Cloudera Runtime 7.0.2

Configuring Apache Kafka

Date published: 2019-12-18
Date modified:

https://docs.cloudera.com/

Legal Notice

Contents

Operating system requirements 4

Performance considerations 4

Quotas 5

JBOD 5

JBOD setup 6

JBOD Disk migration 7

Setting user limits for Kafka 9

Connecting Kafka clients to Data Hub provisioned clusters 9

Operating system requirements

A collection of operating system requirements for Kafka.

SUSE Linux Enterprise Server (SLES)

Unlike CentOS, SLES limits virtual memory by default. Changing this default requires adding the following entries to the /etc/security/limits.conf file:

```
* hard as unlimited
* soft as unlimited
```

Kernel Limits

There are three settings you must configure properly for the kernel.

File Descriptors

You can set file descriptors in Cloudera Manager by going to KafkaConfigurationMaximum Process File Descriptors and setting the required value. Cloudera recommends a configuration of 100000 or higher.

Max Memory Map

You must configure the maximum number of memory maps in your specific kernel settings. Cloudera recommends a configuration of 32000 or higher.

Max Socket Buffer Size

Set the buffer size larger than any Kafka send buffers that you define.

Performance considerations

A collection of basic recommendations for Kafka clusters.

The simplest recommendation for running Kafka with maximum performance is to have dedicated hosts for the Kafka brokers and a dedicated ZooKeeper cluster for the Kafka cluster. If that is not an option, consider these additional guidelines for resource sharing with the Kafka cluster:

Running in VMs

It is common practice in modern data centers to run processes in virtual machines. This generally allows for better sharing of resources. Kafka is sufficiently sensitive to I/O throughput that VMs interfere with the regular operation of brokers. For this reason, it is generally not recommended to run Kafka in VMs. However, if you are running Kafka in a virtual environment you will need to rely on your VM vendor for help with optimizing Kafka performance.

Do not run other processes with Brokers or ZooKeeper

Due to I/O contention with other processes, it is generally recommended to avoid running other such processes on the same hosts as Kafka brokers.

Keep the Kafka-ZooKeeper Connection Stable

Kafka relies heavily on having a stable ZooKeeper connection. Putting an unreliable network between Kafka and ZooKeeper will appear as if ZooKeeper is offline to Kafka. Examples of unreliable networks include:

- Do not put Kafka/ZooKeeper nodes on separated networks
- Do not put Kafka/ZooKeeper nodes on the same network with other high network loads

Quotas

Learn about Quotas and how to set them.

For a quick video introduction to quotas, see Quotas.

Kafka can enforce quotas on produce and fetch requests. Producers and consumers can use very high volumes of data. This can monopolize broker resources, cause network saturation, and generally deny service to other clients and the brokers themselves. Quotas protect against these issues and are important for large, multi-tenant clusters where a small set of clients using high volumes of data can degrade the user experience.

Quotas are byte-rate thresholds, defined per client ID. A client ID logically identifies an application making a request. A single client ID can span multiple producer and consumer instances. The quota is applied for all instances as a single entity. For example, if a client ID has a produce quota of 10 MB/s, that quota is shared across all instances with that same ID.

When running Kafka as a service, quotas can enforce API limits. By default, each unique client ID receives a fixed quota in bytes per second, as configured by the cluster (quota.producer.default, quota.consumer.default). This quota is defined on a per-broker basis. Each client can publish or fetch a maximum of *X* bytes per second per broker before it gets throttled.

The broker does not return an error when a client exceeds its quota, but instead attempts to slow the client down. The broker computes the amount of delay needed to bring a client under its quota and delays the response for that amount of time. This approach keeps the quota violation transparent to clients (outside of client-side metrics). This also prevents clients from having to implement special backoff and retry behavior.

You can override the default quota for client IDs that need a higher or lower quota. The mechanism is similar to per-topic log configuration overrides. Write your client ID overrides to ZooKeeper under /config/clients. All brokers read the overrides, which are effective immediately. You can change quotas without having to do a rolling restart of the entire cluster.

