

Apache Kafka



- [Overview](#)
- [Kafka Use Cases](#)
- [Architecture](#)
- [Kafka](#)
 - [Topics and Partitions](#)
 - [Kafka Brokers](#)
 - [Consumers \(still relevant? - moved to Kafka Connect?\)](#)
 - [Data Replication](#)
- [Kafka Connect](#)
 - [Overview](#)
- [Installation on Kubernetes](#)
- [Kafka Configuration](#)
- [Connectors](#)
- [Functions](#)
 - [Login to the kafka-connect pod](#)
 - [List Connectors](#)
 - [Enable a Connector](#)
 - [Get connector details](#)
 - [Delete a Connector](#)
 - [Create a Topic](#)
 - [Send Message to a topic](#)
 - [Consume a Message from a Topic](#)
- [Kafka Broker Details \(Bitnami\)](#)
 - [Folders](#)
 - [Commands](#)
- [REST Api](#)
 - [Pushing Message to Topic](#)
 - [Consuming Messages from a Topic](#)
- [References](#)

Overview

Apache Kafka is **an open source project** for a distributed publish-subscribe messaging system rethought as a distributed commit log.

Kafka stores messages in topics that are partitioned and replicated across multiple brokers in a cluster. Producers send messages to topics from which consumers read.

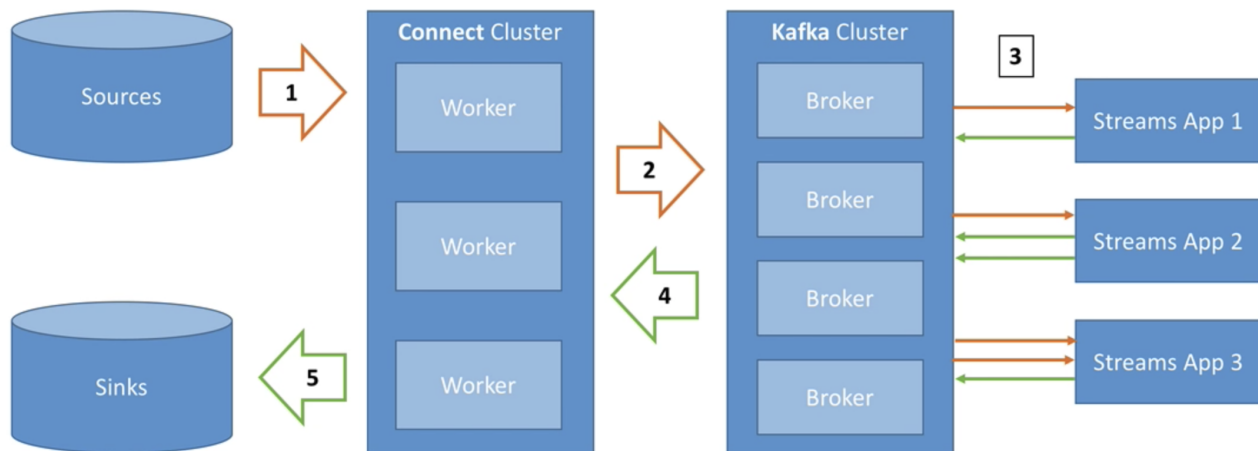
Created by LinkedIn and is now an Open Source project maintained by Confluent.

Kafka Use Cases

Some use cases for using Kafka:

- Messaging System
- Activity Tracking
- Gathering metrics from many different sources
- Application Logs gathering
- Stream processing (with the Kafka Streams API or Spark for example)
- De-coupling of system dependencies
- Integration with Spark, Flink, Storm, Hadoop and many other Big Data technologies

