Cloudera Runtime 7.3.2

Configuring Fault Tolerance

Date published: 2020-07-28
Date modified: 2026-03-31

https://docs.cloudera.com/

Legal Notice

Contents

High Availability on HDFS clusters

Configuring HDFS High Availability (HA) eliminates the NameNode as a potential single point of failure (SPOF) in an HDFS cluster.

Important:

When Apache Iceberg creates tables, it stores the current NameNode address in the metadata. If HDFS is running without High Availability (HA), Iceberg stores the NameNode's hostname and port. If HDFS is configured with HA, it stores the nameservice ID (example, nameservice1).

Consequently, modifying the HDFS configuration such as transitioning from a non-HA to an HA setup or reverting from HA to non-HA, changes the NameNode address and invalidates the stored address, rendering the existing data unreadable. In these scenarios, you must export and re-import any existing Iceberg data after the HDFS changes are finalized to maintain data accessibility.

For more information, see Create table feature using Apache Iceberg.

Configuring HDFS High Availability

The HDFS NameNode High Availability (HA) feature enables you to run redundant NameNodes in the same cluster in an Active/Passive configuration with a hot standby. This eliminates the NameNode as a potential single point of failure (SPOF) in an HDFS cluster.

Important:

When Apache Iceberg creates tables, it stores the current NameNode address in the metadata. If HDFS is running without High Availability (HA), Iceberg stores the NameNode's hostname and port. If HDFS is configured with HA, it stores the nameservice ID (example, nameservice1).

Consequently, modifying the HDFS configuration such as transitioning from a non-HA to an HA setup or reverting from HA to non-HA, changes the NameNode address and invalidates the stored address, rendering the existing data unreadable. In these scenarios, you must export and re-import any existing Iceberg data after the HDFS changes are finalized to maintain data accessibility.

For more information, see Create table feature using Apache Iceberg.

In a standard configuration, the NameNode is a single point of failure (SPOF) in an HDFS cluster. Each cluster has a single NameNode, and if that machine or process becomes unavailable, the cluster as a whole is unavailable until the NameNode is either restarted or brought up on a separate machine. This situation impacts the total availability of the HDFS cluster in two major ways:

- In the case of an unplanned event such as a machine crash, the cluster would be unavailable until an operator restarted the NameNode.
- Planned maintenance events such as software or hardware upgrades on the NameNode machine would result in periods of cluster downtime.

HDFS NameNode HA avoids this by facilitating either a fast failover to a standby NameNode during machine crash, or a graceful administrator-initiated failover during planned maintenance.

NameNode architecture

In a typical HA cluster, two different machines are configured as NameNodes. In a working cluster, one of the NameNode machine is in the Active state, and the other is in the Standby state.

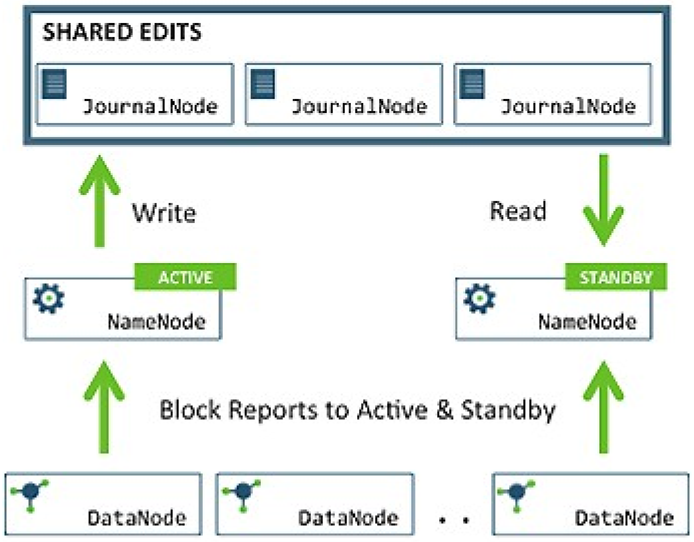
The Active NameNode is responsible for all client operations in the cluster, while the Standby NameNode acts as a backup. The Standby machine maintains enough state to provide a fast failover (if required).