By default, each client ID receives an unlimited quota. The following configuration sets the default quota per producer and consumer client ID to 10 MB/s.

```
quota.producer.default=10485760
quota.consumer.default=10485760
```

To set quotas using Cloudera Manager, open the Kafka Configuration page and search for Quota. Use the fields provided to set the Default Consumer Quota or Default Producer Quota.

JBOD

Overview on Kafka with JBOD.

JBOD refers to a system configuration where disks are used independently rather than organizing them into redundant arrays (RAID). Using RAID usually results in more reliable hard disk configurations even if the individual disks are not reliable. RAID setups like these are common in large scale big data environments built on top of commodity hardware. RAID enabled configurations are more expensive and more complicated to set up. In a large number of environments, JBOD configurations are preferred for the following reasons:

- Reduced storage cost: RAID-10 is recommended to protect against disk failures. However, scaling RAID-10 configurations can become excessively expensive. Storing the data redundantly on each node means that storage space requirements have to be multiplied because the data is also replicated across nodes.
- Improved performance: Just like HDFS, the slowest disk in RAID-10 configuration limits overall throughput. Writes need to go through a RAID controller. On the other hand, when using JBOD, IO performance is increased as a result of isolated writes across disks without a controller.

JBOD setup

Learn how to set up JBOD in your Kafka environment.

Before you begin

Consider the following before using JBOD support in Kafka:

- Manual operation and administration: Monitoring offline directories and JBOD related metrics is done through Cloudera Manager. However, identifying failed disks and rebalancing partitions between disks is done manually.
- Manual load balancing between disks: Unlike with RAID-10, JBOD does not automatically distribute data across disks. The process is fully manual.

To provide robust JBOD support in Kafka, changes in the Kafka protocol have been made. When performing an upgrade to a new version of Kafka, make sure that you follow the recommended rolling upgrade process.

For more information regarding the JBOD related Kafka protocol changes, see KIP-112 and KIP-113.

Procedure

1. Mount the required number of disks on your system.
2. In Cloudera Manager, set up log directories for all Kafka brokers:
   a) Go to the Kafka service, select Instances and select the broker.
   b) Go to Configuration and find the Data Directories property.
   c) Modify the path of the log directories so that they correspond with the newly mounted disks.

      Note: Depending on your, setup you may need to add or remove multiple data directories.

   d) Enter a Reason for change, and then click Save Changes to commit the changes.
3. Go to the Kafka service and select Configuration.
4. Find and configure the following properties depending on your system and use case.
   - Number of I/O Threads
   - Number of Network Threads
   - Number of Replica Fetchers
   - Minimum Number of Replicas in ISR
5. Set replication factor to at least 3.

   Important: If you set replication factor to less than 3, your data will be at risk. In addition, in case of a disk failure, disk maintenance cannot be carried out without system downtime.

6. Restart the service:
   a) Return to the home page by clicking hte Cloudera Manager logo.
   b) Go to the Kafka service and select Actions Rolling Restart
   c) Check the Restart roles with stale configurations only checkbox and click Rolling restart.
   d) Click Close when the restart has finished.

Results

JBOD disks are set up in your Kafka environment.

Related Information

KIP-112
KIP-113

JBOD Disk migration

Learn how to migrate existing Kafka partitions to JBOD configured disks.

About this task

Migrating data from one disk to another is achieved with the kafka-reassign-partitions tool. The following instructions focus on migrating existing Kafka partitions to JBOD configured disks.

Note: Cloudera recommends that you minimize the volume of replica changes per command instance. Instead of moving 10 replicas with a single command, move two at a time in order to save cluster resources.

Before you begin

- Set up JBOD in your Kafka environment. For more information, see JBOD Setup.
- Collect the log directory paths on the JBOD disks where you want to migrate existing data.
- Collect the broker IDs of the brokers you want to migrate data to.
- Collect the name of the topics you want to migrate partitions from.

