

Cloudera Runtime 7.1.0

Securing Apache Hive

Date published: 2019-08-21

Date modified:

CLOUdera

<https://docs.cloudera.com/>

Legal Notice

© Cloudera Inc. 2025. 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

Hive access authorization.....	4
Transactional table access.....	5
External table access.....	5
HWC authorization.....	5

Hive access authorization

As administrator, you need to understand that the Hive default authorization for running Hive queries is insecure and what you need to do to secure your data. You need to set up Apache Ranger.

To limit Apache Hive access to approved users, Cloudera recommends and supports only Ranger. Authorization is the process that checks user permissions to perform select operations, such as creating, reading, and writing data, as well as editing table metadata. Apache Ranger provides centralized authorization for all Cloudera Runtime Services.

You can set up Ranger to protect managed, ACID tables or external tables using a Hadoop SQL policy. You can protect external table data on the file system by using an HDFS policy in Ranger.

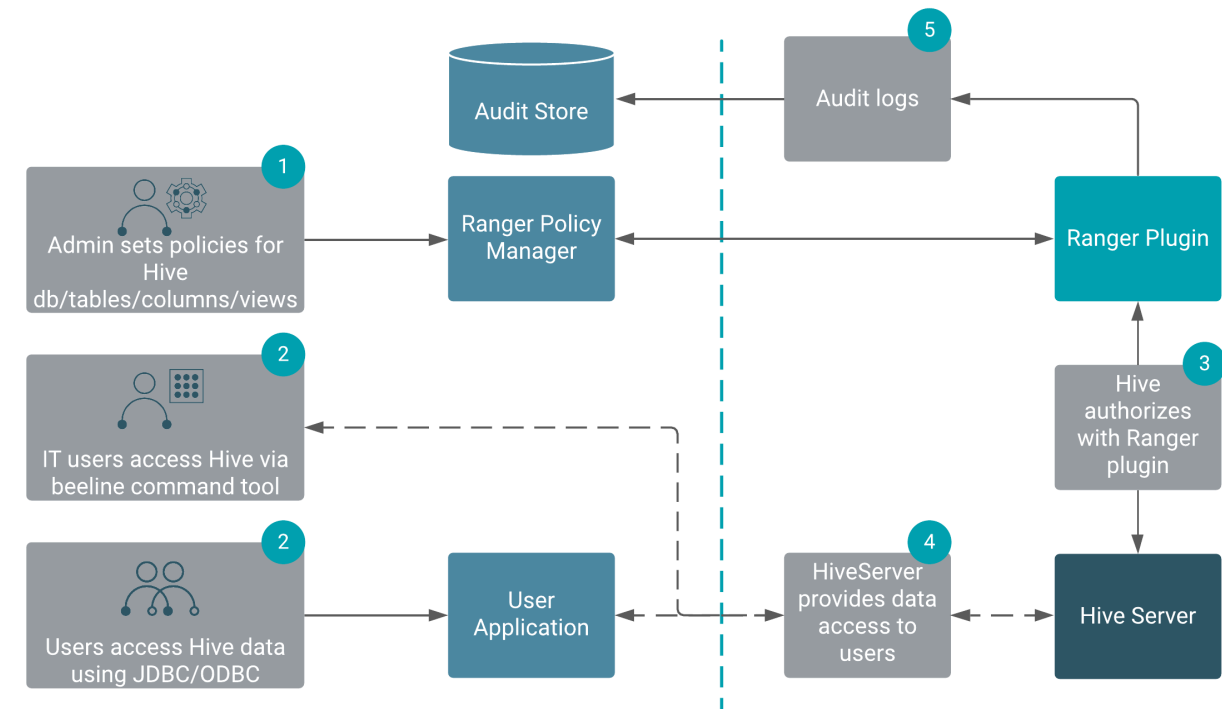
Preloaded Ranger Policies

In Ranger, preloaded Hive policies are available by default. Users covered by these policies can perform Hive operations. All users need to use the default database, perform basic operations such as listing database names, and query the information schema. To provide this access, preloaded default database tables columns and information_ schema database policies are enabled for group public (all users). Keeping these policies enabled for group public is recommended. For example, if the default database tables columns policy is disabled preventing use of the default database, the following error appears:

```
hive> USE default;  
Error: Error while compiling statement: FAILED: HiveAccessControlException  
Permission denied: user [hive] does not have [USE] privilege on [default]
```

Apache Ranger policy authorization

Apache Ranger provides centralized policy management for authorization and auditing of all Cloudera Runtime services, including Hive. All Cloudera Runtime services are installed with a Ranger plugin used to intercept authorization requests for that service, as shown in the following illustration.