In order for the Standby node to keep its state synchronized with the Active node, both nodes communicate with a group of separate daemons called JournalNodes (JNs). When the Active node performs any namespace modification, the Active node durably logs a modification record to a majority of these JNs. The Standby node reads the edits from the JNs and continuously watches the JNs for changes to the edit log. Once the Standby Node observes the edits, it applies these edits to its own namespace. When using QJM, JournalNodes act as the shared editlog storage. In a failover event, the Standby ensures that it has read all of the edits from the JounalNodes before promoting itself to the Active state. (This mechanism ensures that the namespace state is fully synchronized before a failover completes.)

Note:

Secondary NameNode is not required in HA configuration because the Standby node also performs the tasks of the Secondary NameNode.

To provide a fast failover, it is also necessary that the Standby node has up-to-date information on the location of blocks in your cluster. To get accurate information about the block locations, DataNodes are configured with the location of all the NameNodes, and send block location information and heartbeats to all the NameNode machines.



It is vital for the correct operation of an HA cluster that only one of the NameNodes should be Active at a time. Failure to do so would cause the namespace state to quickly diverge between the NameNode machines, thus causing potential data loss. (This situation is referred to as a split-brain scenario.)

To prevent the split-brain scenario, the JournalNodes allow only one NameNode to be a writer at a time. During failover, the NameNode, that is chosen to become active, takes over the role of writing to the JournalNodes. This process prevents the other NameNode from continuing in the Active state and thus lets the new Active node proceed with the failover safely.

Preparing the hardware resources for HDFS High Availability

Make sure that you prepare the required hardware resources for High Availability.

- NameNode machines: The machines where you run Active and Standby NameNodes, should have exactly the same hardware.
- JournalNode machines: The machines where you run the JournalNodes. The JournalNode daemon is relatively lightweight, so these daemons may reasonably be co- located on machines with other Hadoop daemons, for example the NameNodes or the YARN ResourceManager.

  Note:

  There must be at least three JournalNode daemons, because edit log modifications must be written to a majority of JNs. This lets the system tolerate failure of a single machine. You may also run more than three JournalNodes, but in order to increase the number of failures that the system can tolerate, you must run an odd number of JNs (3, 5, 7, and so on).

  Note that when running with N JournalNodes, the system can tolerate at most (N - 1) / 2 failures and continue to function normally.

- ZooKeeper machines: For automated failover functionality, there must be an existing ZooKeeper cluster available. The ZooKeeper service nodes can be co-located with other Hadoop daemons.

In an HA cluster, the Standby NameNode also performs checkpoints of the namespace state. Therefore, do not deploy a Secondary NameNode in an HA cluster.

Using Cloudera Manager to manage HDFS HA

You can use Cloudera Manager to configure your Cloudera cluster for HDFS HA and automatic failover.

Important:

When Apache Iceberg creates tables, it stores the current NameNode address in the metadata. If HDFS is running without High Availability (HA), Iceberg stores the NameNode's hostname and port. If HDFS is configured with HA, it stores the nameservice ID (example, nameservice1).

Consequently, modifying the HDFS configuration such as transitioning from a non-HA to an HA setup or reverting from HA to non-HA, changes the NameNode address and invalidates the stored address, rendering the existing data unreadable. In these scenarios, you must export and re-import any existing Iceberg data after the HDFS changes are finalized to maintain data accessibility.

For more information, see Create table feature using Apache Iceberg.

Enabling HDFS HA

An HDFS high availability (HA) cluster uses two NameNodes -- an active NameNode and a standby NameNode. Only one NameNode can be active at any point in time.

HDFS HA depends on maintaining a log of all namespace modifications in a location available to both NameNodes, so that in the event of a failure, the standby NameNode has up-to-date information about the edits and location of blocks in the cluster.

Important: Enabling and disabling HA causes a service outage for the HDFS service and all services that depend on HDFS. Before enabling or disabling HA, ensure that there are no jobs running on your cluster.

Prerequisites for enabling HDFS HA using Cloudera Manager

There are certain prerequisites that your cluster must satisfy before HDFS HA can be enabled using Cloudera Manager.

You must perform the following tasks *before* enabling HDFS HA:

- Ensure that you perform all the required configuration and setup tasks.
- Enabling and disabling HA causes a service outage for the HDFS service and all services that depend on HDFS. Before enabling or disabling HA, ensure that there are no jobs running on your cluster.
- Ensure that you have a ZooKeeper service configured.

