

Migrating Kudu to CDP Private Cloud

Date published: 2019-08-22

Date modified: 2021-09-21



Legal Notice

© Cloudera Inc. 2024. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

- Kudu migration overview..... 4**
 - Backing up data in Kudu..... 4
 - Applying custom Kudu configuration..... 4
 - Copying the backed up data to the CDP cluster..... 5
 - Restoring Kudu data into the target CDP cluster..... 5

Kudu migration overview

When you migrate your Kudu data from CDH to CDP you have to use the Kudu backup tool to back up and then restore your Kudu data.

Backing up data in Kudu

You can back up all your data in Kudu using the `kudu-backup-tools.jar` Kudu backup tool.

The Kudu backup tool runs a Spark job that builds the backup data file and writes it to HDFS or AWS S3, based on what you specify. Note that if you are backing up to S3, you have to provide S3 credentials to `spark-submit` as described in [Specifying Credentials to Access S3 from Spark](#)

The Kudu backup tool creates a full backup of your data on the first run. Subsequently, the tool creates incremental backups.



Important: Incremental backup and restore functionality is available only CDH 6.3.0 and later. Therefore, if you have active ingest processes, such as Spark jobs, Impala SQL batches, or Nifi inserting or updating data in Kudu, you might need to pause these processes before starting full backup to avoid losing data changes happening after starting the Kudu backup process.

Run the following command to start the backup process:

```
spark-submit --class org.apache.kudu.backup.KuduBackup <path to kudu-
backup2_2.11-1.12.0.jar> \
--kuduMasterAddresses <addresses of Kudu masters> \
--rootPath <path to store the backed up data> \
<table_name>
```

where

- `--kuduMasterAddresses` is used to specify the addresses of the Kudu masters as a comma-separated list. For example, `master1-host,master-2-host,master-3-host` which are the actual hostnames of Kudu masters.
- `--rootPath` is used to specify the path to store the backed up data. It accepts any Spark-compatible path.
 - Example for HDFS: `hdfs:///kudu-backups`
 - Example for AWS S3: `s3a://kudu-backup/`

If you are backing up to S3 and see the “Exception in thread “main” java.lang.IllegalArgumentException: path must be absolute” error, ensure that S3 path ends with a forward slash (/).

- `<table_name>` can be a table or a list of tables to be backed up.

Example:

```
spark-submit --class org.apache.kudu.backup.KuduBackup /opt/cloudera/
parcels/CDH-7.2.1-1.cdh7.2.1.p0.4041380/lib/kudu/kudu-backup2_2.11.jar \
--kuduMasterAddresses cluster-1.cluster_name.root.hwx.site,cluster-2.cluster_name.root.hwx.site \
--rootPath hdfs:///kudu-backups \
my_table
```

Applying custom Kudu configuration

If you applied any custom Kudu configurations in your CDH clusters, then you manually have to apply those configurations in your target CDP cluster.

If you have changed the value of `tablet_history_max_age_sec` and you plan to run incremental backups of Kudu on the target cluster, we recommend resetting `tablet_history_max_age_sec` to the default value of 1 week (see <https://issues.apache.org/jira/browse/KUDU-2677>).

Examples of commonly modified configuration flags:

- `rpc_max_message_size`
- `tablet_transaction_memory`
- `rpc_service_queue_length`
- `raft_heartbeat_interval`
- `heartbeat_interval_ms`
- `memory_limit_hard_bytes`
- `block_cache_capacity_mb`

Once you manually applied custom configurations, restart the Kudu cluster.

For more information about Kudu configuration, see *Configuring Apache Kudu*.

Related Information

[Configuring Apache Kudu](#)

Copying the backed up data to the CDP cluster

You can copy your backed up data to the target CDP cluster in two ways: using `distcp` or using the Replication Manager.

Using `distcp` to copy the backed up data

Use the following command to copy your backed up Kudu data to the target CDP cluster:

```
sudo -u hdfs hadoop distcp hdfs:///kudu/kudu-backups/* hdfs://cluster-2.cluster_name.root.hwx.site/kudu/kudu-backups/
```

Using Replication Manager to copy the backed up data

HDFS replication enables you to copy (replicate) your HDFS data from one HDFS service to another, synchronizing the data set on the destination service with the data set on the source service, based on a specified replication policy.

For more information, see *HDFS replicaton policy*.

Related Information

[HDFS replication policy](#)

Restoring Kudu data into the target CDP cluster

Once you have backed up your data in Kudu, you can copy the data to the target CDP cluster and then restore it using the Kudu backup tool.

Before you begin

- If you applied any custom Kudu configurations in your old clusters, then you manually have to apply those configurations in your target cluster.
- Copy the backed up Kudu data to the target CDP cluster.
- While scanning or reading data from Kudu tables using Impala (for example, through `impala-shell` or Hue) to verify the records in the destination table, remember that the Impala table might point to a Kudu table with a

different name, which is defined by the `kudu.table_name` property. Meanwhile, backup and restore tools work does not depend on Impala table names, but rather depends on actual Kudu table names.

To get the information on the `kudu.table_name` property for a table, you can use the `SHOW CREATE TABLE` statement in `impala-shell` or Hue:

```
> SHOW CREATE TABLE my_kudu_table;

CREATE TABLE my_kudu_table
(
  id BIGINT,
  name STRING,
  PRIMARY KEY(id)
)
PARTITION BY HASH PARTITIONS 16
STORED AS KUDU
TBLPROPERTIES (
  'kudu.table_name' = 'my_kudu_table_renamed'
);
```

Procedure

1. Run the following command to restore the backup on the target cluster:

```
spark-submit --class org.apache.kudu.backup.KuduRestore <path to kudu-
backup2_2.11-1.12.0.jar> \
--kuduMasterAddresses <addresses of Kudu masters> \
--rootPath <path to the stored backed up data> \
<table_name>
```

where

- `--kuduMasterAddresses` is used to specify the addresses of the Kudu masters as a comma-separated list. For example, `master1-host,master-2-host,master-3-host` which are the actual hostnames of Kudu masters.

- `--rootPath` is used to specify the path at which you stored the backed up data.. It accepts any Spark-compatible path.
 - Example for HDFS: `hdfs:///kudu-backups`
 - Example for AWS S3: `s3a://kudu-backup/`

If you are backed up to S3 and see the “Exception in thread "main" java.lang.IllegalArgumentException: path must be absolute” error, ensure that S3 path ends with a forward slash (/).

- `<table_name>` can be a table or a list of tables to be backed up.
- Optional: `--tableSuffix`, if set, adds suffices to the restored table names. It can only be used when the `createTables` property is true.
- Optional: `--timestampMs` is a UNIX timestamp in milliseconds that defined the latest time to use when selecting restore candidates. Its default value is `System.currentTimeMillis()`.

```
sudo -u hdfs spark-submit --class org.apache.kudu.backup.KuduRestore /  
opt/cloudera/parcels/CDH-7.2.0-1.cdh7.2.0.p0.3758356/lib/kudu/kudu-  
backup2_2.11.jar \  
--kuduMasterAddresses cluster-1.cluster_name.root.hwx.site \  
--rootPath hdfs:///kudu/kudu-backups \  
my_table
```

2. Restart the Kudu service in Cloudera Manager.