Cloudera Runtime 7.3.1

Ranger Auditing

Date published: 2020-07-28
Date modified: 2024-12-10

https://docs.cloudera.com/

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

Audit Overview 4

Managing Auditing with Ranger 4
- Viewing audit details 6
- Viewing audit metrics 9
- Creating a read-only Admin user (Auditor) 10
- Configuring Ranger audit properties for Solr 11
  - Limiting solr spool directory growth 12
- Configuring Ranger audit properties for HDFS 12
- Triggering HDFS audit files rollover 13
- Configuring Ranger audit log storage to a local file 15

Ranger Audit Filters 17
- Default Ranger audit filters 18
- Configuring a Ranger audit filter policy 20
- How to set audit filters in Ranger Admin Web UI 23
- Filter service access logs from Ranger UI 24

Configuring audit spool alert notifications 26

Charting spool alert metrics 30

Excluding audits for specific users, groups, and roles 30

Changing Ranger audit storage location and migrating data 31

Configuring Ranger audits to show actual client IP address 35

Audit Overview

Apache Ranger provides a centralized framework for collecting access audit history and reporting data, including filtering on various parameters. Ranger enhances audit information obtained from Hadoop components and provides insights through this centralized reporting capability.

Ranger plugins support storing audit data to multiple audit destinations.

Solr

The Solr audit destination is a short term audit destination (with a default TTL of 90 days) managed by Solr which can be configured by a Ranger Admin user. The Ranger Admin Web UI displays the access audit data from the audit data stored in Solr.

HDFS

The HDFS audit destination is a long term audit destination for archival/compliance purposes. The HDFS audit destination has no default retention/purge period. A customer must manage the storage/retention/purge/archival of audit data stored in HDFS manually.

Related Information

Configuring Ranger audit properties for Solr

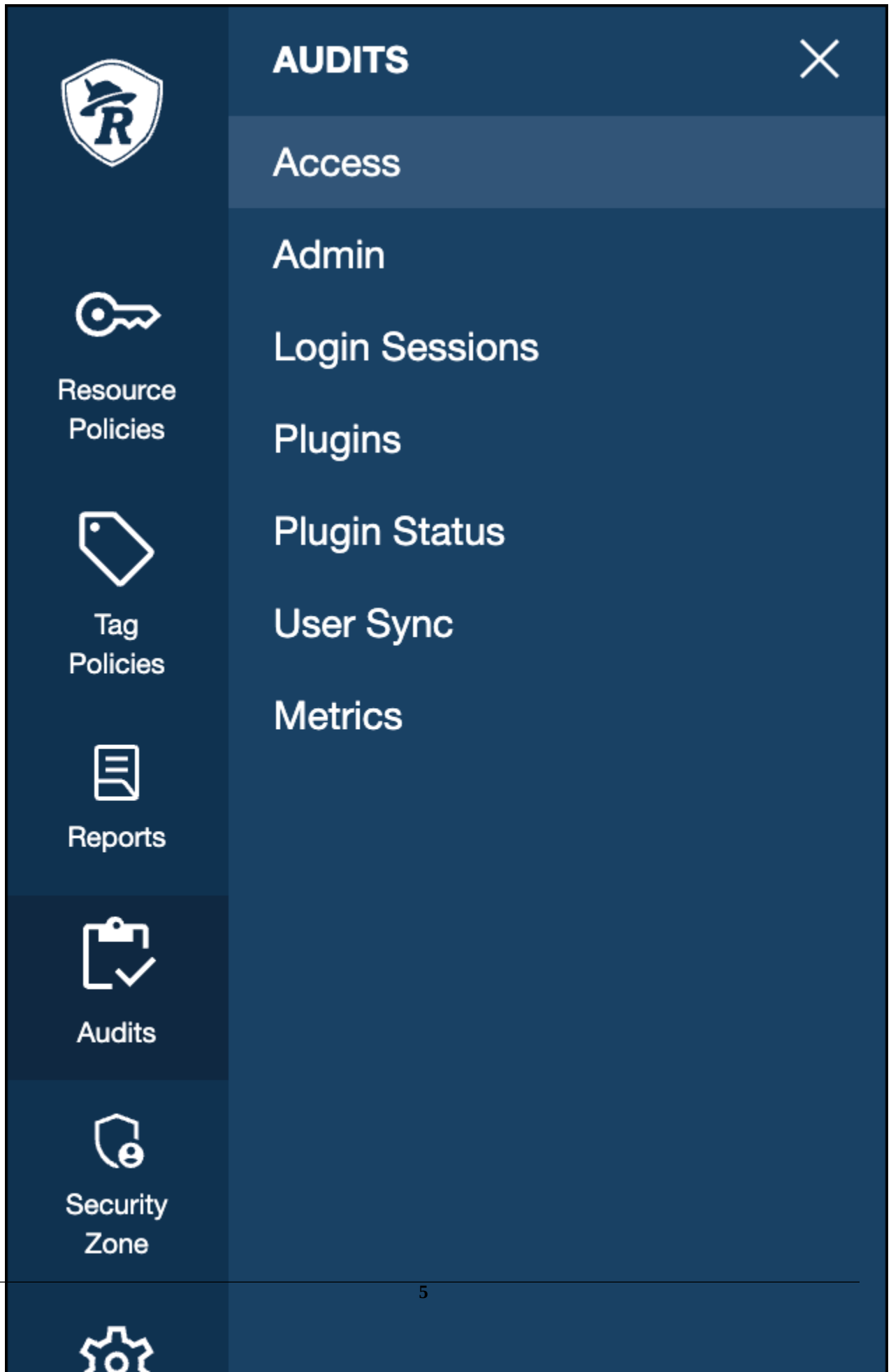
Configuring Ranger audit properties for HDFS

Managing Auditing with Ranger

You can manage auditing using the Audit page in the Ranger Admin Web UI.