Note: Output examples in these instructions are cleaned and formatted to make them easily readable.

Procedure

1. Create a topics-to-move JSON file that specifies the topics you want to reassign. Use the following format:

   Use the following format:

   ```
   {"topics":  [{"topic": "mytopic1"},
                {"topic": "mytopic2"}],
    "version":1
   }
   ```

2. Generate the content for the reassignment configuration JSON with the following command:

   ```
   kafka-reassign-partitions --zookeeper hostname:port --topics-to-move-json-
   file topics to move.json --broker-list broker 1, broker 2 --generate
   ```

   Running the command lists the distribution of partition replicas on your current brokers followed by a proposed partition reassignment configuration.

   Example output:

   ```
   Current partition replica assignment
   {"version":1,
    "partitions":
      [{"topic":"mytopic2","partition":1,"replicas":[2,3],"log_dirs":["any","
   any"]},
       {"topic":"mytopic1","partition":0,"replicas":[1,2],"log_dirs":["any"
   ,"any"]},
       {"topic":"mytopic2","partition":0,"replicas":[1,2],"log_dirs":["any","
   any"]},
       {"topic":"mytopic1","partition":2,"replicas":[3,1],"log_dirs":["any"
   ,"any"]},
       {"topic":"mytopic1","partition":1,"replicas":[2,3],"log_dirs":["any","
   any"]}]
   }

   Proposed partition reassignment configuration
   ```

```
{"version":1,
 "partitions":
   [{"topic":"mytopic1","partition":0,"replicas":[4,5],"log_dirs":["any",
"any"]},
    {"topic":"mytopic1","partition":2,"replicas":[4,5],"log_dirs":["any
","any"]},
    {"topic":"mytopic2","partition":1,"replicas":[4,5],"log_dirs":["any",
"any"]},
    {"topic":"mytopic1","partition":1,"replicas":[5,4],"log_dirs":["any
","any"]},
    {"topic":"mytopic2","partition":0,"replicas":[5,4],"log_dirs":["any",
"any"]}]
}
```

In this example, the tool proposed a configuration which reassigns existing partitions on broker 1, 2, and 3 to brokers 4 and 5.

3. Copy and paste the proposed partition reassignment configuration into an empty JSON file.
4. Modify the suggested reassignment configuration.

   When migrating data you have two choices. You can move partitions to a different log directory on the same broker, or move it to a different log directory on another broker.

   a. 1. To reassign partitions between log directories on the same broker, change the appropriate any entry to an absolute path. For example:

      ```
      {"topic":"mytopic1","partition":0,"replicas":[4,5],"log_dirs":["/
      JBOD-disk/directory1","any"]}
      ```

      2. To reassign partitions between log directories across different brokers, change the broker ID specified in replicas and the appropriate any entry to an absolute path. For example:

      ```
      {"topic":"mytopic1","partition":0,"replicas":[6,5],"log_dirs":["/
      JBOD-disk/directory1","any"]}
      ```

5. Save the file.
6. Start the redistribution process with the following command:

   ```
   kafka-reassign-partitions --zookeeper hostname:port  --reassignment-json-
   file reassignment configuration.json --bootstrap-server hostname:port --e
   xecute
   ```

   Important: The bootstrap server has to be specified with the --bootstrap-server option if an absolute log directory path is specified for a replica in the reassignment configuration JSON file.

   The tool prints a list containing the original replica assignment and a message that reassignment has started. Example output:

   ```
   Current partition replica assignment

   {"version":1,
    "partitions":
      [{"topic":"mytopic2","partition":1,"replicas":[2,3],"log_dirs":["any",
   "any"]},
       {"topic":"mytopic1","partition":0,"replicas":[1,2],"log_dirs":["any
   ","any"]},
       {"topic":"mytopic2","partition":0,"replicas":[1,2],"log_dirs":["any",
   "any"]},
       {"topic":"mytopic1","partition":2,"replicas":[3,1],"log_dirs":["any
   ","any"]},
       {"topic":"mytopic1","partition":1,"replicas":[2,3],"log_dirs":["any",
   "any"]}]
   ```

```
}

Save this to use as the --reassignment-json-file option during rollback
Successfully started reassignment of partitions.
```

