Kafka Commands Primer

Once you have Confluent Platform running, an intuitive next step is try out some basic Kafka commands to create topics and work with producers and consumers. This should help orient Kafka newbies and pros alike that all those familiar Kafka tools are readily available in Confluent Platform, and work the same way. These provide a means of testing and working with basic functionality, as well as configuring and monitoring deployments. The commands surface a subset of the APIs available to you.

Confluent Platform ships with Kafka commands and utilities in `$CONFLUENT_HOME/etc/kafka/bin`. This `bin/` directory includes both Confluent proprietary and open source Kafka utilities.

A few things to note:

- With Confluent Platform installed and running on your system, you can run Kafka commands from anywhere; for example, from your `$HOME` (`~/`) directory. You do *not* have to run these from within `$CONFLUENT_HOME`.
- A full list is provided in CLI Tools for Confluent Platform. Those in the list that begin with `kafka-` are the Kafka open source command utilities.
- Comprehensive command line help is available by typing any of the commands with no arguments; for example, `kafka-topics` or `kafka-producer-perf-test`.

To help get you started, the sections below provide examples for some of the most fundamental and widely-used commands.

Create, list and describe topics

You can use `kafka-topics` for operations on topics (create, list, describe, alter, delete, and so forth).

In a command window, run the following commands to experiment with topics.

1. Create three topics, `cool-topic`, `warm-topic`, `hot-topic`.

```
kafka-topics --create --topic cool-topic --bootstrap-server localhost:9092

kafka-topics --create --topic warm-topic --bootstrap-server localhost:9092

kafka-topics --create --topic hot-topic --partitions 2 --replication-factor 2
--bootstrap-server localhost:9092
```

2. List all topics.

```
kafka-topics --list --bootstrap-server localhost:9092
```

Tip

System topics are prefaced by an underscore in the output. The topics you created are listed at the end.

3. Describe a topic.

   This shows partitions, replication factor, and in-sync replicas for the topic.

```
kafka-topics --describe --topic cool-topic --bootstrap-server localhost:9092
```

   Your output should resemble the following:

```
Topic: cool-topic PartitionCount: 1       ReplicationFactor: 1    Configs: seg
ment.bytes=1073741824
  Topic: cool-topic       Partition: 0    Leader: 0       Replicas: 0     Isr:
0  Offline:
```

Tip

If you run `kafka-topics --describe` with no specified topic, you get a detailed description of every topic on the cluster (system and user topics).

4. Describe another topic, using one of the other brokers in the cluster as the bootstrap server.

```
kafka-topics --describe --topic hot-topic --bootstrap-server localhost:9094
```

   Here is that example output:

```
Topic: hot-topic  PartitionCount: 2       ReplicationFactor: 2    Configs: seg
ment.bytes=1073741824
  Topic: hot-topic        Partition: 0    Leader: 1       Replicas: 1,0   Isr:
1,0        Offline:
  Topic: hot-topic        Partition: 1    Leader: 0       Replicas: 0,2   Isr:
0,2        Offline:
```

   You can connect to any of the brokers in the cluster to run these commands because they all have the same data!

5. Alter a topic’s cofiguration.

   For this example, change the partition count on hot-topic from `2` to `9`.

```
kafka-topics --alter --topic hot-topic --partitions 9 --bootstrap-server local
host:9092
```

Tip

Dynamic topic modification is inherently limited by the current configurations. For example, you cannot decrease the number of partitions or modify the replication factor for a topic, as that would require partition reassignment.

6. Rerun `--describe` on the same topic.

```
kafka-topics --describe --topic hot-topic --bootstrap-server localhost:9092
```

Here is that example output, and verify that the partition count is updated to `9`:

```
Topic: hot-topic  PartitionCount: 9       ReplicationFactor: 2    Configs: seg
ment.bytes=1073741824
  Topic: hot-topic        Partition: 0    Leader: 2       Replicas: 2,1   Isr:
2,1        Offline:
  Topic: hot-topic        Partition: 1    Leader: 1       Replicas: 1,0   Isr:
1,0        Offline:
  Topic: hot-topic        Partition: 2    Leader: 1       Replicas: 1,2   Isr:
1,2        Offline:
  Topic: hot-topic        Partition: 3    Leader: 2       Replicas: 2,1   Isr:
2,1        Offline:
  Topic: hot-topic        Partition: 4    Leader: 0       Replicas: 0,2   Isr:
0,2        Offline:
  Topic: hot-topic        Partition: 5    Leader: 1       Replicas: 1,0   Isr:
1,0        Offline:
  Topic: hot-topic        Partition: 6    Leader: 2       Replicas: 2,0   Isr:
2,0        Offline:
  Topic: hot-topic        Partition: 7    Leader: 0       Replicas: 0,1   Isr:
0,1        Offline:
  Topic: hot-topic        Partition: 8    Leader: 1       Replicas: 1,2   Isr:
1,2        Offline:
```

7. Delete a topic.

```
kafka-topics --delete --topic warm-topic --bootstrap-server localhost:9092
```

8. List all topics.

```
kafka-topics --list --bootstrap-server localhost:9092
```

Run producers and consumers to send and read messages

The command utilities `kafka-console-producer` and `kafka-console-consumer` allow you to manually produce messages to and consume from a topic.