The following table compares authorization models:

Authorization model	Secure?	Fine-grained authorization (column, row level)	Privilege management using GRANT/REVOKE statements	Centralized management GUI
Apache Ranger	Secure	Yes	Yes	Yes
Hive default	Not secure. No restriction on which users can run GRANT statements	Yes	Yes	No

When you run grant/revoke commands and Apache Ranger is enabled, a Ranger policy is created/removed.

Transactional table access

As administrator, you must enable the Apache Ranger service to authorize users who want to work with transactional tables. These types of tables are the default, ACID-compliant tables in Hive 3 and later.

ACID tables reside by default in `/warehouse/tablespace/managed/hive`. Only the Hive service can own and interact with files in this directory. Ranger is the only available authorization mechanism that Cloudera recommends for ACID tables.

External table access

As administrator, you must set up Apache Ranger to allow users to access external tables.

External tables reside by default in `/warehouse/tablespace/external` on your object store. To specify some other location of the external table, you need to include the specification in the table creation statement as shown in the following example:

```
CREATE EXTERNAL TABLE my_external_table (a string, b string)
LOCATION '/users/andrena';
```

Hive assigns a default permission of 777 to the hive user, sets a umask to restrict subdirectories, and provides a default ACL to give Hive read and write access to all subdirectories. External tables must be secured using Ranger.

HWC authorization

The way you configure Hive Warehouse Connector (HWC) affects the query authorization process and your security. There are a number of ways to access Hive through HWC, and not all operations go through HiveServer (HS2). Some operations, such as Spark Direct Reader and Hive Streaming, go to Hive directly through HMS where storage-based permissions generally apply.

As a client user, you must be logged in using kerberos before using HWC. You need appropriate storage permissions to write to destination partition or table location. You need to configure an HWC read option. HWC read configuration options are shown in the following table:

Table 1:

Capabilities	JDBC mode	Spark Direct Reader mode
Ranger integration with fine-grained access control	#	N/A
Hive ACID reads	#	#

Capabilities	JDBC mode	Spark Direct Reader mode
Workloads handled	Non-production workloads, small datasets	Production workloads, ETL without fine-grained access control