7. Verify the status of the reassignment with the following command:

```
kafka-reassign-partitions --zookeeper hostname:port --reassignment-json-
file reassignment configuration.json  --bootstrap-server hostname:port --v
erify
```

The tool prints the reassignment status of all partitions. Example output:

```
Status of partition reassignment:
Reassignment of partition mytopic2-1 completed successfully
Reassignment of partition mytopic1-0 completed successfully
Reassignment of partition mytopic2-0 completed successfully
Reassignment of partition mytopic1-2 completed successfully
Reassignment of partition mytopic1-1 completed successfully
```

Results

Existing Kafka partitions are migrated to JBOD configured disks.

Related Information

JBOD setup

kafka-reassign-partitions

Setting user limits for Kafka

Learn more about Kafka User limits and how to monitor them.

Kafka opens many files at the same time. The default setting of 1024 for the maximum number of open files on most Unix-like systems is insufficient. Any significant load can result in failures and cause error messages such as java.io.IOException...(Too many open files) to be logged in the Kafka or HDFS log files. You might also notice errors such as this:

```
ERROR Error in acceptor (kafka.network.Acceptor)
java.io.IOException: Too many open files
```

Cloudera recommends setting the value to a relatively high starting point, such as 32,768.

You can monitor the number of file descriptors in use on the Kafka Broker dashboard. In Cloudera Manager:

1. Go to the Kafka service.
2. Select a Kafka Broker.
3. Open Charts Library Process Resources and scroll down to the File Descriptors chart.

Connecting Kafka clients to Data Hub provisioned clusters

Learn how to connect Kafka clients to clusters provisioned with Data Hub.

About this task

Use the following steps to connect Kafka clients to clusters provisioned with Data Hub. Configuration examples provided in this list of steps assume that the cluster you are connecting to was provisioned with a Streams Messaging cluster definition.

Before you begin

- If you are connecting your clients from outside the virtual network, verify that both inbound and outbound traffic is enabled on the port used by Kafka brokers for secure communication. The default port is 9093. For more information, see Security Groups for Your VPC in the Amazon Virtual Private Cloud User Guide.
- If you are connecting your clients over the internet, verify that your virtual network has a public IP address assigned to it. For more information, see IP Addressing in Your VPC in the Amazon Virtual Private Cloud User Guide.
- Clients connecting to Data Hub provisioned clusters require a CDP user account that provides access to the required CDP resources. Verify that a CDP user account with the required roles and permissions is available for use. If not, create one. Any type of CDP user account can be used. If you are creating a new account to be used by Kafka clients, Cloudera recommends that you create a machine user account. For more information, see User Management in the Cloudera Management Console documentation.
- In addition to the CDP user account having access to the required CDP resources, the user account also needs to have the correct policies assigned to it in Ranger. Otherwise, the client cannot perform tasks on Kafka resources. These policies are specified within the Ranger instance that provides authorization to the Kafka service you want to connect to. For more information, see the Cloudera Runtime documentation on Apache Ranger and the Kafka specific Ranger documentation.

Procedure

1. Obtain the FreeIPA certificate of your environment:
   a) From the CDP Home Page navigate to Management Console Environments .
   b) Locate and select your environment from the list of available environments.
   c) Go to the Summary tab.
   d) Scroll down to the FreeIPA section.
   e) Click ActionsGet FreeIPA Certificate.

   The FreeIPA certificate file, *[***ENVIRONMENT NAME***]*-env.crt, is downloaded to your computer.
2. Add the FreeIPA certificate to the truststore of the client.

   The certificate needs to be added for all clients that you want to connect to the Data Hub provisioned cluster. The exact steps of adding the certificate to the truststore depends on the platform and key management software used. For example, you can use the Java Keytool command line tool:

   ```
   keytool -import -keystore [***CLIENT TRUSTSTORE.JKS***] -al
   ias [***ALIAS***] -file [***FREEIPA CERTIFICATE***]
   ```

   

   Tip: This command creates a new truststore file if the file specified with the -keystore option does not exist.