Architecture



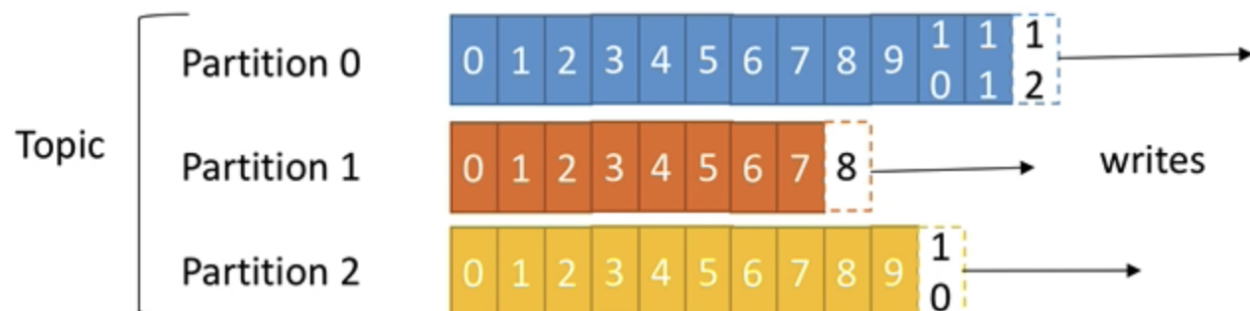
1. Source Connectors pull data from sources
2. Data is sent to Kafka cluster
3. Transformation of topic data into another topic can be done with Streams
4. Sink Connectors in Connect cluster pull data from Kafka
5. Sink Connectors push data to sinks

Kafka

Topics and Partitions

Topics: a particular stream of data

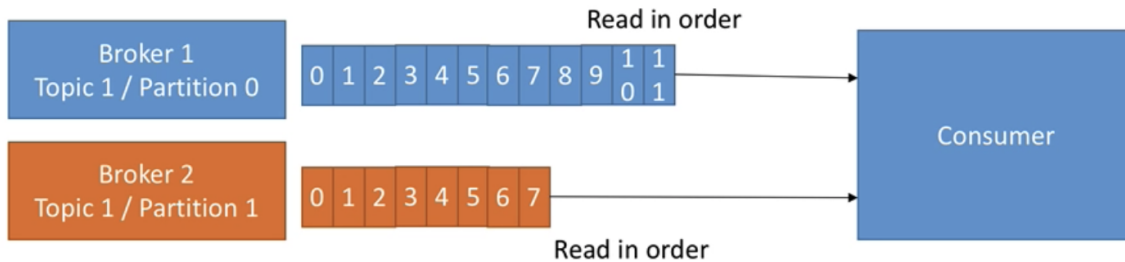
- similar to a table in a database(without constraints)
- you can have as many topics as you want
- a topic is identified by it's name



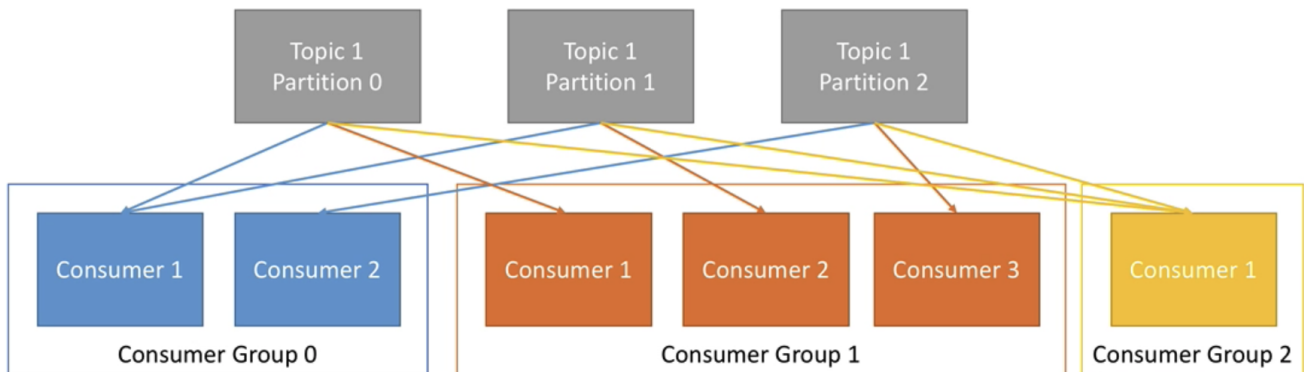
Topics are split into partitions

- each partition is ordered
- each message with a partition gets an incremental id, called offset.
- offsets are only relevant for a particular partition
- order is guaranteed only in a partition (not across partitions)
- data is assigned to a random partition unless a key is provided
- you can have as many partitions per topic as you want
- specifying a key, ensures that your message is written to the same partition (which ensures order).

