AdminManual Configuration

Configuring Hive

A number of configuration variables in Hive can be used by the administrator to change the behavior for their installations and user sessions. These variables can be configured in any of the following ways, shown in the order of preference:

- Using the **set command** in the CLI or Beeline for setting session level values for the configuration variable for all statements subsequent to the set command. For example, the following command sets the scratch directory (which is used by Hive to store temporary output and plans) to `/tmp/mydir` for all subsequent statements:

```
set hive.exec.scratchdir=/tmp/mydir;
```

- Using the **`--hiveconf`** option of the `hive` command (in the CLI) or `beeline` command for the entire session. For example:

```
bin/hive --hiveconf hive.exec.scratchdir=/tmp/mydir
```

- In **`hive-site.xml`**. This is used for setting values for the entire Hive configuration (see hive-site.xml and hive-default.xml.template below). For example:

```
<property>
  <name>hive.exec.scratchdir</name>
  <value>/tmp/mydir</value>
  <description>Scratch space for Hive jobs</description>
</property>
```

- In **server-specific configuration files** (supported starting Hive 0.14). You can set metastore-specific configuration values in **hivemetastore-site.xml**, and HiveServer2-specific configuration values in **hiveserver2-site.xml**.
  The server-specific configuration file is useful in two situations:

  a. You want a different configuration for one type of server (for example – enabling authorization only in HiveServer2 and not CLI).
  b. You want to set a configuration value only in a server-specific configuration file (for example – setting the metastore database password only in the metastore server configuration file).

     HiveMetastore server reads hive-site.xml as well as hivemetastore-site.xml configuration files that are available in the $HIVE_CONF_DIR or in the classpath. If the metastore is being used in embedded mode (i.e., hive.metastore.uris is not set or empty) in `hive` commandline or HiveServer2, the hivemetastore-site.xml gets loaded by the parent process as well.
     The value of hive.metastore.uris is examined to determine this, and the value should be set appropriately in hive-site.xml .
     Certain metastore configuration parameters like hive.metastore.sasl.enabled, hive.metastore.kerberos.principal, hive.metastore.execute.setugi, and hive.metastore.thrift.framed.transport.enabled are used by the metastore client as well as server. For such common parameters it is better to set the values in hive-site.xml, that will help in keeping them consistent.

     HiveServer2 reads hive-site.xml as well as hiveserver2-site.xml that are available in the $HIVE_CONF_DIR or in the classpath.
     If HiveServer2 is using the metastore in embedded mode, hivemetastore-site.xml also is loaded.

     The order of precedence of the config files is as follows (later one has higher precedence) –
     hive-site.xml -> hivemetastore-site.xml -> hiveserver2-site.xml -> '`-hiveconf`' commandline parameters.

hive-site.xml and hive-default.xml.template

`hive-default.xml.template` contains the default values for various configuration variables that come prepackaged in a Hive distribution. In order to override any of the values, create `hive-site.xml` instead and set the value in that file as shown above.

`hive-default.xml.template` is located in the `conf` directory in your installation root, and `hive-site.xml` should also be created in the same directory.

Please note that the template file `hive-default.xml.template` is not used by Hive at all (as of Hive 0.9.0) – the canonical list of configuration options is only managed in the `HiveConf` java class. The template file has the formatting needed for `hive-site.xml`, so you can paste configuration variables from the template file into `hive-site.xml` and then change their values to the desired configuration.

In Hive releases 0.9.0 through 0.13.1, the template file does not necessarily contain all configuration options found in `HiveConf.java` and some of its values and descriptions might be out of date or out of sync with the actual values and descriptions. However, as of Hive 0.14.0 the template file is generated directly from `HiveConf.java` and therefore it is a reliable source for configuration variables and their defaults.

The administrative configuration variables are listed below. User variables are listed in Hive Configuration Properties. As of Hive 0.14.0 you can display information about a configuration variable with the SHOW CONF command.

