Cloudera Runtime 7.0.2

Troubleshooting HBase

Date published: 2019-09-23
Date modified:

https://docs.cloudera.com/

Legal Notice

Contents

Troubleshooting HBase.....4

Using the HBCK2 tool to remediate HBase clusters.....4

Running the HBCK2 tool.....4
Finding issues.....5
Fixing issues.....6
HBCK2 tool command reference.....8

Thrift Server crashes after receiving invalid data.....9

HBase is using more disk space than expected.....9

Troubleshoot RegionServer grouping.....10

Troubleshooting HBase

If you have encountered an error and want to search for speecific keywords in the Apache HBase log, you can use the log search feature in Cloudera Manager. To search for specific keywords in the Apache HBase logs in Cloudera Manager, go to Diagnostics Logs . You can search for specific keywords from various sources, services, role types, and hosts.

Using the HBCK2 tool to remediate HBase clusters

The HBCK2 tool is a repair tool that you can use to remediate Apache HBase clusters. The HBCK2 tool is the next version of the Apache HBase hbck tool.

The HBCK2 tool is a repair tool to remediate Apache HBase clusters in CDH. The HBCK2 tool is the next version of the Apache HBase hbck tool.

To identify a list of inconsistencies or blockages in a running HBase cluster, you can view the Master logs. Once you have identified the issue, you can then use the HBCK2 tool to fix the defect or to skip-over a bad state. The HBCK2 tool uses an interactive fix-it process by asking the Master to make the fixes rather than carry out the repair locally.

The HBCK2 performs a single, discrete task each time it is run. The HBCK2 tool does not analyze everything in a running cluster and repair all the problems. Instead, you can use the HBCK2 tool to iteratively find and fix issues in your cluster. The HBCK2 tool lets you use interactive commands to fix one issue at a time.

Important: The HBCK2 tool is specific to internals of Apache HBase. Using this tool requires binaries that are specific to your version of CDP Runtime, and you must always use it with the assistance of Cloudera Support and/or Cloudera Professional Services. Please contact Cloudera Support if you believe you have an issue that requires using the HBCK2 tool.

Running the HBCK2 tool

You can run the HBCK2 tool from the command-line interface in your target cluster.

About this task

The HBCK2 tool is a part of the hbase-operator-tools binary. Once you get the hbase-operator-tools binary from Cloudera, upload the binary tarball to the target cluster and extract the tarball. The HBCK2 JAR file is contained in the operator tools tarball provided by Cloudera Support at hbase-operator-tools-<version>/hbase-hbck2/hbase-hbck2-<version>.jar.

Procedure

- Run the HBCK2 tool by specifying the JAR path with the “-j” option as shown here:

```
$ hbase hbck -j $HOME/hbase-operator-tools-<version>/hbase-hbck2/hbase-h
bck2-<version>.jar
```

When you run the command, the HBCK2 tool command-line menu appears.

As a Cloudera Support or Professional Services personnel using this tool to remediate an HBase cluster, you can gather useful information using these commands as an HBase super user (typically, hbase), or an HBase principal if Kerberos is enabled:

```
$ hdfs dfs -ls -R /hbase/ 2>&1 | tee /tmp/hdfs-ls.txt
$ hbase hbck -details 2>&1 | tee /tmp/hbase-hbck.txt
```

```
$ echo "scan 'hbase:meta'" | hbase shell 2>&1 | tee /tmp/hbase-meta.txt
```

Finding issues

The HBCK2 tool enables you to use interactive commands to fix one issue at a time. If you have multiple issues, you may have to run the tool iteratively to find and resolve all the issues. Use the following utilities and commands to find the issues.

Find issues using diagnostic tools

Master logs

The Apache HBase Master runs all the cluster start and stop operations, RegionServer assignment, and server crash handling. Everything that the Master does is a procedure on a state machine engine and each procedure has an unique procedure ID (PID). You can trace the lifecycle of a procedure by tracking its PID through the entries in the Master log. Some procedures may spawn sub-procedures and wait for the sub-procedure to complete.

You can trace the sub-procedure by tracking its PID and the parent PID (PPID).