Kafka Brokers

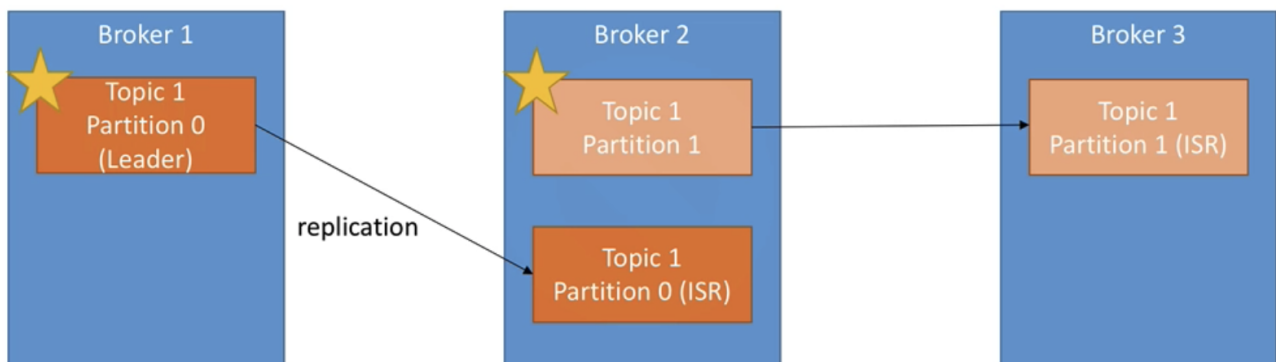


Consumers (still relevant? - moved to Kafka Connect?)



Data Replication

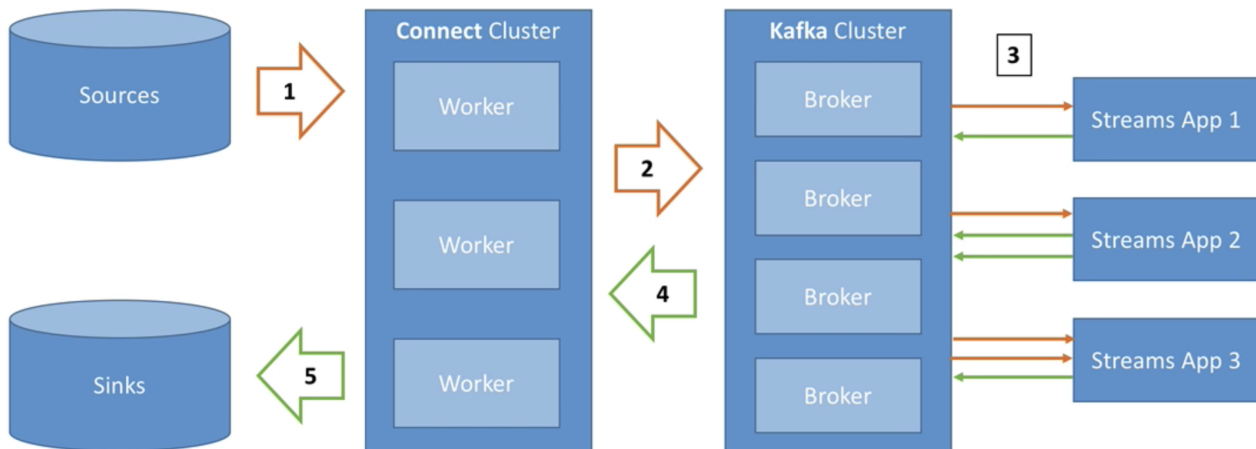
- topics should have a replication factor greater than 1 (usually 2 or 3)
- this ensures that if a broker goes down, another broker can serve the data
- ISR - in-synch replica



Kafka Connect

Overview

- Source connectors to get data from common data sources
- Sink connectors to publish that data in common data sources
- Make it easy for non-experienced dev to quickly get their data reliably into Kafka
- Re-usable code!