Temporary Folders

Hive uses temporary folders both on the machine running the Hive client and the default HDFS instance. These folders are used to store per-query temporary/intermediate data sets and are normally cleaned up by the hive client when the query is finished. However, in cases of abnormal hive client termination, some data may be left behind. The configuration details are as follows:

- On the HDFS cluster this is set to */tmp/hive-<username>* by default and is controlled by the configuration variable *hive.exec.scratchdir*
- On the client machine, this is hardcoded to */tmp/<username>*

Note that when writing data to a table/partition, Hive will first write to a temporary location on the target table's filesystem (using hive.exec.scratchdir as the temporary location) and then move the data to the target table. This applies in all cases - whether tables are stored in HDFS (normal case) or in file systems like S3 or even NFS.

Log Files

Hive client produces logs and history files on the client machine. Please see Hive Logging for configuration details.

For WebHCat logs, see Log Files in the WebHCat manual.

Derby Server Mode

Derby is the default database for the Hive metastore (Metadata Store). To run Derby as a network server for multiple users, see Hive Using Derby in Server Mode.

Configuration Variables

Broadly the configuration variables for Hive administration are categorized into:

- Hive Configuration Variables
- Hive Metastore Configuration Variables
- Configuration Variables Used to Interact with Hadoop
- Hive Variables Used to Pass Run Time Information

Also see Hive Configuration Properties in the Language Manual for non-administrative configuration variables.

> **Version information: Metrics**
>
> A new Hive metrics system based on Codahale is introduced in releases 1.3.0 and 2.0.0 by HIVE-10761. To configure it or revert to the old metrics system, see the Metrics section of Hive Configuration Properties.

Hive Configuration Variables

Variable Name	Description	Default Value
hive.ddl.output.format	The data format to use for DDL output (e.g. `DESCRIBE table`). One of "text" (for human readable text) or "json" (for a json object). (As of Hive 0.9.0.)	text
hive.exec.script.wrapper	Wrapper around any invocations to script operator e.g. if this is set to python, the script passed to the script operator will be invoked as `python <script command>`. If the value is null or not set, the script is invoked as `<script command>`.	null
hive.exec.plan		null

hive.exec.scratchdir	This directory is used by Hive to store the plans for different map/reduce stages for the query as well as to stored the intermediate outputs of these stages. *Hive 0.14.0 and later:* HDFS root scratch directory for Hive jobs, which gets created with write all (733) permission. For each connecting user, an HDFS scratch directory ${hive.exec.scratchdir}/<username> is created with ${hive.scratch.dir.permission}.	/tmp/<user.name>/hive (Hive 0.8.0 and earlier) /tmp/hive-<user.name> (as of Hive 0.8.1 to 0.14.0) /tmp/hive (Hive 0.14.0 and later)
hive.scratch.dir.permission	The permission for the user-specific scratch directories that get created in the root scratch directory ${hive.exec.scratchdir}. (As of Hive 0.12.0.)	700 (Hive 0.12.0 and later)
hive.exec.local.scratchdir	This directory is used for temporary files when Hive runs in local mode. (As of Hive 0.10.0.)	/tmp/<user.name>
hive.exec.submitviachild	Determines whether the map/reduce jobs should be submitted through a separate jvm in the non local mode.	false - By default jobs are submitted through the same jvm as the compiler
hive.exec.script.maxerrsize	Maximum number of serialization errors allowed in a user script invoked through `TRANSFORM` or `MAP` or `REDUCE` constructs.	100000
hive.exec.compress.output	Determines whether the output of the final map/reduce job in a query is compressed or not.	false
hive.exec.compress.intermediate	Determines whether the output of the intermediate map/reduce jobs in a query is compressed or not.	false
hive.resource.use.hdfs.location	Reference HDFS based files/jars directly instead of copying to session based HDFS scratch directory. (As of Hive 2.2.1.)	true
hive.jar.path	The location of hive_cli.jar that is used when submitting jobs in a separate jvm.	
hive.aux.jars.path	The location of the plugin jars that contain implementations of user defined functions and SerDes.	
hive.reloadable.aux.jars.path	The location of plugin jars that can be renewed (added, removed, or updated) by executing the Beeline reload command, without having to restart HiveServer2. These jars can be used just like the auxiliary classes in hive.aux.jars.path for creating UDFs or SerDes. (As of Hive 0.14.0.)	
hive.partition.pruning	A strict value for this variable indicates that an error is thrown by the compiler in case no partition predicate is provided on a partitioned table. This is used to protect against a user inadvertently issuing a query against all the partitions of the table.	nonstrict
hive.map.aggr	Determines whether the map side aggregation is on or not.	true
hive.join.emit.interval		1000
hive.map.aggr.hash.percentmemory		(float)0.5
hive.default.fileformat	Default file format for CREATE TABLE statement. Options are TextFile, SequenceFile, RCFile, and Orc.	TextFile