To explore options for auditing policies, click Audits

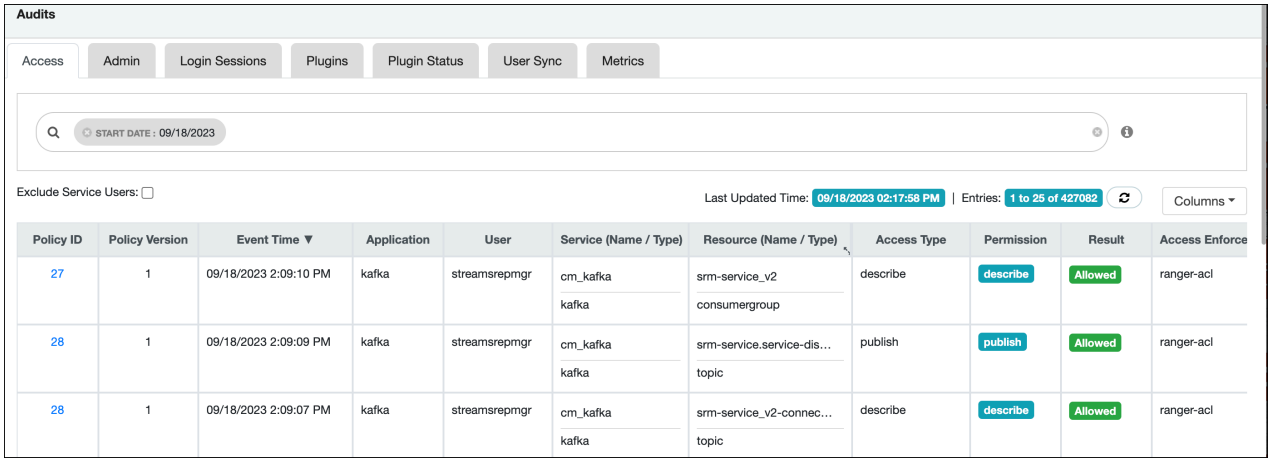
Figure 1: Ranger Admin Audits Menu


AUDITS
Access
Admin
Login Sessions
Plugins
Plugin Status
User Sync
Metrics
Resource Policies
Tag Policies
Reports
Audits
Security Zone

in the left menu of the Ranger Admin Web UI,

then, choose one of the following options:

- Access
- Admin
- Login sessions
- Plugins
- Plugin Status
- User Sync
- Metrics



See related topics for more information about using the Audits pages to manage auditing.

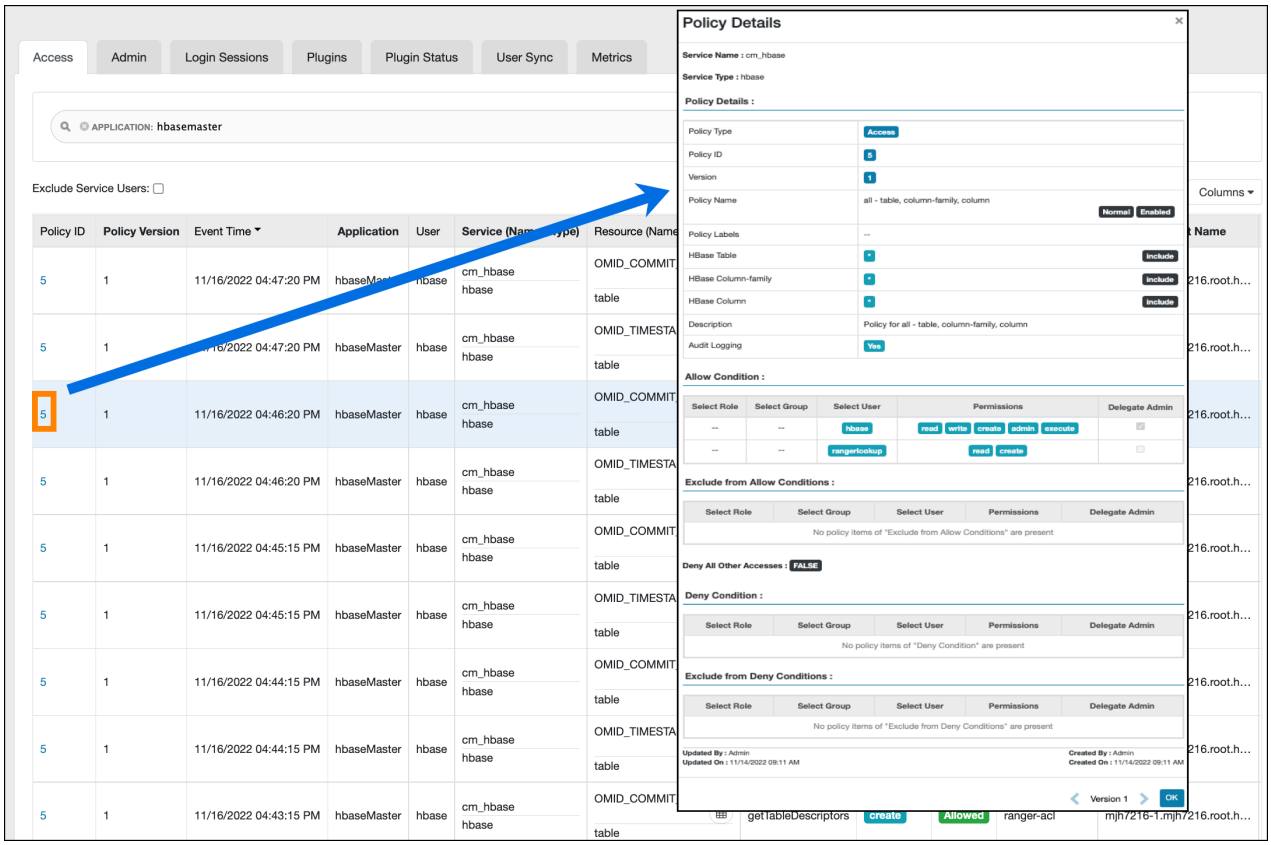
Viewing audit details

How to view policy and audit log details in Ranger audits.