Installation on Kubernetes

See <https://github.com/confluentinc/cp-helm-charts>

```
$ helm repo add confluentinc https://confluentinc.github.io/cp-helm-charts/
```

"confluentinc" has been added to your repositories

```
> helm repo update
```

Hang tight while we grab the latest from your chart repositories...

...Skip local chart repository

...Successfully got an update from the "confluentinc" chart repository

...Successfully got an update from the "stable" chart repository

Update Complete. Happy Helming!

```
$ helm install confluent-kafka confluentinc/cp-helm-charts
```

NAME: confluent-kafka

LAST DEPLOYED: Tue Jul 27 18:42:52 2021

NAMESPACE: default

STATUS: deployed

REVISION: 1

NOTES:

Zookeeper

Connection string for Confluent Kafka:

confluent-kafka-cp-zookeeper-0.confluent-kafka-cp-zookeeper-headless:2181,confluent-kafka-cp-zookeeper-1.

confluent-kafka-cp-zookeeper-headless:2181,...

To connect from a client pod:

1. Deploy a zookeeper client pod with configuration:

```
apiVersion: v1
kind: Pod
metadata:
  name: zookeeper-client
  namespace: default
spec:
  containers:
  - name: zookeeper-client
    image: confluentinc/cp-zookeeper:6.1.0
    command:
      - sh
      - -c
      - "exec tail -f /dev/null"
```

2. Log into the Pod

```
kubectrl exec -it zookeeper-client -- /bin/bash
```

3. Use zookeeper-shell to connect in the zookeeper-client Pod:

```
zookeeper-shell confluent-kafka-cp-zookeeper:2181
```

4. Explore with zookeeper commands, for example:

```
# Gives the list of active brokers
ls /brokers/ids
```

```
# Gives the list of topics
ls /brokers/topics
```

```
# Gives more detailed information of the broker id '0'
get /brokers/ids/0## -----
```

Kafka

To connect from a client pod:

1. Deploy a kafka client pod with configuration:

```
apiVersion: v1
kind: Pod
metadata:
  name: kafka-client
  namespace: default
spec:
  containers:
  - name: kafka-client
    image: confluentinc/cp-enterprise-kafka:6.1.0
    command:
      - sh
      - -c
      - "exec tail -f /dev/null"
```

2. Log into the Pod

```
kubectrl exec -it kafka-client -- /bin/bash
```

3. Explore with kafka commands:

```
# Create the topic
kafka-topics --zookeeper confluent-kafka-cp-zookeeper-headless:2181 --topic confluent-kafka-topic --create --
partitions 1 --replication-factor 1 --if-not-exists
```

```
# Create a message
MESSAGE="`date -u`"
```

```
# Produce a test message to the topic
echo "$MESSAGE" | kafka-console-producer --broker-list confluent-kafka-cp-kafka-headless:9092 --topic confluent-kafka-topic

# Consume a test message from the topic
kafka-console-consumer --bootstrap-server confluent-kafka-cp-kafka-headless:9092 --topic confluent-kafka-topic --from-beginning --timeout-ms 2000 --max-messages 1 | grep "$MESSAGE"
```

```
$ kubectl get pods
```

```
$ kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
confluent-kafka-cp-control-center-5cf9477c94-6j2tz	1/1	Running	7	49m
confluent-kafka-cp-kafka-0	2/2	Running	0	49m
confluent-kafka-cp-kafka-1	2/2	Running	0	38m
confluent-kafka-cp-kafka-2	2/2	Running	0	38m
confluent-kafka-cp-kafka-connect-7646bbff9-hhdqq	2/2	Running	3	49m
confluent-kafka-cp-kafka-rest-57588d5cb5-4sfnp	2/2	Running	6	49m
confluent-kafka-cp-ksql-server-5bdd6b999c-djrj2	2/2	Running	5	49m
confluent-kafka-cp-schema-registry-8db5569b8-2qgm9	2/2	Running	6	49m
confluent-kafka-cp-zookeeper-0	2/2	Running	0	49m
confluent-kafka-cp-zookeeper-1	2/2	Running	0	38m
confluent-kafka-cp-zookeeper-2	2/2	Running	0	38m

Kafka Configuration

<https://docs.confluent.io/platform/current/installation/configuration/broker-configs.html#>

Sample values.yaml