Important:

- Enabling or disabling HA causes the previous monitoring history to become unavailable.
- Some parameters will be automatically set as follows once you have enabled JobTracker HA. If you want to change the value from the default for these parameters, use an advanced configuration snippet.
    - mapred.jobtracker.restart.recover: true
    - mapred.job.tracker.persist.jobstatus.active: true
    - mapred.ha.automatic-failover.enabled: true
    - mapred.ha.fencing.methods: shell(true)

Enabling High Availability and automatic failover

You can use Cloudera Manager to configure your Cloudera cluster for HDFS HA and automatic failover. In Cloudera Manager, HA is implemented using Quorum-based storage. Quorum-based storage relies upon a set of JournalNodes, each of which maintains a local edits directory that logs the modifications to the namespace metadata. Enabling HA enables automatic failover as part of the same command.

About this task

- Cluster Administrator (also provided by Full Administrator)
- The Enable High Availability workflow leads you through adding a second (standby) NameNode and configuring JournalNodes.

Procedure

1. Go to the HDFS service.
2. Select Actions Enable High Availability .

   A screen showing the hosts that are eligible to run a standby NameNode and the JournalNodes displays.

   a) Specify a name for the nameservice and click Continue.

      Attention: Use unique names for the nameservices.

   b) In the NameNode Hosts field, click Select a host.

      The host selection dialog box displays.

   c) Select the checkbox next to the hosts where you want the standby NameNode to be set up and click OK.

      The standby NameNode cannot be on the same host as the active NameNode, and the host that is chosen should have the same hardware configuration (RAM, disk space, number of cores, and so on) as the active NameNode.

   d) In the JournalNode Hosts field, click Select hosts.

      The host selection dialog box displays.

   e) Check the checkboxes next to an odd number of hosts (a minimum of three) to act as JournalNodes and click OK.

      JournalNodes should be hosted on hosts with similar hardware specification as the NameNodes. Cloudera recommends that you put a JournalNode each on the same hosts as the active and standby NameNodes, and the third JournalNode on similar hardware, such as the JobTracker.

   f) Click Continue.
   g) In the JournalNode Edits Directory property, enter a directory location for the JournalNode edits directory into the fields for each JournalNode host.

      - You may enter only one directory for each JournalNode. The paths do not need to be the same on every JournalNode.
      - The directories you specify should be empty.
      - The directory owner should be hdfs:hadoop and must have read, write, and r permission (drwx------).

   h) Extra Options: Decide whether Cloudera Manager should clear existing data in ZooKeeper, standby NameNode, and JournalNodes.

      If the directories are not empty (for example, you are re-enabling a previous HA configuration), Cloudera Manager will not automatically delete the contents—you can select to delete the contents by keeping the default checkbox selection. The recommended default is to clear the directories. If you choose not to do so, the data should be in sync across the edits directories of the JournalNodes and should have the same version data as the NameNodes.

i) Click Continue.

Cloudera Manager executes a set of commands that stop the dependent services, delete, create, and configure roles and directories as required, create a nameservice and failover controller, restart the dependent services, and deploy the new client configuration.

Important: Some steps, such as formatting the NameNode may report failure if the action was already completed. However, the configuration steps continue to run after reporting non-critical failed steps.

Note: The HDFS Instances page denotes the active and standby NameNodes in parentheses next to the respective role instances.

3. Restart Ranger KMS, if configured for your cluster.
4. Configure HDFS HA for other Cloudera services, if required.

   Configuring other Cloudera components to use HDFS HA on page 9

What to do next

Important: If you change the NameNode Service RPC Port (dfs.namenode.servicerpc-address) while automatic failover is enabled, this will cause a mismatch between the NameNode address saved in the ZooKeeper /hadoop-ha znode and the NameNode address that the Failover Controller is configured with. This will prevent the Failover Controllers from restarting. If you need to change the NameNode Service RPC Port after Auto Failover has been enabled, you must do the following to re-initialize the znode:

1. Stop the HDFS service.
2. Configure the service RPC port:
   a. Go to the HDFS service.
   b. Click the Configuration tab
   c. Select ScopeNameNode.
   d. Select CategoryPorts and Addresses.
   e. Locate the NameNode Service RPC Port property or search for it by typing its name in the Search box.
   f. Change the port value as needed.