If there is a problem with RegionServer assignment, the Master prints a STUCK log entry similar to the following:

```
2018-09-12 15:29:06,558 WARN
org.apache.hadoop.hbase.master.assignment.AssignmentManager: STUCK
Region-In-Transition rit=OPENING, location=va1001.example.org,00001,10001732
30599,
table=IntegrationTestBigLinkedList_20180626110336,
region=dbdb56242f17610c46ea044f7a42895b
```

Master user interface

Status tables

You can find issues in your HBase tables by looking at the status tables section in the Master user interface home page. Look through the list of tables to identify if a table is ENABLED, ENABLING, DISABLED, or DISABLING. You can also take a look at the regions in transition states: OPEN, CLOSED. For example, there may be an issue if a table is ENABLED, some regions are not in the OPEN state, and the Master log entries do not have any ongoing assignments.

Procedures and locks

When an Apache HBase cluster is started, the Procedures & Locks page in the Master user interface is populated with information about the procedures, locks, and the count of WAL files. After the cluster settles, if the WAL file count does not reduce, it leads to procedure blocks. You can identify those procedures and locks on this page.

You can also get a list of locks and procedures using this command in the HBase shell:

```
$ echo "list_locks"| hbase shell &> /tmp/locks.txt
$ echo "list_procedures"| hbase shell &> /tmp/procedures.txt
```

Apache HBase canary tool

Use the HBase canary tool to verify the state of the assigns in your cluster. You can run this tool to focus on just one table or the entire cluster. You can check the cluster assign using this command:

```
$ hbase canary -f false -t 6000000 &>/tmp/canary.log
```

Use the -f parameter to look for failed region fetches, and set the -t parameter to run for a specified time.

Fixing issues

You can fix issues using the HBCK2 tool.

You must keep these in mind when fixing issues using HBCK2. Ensure that:

- A region is not in the CLOSING state during “assign”, and in the OPENING state during “unassign”. You can change the state using the setRegionState command. See the HBCK2 tool Command Reference section for more information.
- You fix only one table at a time.

Important: Contact Cloudera Support before using any of the HBCK2 tool commands.

Fix assign and unassign issues

You can fix assign and unassign issues by monitoring the current list of outstanding locks. An assign against a locked region will wait till the lock is released. An assignment gets an exclusive lock on the region.

Fix master startup cannot progress error

If you see a master startup cannot progress holding-pattern until region online error in the Master log, it means that the Master is unable to start because there is no procedure to assign hbase:meta. You will see an error message similar to this:

```
2020-04-01 22:07:42,792 WARN org.apache.hadoop.hbase.master.HMaster:
 hbase:meta,,1.1588230740 is NOT online; state={1588230740 state=CLOSING,
ts=1538456302300, server=ve1017.example.org,22101,1234567891012};
ServerCrashProcedures=true. Master startup cannot progress in holding-pat
tern until region onlined.
```

To fix this issue, run the following command:

```
$ hbase hbck -j $HOME/hbase-operator-tools-<version>/hbase-hbck2/hbase-hbck2
-<version>.jar
assigns 1588230740
```

The same issue can occur with a hbase:namespace system table. To fix this issue, run the following command:

```
$ hbase hbck -j $HOME/hbase-operator-tools-<version>/hbase-hbck2/hbase-hbck2
-<version>.jar
assigns <hbase:namespace encoded region id>
```

You can find the namespace encoded region id using this command:

```
$ echo "scan 'hbase:meta',{COLUMNS=>'info:regioninfo',
FILTER=>\"PrefixFilter('hbase:namespace')\"}" | hbase shell
```

The namespace encoded region id is the value under the "ENCODED" field in the results.

Fix missing regions in hbase:meta region/table

If you encounter an issue where table regions have been removed from the hbase:meta table, you can use the addFsRegionsMissingInMeta to resolve this issue. Ensure that the Master is online. This command is not as disruptive as the hbase:meta rebuild command.

To fix this issue, run this command:

```
$ hbase hbck -j $HOME/hbase-operator-tools-<version>/hbase-hbck2/hbase-hbck2
-<version>.jar
addFsRegionsMissingInMeta <NAMESPACE|NAMESPACE:TABLENAME>
```

The command returns an HBCK2 “assigns” command with all the listed re-inserted regions.You must restart the Master, and then run the HBCK2 'assigns' command returned by the addFsRegionsMissingInMeta command to complete your fix.