```

cp-zookeeper:
  prometheus:
    jmx:
      enabled: false
  securityContext:
    runAsUser: 1000

cp-kafka:
  nodeport:
    enabled: true
    servicePort: 19092
    firstListenerPort: 31090
  persistence:
    enabled: true
    size: 10Gi
  configurationOverrides:
    "log.retention.bytes": "7516192768"
    "delete.topic.enable": "true"
    "message.max.bytes": "1048588"
    "advertised.listeners": |-
      EXTERNAL://localhost:${(31090 + ${KAFKA_BROKER_ID})}
  prometheus:
    jmx:
      enabled: false
  securityContext:
    runAsUser: 1000

cp-schema-registry:
  prometheus:
    jmx:
      enabled: false

cp-kafka-rest:
  prometheus:
    jmx:
      enabled: false

cp-ksql-server:
  prometheus:
    jmx:
      enabled: false

cp-control-center:
  prometheus:
    jmx:
      enabled: false

```

Connectors

Connector	Type	URL
Azure	Kafka Sink Connector	https://www.confluent.de/hub/chaitalisagesh/kafka-connect-log-analytics
Fluent	Output Plugin for fluentbit	https://github.com/fluent/fluent-plugin-kafka
Elasticsearch	Kafka Source Connector	https://www.confluent.io/hub/dariobalinzo/kafka-connect-elasticsearch-source
Elasticsearch	Kafka Sink Connector	https://github.com/confluentinc/kafka-connect-elasticsearch
*Fluent	Output Plugin for fluentbit	https://docs.fluentbit.io/manual/pipeline/outputs/kafka

Functions

These call can all be made from the kafka-connect pod

Login to the kafka-connect pod

```
kubectl exec -it kafka-cp-kafka-connect-<ID> -c cp-kafka-connect-server bash
```

List Connectors

```
$ curl -s -X GET -H "Content-Type: application/json" http://kafka-cp-kafka-connect:8083/connectors

["azure-sink-connector"]
```

Enable a Connector

```
$ curl -s -X POST -H "Content-Type: application/json" --data '{"name":"azure-sink-connector","config":
{"connector.class":
ss": "io.kafka.connect.log.analytics.sink.LogAnalyticsSinkConnector","topics" : "john-test", "key.converter":
"org.apache.kafka.connect.storage.StringConverter", "value.
converter": "org.apache.kafka.connect.json.JsonConverter", "value.converter.schemas.enable": "false",
"workspace.id" : "7e0d2c8e-a46c-4fd9-b274-4b07f0ba555c", "workspace
.key" : "y3n6lvRaKhDIaV6UuGn6+nuh/BoRsQI0fy9S13ZdrL/w56LU0uqrRK3ajAAnxjo8W4PAzxId0V09bJWxmtrNLA==" }}'
http://kafka-cp-kafka-connect:8083/connectors

{"name":"azure-sink-connector","config":{"connector.class":"io.kafka.connect.log.analytics.sink.
LogAnalyticsSinkConnector","topics":"john-test","key.converter":"org.apache.kafka.connect.storage.
StringConverter","value.converter":"org.apache.kafka.connect.json.JsonConverter","value.converter.schemas.
enable":"false","workspace.id":"7e0d2c8e-a46c-4fd9-b274-4b07f0ba555c","workspace.key":"y3n6lvRaKhDIaV6UuGn6+nuh
/BoRsQI0fy9S13ZdrL/w56LU0uqrRK3ajAAnxjo8W4PAzxId0V09bJWxmtrNLA==","name":"azure-sink-connector"},"tasks":[,"
type":"sink"}
```

Get connector details

