vagrant@ubuntu-bionic:~$ echo "Access at: http://$(hostname -I | awk '{print $1}'):8080"

Access at: <http://10.0.2.15:8080>

sudo cat /home/vagrant/.jenkins/secrets/initialAdminPassword

Here's a consolidated list of all working Docker commands from our troubleshooting, organized by service with explanations:

1. **Kafka Broker (KRaft Mode)**

bash

Copy

Download

docker run -d \

--name kafka \

--network host \

-e KAFKA\_ENABLE\_KRAFT=yes \

-e KAFKA\_CFG\_NODE\_ID=1 \

-e KAFKA\_CFG\_PROCESS\_ROLES=controller,broker \

-e KAFKA\_CFG\_CONTROLLER\_QUORUM\_VOTERS="1@$(hostname -I | awk '{print $1}'):9093" \

-e KAFKA\_CFG\_LISTENERS="PLAINTEXT://0.0.0.0:9092,CONTROLLER://0.0.0.0:9093" \

-e KAFKA\_CFG\_ADVERTISED\_LISTENERS="PLAINTEXT://$(hostname -I | awk '{print $1}'):9092" \

-e KAFKA\_CFG\_LOG\_DIRS="/tmp/kraft-combined-logs" \

apache/kafka:3.7.0 \

sh -c "

/opt/kafka/bin/kafka-storage.sh format -t \$(\

/opt/kafka/bin/kafka-storage.sh random-uuid\

) -c /opt/kafka/config/kraft/server.properties;

/opt/kafka/bin/kafka-server-start.sh /opt/kafka/config/kraft/server.properties

"

**Key Options**:

* --network host for simpler networking
* KRaft mode (no ZooKeeper)
* Advertises the host's IP automatically
* Auto-formats storage on first run

2. **Kafka-UI (Port 9080)**

bash

Copy

Download

docker run -d \

--name kafka-ui \

-p 9080:8080 \

--add-host="kafka:$(hostname -I | awk '{print $1}')" \

-e KAFKA\_CLUSTERS\_0\_NAME=local \

-e KAFKA\_CLUSTERS\_0\_BOOTSTRAPSERVERS=kafka:9092 \

-e KAFKA\_CLUSTERS\_0\_PROPERTIES\_SECURITY\_PROTOCOL=PLAINTEXT \

provectuslabs/kafka-ui:latest

**Key Options**:

* Maps to port **9080** to avoid Jenkins conflicts
* Uses --add-host for DNS resolution
* Plaintext protocol for simplicity

3. **Kafdrop (Port 9081)**

bash

Copy

Download

docker run -d \

--name kafdrop \

-p 9081:9000 \

--add-host="kafka:$(hostname -I | awk '{print $1}')" \

-e KAFKA\_BROKERCONNECT=kafka:9092 \

-e JVM\_OPTS="-Xms32M -Xmx64M" \

obsidiandynamics/kafdrop:latest

**Key Options**:

* Lightweight JVM settings
* Port **9081** to avoid conflicts
* Same host resolution as Kafka-UI

4. **Jenkins (Existing Setup)**

bash

Copy

Download

docker run -d \

--name jenkins \

-p 8080:8080 \

-p 50000:50000 \

-v jenkins\_home:/var/jenkins\_home \

jenkins/jenkins:lts

*(Assuming you already have this running on port 8080)*

**Verification Commands**

Check Kafka Topics/Messages:

bash

Copy

Download

docker run --rm confluentinc/cp-kafkacat \

kafkacat -b $(hostname -I | awk '{print $1}'):9092 -t jenkins-events -C

Check Container Logs:

bash

Copy

Download

docker logs kafka-ui --tail 20

docker logs kafdrop --tail 20

List All Services:

bash

Copy

Download

docker ps --format "table {{.Names}}\t{{.Status}}\t{{.Ports}}"

**Troubleshooting Fixes Applied**

1. **Fixed**advertised.listeners**in Kafka**:

bash

Copy

Download

docker exec kafka sed -i "s/localhost/$(hostname -I | awk '{print $1}')/" /opt/kafka/config/kraft/server.properties

docker restart kafka

1. **Resolved Port Conflicts** by assigning:
   * Jenkins: **8080**
   * Kafka-UI: **9080**
   * Kafdrop: **9081**
2. **Network Isolation**: Used --add-host for consistent DNS across containers.