      To apply this configuration property to other role groups as needed, edit the value for the appropriate role group.
3. On a ZooKeeper server host, run zookeeper-client.
   a. Run the following to remove the configured nameservice. This example assumes the name of the nameservice is nameservice1. You can identify the nameservice from the Federation and High Availability section on the HDFS Instances tab:

      ```
      rmr /hadoop-ha/nameservice1
      ```
4. Click the Instances tab.
5. Select ActionsInitialize High Availability State in ZooKeeper.
6. Start the HDFS service.

Disabling and redeploying HDFS HA

When you disable HDFS HA, Cloudera Manager ensures that one NameNode is active, and saves the namespace. Then it stops the standby NameNode, creates a Secondary NameNode, removes the standby NameNode role, and restarts all the HDFS services.

About this task

Important: Disabling HDFS HA makes the existing Iceberg tables unreadable. If HDFS is running with HA, export existing Iceberg tables to a non-Iceberg format before disabling HA, then re-create the Iceberg tables and reload the exported data. For more information, see Create table feature using Apache Iceberg.

Procedure

1. Go to the HDFS service.
2. Select ActionsDisable High Availability.
3. Select the hosts for the NameNode and the Secondary NameNode and click Continue.
4. Select the HDFS checkpoint directory and click Continue.
5. Confirm that you want to take this action.
6. Configure the Hive Metastore to use HDFS HA.

Configuring other Cloudera components to use HDFS HA

You can use the HDFS high availability NameNodes with other components of Cloudera.

Configuring HBase to use HDFS HA

If you configure HBase to use an HA-enabled HDFS instance, Cloudera Manager automatically handles HA configuration for you.

Configuring the Hive Metastore to use HDFS HA

The Hive metastore can be configured to use HDFS high availability by using Cloudera Manager or by using the command line for unmanaged clusters.

Procedure

1. In the Cloudera Admin Console, go to the Hive service.
2. Select Actions Stop .

   Note: You may want to stop the Cloudera Data Explorer (Hue) and Impala services first, if present, as they depend on the Hive service.

   Click Stop again to confirm the command.
3. Back up the Hive metastore database.
4. Select ActionsUpdate Hive Metastore NameNodes and confirm the command.
5. Select ActionsStart and click Start to confirm the command.
6. Restart the Cloudera Data Explorer (Hue) and Impala services if you stopped them prior to updating the metastore.

Configuring Impala to work with HDFS HA

You must reconfigure the Hive metastore database and issue from the INVALIDATE   METADATA statement from the Impala shell.

Procedure

1. Complete the steps to reconfigure the Hive metastore database.

   Impala shares the same underlying database with Hive, to manage metadata for databases, tables, and so on.
2. Issue the INVALIDATE METADATA statement from an Impala shell.

   This one-time operation makes all Impala daemons across the cluster aware of the latest settings for the Hive metastore database.

   Alternatively, restart the Impala service.

Configuring oozie to use HDFS HA

You can configure an Oozie workflow to use HDFS HA.

Procedure

- Use the HDFS nameservice instead of the NameNode URI in the <name-node> element of the workflow.

  ```
  <action name="mr-node">
    <map-reduce>
      <job-tracker>${jobTracker}</job-tracker>
      <name-node>hdfs://ha-nn
  ```

  where *HA-NN* is the value of dfs.nameservices in hdfs-site.xml.

Changing a nameservice name for Highly Available HDFS using Cloudera Manager

Based on your requirements, you can change a nameservice name for a cluster that has HDFS HA configured.

Before you begin

Before you start, make note of the NameNode data directory and JournalNode edits directory. You can find dfs.name node.name.dir and dfs.journalnode.edits.dir from the configuration of HDFS service in the Cloudera Admin Console.

Cloudera recommends you to take a backup for the above directories so that if there is any operator error, you still have a chance to recover.

In this example, the following values are assumed:

- The current nameservice name is nameservice1.
- The NameNode data directory is /data/dfs/nn.
- The JournalNode edits directory is /data/dfs/jn.

About this task

Important:

When Apache Iceberg creates tables, it stores the current NameNode address in the metadata. If HDFS is running without High Availability (HA), Iceberg stores the NameNode's hostname and port. If HDFS is configured with HA, it stores the nameservice ID (example, nameservice1).