Procedure

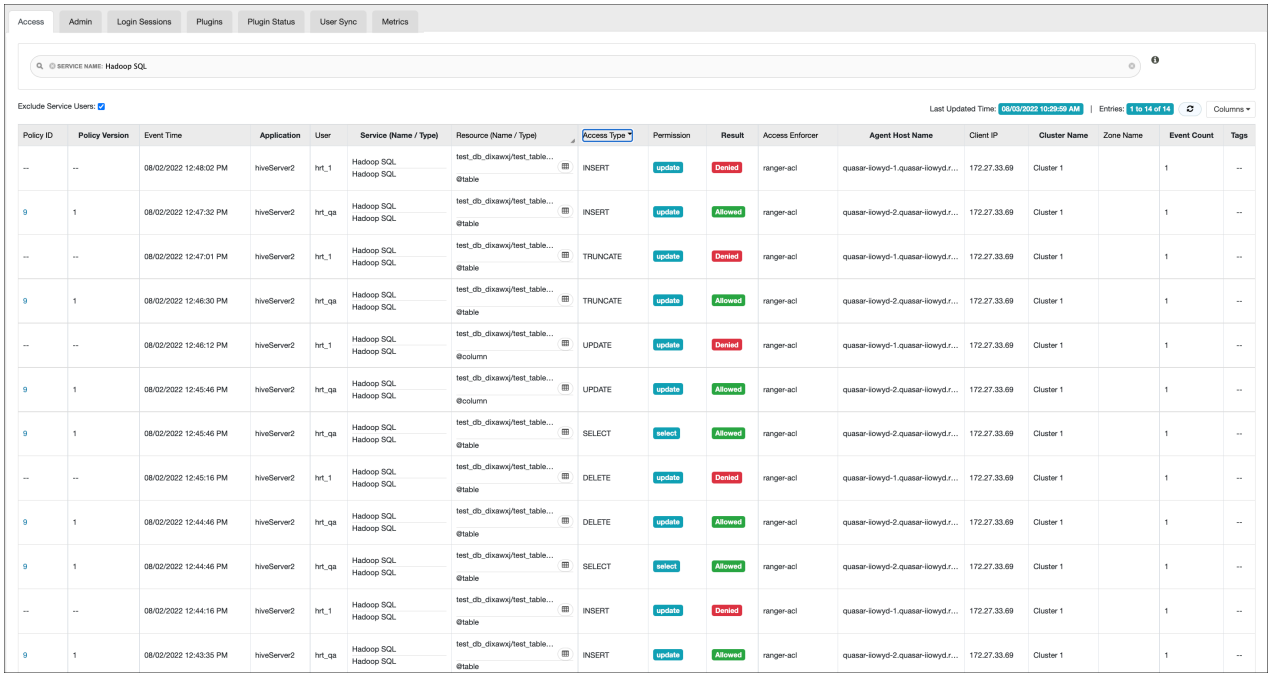
To view policy details for a specific audit log, click Access Policy ID .