Example output:

```
Regions re-added into Meta: 2
WARNING:
2 regions were added to META, but these are not yet on Masters cache.
You need to restart Masters, then run hbck2 'assigns' command below:
assigns 7be03127c5e0e2acfc7cae7ddfa9e29e e50b8c1adc38c942e226a8b2976f0c8c
```

Fix extra regions in hbase:meta region/table

If there are extra regions in hbase:meta, it may be because of problems in splitting, deleting/moving the region directory manually, or in rare cases because of the loss of metadata.

To fix this issue, run this command:

```
$ hbase hbck -j $HOME/hbase-operator-tools-<version>/hbase-hbck2/hbase-hbck2
-<version>.jar
extraRegionsInMeta --fix  <NAMESPACE|NAMESPACE:TABLENAME>...
```

Important: Use the --fix option only when the extra regions are overlapping the existing valid regions. Otherwise, use the assigns command to recreate the regions.

Rebuild hbase:meta

If hbase:meta is offline because it is corrupted, you can bring it back online if the corruption is not too critical. If the namespace region is among the mission regions, scan hbase:meta during initialization to check if hbase:meta is online.

To check if hbase:meta is online, run this command in the Apache HBase shell:

```
$ echo "scan 'hbase:meta', {COLUMN=>'info:regioninfo'}" | hbase shell
```

If this scan does not throw any errors, then you can run the following command to validate that the tables are present:

```
$ hbase hbck -j $HOME/hbase-operator-tools-<version>/hbase-hbck2/hbase-hbck2
-<version>.jar
addFsRegionsMissingInMeta <NAMESPACE|NAMESPACE:TABLENAME>
```

The addFsRegionsMissingInMeta command adds regions back to the hbase:meta table if the regioninfo file is present in the storage but the regions were deleted because of an issue.

Fix dropped references and corrupted HFiles

To fix hanging references and corrupt HFiles, run the following command:

```
$ hbase hbck -j $HOME/hbase-operator-tools-<version>/hbase-hbck2/hbase-hbck2
-<version>.jar
filesystem --fix [<TABLENAME>...]
```

HBCK2 tool command reference

You can use this list of HBCK2 commands in your target cluster's command-line interface.

HBCK2 commands

- `addFsRegionsMissingInMeta <NAMESPACE|NAMESPACE:TABLENAME>...`

  Options: -d,--force_disable Use this option to abort fix for table if disable fails.

- `assigns [OPTIONS] <ENCODED_REGIONNAME>...`

  Options:-o,--override Use this option to override ownership by another procedure.

- `bypass [OPTIONS] <PID>...`

  Options: -o,--override Use this option to override if procedure is running/stuck -r,--recursive Use this option to bypass parent and its children.

  -w,--lockWait Use this option to wait (in milliseconds) before giving up; default=1.

- `extraRegionsInMeta <NAMESPACE|NAMESPACE:TABLENAME>...`

  Options:-f, --fix Use this option to fix meta by removing all extra regions found.

- `filesystem [OPTIONS] [<TABLENAME>...]`

  Options:-f, --fix Use this option to sideline corrupt HFiles, bad links, and references.

- `replication [OPTIONS] [<TABLENAME>...]`

  Options:-f, --fix Use this option to fix replication issues.

- `reportMissingRegionsInMeta <NAMESPACE|NAMESPACE:TABLENAME>...`

  Use this command when regions missing from hbase:meta but directories are still present in HDFS.

- `setRegionState <ENCODED_REGIONNAME> <STATE>`

  Possible region states: OFFLINE, OPENING, OPEN, CLOSING, CLOSED, SPLITTING, SPLIT, FAILED_OPEN, FAILED_CLOSE, MERGING, MERGED, SPLITTING_NEW, MERGING_NEW, ABNORMALLY_CLOSED.

  Caution: This command is recommended to be used only as a last resort. Example scenarios include unassigns/assigns that does not happen because the region is in an inconsistent state in hbase:meta.

- `setTableState <TABLENAME> <STATE>`

  Possible table states and representations in hbase:meta table: ENABLED (\x08\x00), DISABLED (\x08\x01), DISABLING (\x08\x02), ENABLING (\x08\x03).