Consequently, modifying the HDFS configuration such as transitioning from a non-HA to an HA setup or reverting from HA to non-HA, changes the NameNode address and invalidates the stored address, rendering the existing data unreadable. In these scenarios, you must export and re-import any existing Iceberg data after the HDFS changes are finalized to maintain data accessibility.

For more information, see Create table feature using Apache Iceberg.

Procedure

1. Ensure that both NameNodes enter safemode and you save the current namespace and merge all the recent edits to the same fsimage id for both NameNodes.
   a) Go to HDFS service Instance .
   b) Click NameNodes (Active) Actions Enter Safemode .
   c) Click NameNodes (Active) Actions Save Namespace .
   d) Click NameNodes (Standby) Actions Enter Safemode .
   e) Click NameNodes (Standby) Actions Save Namespace .
   f) ssh to both NameNodes data directories to ensure that the same fsimage is generated.

2. Delete the current nameservice znode from ZooKeeper CLI.
   a) Stop all services except ZooKeeper.
   b) On a ZooKeeper host, run zookeeper-client.
   c) Execute the following to remove the configured nameservice:

      ```
      deleteall /hadoop-ha/nameservice1
      ```

      Login as ZooKeeper super user if Kerberos is enabled, so that you have the authorization to delete the protected znodes.
3. In the Cloudera Admin Console, change the NameNode nameservice name.
   a) Go to the HDFS service.
   b) Click Configuration.
   c) Type nameservice in the Search field.
   d) For the NameNode Nameservice property, change the nameservice name in the NameNode (instance_name) field.

      The name must be unique and can contain only alphanumeric characters.
   e) Type quorum in the Search field.
   f) For the Quorum-based Storage Journal name property, change the nameservice name in the NameNode (instance_name) field.
   g) Save changes.
4. Create a new nameservice znode.
   a) Click Instances.
   b) In the Federation and High Availability pane, select Actions Initialize High Availability State in ZooKeeper .
   c) On a ZooKeeper host, run zookeeper-client and verify that the new nameservice znode is created.

      ```
      ls /hadoop-ha
      ```
5. Update Hive Metastore NameNodes.
   a) Go to the Hive service.
   b) Select Actions Update Hive Metastore NameNodes .
6. If you have an Impala service, restart the Impala service or run an INVALIDATE METADATA query.
7. Initialize JournalNode shared edits directory.
   a) Click Instances.
   b) Select the checkboxes next to the JournalNode role instances.
   c) Select Actions for Selected Start .
   d) Click any NameNode, select Actions Initialize Shared Edits Directory .
   e) ssh to JournalNode host to ensure that the JournalNode data directory has the new nameservice folder generated.
8. Start other HDFS roles.
   a) Click the Instance tab, select the checkboxes next to all the HDFS role instances except for datanodes.
   b) Select Actions for Selected Start .

      You can see both NameNodes web UIs show Active and Standby, also nameservice names are updated. However both NameNodes are in safemode waiting for DN to startup and send block reports.
   c) Click Instances, start all the DN roles, wait for both NN to exit safemode automatically.
9. Redeploy client configuration files.
10. Start all services except ZooKeeper.

11. For clusters having Ranger service, change the references to new nameservice name, if any.

a) Go to Ranger Admin Web UI, click cm_hdfs, check available policies and change the references to new nameservice name on HDFS path, if any.

b) On Ranger Admin Web UI, click cm_hive, check available policies and change the references to new nameservice name on Impala URI, if any.

Manually failing over to the standby NameNode

If you are running a HDFS service with HA enabled, you can manually cause the active NameNode to failover to the standby NameNode. This is useful for planned downtime—for hardware changes, configuration changes, or software upgrades of your primary host.

About this task

Procedure

1. Go to the HDFS service.
2. Click the Instances tab.
3. Click Federation and High Availability.
4. Locate the row for the Nameservice where you want to fail over the NameNode.

   Multiple rows display only when using HDFS federation.
5. Select Actions Manual Failover .

   Note: This option does not appear if HA is not enabled for the cluster.
6. From the pop-up, select the NameNode that should be made active, then click Manual Failover.