Audit > Access: hbasemaster

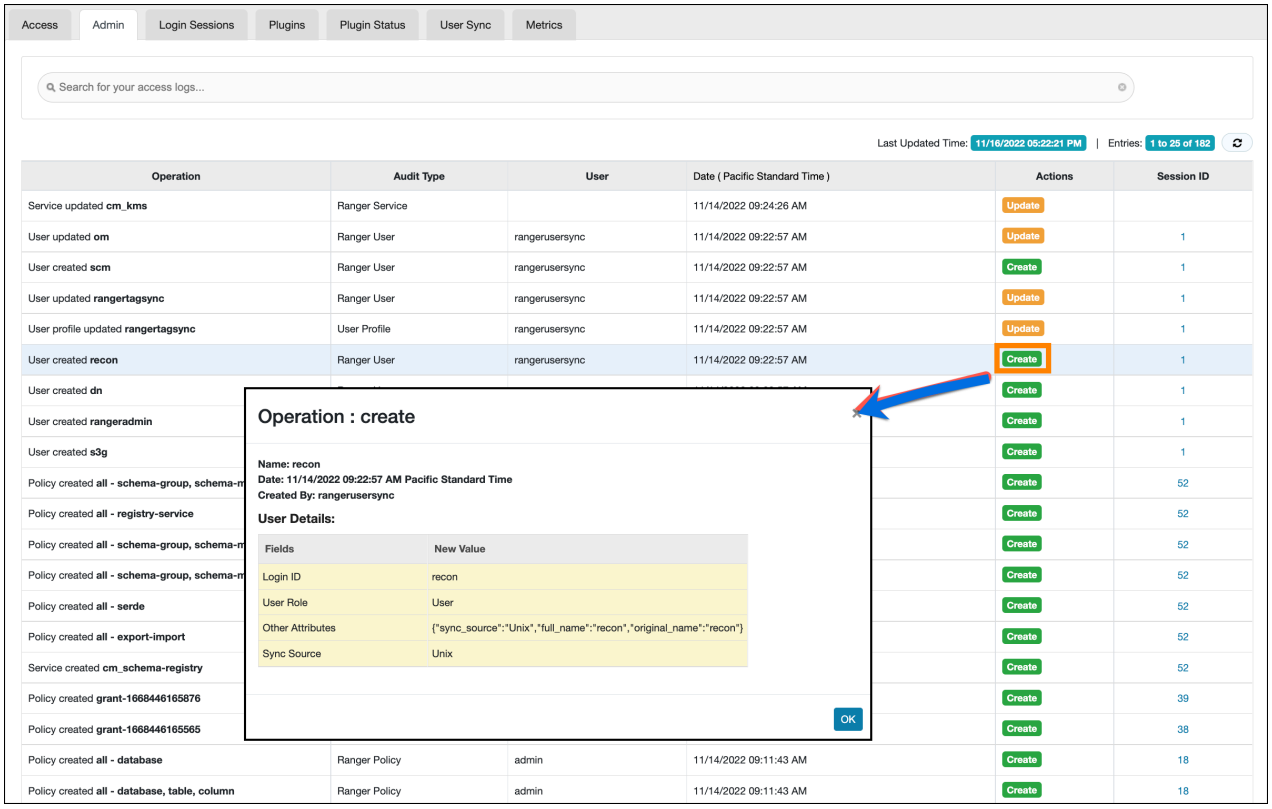


Audit > Access: HadoopSQL

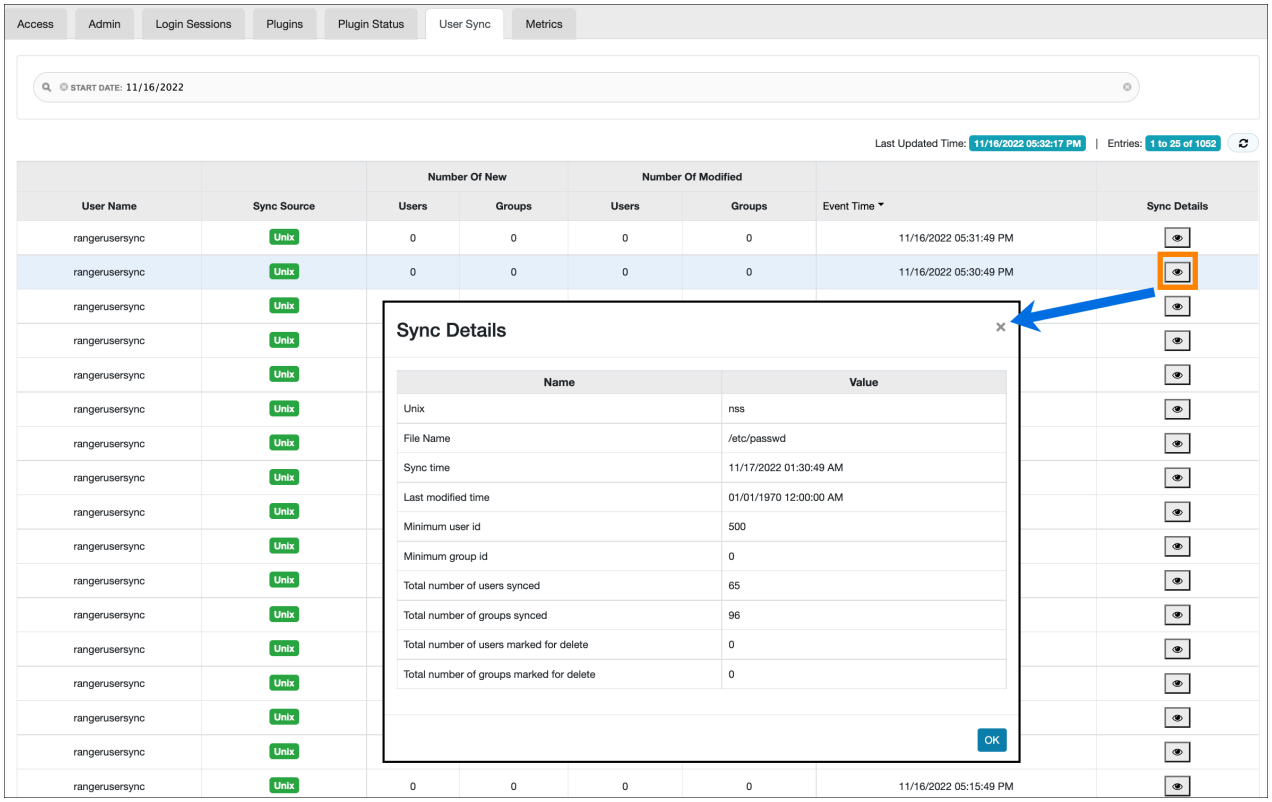
Note: The Hive plugin audit handler now logs UPDATE operations as INSERT, UPDATE, DELETE, and TRUNCATE specifically.



Audit > Admin: Create



Audit > User Sync: Sync details


Access
Admin
Login Sessions
Plugins
Plugin Status
User Sync
Metrics
START DATE: 11/16/2022
Last Updated Time: 11/16/2022 05:32:17 PM | Entries: 1 to 25 of 1052
User Name
Sync Source
Number Of New
Users
Groups
Number Of Modified
Users
Groups
Event Time
Sync Details
rangerusersync
Unix
0
0
0
0
11/16/2022 05:31:49 PM
rangerusersync
Unix
0
0
0
0
11/16/2022 05:30:49 PM
rangerusersync
Unix
rangerusersync
Unix
rangerusersync
Unix

Viewing audit metrics

How to view audit metrics information using the Ranger Admin Web UI.

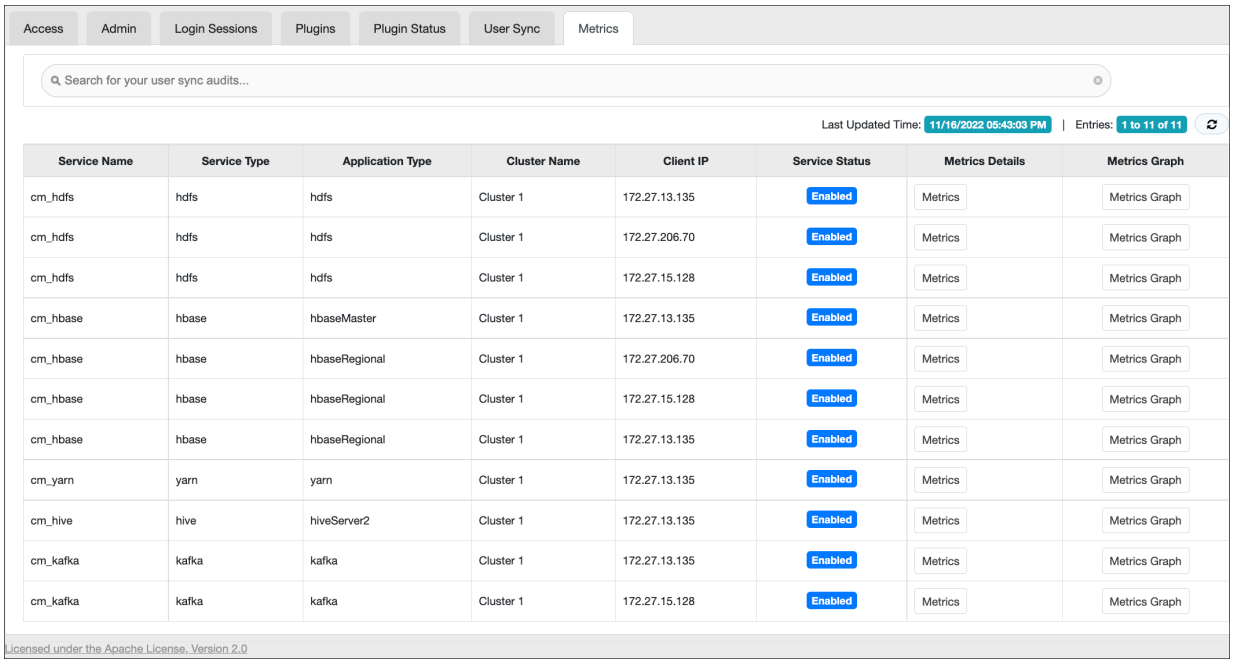
About this task

Metrics provides a high-level view of audit logs as they generate and update in Ranger. Ranger captures audit metrics throughput from the following Ranger services:

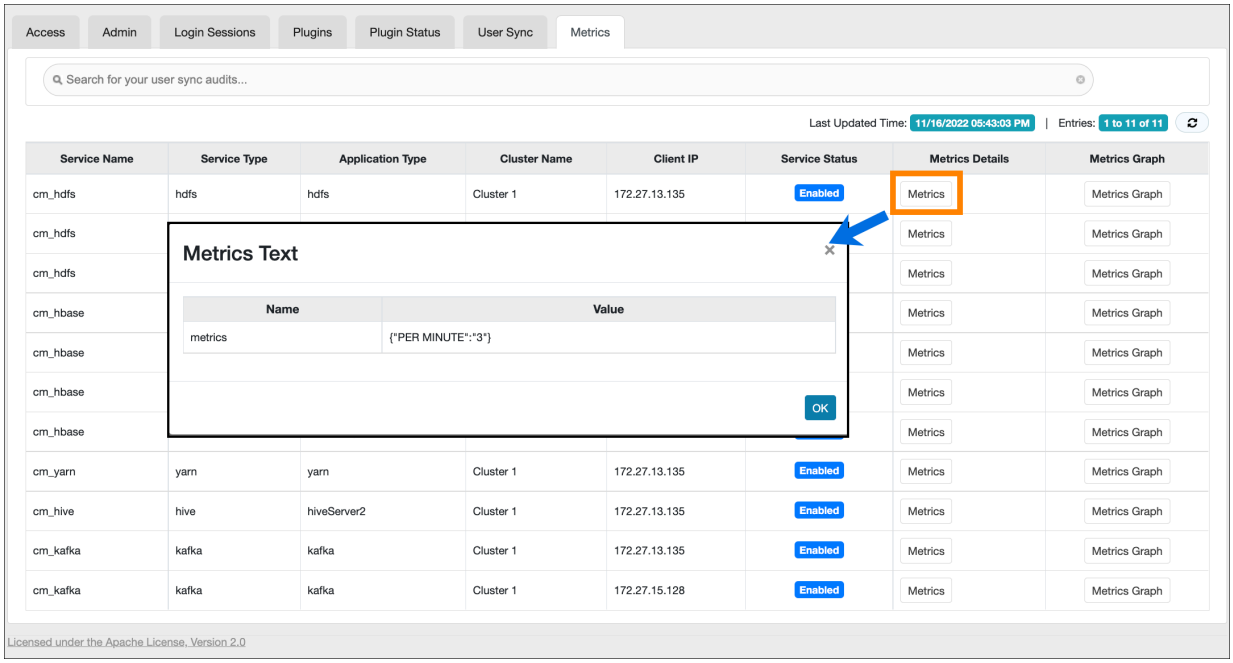
- Atlas
- HBase
- Hdfs
- Hive
- Impala
- Kafka
- Knox
- Kudu
- NiFi
- Schema-registry
- Solr
- Streams Messaging Manager
- Yarn

Procedure

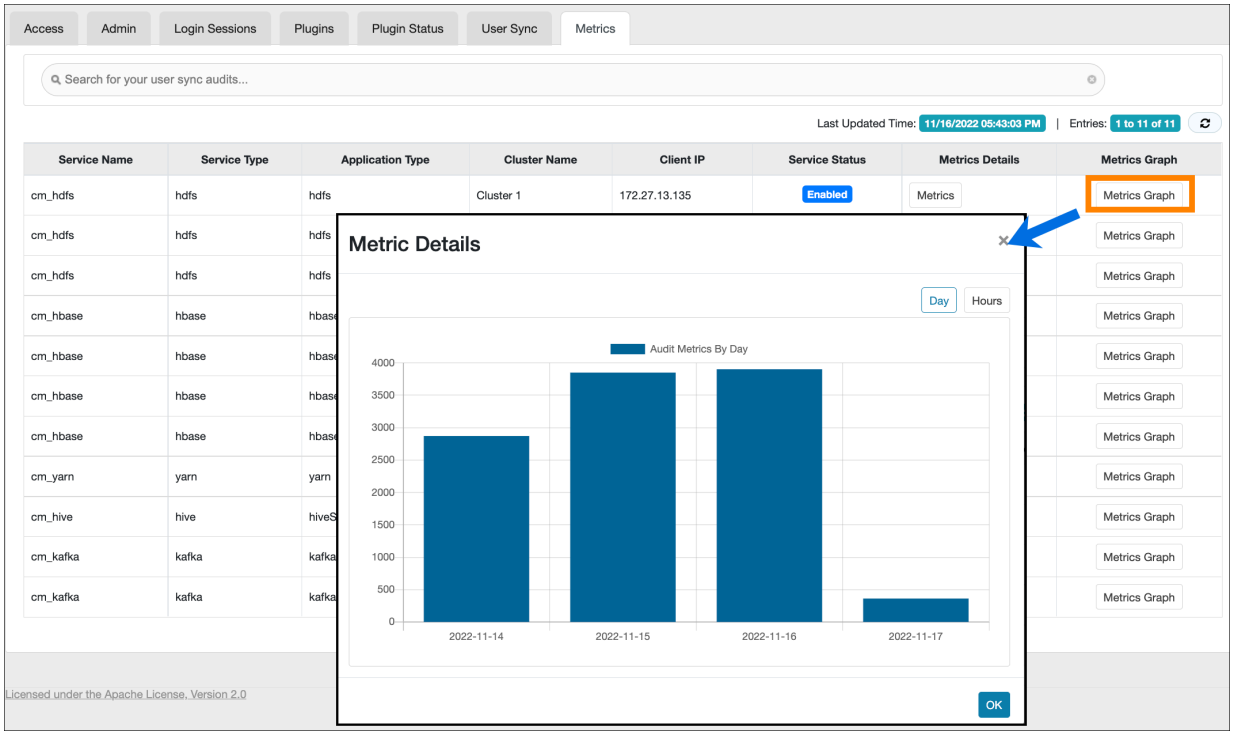
1. To view audit metrics, in the Ranger Admin Web UI, click Audits Metrics .



2. To view metrics details for a specific service, click Metrics.



3. To view hourly or daily metrics as a graphic for a specific service, click Metrics Graph.



Creating a read-only Admin user (Auditor)