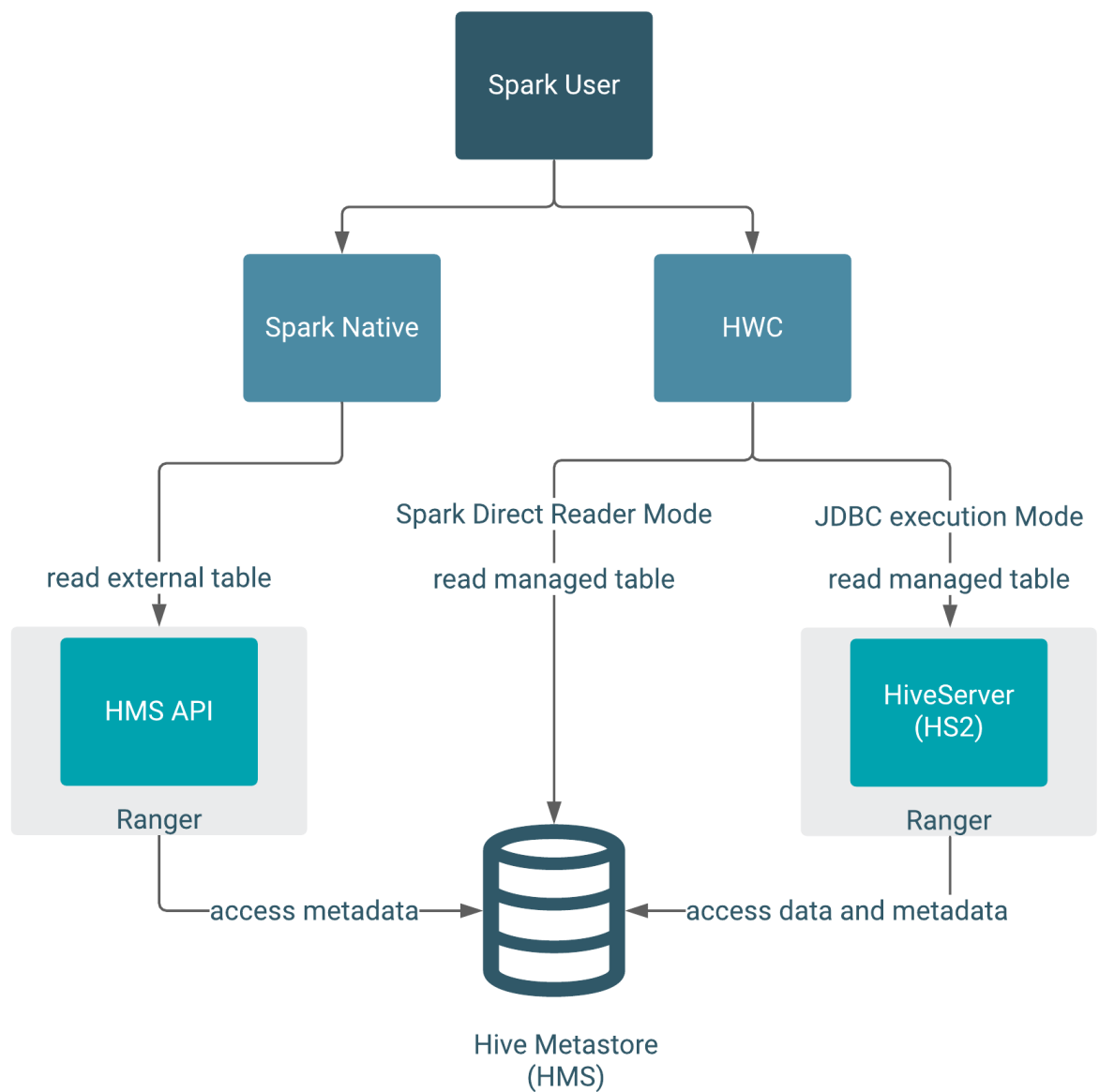
These read configuration options require connections to different Hive components:

- Direct Reader configuration: Connects to Hive Metastore (HMS)
- JDBC configuration: Connects to HiveServer (HS2)