```
$ curl -s -X GET -H "Content-Type: application/json" http://kafka-cp-kafka-connect:8083/connectors/azure-sink-
connector

{
  "name": "azure-sink-connector",
  "config": {
    "connector.class": "io.kafka.connect.log.analytics.sink.LogAnalyticsSinkConnector",
    "workspace.id": "7e0d2c8e-a46c-4fd9-b274-4b07f0ba555c",
    "topics": "john-test",
    "value.converter.schemas.enable": "false",
    "name": "azure-sink-connector",
    "workspace.key": "y3n6lvRaKhDIaV6UuGn6+nuh/BoRsQI0fy9S13ZdrL/w56LU0uqrRK3ajAAnxjo8W4PAzxId0V09bJWxmtrNLA==",
    "value.converter": "org.apache.kafka.connect.json.JsonConverter",
    "key.converter": "org.apache.kafka.connect.storage.StringConverter"
  },
  "tasks": [
    {
      "connector": "azure-sink-connector",
      "task": 0
    }
  ],
  "type": "sink"
}
```


Delete a Connector

```
curl -s -X DELETE -H "Content-Type: application/json" http://kafka-cp-kafka-connect:8083/connectors/azure-sink-connector
```

Create a Topic

```
kafka-topics --zookeeper kafka-cp-zookeeper-headless:2181 --topic john-test --create --partitions 1 --replication-factor 1 --if-not-exists
```

Send Message to a topic

```
echo '{"test": 213}' | kafka-console-producer --broker-list kafka-cp-kafka-headless:9092 --topic john-test
```

Consume a Message from a Topic

```
kafka-console-consumer --bootstrap-server kafka-cp-kafka-headless:9092 --topic john-test --from-beginning --timeout-ms 2000 --max-messages 1
```

Debugging

Kafka Connect

Should see the following log if running:

```
[2021-11-16 15:24:32,204] INFO Kafka Connect started (org.apache.kafka.connect.runtime.Connect)
```

Kafka Broker Details (Bitnami)

Folders

Kafka Home Folder: **/opt/bitnami/kafka**

```
$ ls -l

total 48
-rw-rw-r-- 1 root root 14520 Sep 21 09:58 LICENSE
-rw-rw-r-- 1 root root  953 Sep 21 09:58 NOTICE
drwxrwxr-x 1 root root 4096 Sep 21 10:06 bin
drwxrwxr-x 1 root root 4096 Oct  5 03:34 config
drwxrwxr-x 1 root root 4096 Sep 21 10:06 libs
drwxrwxr-x 1 root root 4096 Sep 21 10:06 licenses
drwxrwxr-x 1 root root 4096 Oct  6 17:14 logs
drwxrwxr-x 1 root root 4096 Sep 21 10:06 site-docs
```

Commands

```
## Creating new Topics

./kafka-topics.sh \
  --zookeeper zookeeper:2181 \
  --create \
  --topic kafka.learning.tweets \
  --partitions 1 \
  --replication-factor 1

./kafka-topics.sh \
  --zookeeper zookeeper:2181 \
  --create \
  --topic kafka.learning.alerts \
  --partitions 1 \
  --replication-factor 1

## Listing Topics

./kafka-topics.sh \
  --zookeeper zookeeper:2181 \
  --list

## Getting details about a Topic

./kafka-topics.sh \
  --zookeeper zookeeper:2181 \
  --describe

## Publishing Messages to Topics

./kafka-console-producer.sh \
  --bootstrap-server localhost:29092 \
  --topic kafka.learning.tweets

## Consuming Messages from Topics

./kafka-console-consumer.sh \
  --bootstrap-server localhost:29092 \
  --topic kafka.learning.tweets \
  --from-beginning

## Deleting Topics

./kafka-topics.sh \
  --zookeeper zookeeper:2181 \
  --delete \
  --topic kafka.learning.alerts

