Setting Up and Testing a 2-Node Apache Kafka Cluster (KRaft Mode)

Version: Kafka 4.0.0

Date: August 4, 2025

Author: [Your Name/Organization]

⚠️ CRITICAL WARNING: Not for Production Use

A 2-node KRaft cluster **lacks fault tolerance** for controller services. If one of the two nodes fails, the remaining node cannot form a quorum, and the cluster will become unavailable. No new topics can be created, and partition leadership cannot be re-assigned.

This guide is for **training and development environments only** to understand how a quorum-based system works. For any production environment, a minimum of three nodes is required to ensure fault tolerance.

Table of Contents

1. Introduction

This document provides a detailed guide for setting up a 2-node Apache Kafka cluster using the KRaft (Kafka Raft) consensus protocol. This setup is useful for learning and testing purposes to demonstrate how Kafka and the KRaft quorum function. Due to its inherent lack of fault tolerance, it must not be used for production workloads.

2. Prerequisites

Ensure the following prerequisites are met on **both server machines** that will host Kafka:

- **Operating System:** Ubuntu Server 24.04.2 LTS (or similar Debian-based Linux distribution).
- **Java Development Kit (JDK):** OpenJDK 17 or later.
- **Network Connectivity:** Both nodes must be able to communicate with each other via their hostnames or IP addresses.
- **Sufficient Resources:** Minimum 2 GB RAM, 2 CPU Cores, and adequate disk space.
- **User Permissions:** sudo access for installation steps.

3. Cluster Node Information

We will use the following naming convention and assume corresponding IP addresses. **Replace these example IPs with your actual server IPs.**

Node Name	Role	IP Address (Example)	KRaft Node ID	Client Port (PLAINTEXT)	Controller Port (CONTROLLER)
kafka1	Broker & Controller	192.168.1.199	1	9092	9093
kafka2	Broker & Controller	192.168.1.200	2	9092	9093

4. Step-by-Step Installation & Configuration

4.1. Preparation (On All 2 Nodes: kafka1, kafka2)

Perform these steps identically on both machines.

4.1.1. System Update & Java Installation

```
sudo apt update && sudo apt upgrade -y
sudo apt install -y openjdk-17-jdk wget tar
```

4.1.2. Configure JAVA_HOME (System-Wide)

1. **Find your Java 17 path:** readlink -f /usr/bin/java | sed "s:bin/java::"
2. **Edit /etc/environment** (sudo nano /etc/environment) and add the lines:
   ```
   JAVA_HOME="/usr/lib/jvm/java-17-openjdk-amd64"
   PATH="$PATH:$JAVA_HOME/bin"
   ```
3. **Save, exit, and apply changes:** source /etc/environment

4.1.3. Configure Hostnames (/etc/hosts)

This is **CRUCIAL** for inter-node communication.

1. **Edit /etc/hosts on EACH node:** sudo nano /etc/hosts
2. **Add entries for both Kafka nodes** at the end:
   ```
   # Kafka Cluster Nodes
   192.168.1.199   kafka1
   192.168.1.200   kafka2
   ```
3. **Test connectivity from each node:** ping kafka1 and ping kafka2.

4.1.4. Create Kafka User & Directories

```
KAFKA_USER="kafka"
INSTALL_DIR="/usr/local/kafka"
sudo useradd -m -s /bin/bash "$KAFKA_USER"
sudo mkdir -p "$INSTALL_DIR" "$INSTALL_DIR/logs" "$INSTALL_DIR/data"
sudo chown -R "$KAFKA_USER":"$KAFKA_USER" "$INSTALL_DIR"
```

4.1.5. Download & Extract Kafka

```
KAFKA_VERSION="4.0.0"
SCALA_VERSION="2.13"
DOWNLOAD_URL="https://downloads.apache.org/kafka/${KAFKA_VERSION}/kafka_${
SCALA_VERSION}-${KAFKA_VERSION}.tgz"

wget "$DOWNLOAD_URL" -O /tmp/kafka.tgz
```

```
tar -xzf /tmp/kafka.tgz -C /tmp/
sudo mv /tmp/kafka_${SCALA_VERSION}-${KAFKA_VERSION}/* "$INSTALL_DIR/"
sudo chown -R "$KAFKA_USER":"$KAFKA_USER" "$INSTALL_DIR"
```

4.2. Kafka Configuration (server.properties - Per Node)

Each node needs its own server.properties file with a unique node.id.

On kafka1:

```
sudo tee /usr/local/kafka/config/server.properties > /dev/null <<EOF
# KRaft Node Configuration
process.roles=broker,controller
node.id=1

# Controller Quorum
controller.quorum.voters=1@kafka1:9093,2@kafka2:9093

# Listeners for Clients and Inter-Broker Communication
listeners=PLAINTEXT://kafka1:9092,CONTROLLER://kafka1:9093
advertised.listeners=PLAINTEXT://kafka1:9092,CONTROLLER://kafka1:9093
controller.listener.names=CONTROLLER
inter.broker.listener.name=PLAINTEXT

# Log Directories for Topic Data
log.dirs=/usr/local/kafka/logs
log.retention.hours=168
log.segment.bytes=1073741824
num.recovery.threads.per.data.dir=1

# Data Directory for KRaft Metadata Logs
metadata.log.dir=/usr/local/kafka/data
EOF
sudo chown kafka:kafka /usr/local/kafka/config/server.properties
```