**Final Access URLs**

* Jenkins: http://<your-ip>:8080
* Kafka-UI: http://<your-ip>:9080
* Kafdrop: http://<your-ip>:9081

All services should now coexist without conflicts. Let me know if you'd like to add monitoring or backups!

we also create .sh files to store kafka properties and other also chagne permission can you also list that any other small debugging things

Here's a comprehensive set of helper scripts and debugging tools to manage your Kafka + Jenkins setup:

**1. Configuration Files**

\*\*kafka-setup.sh\*\* (Save and run with chmod +x kafka-setup.sh`)

bash

Copy

Download

#!/bin/bash

*# Sets up Kafka with proper advertised listeners*

KAFKA\_IP=$(hostname -I | awk '{print $1}')

docker run -d \

--name kafka \

--network host \

-v /tmp/kraft-combined-logs:/tmp/kraft-combined-logs \

-e KAFKA\_ENABLE\_KRAFT=yes \

-e KAFKA\_CFG\_ADVERTISED\_LISTENERS="PLAINTEXT://$KAFKA\_IP:9092" \

apache/kafka:3.7.0 \

sh -c "

/opt/kafka/bin/kafka-storage.sh format -t \$(

/opt/kafka/bin/kafka-storage.sh random-uuid

) -c /opt/kafka/config/kraft/server.properties;

/opt/kafka/bin/kafka-server-start.sh /opt/kafka/config/kraft/server.properties

"

echo "Kafka started with IP: $KAFKA\_IP"

**`kafka-debug.sh** (Debugging tool)

bash

Copy

Download

#!/bin/bash

*# Checks Kafka broker health*

KAFKA\_IP=$(docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' kafka 2>/dev/null || hostname -I | awk '{print $1}')

echo "➤ Testing Kafka connectivity:"

nc -zv $KAFKA\_IP 9092 && echo "✅ Success" || echo "❌ Failed"

echo "➤ Listing topics:"

docker run --rm confluentinc/cp-kafka:7.0.0 \

kafka-topics --bootstrap-server $KAFKA\_IP:9092 --list

echo "➤ Consumer group status:"

docker run --rm confluentinc/cp-kafka:7.0.0 \

kafka-consumer-groups --bootstrap-server $KAFKA\_IP:9092 --list

**2. Permission Fixes**

bash

Copy

Download

*# Make scripts executable*

chmod +x \*.sh

*# Fix volume permissions (if Kafka logs fail)*

sudo chown -R 1000:1000 /tmp/kraft-combined-logs

**3. Jenkins Plugin Debug Script**

**`jenkins-kafka-test.sh** (Test pipeline from CLI)

bash

Copy

Download

#!/bin/bash

*# Tests Jenkins → Kafka integration*

JENKINS\_URL="http://localhost:8080"

JOB\_NAME="kafka\_notify\_test\_plugin"

curl -X POST "$JENKINS\_URL/job/$JOB\_NAME/build" \

--user "admin:<your-api-token>" \

--data-urlencode json='{"parameter": []}'

echo -e "\n➤ Pipeline triggered. Check logs at:"

echo "$JENKINS\_URL/job/$JOB\_NAME/lastBuild/console"

**4. Common Debugging Commands**

**Network Checks**

bash

Copy

Download

*# Check container-to-container connectivity*

docker exec kafka-ui ping kafka

*# Check exposed ports*

netstat -tulnp | grep -E '8080|9080|9081|9092'

**Log Inspection**

bash

Copy

Download

*# Tail all relevant logs*

multitail \

-l 'docker logs -f kafka' \

-l 'docker logs -f kafka-ui' \

-l 'docker logs -f kafdrop' \

-l 'tail -f /var/log/jenkins/jenkins.log'

