**Step 1: Install Java**

1. Install Java on your system using the package manager. For example, on CentOS, you can use the following command:

sudo dnf install java-11-openjdk wget vim

**Step 2: Download Apache Kafka**

1. Go to the Apache Kafka website (<https://kafka.apache.org/downloads>) and download the binary distribution of Kafka.
2. Extract the downloaded archive to a directory of your choice. For example:

tar xzf kafka\_2.12-3.4.0.tgz mv kafka\_2.12-3.4.0 /usr/local/kafka

**Step 3: Configure Kafka Server**

1. Open the Kafka server properties file for editing:

vi /usr/local/kafka/config/server.properties

1. Update the following properties:

listeners=PLAINTEXT://192.168.222.128:9092

advertised.listeners=PLAINTEXT://192.168.222.128:9092

Replace **192.168.222.128** with the IP address of your Kafka server.

**Step 4: Setup Kafka Systemd Unit Files**

1. Create a ZooKeeper systemd unit file:

vi /etc/systemd/system/zookeeper.service

1. Add the following content:

[Unit]

Description=Apache ZooKeeper server

Documentation=http://zookeeper.apache.org

Requires=network.target remote-fs.target

After=network.target remote-fs.target

[Service]

Type=simple

ExecStart=/usr/bin/bash /usr/local/kafka/bin/zookeeper-server-start.sh /usr/local/kafka/config/zookeeper.properties

ExecStop=/usr/bin/bash /usr/local/kafka/bin/zookeeper-server-stop.sh

Restart=on-abnormal

[Install]

WantedBy=multi-user.target

1. Create a Kafka systemd unit file:

vi /etc/systemd/system/kafka.service

1. Add the following content:

[Unit]

Description=Apache Kafka Server

Documentation=http://kafka.apache.org/documentation.html

Requires=zookeeper.service

[Service]

Type=simple

Environment="JAVA\_HOME=/usr/lib/jvm/jre-11-openjdk"

ExecStart=/usr/bin/bash /usr/local/kafka/bin/kafka-server-start.sh /usr/local/kafka/config/server.properties

ExecStop=/usr/bin/bash /usr/local/kafka/bin/kafka-server-stop.sh

[Install]

WantedBy=multi-user.target

1. Reload systemd configuration:

systemctl daemon-reload

**Step 5: Start Kafka Server**

1. Start ZooKeeper:

systemctl start zookeeper

1. Start Kafka:

systemctl start kafka

1. Check the status of Kafka:

systemctl status kafka

**Step 6: Create Topics in Apache Kafka**

1. Create a topic:

cd /usr/local/kafka

bin/kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic testTopic

Replace **testTopic** with the desired topic name.

1. Verify the topic creation:

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

**Step 7: Setup Kafdrop**

1. Download Kafdrop from the GitHub repository (<https://github.com/obsidiandynamics/kafdrop/releases>).
2. Extract the downloaded archive to a directory of your choice. For example:

cd /docker/kafka

tar xzf kafdrop-3.31.0.jar mv kafdrop-3.31.0.jar kafdrop.jar

1. Start Kafdrop:

java --add-opens=java.base/sun.nio.ch=ALL-UNNAMED -jar /docker/kafka/kafdrop.jar --kafka.brokerConnect=localhost:9092

1. Open a browser and access Kafdrop using the following URL: **http://192.168.222.128:9000**

**Step 8: Oracle Setup for Debezium**

1. Connect to the Oracle database using SQL\*Plus as a sysdba:

sqlplus / as sysdba

1. Create a tablespace for the Debezium user:

CREATE TABLESPACE ahosan DATAFILE '/u01/app/oracle/oradata/AHOSAN/ahosan.dbf' SIZE 1G AUTOEXTEND ON NEXT 100M MAXSIZE UNLIMITED;

1. Create a user and grant necessary privileges to the user:

CREATE USER ahosan IDENTIFIED BY ahosan DEFAULT TABLESPACE ahosan TEMPORARY TABLESPACE TEMP PROFILE default ACCOUNT UNLOCK;

GRANT CONNECT, RESOURCE TO ahosan;

ALTER USER ahosan DEFAULT ROLE ALL;

ALTER USER ahosan QUOTA UNLIMITED ON ahosan;

-- Grant additional privileges required by Debezium

GRANT SELECT ON V\_$DATABASE TO ahosan;