1. Open two new command windows, one for a producer, and the other for a consumer.
2. Run a producer to produce to `cool-topic`.

```
kafka-console-producer --topic cool-topic --bootstrap-server localhost:9092
```

3. Send some messages.

   Type your messages at the prompt (`>`), and hit Return after each one.

   Your command window will resemble the following:

```
$ kafka-console-producer --broker-list localhost:9092 --topic cool-topic
>hi cool topic
>did you get this message?
>first
>second
>third
>yes! I love you cool topic
>
```

Tip

You can use the `--broker-list` flag in place of `--bootstrap-server` for the producer, typically used to send data to specific brokers; shown here as an example.

4. In the other command window, run a consumer to read messages from `cool-topic`. Specify that you want to start consuming from the beginning, as shown.

```
kafka-console-consumer --topic cool-topic --from-beginning --bootstrap-server
localhost:9092
```

   Your output will resemble the following:

```
$ kafka-console-consumer --bootstrap-server localhost:9092 --from-beginning --
topic cool-topic
hi cool topic on origin cluster
is this getting to your replica?
first
second
third
yes! I love you cool topic
```

5. When you want to stop the producer and consumer, type Ctl-C in their respective command windows.

Tip

You may want to leave at least the producer running for now, in case you want to send more messages when we revisit topics on the Control Center.

Produce auto-generated message data to topics

You can use `kafka-consumer-perf-test` in its own command window to generate test data to topics.

- For example, open a new command window and type the following command to send data to `hot-topic`, with the specified throughput and record size.

```
kafka-producer-perf-test \
   --producer-props bootstrap.servers=localhost:9092 \
   --topic hot-topic \
   --record-size 1000 \
   --throughput 1000 \
   --num-records 3600000
```

  The command provides status output on messages sent, as shown:

```
4999 records sent, 999.8 records/sec (0.95 MB/sec), 1.1 ms avg latency, 240.0
ms max latency.
5003 records sent, 1000.2 records/sec (0.95 MB/sec), 0.5 ms avg latency, 4.0 m
s max latency.
5003 records sent, 1000.2 records/sec (0.95 MB/sec), 0.6 ms avg latency, 5.0 m
s max latency.
5001 records sent, 1000.2 records/sec (0.95 MB/sec), 0.3 ms avg latency, 3.0 m
s max latency.
5001 records sent, 1000.0 records/sec (0.95 MB/sec), 0.3 ms avg latency, 4.0 m
s max latency.
5000 records sent, 1000.0 records/sec (0.95 MB/sec), 0.8 ms avg latency, 24.0
ms max latency.
5001 records sent, 1000.2 records/sec (0.95 MB/sec), 0.6 ms avg latency, 3.0 m
s max latency.
...
```

- Open a new command window to consume the messages from hot-topic as they are sent (not from the beginning).

```
kafka-console-consumer --topic hot-topic --bootstrap-server localhost:9092
```

  Type Ctl-C to stop the consumer.

Tip

You may want to leave the producer running for a moment, as you are about to revisit Topics on the Control Center.

Revisit Control Center (Optional)

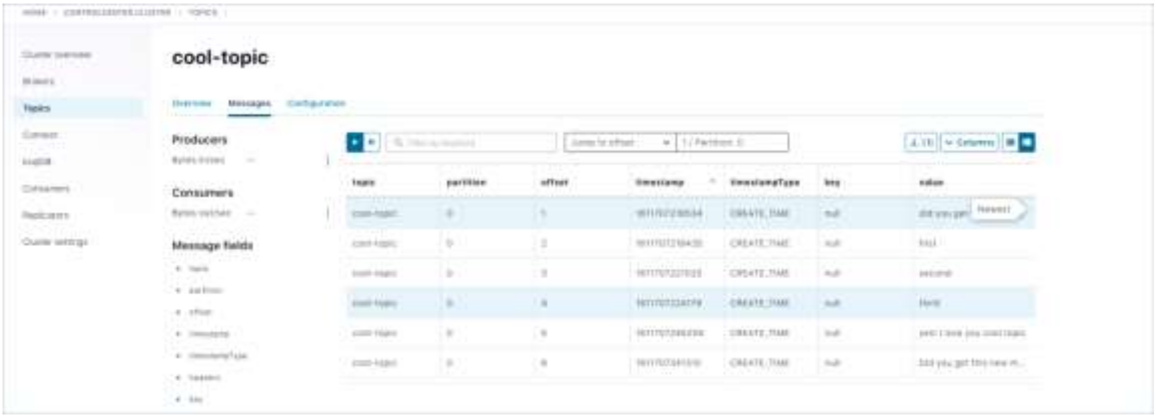
Now that you have created some topics and produced message data to a topic (both manually and with auto-generated), take another look at Control Center, this time to inspect the existing topics.