**Kafka Topic Operations**

bash

Copy

Download

*# Manually create a topic*

docker exec kafka \

kafka-topics --create \

--topic jenkins-events \

--bootstrap-server localhost:9092 \

--partitions 1 \

--replication-factor 1

*# Watch messages in real-time*

docker run --rm -it confluentinc/cp-kafkacat \

kafkacat -b $(hostname -I | awk '{print $1}'):9092 -t jenkins-events -C

**5. Reset Script**

**`reset-all.sh** (Nuclear option)

bash

Copy

Download

#!/bin/bash

*# Full cleanup and fresh start*

docker rm -f kafka kafka-ui kafdrop jenkins 2>/dev/null

sudo rm -rf /tmp/kraft-combined-logs /var/jenkins\_home

docker network prune -f

echo "✅ All containers and volumes removed"

**Key Permissions to Always Check**

bash

Copy

Download

*# Jenkins volume*

sudo chown -R 1000:1000 /var/jenkins\_home

*# Kafka logs*

sudo chown -R 1000:1000 /tmp/kraft-combined-logs

*# Docker sockets (if needed)*

sudo chmod 666 /var/run/docker.sock

**Debugging Flowchart**

1. **Kafka Unreachable?**  
   → Run kafka-debug.sh  
   → Check advertised.listeners
2. **UI Not Loading?**  
   → Verify ports (netstat -tulnp)  
   → Check container logs (docker logs <name>)
3. **Jenkins Plugin Failing?**  
   → Test manually with kafkacat  
   → Check Jenkins system logs (/var/log/jenkins/jenkins.log)

These scripts cover 95% of debugging scenarios. For advanced cases (SSL, ACLs), let me know!

++++++++++

1. Stop and Remove Existing Kafdrop

bash

Copy

Download

docker rm -f kafdrop

2. Run Kafdrop with Correct Configuration

bash

Copy

Download

docker run -d \

--name kafdrop \

-p 9081:9000 \

--add-host="kafka:10.0.2.15" \

-e KAFKA\_BROKERCONNECT=kafka:9092 \

-e JVM\_OPTS="-Xms32M -Xmx64M" \

-e SERVER\_SERVLET\_CONTEXTPATH="/" \

obsidiandynamics/kafdrop:latest

3. Verify It's Running

bash

Copy

Download

docker ps | grep kafdrop

curl -I http://localhost:9081

4. Check Connection Status

bash

Copy

Download

docker logs kafdrop --tail 20 | grep -i "kafka\|connected"

5. Access Kafdrop UI

Open in your browser:

Copy

Download

http://<your-server-ip>:9081

++++++++++++

1. Run Kafka-UI on Port 9080 (or any free port)

bash

Copy

Download

docker rm -f kafka-ui

docker run -d \

--name kafka-ui \

-p 9080:8080 \

--add-host="kafka:10.0.2.15" \

-e KAFKA\_CLUSTERS\_0\_NAME=local \

-e KAFKA\_CLUSTERS\_0\_BOOTSTRAPSERVERS=kafka:9092 \

-e KAFKA\_CLUSTERS\_0\_PROPERTIES\_SECURITY\_PROTOCOL=PLAINTEXT \

provectuslabs/kafka-ui:latest

2. Verify Kafka-UI is Running

bash

Copy

Download

docker ps | grep kafka-ui

curl -I http://localhost:9080

3. Check Connection Status

bash

Copy

Download

docker logs kafka-ui --tail 20 | grep -i "cluster\|connected"

4. Access the UI

Open in your browser:

Copy

Download

http://<your-server-ip>:9080

Key Notes:

1. Uses port 9080 externally (mapped to 8080 inside container)
2. Maintains connectivity to Kafka via --add-host
3. Explicitly sets security protocol to PLAINTEXT

If you need to use a different port, just change the first number in -p 9080:8080 (e.g., -p 9081:8080). The second number (8080) must stay as-is since it's Kafka-UI's internal port

1. Find and free port 8080

bash

Copy

Download

sudo lsof -i :8080

sudo kill -9 <PID> *# Replace <PID> with the process ID from above*