#Create a Topic with multiple partitions

./kafka-topics.sh \
  --zookeeper zookeeper:2181 \
  --create \
  --topic kafka.learning.orders \
  --partitions 3 \
  --replication-factor 1

#Check topic partitioning

./kafka-topics.sh \
  --zookeeper zookeeper:2181 \
  --topic kafka.learning.orders \
  --describe
```

```

## Publishing Messages to Topics with keys

./kafka-console-producer.sh \
  --bootstrap-server localhost:29092 \
  --property "parse.key=true" \
  --property "key.separator=:" \
  --topic kafka.learning.orders

## Consume messages using a consumer group

./kafka-console-consumer.sh \
  --bootstrap-server localhost:29092 \
  --topic kafka.learning.orders \
  --group test-consumer-group \
  --property print.key=true \
  --property key.separator=" = " \
  --from-beginning

## Check current status of offsets

./kafka-consumer-groups.sh \
  --bootstrap-server localhost:29092 \
  --describe \
  --all-groups

## Creating the Topic

./kafka-topics.sh \
  --zookeeper zookeeper:2181 \
  --create \
  --topic kafka.usecase.students \
  --partitions 2 \
  --replication-factor 1

## Describe the Topic

./kafka-topics.sh \
  --zookeeper zookeeper:2181 \
  --topic kafka.usecase.students \
  --describe

## Publish to the Topic

./kafka-console-producer.sh \
  --bootstrap-server localhost:29092 \
  --property "parse.key=true" \
  --property "key.separator=:" \
  --topic kafka.usecase.students

## Consume Message from the Topic

./kafka-console-consumer.sh \
  --bootstrap-server localhost:29092 \
  --topic kafka.usecase.students \
  --group usecase-consumer-group \
  --property print.key=true \
  --property key.separator=" = " \
  --from-beginning

```

REST Api

Pushing Message to Topic

Publish a Message

```
curl -X POST -H "Content-Type: application/vnd.kafka.json.v2+json" \
  --data ' { "records": [ { "value": {"log":"value"} } ] }' \
  "http://localhost:8082/topics/ncyd_test_in"
```

```
{ "offsets": [ { "partition": 0, "offset": 0, "error_code": null, "error": null }, { "key_schema_id": null, "value_schema_id": null } ] }
```

Consuming Messages from a Topic

Create a Consumer

```
curl -X POST -H "Content-Type: application/vnd.kafka.v2+json" \
  --data '{"name": "test_consumer", "format": "json", "auto.offset.reset": "earliest"}' \
  http://localhost:8082/consumers/test_consumer
```

```
{ "instance_id": "test_consumer", "base_uri": "http://ckaf-rest-0.ckaf-rest-headless.default.svc.cluster.local:8082/consumers/test_consumer/instances/test_consumer" }
```

Subscribe to a Topic

```
curl -X POST -H "Content-Type: application/vnd.kafka.v2+json" \
  --data '{"topics": ["ncyd_test_in"]}' \
  http://localhost:8082/consumers/test_consumer/instances/test_consumer/subscription
```

Consume the Messages

```
curl -X GET -H "Accept: application/vnd.kafka.json.v2+json" \
  http://localhost:8082/consumers/test_consumer/instances/test_consumer/records
```

```
[ { "topic": "ncyd_test_in", "key": null, "value": { "log": "value" }, "partition": 0, "offset": 0 } ]
```

References

Reference	URL
Apache Kafka in 5 minutes	https://www.youtube.com/watch?v=PzPXRmVHMxI
Nokia Learning - Kafka	https://nokialearn.csod.com/ui/lms-learning-details/app/course/dc425d19-5642-535f-916c-211768f90a00
Kafka Topics, Partitions and Offsets Explained	https://www.youtube.com/watch?v=_q1IjK5jyU
Kafka Helm Charts	https://github.com/confluentinc/cp-helm-charts
Confluent for Kubernetes	https://docs.confluent.io/operator/current/overview.html#operator-about-intro