3. Obtain CDP workload credentials:

   A valid workload username and password has to be provided to the client, otherwise it cannot connect to the cluster. Credentials can be obtained from Management Console.

   a) From the CDP Home Page navigate to Management Console User Management.
   b) Locate and select the user account you want to use from the list of available accounts.
      The user details page displays information about the user.
   c) Find the username found in the Workload Username entry and note it down.
   d) Find the Workload Password entry and click Set Workload Password.
   e) In the dialog box that appears, enter a new workload password, confirm the password and note it down.
   f) Fill out the Environment text box.
   g) Click Set Workload Password and wait for the process to finish.
   h) Click close.

4. Configure clients.

   In order for clients to be able to connect to Kafka brokers, all required security related properties have to be added to the client's properties file. The following example configuration lists the default properties that are needed when connecting clients to a cluster provisioned by Data Hub with a Streams Messaging cluster definition. If you made changes to the security configuration of the brokers, or provisioned a custom cluster with non-default Kafka security settings, make sure to change the appropriate parameters in the client configuration as well.

   ```
   security.protocol=SASL_SSL
   sasl.mechanism=PLAIN
   ssl.truststore.location=[***CLIENT TRUSTSTORE.JKS***]
   ssl.truststore.password=[***TRUSTSTORE PASSWORD***]
   sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
    required \
       username="[***USERNAME***]" \
       password="[***PASSWORD***]";
   ```

   Replace *[***CLIENT TRUSTSTORE.JKS***]* with the path to the client's truststore file.This is the same file that you added the FreeIPA certificate to in Step 2 on page 10.

   Replace *[***TRUSTSTORE PASSWORD***]* with the password of the truststore file.

   Replace *[***USERNAME***]* and *[***PASSWORD***]* with the workload username and password obtained in Step 3 on page 11.

5. Obtain Kafka broker hostnames:

   You can obtain the Kafka broker hostnames from the Cloudera Manager UI.

   a) From the CDP Home Page navigate to Management Console Environments .
   b) Locate and select your environment from the list of available environments.
   c) Select the Data Hub cluster you want to connect to from the list of available clusters.
   d) Click the link found in the Cloudera Manger Info section.
      You are redirected to the Cloudera Manager web UI.
   e) Click Clusters and select the cluster that the Kafka service is running on.

      The default name for clusters created with a Streams Messaging Cluster definition is streams-messaging.
   f) Select the Kafka service.
   g) Go to Instances.

      The Kafka broker hostnames are listed in the Hostname column.

6. Connect clients to brokers.

   Connect the clients by supplying them with the broker hostnames obtained in step 5 on page 11. The actions you need to take differ depending on the type of client you are using.

   Custom developed Kafka Applications

   When producing or consuming messages with your own Kafka client application, you have to provide the Kafka broker hostnames within the client code.

   Kafka console producer and consumer

   When producing or consuming messages with the kafka-console-consumer or kafka-console-producer command line tools, run the producer or consumer with the appropriate hostnames. Additionally, you must also pass the client properties file containing the security related properties with --producer.config or --consumer.config. For example:

   ```
   kafka-console-producer --broker-
   list [***HOSTNAME***]:[***PORT***] --topic [***TOPIC***] --produ
   cer.config [***CLIENT PROPERTIES FILE***]
   ```

   ```
   kafka-console-consumer --bootstrap-serve
   r [***HOSTNAME***]:[***PORT***] --topic [***TOPIC***] --from-begi
   nning --consumer.config [***CLIENT PROPERTIES FILE***]
   ```

   Replace *[***CLIENT PROPERTIES FILE***]* with the path to the client's properties file. This is the same file that you updated in Step 4 on page 11.

Results

Kafka clients are configured and are able to connect to Data Hub provisioned clusters.