1. Open a web browser and go to http://localhost:9021/, the default URL for Control Center on a local system.
2. Select the cluster, and click **Topics** from the menu.
3. Choose `cool-topic`, then select the **Messages** tab.

   Select **Jump to offset** and type `1`, `2`, or `3` to display previous messages.

   These messages do not show in the order they were sent because the consumer here is not reading `--from-beginning`.

   Try manually typing some more messages to `cool-topic` with your command line producer, and watch them show up here.

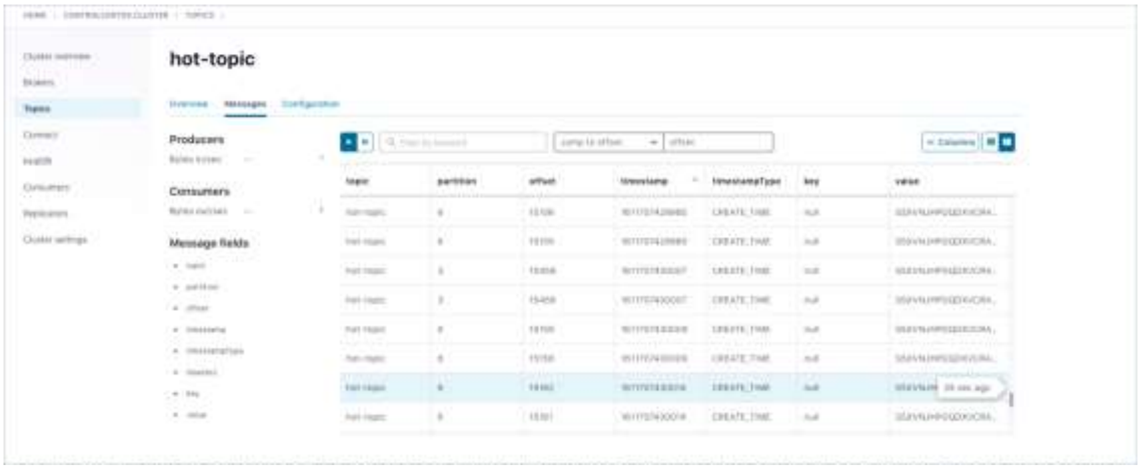
   

   cool-topic

topic	partition	offset	timestamp	timestampType	key	value
cool-topic	0	1	[illegible]	CREATE_TIME	null	did you get
cool-topic	0	2	[illegible]	CREATE_TIME	null	[illegible]
cool-topic	0	3	[illegible]	CREATE_TIME	null	second
cool-topic	0	[illegible]	[illegible]	CREATE_TIME	null	[illegible]
cool-topic	0	[illegible]	[illegible]	CREATE_TIME	null	[illegible]
cool-topic	0	[illegible]	[illegible]	CREATE_TIME	null	[illegible]

4. Navigate to **Topics** > `hot-topic` > **Messages** tab.

   Auto-generated messages from your `kafka-producer-perf-test` are shown here as they arrive.

   hot-topic

topic	partition	offset	timestamp	timestampType	key	value
hot-topic	[illegible]	[illegible]	[illegible]	CREATE_TIME	null	[illegible]
hot-topic	[illegible]	[illegible]	[illegible]	CREATE_TIME	null	[illegible]
hot-topic	[illegible]	[illegible]	[illegible]	CREATE_TIME	null	[illegible]
hot-topic	[illegible]	15459	[illegible]	CREATE_TIME	null	[illegible]
hot-topic	[illegible]	[illegible]	[illegible]	CREATE_TIME	null	[illegible]
hot-topic	[illegible]	[illegible]	[illegible]	CREATE_TIME	null	[illegible]
hot-topic	[illegible]	[illegible]	[illegible]	CREATE_TIME	null	[illegible]
hot-topic	[illegible]	[illegible]	[illegible]	CREATE_TIME	null	[illegible]

Shutdown and cleanup tasks

Run the following shutdown and cleanup tasks.

- Stop the `kafka-producer-perf-test` with Ctl-C in its respective command window.
- Stop the all of the other components with Ctl-C in their respective command windows, in reverse order in which you started them. For example, stop Control Center first, then other components, followed by Kafka brokers, and finally ZooKeeper.