Cloudera Runtime 7.1.2

Managing Cruise Control

Date published: 2020-05-04
Date modified: 2020-06-16

https://docs.cloudera.com/

Legal Notice

Contents

Rebalancing with Cruise Control

The rebalancing process of Cruise Control works in the background. You do not need to adjust or set anything during the process. You can use the REST endpoints to communicate with Cruise Control in the command line.

The default rebalance command for Cruise Control is /rebalance?dryrun=true. It is recommended to leave the dryrun value as the default when running the rebelancing command for the first time, and then changing the value to false when later using Cruise Control rebalancing.

For the list of REST endpoints, see the Cruise Control API reference document.

Note: If the rebalancing process does not start, then Cruise Control does not have enough metric information to start the process. In this case, you need to set the resources and restart the metrics fetching.

Related Information

Cruise Control REST API reference document

Cruise Control REST API endpoints

You can configure and check the status of Cruise Control using the REST API with GET and POST endpoints. You must be aware of the required action or query, and submit a curl command in the command line to use the Cruise Control REST API.

There are two types of REST endpoints for Cruise Control: GET and POST .

For GET

With GET endpoints, you can query information about the rebalancing process, the status of Cruise Control and Kafka brokers. The GET endpoints are read-only operations, and do not have any impact on Cruise Control, Kafka, or the rebalancing process.

For POST

You can use the POST endpoints to change the rebalancing process, modify the number of Kafka brokers, configure Cruise Control, and customize certain sampling and proposal tasks.

In the command line, you need to create the POST or GET commands with curl, and also include the Cruise Control hostname, port, and the required endpoint.

For GET

```
curl -X GET "http://<cruise_control_hostname>:8899/kafkacruisecontrol/ka
fka_cluster_state”
```

For POST

```
curl -X POST “http://<cruise_control_hostname>:8899/kafkacruisecontrol/a
dd_broker?brokerid=24”
```

The following table summarizes the available operations for GET and POST endpoints. The description column only provides a brief summary of the endpoints. For the detailed description of the Cruise Control REST API, see the official REST API documentation.

For GET

Endpoint	Action	Description
/kafkacruisecontrol/state	You can query the state of Kafka Cruise Control.	This includes the state of monitor, executor, analyzer, and anomaly detector.
/kafkacruisecontrol/load	You can query the current cluster load.	The Load Monitor needs to be in RUNNING state. This includes the load-per-broker and load-per-host information. Only shows valid partitions with enough metric samples.
/kafkacruisecontrol/partition_load	You can query partition resource utilization.	The partition load is sorted based on the given resource.
/kafkacruisecontrol/kafka_cluster_state	You can query partition and replica state.	This includes information for each broker and each partition.
/kafkacruisecontrol/proposals	You can query the optimization proposals generated on the workload model.	The proposals can be generated based on either the snapshot window or the valid partitions.
/kafkacruisecontrol/user_tasks	You can query the user request result.	This includes a full list of all the active and completed tasks in Cruise Control.

For POST

Endpoint	Action	Description
/kafkacruisecontrol/rebalance	You can trigger a workload balance for a Kafka cluster.	Can be based on valid windows or valid partitions. You can specify the goal you want to use.
/kafkacruisecontrol/add_broker?brokerid=[id1,id2...]	You can add a list of brokers from a Kafka cluster.	Only moves replicas from existing brokers to the newly added brokers. You can choose to throttle the movement of the replica to the newly added broker. The replica movement is throttled on the existing broker.
/kafkacruisecontrol/remove_broker?brokerid=[id1,id2...]	You can remove a list of brokers from a Kafka cluster.	Only moves partitions from the brokers to be removed to the other existing brokers. You can also specify the destination broker. You can choose to throttle the removed broker during the partition movement.
/kafkacruisecontrol/fix_offline_replicas	You can fix offline replicas in a Kafka cluster.	If the specified topic has offline replicas, they are still moved to healthy brokers.
/kafkacruisecontrol/demote_broker?brokerid=[id1, id2...]	You can move all the leader replicas from a list of brokers.	This makes all the replicas on a given broker to be preferred the least, and triggers a preferred leader replica election to migrate the leader replicas off the broker.
/kafkacruisecontrol/demote_broker?brokerid_and_logdirs=[id1-logdir1, id2-logdir2...]	You can move all the leader replicas from a list of disks.	This makes all the replicas on a given disk to be preferred the least, and triggers a preferred leader replica election to migrate the leader replicas off the disk.
/kafkacruisecontrol/stop_proposal_execution	You can stop an ongoing proposal execution task.	You can stop a proposal when the execution of a task is still ongoing.
/kafkacruisecontrol/pause_sampling	You can pause an ongoing metrics sampling process.	The cause of the pause is recorded and shows up when querying the state of Cruise Control.
/kafkacruisecontrol/resume_sampling	You can resume a paused metrics load sampling.	The cause of the resumption is recorded and shows up when querying the state of Cruise Control.
/kafkacruisecontrol/topic_configuration?topic=[topic_regex]&replication_factor=[target_replication_factor]	You can change the replication factor of a topic.	Existing replicas are not moved. You can determine which replica should be deleted and which broker should be assigned to the new replica.