- `scheduleRecoveries <SERVERNAME>...`

  Schedule ServerCrashProcedure(SCP) for list of RegionServers. Format server name as '<HOSTNAME>,<PORT>,<STARTCODE>' .

- `unassigns <ENCODED_REGIONNAME>...`

  Options:-o,--override Use this option to override ownership by another procedure.

Thrift Server crashes after receiving invalid data

The Thrift server may crash if it receives a large amount of invalid data, due to a buffer overrun.

Why this happens

The Thrift server allocates memory to check the validity of data it receives. If it receives a large amount of invalid data, it may need to allocate more memory than is available. This is due to a limitation in the Thrift library itself.

What to do

To prevent the possibility of crashes due to buffer overruns, use the framed and compact transport protocols. These protocols are disabled by default, because they may require changes to your client code. The two options to add to your hbase-site.xml are hbase.regionserver.thrift.framed and hbase.regionserver.thrift.compact. Set each of these to true, as in the XML below. You can also specify the maximum frame size, using the hbase.regionserver.thrift.framed .max_frame_size_in_mb option.

```
<property>
  <name>hbase.regionserver.thrift.framed</name>
  <value>true</value>
</property>
<property>
  <name>hbase.regionserver.thrift.framed.max_frame_size_in_mb</name>
  <value>2</value>
</property>
<property>
  <name>hbase.regionserver.thrift.compact</name>
  <value>true</value>
</property>
```

HBase is using more disk space than expected

HBase StoreFiles (also called HFiles) store HBase row data on disk. HBase stores other information on disk, such as write-ahead logs (WALs), snapshots, data that would otherwise be deleted but would be needed to restore from a stored snapshot.

Warning: The following information is provided to help you troubleshoot high disk usage only. Do not edit or remove any of this data outside the scope of the HBase APIs or HBase Shell, or your data is very likely to become corrupted.

Table 1: HBase Disk Usage

Location	Purpose	Troubleshooting Notes
/hbase/.snapshots	Contains one subdirectory per snapshot.	To list snapshots, use the HBase Shell command list_snapshots. To remove a snapshot, use delete_snapshot.
/hbase/.archive	Contains data that would otherwise have been deleted (either because it was explicitly deleted or expired due to TTL or version limits on the table) but that is required to restore from an existing snapshot.	To free up space being taken up by excessive archives, delete the snapshots that refer to them. Snapshots never expire so data referred to by them is kept until the snapshot is removed. Do not remove anything from /hba se/.archive manually, or you will corrupt your snapshots.

Location	Purpose	Troubleshooting Notes
/hbase/.logs	Contains HBase WAL files that are required to recover regions in the event of a RegionServer failure.	WALs are removed when their contents are verified to have been written to StoreFiles. Do not remove them manually. If the size of any subdirectory of /hbase/.logs/ is growing, examine the HBase server logs to find the root cause for why WALs are not being processed correctly.
/hbase/logs/.oldWALs	Contains HBase WAL files that have already been written to disk. A HBase maintenance thread removes them periodically based on a TTL.	To tune the length of time a WAL stays in the .oldWALs before it is removed, configure the hbase.master.logcleaner.ttl property, which defaults to 60000 milliseconds, or 1 hour.
/hbase/.logs/.corrupt	Contains corrupted HBase WAL files.	Do not remove corrupt WALs manually. If the size of any subdirectory of /hbase/.logs/ is growing, examine the HBase server logs to find the root cause for why WALs are not being processed correctly.

Troubleshoot RegionServer grouping

When using rsgroup, if you encounter an issue such as the one in the following example, check the log to see what is causing the issue. If a rsgroup operation is unresponsive, you must restart the HBase Master.

An example error of what you will see if you start using rsgroup but the required coprocessor is not found:

```
ERROR: org.apache.hadoop.hbase.exceptions.UnknownProtocolException: No regis
tered master coprocessor service found for name RSGroupAdminService
    at org.apache.hadoop.hbase.master.MasterRpcServices.execMasterService(Ma
sterRpcServices.java:604)
    at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$Mast
erService$2.callBlockingMethod(MasterProtos.java)
    at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:1140)
    at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:133)
    at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:
277)
    at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java
:257)
```