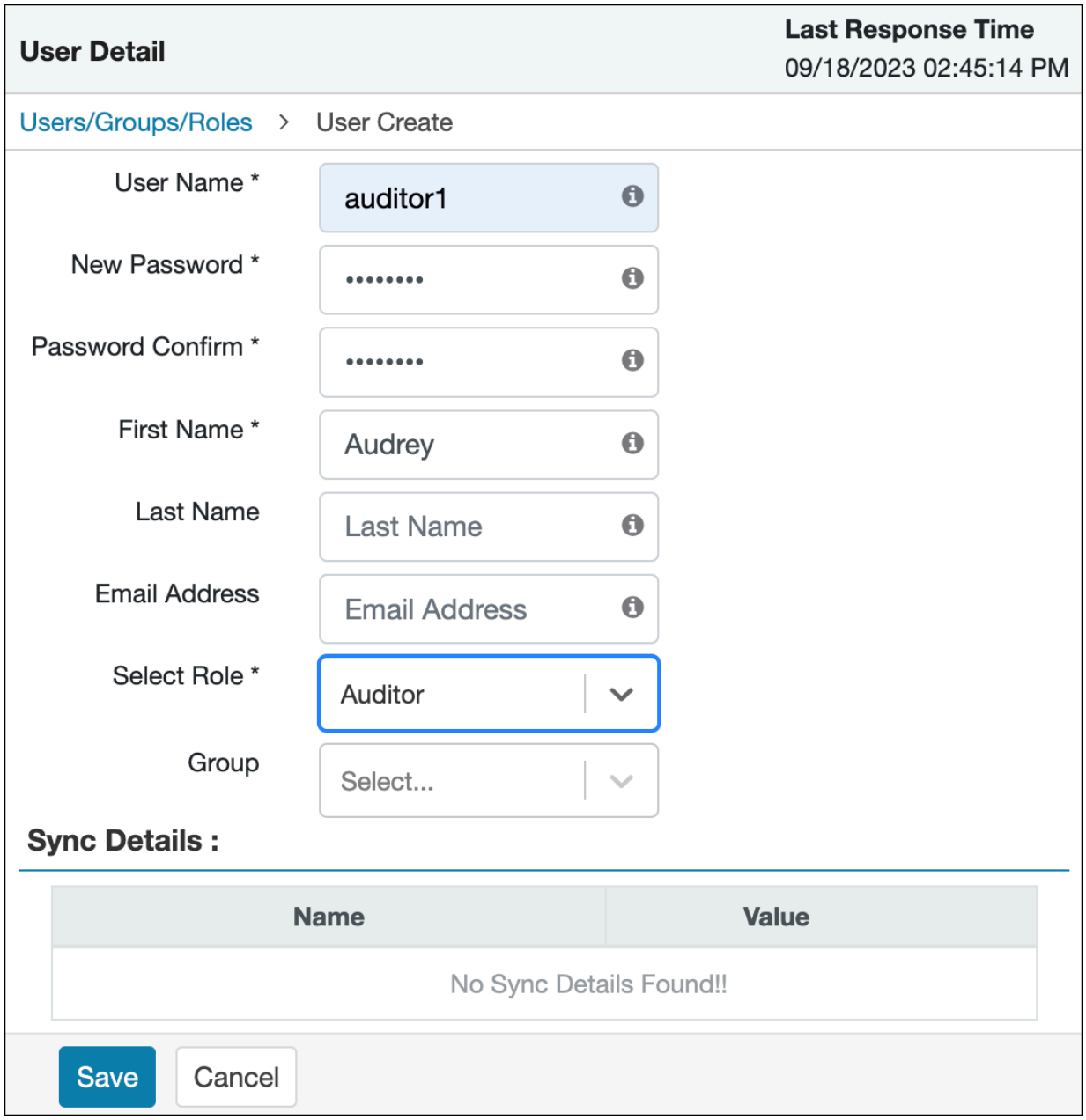
Creating a read-only Admin user (Auditor) enables compliance activities because this user can monitor policies and audit events, but cannot make changes.

About this task

When a user with the Auditor role logs in, they see a read-only view of Ranger policies and audit events. An Auditor can search and filter on access audit events, and access and view all tabs under Audit to understand access events. They cannot edit users or groups, export/import policies, or make changes of any kind.

Procedure

1. In Ranger Admin Web UI, select Settings > Users.
2. Click Add New User.
3. Complete the **User Detail** section, selecting Auditor as the role:



4. Click Save.

Configuring Ranger audit properties for Solr

How to change the default time settings that control how long Ranger keeps audit data collected by Solr.

About this task

The Solr audit destination is intended to store short term audit records .You can configure parameters that control how much data collected by Solr that Ranger will store for auditing purposes.

Table 1: Ranger Audit Configuration Parameters for Solr

Parameter Name	Description	Default Setting	Units
ranger.audit.solr.config.ttl	Time To Live for Solr Collection of Ranger Audits **Note:** "Time To Live for Solr Collection of Ranger Audits" is also known as the Max Retention Days attribute.	90	days
ranger.audit.solr.config.delete.trigger	Auto Delete Period in seconds for Solr Collection of Ranger Audits for expired documents	1	days (configurable)

Procedure

1. From Cloudera Manager choose Ranger Configuration .
2. In Search, type ranger.audit.solr.config, then press Return.
3. In ranger.audit.solr.config.ttl, set the the number of days to keep audit data.
4. In ranger.audit.solr.config.delete.trigger set the number and units (days, minutes, hours, or seconds) to keep data for expired documents
5. Refresh the configuration:
   a) Click Refresh Configuration, as prompted.
   b) In Actions, click Update Solr config-set for Ranger, then confirm.

Limiting solr spool directory growth

To limit stored audit logs, you may set a maximum limit on the solr spool directory size for each service.

Procedure

1. Manually delete the logs under the archive path for the service.
2. Set the log retention value of the archive path from default 100 to 2 .
   a) In Cloudera Manager <service_name> Configuration <service_name> Service Advanced Configuration Snippet (Safety Valve) for <service_name>.xml
   b) Click +
   c) Add the following parameter:

      xasecure.audit.destination.solr.batch.filespool.archive.max.files

      2
3. Restart the service.

Configuring Ranger audit properties for HDFS

How to change the settings that control how Ranger writes audit records to HDFS.

About this task

The HDFS audit destination is intended to store long-term audit records.

You can configure whether Ranger stores audit records in HDFS and at which location.

You must purge long term audit records stored in HDFS manually.

Table 2: Ranger Audit Configuration Parameters for HDFS

Parameter Name	Description	Default Setting	Units
ranger_plugin_hdfs_audit_enabled	controls whether Ranger writes audit records to HDFS	true	T/F
ranger_plugin_hdfs_audit_url	location at which you can access audit records written to HDFS	<hdfs.host_name>/ranger/audit	string

Note: You can also disable storing ranger audit data to hdfs in each service specifically by setting xasecure.audit.destination.hdfs=false in that service.

Procedure

1. From Cloudera Manager choose Ranger Configuration .
2. In Search, type ranger_plugin, then press Return.
3. In ranger_plugin_hdfs_audit_enabled, check/uncheck RANGER-1 (Service Wide)
4. In ranger_plugin_hdfs_audit_url type a valid directory on the hdfs host.
5. Refresh the configuration, using one of the following two options:
   a) Click Refresh Configuration, as prompted or, if Refresh Configuration does not appear,
   b) In Actions, click Update Solr config-set for Ranger, then confirm.

What to do next

(Optional)

You may want to delete older logs from HDFS. Cloudera provides no feature to do this. You may accomplish this task manually, using a script.