Endpoint	Action	Description
/kafkacruisecontrol/admin?disable_self_healing_for=[anomaly_type]	You can enable or disable self-healing for Cruise Control.	The configuration is only possible with administrator rights.
/kafkacruisecontrol/admin?concurrent_partition_movements_per_broker=[integer]	You can increase or decrease the execution concurrency.	The configuration is only possible with admininstrator rights.
/kafkacruisecontrol/admin?drop_recently_removed_brokers=[broker_ids]	You can drop recently removed or demoted Kafka brokers.	The configuration is only possible with administrator rights.

Rebalance after adding Kafka broker

When your application requires more Kafka brokers, you can add new ones without interferring with the rebalancing process using the command line.

Procedure

1. Open Cloudera Manager.
2. Add a new Kafka broker.
3. In the command line, check the status of the Kafka cluster with the /kafka_cluster_state command.
4. Check the status of Cruise Control with /state command to make sure that everything is set before adding the broker.
5. Call the /add_broker command.

   See the REST API reference document for the needed endpoint parameter.

   Cruise Control displays the rebalancing plan.
6. Check the status of the Kafka cluster with the /kafka_cluster_state command.
7. Check the status of Cruise Control with the /state command.

Results

The new broker is added to the cluster.

Related Information

Cruise Control REST API reference document

Rebalance after demoting Kafka broker

When a broker needs maintenance during the rebalance process, you can demote the Kafka broker using the command line. After the maintenance is completed, the rebalancing process can be continued.

About this task

When a broker must undergo maintenance, demoting can be used for removing partition leadership from the given broker.

Procedure

1. In the command line, check the status of the Kafka cluster with the /kafka_cluster_state command.
2. Check the status of Cruise Control with /state command to make sure that everything is set before demoting the broker.

3. Run the /demote_broker command.

   See the REST API reference document for the needed endpoint parameter.

   Note: If there is one replica of a partition, the partition leadership cannot be moved by demoting. Demoting only moves leaders.

   Cruise Control displays the cluster load after the demoting operation.
4. Check the status of the Kafka cluster with the /kafka_cluster_state command.
5. Check the status of Cruise Control with the /state command.

Results

The demoted broker does not contain any partition leaders.

Related Information

Cruise Control REST API reference document

Rebalance after removing Kafka broker

When your application requires that you remove certain Kafka brokers, you can do so using the command line, and continue the rebalancing process with the remaining ones.

Procedure

1. In the command line, check the status of the Kafka cluster with the /kafka_cluster_state command.
2. Check the status of Cruise Control with the /state command to make sure that everything is set before removing the broker.
3. Run the /remove_broker command.

   See the REST API reference document for the needed endpoint parameter.

   Cruise Control displays the plan to move all the partitions from the broker that is removed to the other available brokers.
4. Check the status of Cruise Control using the /state command.

   Note: Cruise Control displays the inter_broker_reaplica_movement task as in-progress and gives you a general summary of the rebalancing result after removing the broker.
5. Check the status of the Kafka cluster with the /kafka_cluster_state command.
6. Open Cloudera Manager.
7. Remove the Kafka role from the cluster.

Results

The Kafka cluster is fully functional even after the broker is removed.

Related Information

Cruise Control REST API reference document