sh[28897]: [2026-01-26 19:20:58,086] INFO Opening s...n)
ene 26 19:21:01 nodo1 kafka-server-start.sh[28897]: [2026-01-26 19:21:01,225] WARN Session 0...n)
ene 26 19:21:01 nodo1 kafka-server-start.sh[28897]: java.net.NoRouteToHostException: No exis...t'
ene 26 19:21:01 nodo1 kafka-server-start.sh[28897]: at sun.nio.ch.SocketChannelImpl.checkCon...d)
ene 26 19:21:01 nodo1 kafka-server-start.sh[28897]: at sun.nio.ch.SocketChannelImpl.finishCo...6)
ene 26 19:21:01 nodo1 kafka-server-start.sh[28897]: at org.apache.zookeeper.ClientCnxnSocket...4)
ene 26 19:21:01 nodo1 kafka-server-start.sh[28897]: at org.apache.zookeeper.ClientCnxn$SendT...9)
ene 26 19:21:02 nodo1 kafka-server-start.sh[28897]: [2026-01-26 19:21:02,334] INFO Opening s...n)
Hint: Some lines were ellipsized, use -l to show in full.
[root@nodo1 opt]#
```

Figura 4: Estado del servicio Kafka activo

4.3. Verificar logs de arranque

```
tail -50 /opt/kafka-2.8.1/logs/server.log
```

Líneas clave:

```
INFO Registered broker 10 at path /brokers/ids/10
INFO Kafka version: 2.8.1
INFO [KafkaServer id=10] started
```

5. Configuración del Cluster Ambari

5.1. Problema: Kafka escuchando en IPv6

Al verificar con netstat en ambari10:

```
netstat -tuln | grep 9092
# Salida: tcp6  :::9092  :::*  LISTEN
```

Problema: El broker escuchaba en IPv6, pero nodo1 utiliza IPv4.

5.2. Modificar Listeners en Ambari

Acceder a Ambari Web UI (`http://172.16.200.10:8080`) y navegar a: Servicios → Kafka → Configs.

Modificar:

```
# Antes:
listeners=PLAINTEXT://localhost:9092

# Despues:
listeners=PLAINTEXT://0.0.0.0:9092
advertised.listeners=PLAINTEXT://172.16.200.10:9092
```

5.3. Forzar IPv4 en Java

En **Advanced kafka-env**:

```
#!/bin/bash
export JAVA_HOME={{java64_home}}
export PATH=$PATH:$JAVA_HOME/bin
export PID_DIR={{kafka_pid_dir}}
export LOG_DIR={{kafka_log_dir}}

{% if kerberos_security_enabled or kafka_other_sasl_enabled %}
export KAFKA_OPTS="-Djava.net.preferIPv4Stack=true \
-Djavax.security.auth.useSubjectCredsOnly=false \
{{kafka_kerberos_params}}"
{% else %}
export KAFKA_OPTS="-Djava.net.preferIPv4Stack=true \
{{kafka_kerberos_params}}"
{% endif %}
```

Guardar, confirmar y reiniciar servicios afectados.

Verificación:

```
netstat -tuln | grep 9092
# Salida: tcp  0.0.0.0:9092  0.0.0.0:*  LISTEN
```

6. Verificación y Pruebas

6.1. Crear Topic de Prueba

```
cd /opt/kafka-2.8.1

bin/kafka-topics.sh --create --topic test-productor \
  --bootstrap-server 172.16.200.28:9092 \
  --partitions 1 --replication-factor 1

bin/kafka-topics.sh --list \
  --bootstrap-server 172.16.200.28:9092

bin/kafka-topics.sh --describe --topic test-productor \
  --bootstrap-server 172.16.200.28:9092
```

Salida:

```
Created topic test-productor.
Topics: ambari_kafka_service_check, test-productor
Leader: 10  Replicas: 10  Isr: 10
```

6.2. Verificar desde ambari10