GRANT FLASHBACK ANY TABLE TO ahosan;

-- Add more necessary privileges as per your requirements

1. Enable supplemental logging and GoldenGate replication:

ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;

ALTER SYSTEM SET enable\_goldengate\_replication = true;

1. Check the value of **enable\_goldengate\_replication** parameter:

show parameters enable\_goldengate\_replication;

**Step 9: Configure Debezium Connector**

1. Copy the Debezium Oracle connector JAR files to the Kafka **libs** directory:

cp /docker/debezium/debezium-connector-oracle/\* /usr/local/kafka/libs/

1. Install the Oracle Instant Client on your system.
2. Configure the **tnsnames.ora** file with the connection details of your Oracle database.
3. Open the Kafka Connect distributed properties file for editing:

vi /usr/local/kafka/config/connect-distributed.properties

1. Update the following properties:

bootstrap.servers=192.168.222.128:9092

listeners=HTTP://192.168.222.128:8083

1. Copy the Oracle client's **ojdbc.jar** file to the Kafka **libs** directory:

cp /usr/lib/oracle/19.19/client64/lib/ojdbc8.jar /usr/local/kafka/libs/ojdbc.jar

**Step 10: Start Debezium Connector**

1. Start the Kafka Connect distributed service:

/usr/local/kafka/bin/connect-distributed.sh /usr/local/kafka/config/connect-distributed.properties

1. Create a configuration file for the Oracle connector:

vi /docker/debezium/connector-config.json

1. Add the following content to the file:

{

"name": "inventory-connector",

"config": {

"connector.class": "io.debezium.connector.oracle.OracleConnector",

"tasks.max": "1",

"topic.prefix": "server1",

"database.hostname": "192.168.222.128",

"database.port": "1521",

"database.user": "ahosan",

"database.password": "ahosan",

"database.dbname": "ahosan",

"bootstrap.servers": "192.168.222.128:9092",

"schema.history.kafka.bootstrap.servers": "192.168.222.128:9092",

"schema.history.kafka.topic": "testTopic",

"table.include.list": "ahosan.test"

}

}

1. Register the Oracle connector by sending a POST request to Kafka Connect's REST API:

cd /docker/debezium

curl -X POST -H "Content-Type: application/json" --data @connector-config.json http://192.168.222.128:8083/connectors

1. Check the status of the connector:

curl -X GET http://192.168.222.128:8083/connectors/inventory-connector

**Step 11: Create a Kafka Consumer**

1. Install Python 3.8 and the required dependencies:

dnf install python3.8

python3.8 -m pip install kafka-python mysql-connector-python

1. Create a file for the Kafka consumer:

vi /docker/debezium/consumer.py

1. Add the following Python code to the file:
2. import json
3. from kafka import KafkaConsumer
4. import mysql.connector
5. import time
6. consumer = KafkaConsumer('server1.AHOSAN.TEST', bootstrap\_servers='192.168.222.128:9092')
7. mysql\_table = 'todo'
8. mysql\_connection = mysql.connector.connect(
9. host="192.168.222.128",
10. user="TodoList",
11. password="Todo\_List123",
12. database="TodoList"
13. )
14. mysql\_cursor = mysql\_connection.cursor()
15. while True:
16. for message in consumer:
17. change\_event = json.loads(message.value)
18. print(change\_event)
19. # Get and serialize data from Kafka
20. payload = change\_event.get('payload', {})
21. data = payload.get('after', {})
22. name = data.get('NAME', '')
23. address = data.get('ADDRESS', '')
24. phone = data.get('PHONE', '')
25. # Insert the data into MySQL
26. insert\_query = "INSERT INTO {} (name, address, phone) VALUES (%s, %s, %s)".format(mysql\_table)
27. values = (name, address, phone)
28. mysql\_cursor.execute(insert\_query, values)
29. mysql\_connection.commit()

time.sleep(1)

**Step 12: Run the Kafka Consumer**

1. Run the Kafka consumer using the Python script:

cd /docker/debezium

python3.8 consumer.py

Congratulations! You have completed the setup process for Kafka, ZooKeeper, Kafdrop, Oracle, and Debezium. You can now start producing and consuming messages from Kafka and observe the data being captured by the Debezium Oracle connector and inserted into the MySQL database.