   Note: For advanced use only: You can set the Force Failover checkbox to force the selected NameNode to be active, irrespective of its state or the other NameNode's state. Forcing a failover will first attempt to failover the selected NameNode to active mode and the other NameNode to standby mode. It will do so even if the selected NameNode is in safe mode. If this fails, it will proceed to transition the selected NameNode to active mode. To avoid having two NameNodes be active, use this only if the other NameNode is either definitely stopped, or can be transitioned to standby mode by the first failover step.
7. Click Finish.

Results

Cloudera Manager transitions the NameNode you selected to be the active NameNode, and the other NameNode to be the standby NameNode. HDFS should never have two active NameNodes.

Additional HDFS haadmin commands to administer the cluster

After your HA NameNodes are configured and started, you will have access to some additional commands to administer your HA HDFS cluster.

The following are high-level uses of some important subcommands. For specific usage information of each subcommand, you should run hdfs haadmin -help   <command>.

getservicestate

Determine whether the given NameNode is active or standby

Connect to the provided NameNode to determine its current state, printing either "standby" or "active" to STDOUT as appropriate. This subcommand might be used by `cron` jobs or monitoring scripts which need to behave differently based on whether the NameNode is currently active or standby.

checkhealth

Check the health of the given NameNode

Connect to the provided NameNode to check its health. The NameNode is capable of performing some diagnostics on itself, including checking if internal services are running as expected. This command will return 0 if the NameNode is healthy, non-zero otherwise. One might use this command for monitoring purposes.

Turning safe mode on HA NameNodes

You can use the applicable `dfsadmin` command options to turn safe mode on for a single or both the active and standby NameNodes.

Procedure

- To turn safe mode on for both NameNodes, run hdfs dfsadmin -safemode enter.
- To turn safe mode on for a single NameNode, run hdfs dfsadmin -fs hdfs://<host>:<port> -safemode enter

Note: For a full list of `dfsadmin` command options, run: hdfs dfsadmin -help.

Converting from an NFS-mounted shared edits directory to Quorum-Based Storage

Converting a HA configuration from using an NFS-mounted shared edits directory to Quorum-based storage involves disabling the current HA configuration then enabling HA using Quorum-based storage.

Procedure

1. Disable HA.
2. Empty the name directories of the standby NameNode.
3. Enable HA with Quorum-based Storage.

Administrative commands

The subcommands of `hdfs haadmin` are extensively used for administering an HA cluster.

Running the `hdfs haadmin` command without any additional arguments will display the following usage information:

```
Usage: HAAdmin [-ns <nameserviceId>]
 [-transitionToActive <serviceId>]
 [-transitionToStandby <serviceId>]
 [-failover [--forcefence] [--forceactive] <serviceId> <serviceId>]
 [-getServiceState <serviceId>]
 [-checkHealth <serviceId>]
 [-help <command>]
```

This section provides high-level uses of each of these subcommands.

- transitionToActive and transitionToStandby: Transition the state of the given NameNode to Active or Standby.

  These subcommands cause a given NameNode to transition to the Active or Standby state, respectively. These commands do not attempt to perform any fencing, and thus should be used rarely. Instead, Cloudera recommends using the following subcommand:

  ```
  hdfs haadmin -failover
  ```

- failover: Initiate a failover between two NameNodes.

  This subcommand causes a failover from the first provided NameNode to the second.

  - If the first NameNode is in the Standby state, this command transitions the second to the Active state without error.
  - If the first NameNode is in the Active state, an attempt will be made to gracefully transition it to the Standby state. If this fails, the fencing methods (as configured by dfs.ha.fencing.methods) will be attempted in order until one succeeds. Only after this process will the second NameNode be transitioned to the Active state. If the fencing methods fail, the second NameNode is not transitioned to Active state and an error is returned.

- getServiceState: Determine whether the given NameNode is Active or Standby.

  This subcommand connects to the provided NameNode, determines its current state, and prints either "standby" or "active" to STDOUT appropriately. This subcommand might be used by cron jobs or monitoring scripts.
- checkHealth: Check the health of the given NameNode.

  This subcommand connects to the NameNode to check its health. The NameNode is capable of performing some diagnostics that include checking if internal services are running as expected. This command will return 0 if the NameNode is healthy else it will return a non-zero code.