Note:

The following example script is not supported by Cloudera. It is shown for reference only. You must test this successfully in a test environment before implementing it in a production cluster.

You must specify the audit log directory by replacing the 2nd line hdfs dfs -ls /<path_to>/<audit_logs> in the example script.

You may also include the -skipTrash option, if you choose, on 7th line in the script.

```
###################
today=`date +'%s'`
hdfs dfs -ls /<path_to>/<audit_logs> | grep "^d" | while read line ; do
dir_date=$(echo ${line} | awk '{print $6}')
difference=$(( ( ${today} - $(date -d ${dir_date} +%s) ) / ( 24*60*60 ) ))
filePath=$(echo ${line} | awk '{print $8}')

if [ ${difference} -gt 30 ]; then
    hdfs dfs -rm -r $filePath
fi
done
###################
```

Related Information

How to do a cleanup of hdfs files older than a certain date using a bash script

Triggering HDFS audit files rollover

How to configure when HDFS audit files close for each service.

About this task

By default, the Ranger Audit framework closes audit files created in HDFS or other cloud storage inline with audit event triggers. In other words, when an audit event occurs, Ranger checks the configured rollout time and then closes the file if the threshold has reached. Default audit rollout time is 24 hours. If no audit event occurs in a 24 hour period, files remain open beyond the 24 hour period. In some environments, audit log analysis that encounter an audit file open beyond the current date can cause system exceptions. If you want the files to be closed every day, so that the audit log file will have only that day's log and the next day's log will be in the next day's file, you can configure the audit framework to close files every day. To do this, you must add several configuration parameters to the ranger-<service_name>-audit.xml (safety valve) file for each service, using Cloudera Manager.

Procedure

1. From Cloudera Manager choose <service_name> Configuration .
2. In <service_name> Configuration Search , type ranger-<service_name>, then press Return.
3. In <service_name> Server Advanced Configuration Snippet (Safety Valve) for ranger-<service_name>-audit.xml, do the following steps:
   a) Click + (Add).
   b) In Name, type xasecure.audit.destination.hdfs.file.rollover.enable.periodic.rollover
   c) In Value, type true.

      When this is enabled Ranger Audit Framework will spawn a Scheduler thread which monitors the occurrence of closing threshold and closes the file. By default every night the file gets closed.
   d) Click + (Add another).
   e) In Name, type xasecure.audit.destination.hdfs.file.rollover.sec
   f) In Value, type an integer value in seconds.

      This is the time in seconds when the file has to be closed. The default value is 86400 sec (1 day) which triggers the file to be closed at midnight and opens a new audit log for the next day. You can override the default value

can be overridden by setting this parameter. For example, if you set the value 3600 (1 hr), the file gets closed every hour.

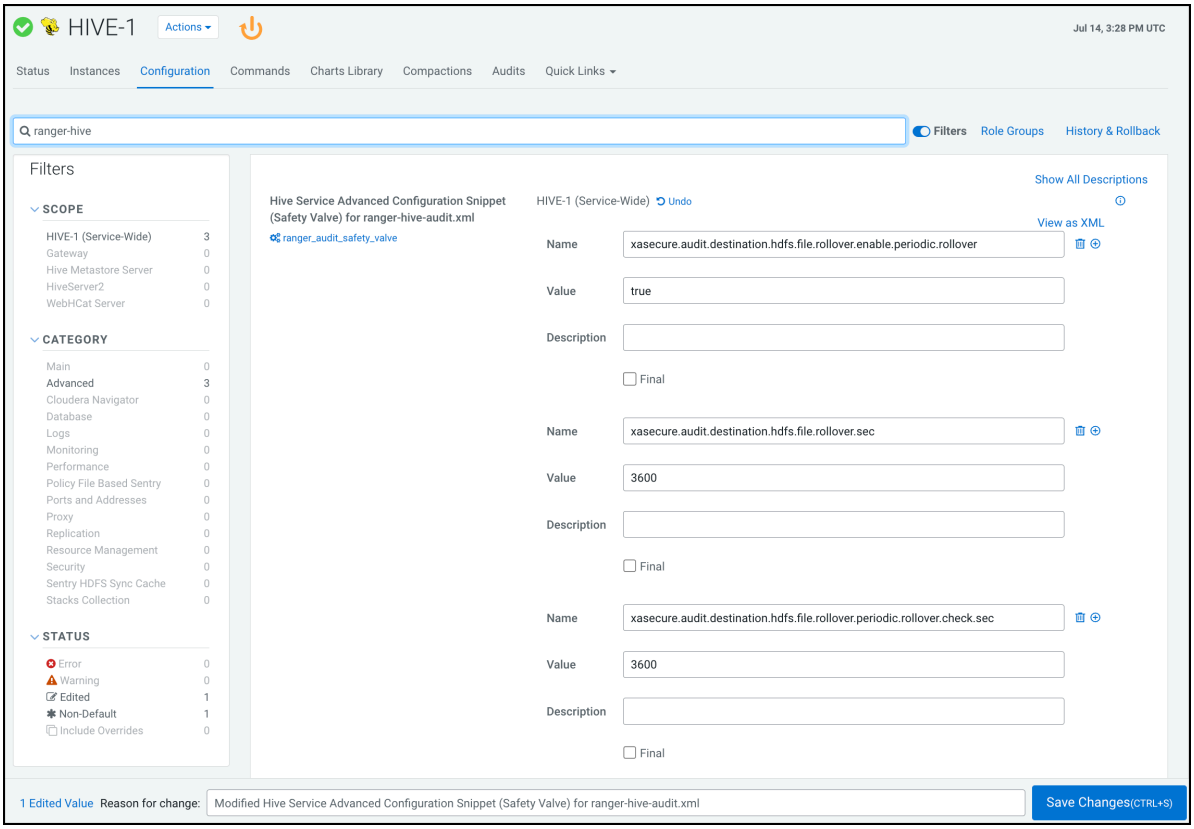
g) Click + (Add another).

h) In Name, type xasecure.audit.destination.hdfs.file.rollover.periodic.rollover.check.sec

i) In Value, type an integer value in seconds.

This is the time frequency of the check to be done whether the threshold time for rollover has occurred. By default the check is done every 60 secs. You can configure this parameter to delay the check time.

Figure 2: Example: Hive service configured to trigger rollover of hdfs audit files



j) Click Save Changes (CTRL+S).

4. Repeat steps 1-3 for each service.
5. Restart the service.

Configuring Ranger audit log storage to a local file

How to configure a service to write Ranger audit log records to a local file, in addition to the default Solr and HDFS locations.