hive.merge.mapfiles	Merge small files at the end of a map-only job.	true
hive.merge.mapredfiles	Merge small files at the end of a map-reduce job.	false
hive.merge.size.per.task	Size of merged files at the end of the job.	256000000
hive.merge.smallfiles.avgsize	When the average output file size of a job is less than this number, Hive will start an additional map-reduce job to merge the output files into bigger files. This is only done for map-only jobs if hive.merge.mapfiles is true, and for map-reduce jobs if hive.merge.mapredfiles is true.	16000000
hive.querylog.enable.plan.progress	Whether to log the plan's progress every time a job's progress is checked. These logs are written to the location specified by `hive.querylog.location`. (As of Hive 0.10.)	true
hive.querylog.location	Directory where structured hive query logs are created. One file per session is created in this directory. If this variable set to empty string structured log will not be created.	/tmp/<user.name>
hive.querylog.plan.progress.interval	The interval to wait between logging the plan's progress in milliseconds. If there is a whole number percentage change in the progress of the mappers or the reducers, the progress is logged regardless of this value. The actual interval will be the ceiling of (this value divided by the value of `hive.exec.counters.pull.interval`) multiplied by the value of `hive.exec.counters.pull.interval` i.e. if it is not divide evenly by the value of `hive.exec.counters.pull.interval` it will be logged less frequently than specified. This only has an effect if `hive.querylog.enable.plan.progress` is set to `true`. (As of Hive 0.10.)	60000
hive.stats.autogather	A flag to gather statistics automatically during the INSERT OVERWRITE command. (As of Hive 0.7.0.)	true
hive.stats.dbclass	The default database that stores temporary hive statistics. Valid values are `hbase` and `jdbc` while `jdbc` should have a specification of the Database to use, separated by a colon (e.g. `jdbc:mysql`). (As of Hive 0.7.0.)	jdbc:derby
hive.stats.dbconnectionstring	The default connection string for the database that stores temporary hive statistics. (As of Hive 0.7.0.)	jdbc:derby:;databaseName=TempStatsStore;create=true
hive.stats.jdbcdriver	The JDBC driver for the database that stores temporary hive statistics. (As of Hive 0.7.0.)	org.apache.derby.jdbc.EmbeddedDriver
hive.stats.reliable	Whether queries will fail because stats cannot be collected completely accurately. If this is set to true, reading/writing from /into a partition may fail becuase the stats could not be computed accurately. (As of Hive 0.10.0.)	false
hive.enforce.bucketing	If enabled, enforces inserts into bucketed tables to also be bucketed. (Hive 0.6.0 through Hive 1.x.x only)	false
hive.variable.substitute	Substitutes variables in Hive statements which were previously set using the `set` command, system variables or environment variables. See HIVE-1096 for details. (As of Hive 0.7.0.)	true
hive.variable.substitute.depth	The maximum replacements the substitution engine will do. (As of Hive 0.10.0.)	40
hive.vectorized.execution.enabled	This flag controls the vectorized mode of query execution as documented in HIVE-4160. (As of Hive 0.13.0.)	`false`

Hive Metastore Configuration Variables

Please see Hive Metastore Administration for information about the configuration variables used to set up the metastore in local, remote, or embedded mode. Also see descriptions in the Metastore section of the Language Manual's Hive Configuration Properties.

For security configuration (Hive 0.10 and later), see the Hive Metastore Security section in the Language Manual's Hive Configuration Properties.