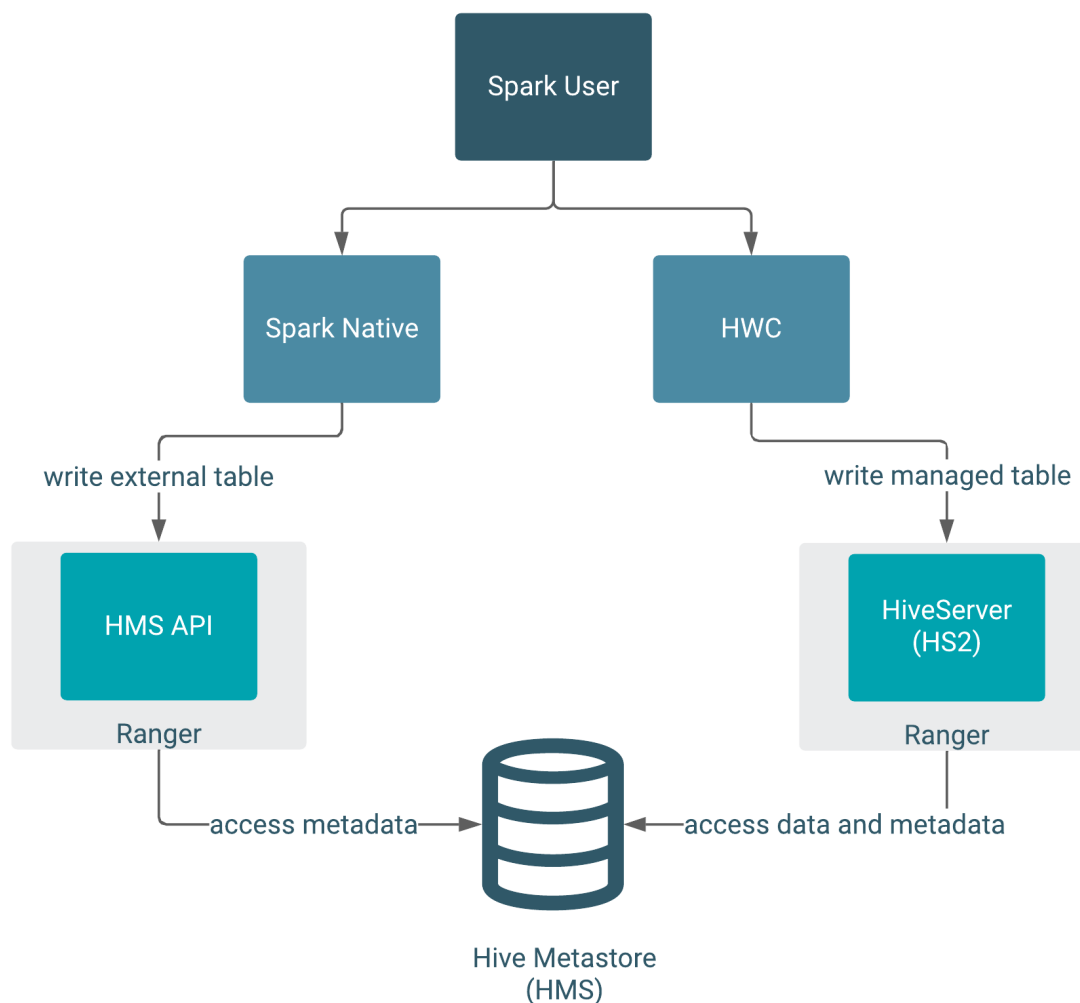
Ranger authorizes access to Hive tables from Spark through HiveServer (HS2), or the Hive metastore API (HMS API).

To write ACID managed tables from Spark to Hive, you must use HWC. To write external tables from Spark to Hive, you can use native Spark or HWC.

The following diagram shows the typical read authorization process:



The following diagram shows the typical write authorization process:



When writing, HWC always enforces authorization through HiveServer (HS2). Reading managed tables in JDBC mode enforces Ranger authorization, including fine-grained features, such as column mapping. In Direct Reader mode, the Ranger and HMS integration provides authorization.

External table queries go through the HMS API, which is also integrated with Ranger. If you do not use HWC, the Hive metastore (HMS) API, integrated with Ranger, authorizes external table access. HMS API-Ranger integration enforces the Ranger Hive ACL in this case. When you use HWC, queries such as DROP TABLE affect file system data as well as metadata in HMS.

Using the Direct Reader option, SparkSQL queries read managed table metadata directly from the HMS, but only if you have permission to access files on the file system. You cannot write to managed tables using the Direct Reader option.

Managed table authorization

A Spark job impersonates the end user when attempting to access an Apache Hive managed table. As an end user, you do not have permission to access, managed files in the Hive warehouse. Managed tables have default file system permissions that disallow end user access, including Spark user access.

As Administrator, you set permissions in Ranger to access the managed tables when you configure HWC for JDBC reads. You can fine-tune Ranger to protect specific data. For example, you can mask data in certain columns, or set up tag-based access control.

When you configure HWC for Direct Reader mode, you cannot use Ranger in this way. You must set read access to the file system location for managed tables. You must have Read and Execute permissions on the Hive warehouse location (`hive.metastore.warehouse.dir`).

External table authorization

Ranger authorization of external table reads and writes is supported. You need to configure a few properties in Cloudera Manager for authorization of external table writes. You must be granted file system permissions on external table files to allow Spark direct access to the actual table data instead of just the table metadata.

Direct Reader Authorization Limitation

As Spark allows users to run arbitrary code, Ranger fine grained access control, such as row level filtering or column level masking, is not possible within Spark itself. This limitation extends to data read using Direct Reader.

To restrict data access at a fine-grained level, use a read option that supports Ranger. Only consider using the Direct Reader option to read Hive data from Spark if you do not require fine-grained access. For example, use Direct Reader for ETL use cases.