On kafka2:

```
sudo tee /usr/local/kafka/config/server.properties > /dev/null <<EOF
# KRaft Node Configuration
```

```
process.roles=broker,controller
node.id=2

# Controller Quorum
controller.quorum.voters=1@kafka1:9093,2@kafka2:9093

# Listeners for Clients and Inter-Broker Communication
listeners=PLAINTEXT://kafka2:9092,CONTROLLER://kafka2:9093
advertised.listeners=PLAINTEXT://kafka2:9092,CONTROLLER://kafka2:9093
controller.listener.names=CONTROLLER
inter.broker.listener.name=PLAINTEXT

# Log Directories for Topic Data
log.dirs=/usr/local/kafka/logs
log.retention.hours=168
log.segment.bytes=1073741824
num.recovery.threads.per.data.dir=1

# Data Directory for KRaft Metadata Logs
metadata.log.dir=/usr/local/kafka/data
EOF
sudo chown kafka:kafka /usr/local/kafka/config/server.properties
```

4.3. Generate KRaft Cluster ID (ONLY on ONE node, e.g., kafka1)

```
sudo -u kafka /usr/local/kafka/bin/kafka-storage.sh random-uuid
```

Copy the output. This is your CLUSTER_ID.

4.4. Format KRaft Storage (On All 2 Nodes)

Use the CLUSTER_ID from the previous step.

On kafka1 (replace YOUR_CLUSTER_ID):

```
sudo -u kafka /usr/local/kafka/bin/kafka-storage.sh format \
  --cluster-id "YOUR_CLUSTER_ID" \
  --config /usr/local/kafka/config/server.properties \
  --ignore-formatted
```

On kafka2 (replace YOUR_CLUSTER_ID):

```
sudo -u kafka /usr/local/kafka/bin/kafka-storage.sh format \
  --cluster-id "YOUR_CLUSTER_ID" \
  --config /usr/local/kafka/config/server.properties \
  --ignore-formatted
```

4.5. Create Systemd Service (On All 2 Nodes)

Create this service file on both machines.

```
sudo tee /etc/systemd/system/kafka.service > /dev/null <<EOF
[Unit]
Description=Apache Kafka 4.0 Broker (KRaft Mode)
Documentation=https://kafka.apache.org/documentation/
Wants=network-online.target
After=network-online.target

[Service]
User=kafka
Environment="JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64"
ExecStart=/usr/local/kafka/bin/kafka-server-start.sh
/usr/local/kafka/config/server.properties
ExecStop=/usr/local/kafka/bin/kafka-server-stop.sh
Restart=on-failure
RestartSec=5

[Install]
WantedBy=multi-user.target
EOF

sudo systemctl daemon-reload
```

4.6. Enable & Start Kafka (On All 2 Nodes)

```
sudo systemctl enable kafka
sudo systemctl start kafka
```

5. Testing the Kafka Cluster

After both nodes are started, wait a minute for them to form a quorum.

5.1. Verify Connectivity to All Brokers

Run these from any node:

```
sudo -u kafka /usr/local/kafka/bin/kafka-broker-api-versions.sh --bootstrap-server kafka1:9092
sudo -u kafka /usr/local/kafka/bin/kafka-broker-api-versions.sh --bootstrap-server kafka2:9092
```

Expected: Output showing both brokers (IDs 1 and 2) are online.

5.2. Create a Test Topic

Use a replication-factor of 2.

```
sudo -u kafka /usr/local/kafka/bin/kafka-topics.sh --create \
  --topic my_cluster_test_topic \
  --partitions 2 \
  --replication-factor 2 \
  --bootstrap-server kafka1:9092
```

5.3. Verify Topic Creation and Replication

```
sudo -u kafka /usr/local/kafka/bin/kafka-topics.sh --describe \
  --topic my_cluster_test_topic \
  --bootstrap-server kafka1:9092
```

Expected: You should see both brokers (IDs 1 and 2) listed in the Replicas and Isr for both partitions.

5.4. Produce Messages to the Topic

```
sudo -u kafka /usr/local/kafka/bin/kafka-console-producer.sh \
  --topic my_cluster_test_topic \
  --bootstrap-server kafka1:9092
```

Type messages and press Enter.

5.5. Consume Messages from the Topic

Open a new terminal on a different node.

```
sudo -u kafka /usr/local/kafka/bin/kafka-console-consumer.sh \
  --topic my_cluster_test_topic \
  --bootstrap-server kafka1:9092 \
  --from-beginning
```

5.6. Test Fault Tolerance (CRITICAL TEST)

This is the most important part of the exercise to understand the limitations of a 2-node cluster.

1. **Stop one broker:** On kafka2, stop the Kafka service:
   sudo systemctl stop kafka

2. **Try to produce a message:** Go back to your producer. It will likely fail or time out after a delay because it can't find a new leader.
3. **Try to create a new topic:**
   sudo -u kafka /usr/local/kafka/bin/kafka-topics.sh --create --topic new_topic --bootstrap-server kafka1:9092

   This will fail.

Reason: With only one node running, the KRaft quorum is lost (1 out of 2 is not a majority). The single remaining node cannot elect itself as the controller leader, so the cluster is effectively frozen. This demonstrates why a 2-node cluster is not fault-tolerant.

4. **Restart kafka2** to restore the cluster: sudo systemctl start kafka.

6. Troubleshooting Common Issues

Refer to the troubleshooting section of the 3-node guide. The causes and fixes for issues like ConfigException, Connection to node -1, and service startup failures are the same.

7. Conclusion

You have successfully set up and tested a 2-node Kafka cluster. This exercise should clearly demonstrate the importance of a 3-node minimum for a fault-tolerant KRaft quorum. For any reliable or production-critical deployment, always use an odd number of controller nodes (3, 5, etc.) to ensure a majority can be reached even if one or more nodes fail.