```
ssh hadoop@172.16.200.10
/usr/bin/kafka-topics.sh --list \
  --bootstrap-server 172.16.200.10:9092
```

Ambos brokers ven el mismo topic (comparten Zookeeper).

7. Comandos Útiles

7.1. Gestión del Servicio

```
sudo systemctl status kafka
sudo systemctl start kafka
sudo systemctl stop kafka
sudo systemctl restart kafka
sudo journalctl -u kafka -f
```

7.2. Gestión de Topics

```
# Listar
bin/kafka-topics.sh --list \
  --bootstrap-server 172.16.200.28:9092

# Crear
bin/kafka-topics.sh --create --topic NOMBRE \
  --bootstrap-server 172.16.200.28:9092 \
  --partitions 3 --replication-factor 1

# Eliminar
bin/kafka-topics.sh --delete --topic NOMBRE \
  --bootstrap-server 172.16.200.28:9092
```

7.3. Productores y Consumidores

```
# Productor
bin/kafka-console-producer.sh \
  --broker-list 172.16.200.28:9092 \
```

```
  --topic test-productor

# Consumidor
bin/kafka-console-consumer.sh \
  --bootstrap-server 172.16.200.28:9092 \
  --topic test-productor --from-beginning
```

8. Solución de Problemas

8.1. Kafka no arranca

Diagnóstico:

```
sudo journalctl -u kafka -n 100 --no-pager
tail -100 /opt/kafka-2.8.1/logs/server.log
```

Causas comunes:

- Puerto ocupado: `sudo lsof -ti:9092 | xargs kill -9`
- Java no encontrado: `java -version`
- Zookeeper inaccesible: `nc -zv 172.16.200.10 2181`
- Permisos: `sudo chown -R hadoop:hadoop /opt/kafka-2.8.1`

8.2. Timeout al listar topics

```
sudo firewall-cmd --list-ports
telnet 172.16.200.10 9092
netstat -tuln | grep 9092
```

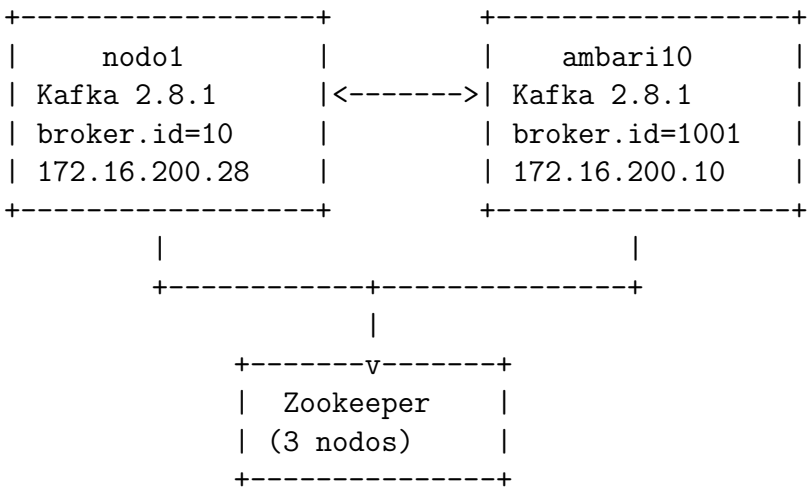
9. Conclusión

Se ha instalado correctamente Apache Kafka 2.8.1 en nodo1, integrado con el cluster Ambari HDP.

9.1. Logros

- Kafka como servicio systemd con inicio automático
- Integración con Zookeeper del cluster Ambari
- Comunicación bidireccional nodo1-ambari10
- Topics compartidos entre brokers
- IPv4 configurado correctamente

9.2. Arquitectura Final



9.3. Próximos Pasos

1. Productor Python para eventos NFC
2. Consumidor n8n
3. Integración PostgreSQL + Redis
4. Procesamiento Spark Streaming
5. Dashboards y Analytics