Configuration Variables Used to Interact with Hadoop

Variable Name	Description	Default Value
hadoop.bin.path	The location of the Hadoop script which is used to submit jobs to Hadoop when submitting through a separate JVM.	$HADOOP_HOME/bin/hadoop
hadoop.config.dir	The location of the configuration directory of the Hadoop installation.	$HADOOP_HOME/conf
fs.default.name	The default name of the filesystem (for example, localhost for hdfs://<clustername>:8020). For YARN this configuration variable is called fs.defaultFS.	file:///
map.input.file	The filename the map is reading from.	null
mapred.job.tracker	The URL to the jobtracker. If this is set to local then map/reduce is run in the local mode.	local
mapred.reduce.tasks	The number of reducers for each map/reduce stage in the query plan.	1
mapred.job.name	The name of the map/reduce job.	null
mapreduce.input.fileinputformat.split.maxsize	For splittable data this changes the portion of the data that each mapper is assigned. By default, each mapper is assigned based on the block sizes of the source files. Entering a value larger than the block size will decrease the number of splits which creates fewer mappers. Entering a value smaller than the block size will increase the number of splits which creates more mappers.	empty
fs.trash.interval	The interval, in minutes, after which a trash checkpoint directory is deleted. (This is also the interval between checkpoints.) The checkpoint directory is located in `.Trash` under the user's home directory and contains files and directories that were removed since the previous checkpoint. Any setting greater than 0 enables the trash feature of HDFS. When using the Transparent Data Encryption (TDE) feature, set this to 0 in Hadoop core-site.xml as documented in HIVE-10978.	0

Hive Variables Used to Pass Run Time Information

Variable Name	Description	Default Value
hive.session.id	The id of the Hive Session.	
hive.query.string	The query string passed to the map/reduce job.	
hive.query.planid	The id of the plan for the map/reduce stage.	
hive.jobname.length	The maximum length of the jobname.	50
hive.table.name	The name of the Hive table. This is passed to the user scripts through the script operator.	
hive.partition.name	The name of the Hive partition. This is passed to the user scripts through the script operator.	
hive.alias	The alias being processed. This is also passed to the user scripts through the script operator.	

Removing Hive Metastore Password from Hive Configuration

Support for this was added in Hive 0.14.0 with HIVE-7634 and HADOOP-10904. By setting up a CredentialProvider to handle storing/retrieval of passwords, you can remove the need to keep the Hive metastore password in cleartext in the Hive configuration.

1. Set up the CredentialProvider to store the Hive Metastore password, using the key javax.jdo.option.ConnectionPassword (the same key as used in the Hive configuration). For example, the following command adds the metastore password to a JCEKS keystore file at /usr/lib/hive/conf/hive.jceks:

```
$ hadoop credential create javax.jdo.option.ConnectionPassword -provider jceks://file/usr/lib/hive/conf
/hive.jceks
Enter password:
Enter password again:
javax.jdo.option.ConnectionPassword has been successfully created.
org.apache.hadoop.security.alias.JavaKeyStoreProvider has been updated.
```

Make sure to restrict access to this file to just the user running the Hive Metastore server/HiveServer2.
See http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/CommandsManual.html#credential for more information.

2. Update the Hive configuration to use the designated CredentialProvider. For example to use our /usr/lib/hive/conf/hive.jceks file:

```
  <!-- Configure credential store for passwords-->
  <property>
    <name>hadoop.security.credential.provider.path</name>
    <value>jceks://file/usr/lib/hive/conf/hive.jceks</value>
  </property>
```

   This configures the CredentialProvider used by http://hadoop.apache.org/docs/current/api/org/apache/hadoop/conf/Configuration.html#getPassword(java.lang.String), which is used by Hive to retrieve the metastore password.
3. Remove the Hive Metastore password entry (javax.jdo.option.ConnectionPassword) from the Hive configuration. The CredentialProvider will be used instead.
4. Restart Hive Metastore Server/HiveServer2.

Configuring HCatalog and WebHCat

HCatalog

Starting in Hive release 0.11.0, HCatalog is installed and configured with Hive. The HCatalog server is the same as the Hive metastore.

- See Hive Metastore Administration for metastore configuration properties.
- See HCatalog Installation from Tarball for additional information.

For Hive releases prior to 0.11.0, see the "Thrift Server Setup" section in the HCatalog 0.5.0 document Installation from Tarball.

WebHCat

For information about configuring WebHCat, see WebHCat Configuration.