About this task

The Solr and HDFS audit destinations are intended to store short- and long-term audit records respectively, by default. You can configure whether a service also stores Ranger audit records in a local file destination. You may also further configure a service by appending additional, service-specific configuration properties. To do this, first modify ranger-<service_name>-audit.xml file, using Cloudera Manager. Then, add additional appenders to the configuration. This topic shows how to add appenders to the Hive service, for example.

Table 3: Ranger Audit Destination Configuration Parameters for services

Parameter Name	Description	Default Setting	Units
xasecure.audit.destination.log4j	controls whether Ranger writes audit records to a service	true	T/F
xasecure.audit.destination.log4j.logger	location at which you can access audit records written to HDFS	rangerauditlogger	string

Note: You can also disable storing ranger audit data to hdfs in each service specifically by setting xasecure.audit.destination.hdfs=false in that service.

Procedure

1. Configure Ranger Log4JAuditDestination.
   a) From Cloudera Manager choose <service_name> Configuration .
   b) In Search, type ranger-<service_name>-audit, then press Return.
   c) In Hive Service Advanced Configuration Snippet (Safety Valve) for ranger-hive-audit.xml, click +.
   d) Add the following property/value:
   Name

   xasecure.audit.destination.log4j

   Value

   true
   e) Click Save Changes (CTRL+S).
   f) In Hive Service Advanced Configuration Snippet (Safety Valve) for ranger-hive-audit.xml, click +.
   g) Add the following property/value
   Name

   xasecure.audit.destination.log4j.logger

   Value

   rangerauditlogger
   h) Click Save Changes (CTRL+S).
2. Configure HIVE log4 for ranger audit log appender.

   Add the following properties to the configuration: (?? HiveServer2 Logging Advanced Configuration Snippet (Safety Valve) ??)

```
appenders=console, DRFA, redactorForRootLogger, RANGERAUDIT
appenders=console, DRFA, redactorForRootLogger, RANGERAUDIT
appender.RANGERAUDIT.type=RollingRandomAccessFile
appender.RANGERAUDIT.name=RANGERAUDIT
appender.RANGERAUDIT.fileName=${log.dir}/hive-ranger-audit.log
appender.RANGERAUDIT.filePattern=${log.dir}/hive-ranger-audit.log.%d{yyyy-
MM-dd}-%i
appender.RANGERAUDIT.filePermissions=rwxr--r--
appender.RANGERAUDIT.layout.type=PatternLayout
appender.RANGERAUDIT.layout.pattern=%d{ISO8601} %q %5p [%t] %c{2} (%F:%
M(%L)) - %m%n
appender.RANGERAUDIT.policies.type=Policies
appender.RANGERAUDIT.policies.size.type=SizeBasedTriggeringPolicy
appender.RANGERAUDIT.policies.size.size=250MB
appender.RANGERAUDIT.strategy.type=DefaultRolloverStrategy
appender.RANGERAUDIT.strategy.max=10
appender.RANGERAUDIT.strategy.action.type=DELETE
appender.RANGERAUDIT.strategy.action.basepath=${log.dir}
appender.RANGERAUDIT.strategy.action.maxdepth=1
```

```
appender.RANGERAUDIT.strategy.action.PathConditions.glob=hive-ranger-aud
it.*
appender.RANGERAUDIT.strategy.action.PathConditions.type=IfFileName
appender.RANGERAUDIT.strategy.action.PathConditions.nestedConditions.ty
pe=IfAccumulatedFileCount
appender.RANGERAUDIT.strategy.action.PathConditions.nestedConditions.excee
ds=2GB
loggers=Ranger
logger.Ranger.name=rangerauditlogger
logger.Ranger.level=INFO
logger.Ranger.appenderRefs=RANGERAUDIT
logger.Ranger.appenderRef.RANGERAUDIT.ref=RANGERAUDIT
```

3. Restart the service to refresh the configuration.

   The result will be hive-ranger-audit.log getting created in the configured folder. Ranger audit logs for hive operations will be appended.

   Note:

   Change the max number of files to retain the files in the folder. Scripting has to be done to periodically move the hive-ranger-audit.log.* to the processing directory if this feature is used to send these audit logs to other destinations via external applications.

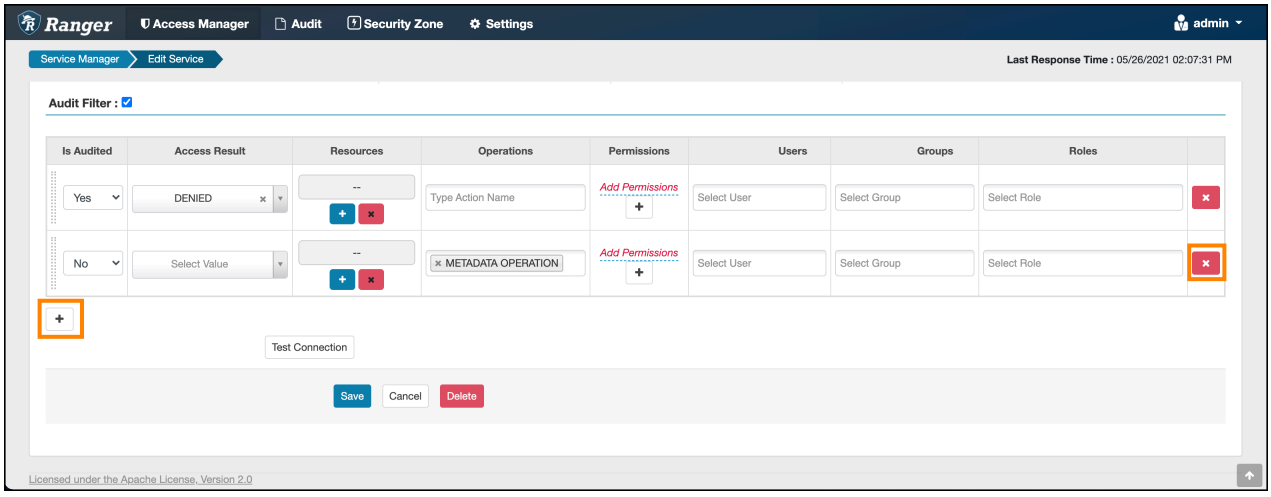
Ranger Audit Filters

You can use Ranger audit filters to control the amount of audit log data collected and stored on your cluster.

About Ranger audit filters

Ranger audit filters allow you to control the amount of audit log data for each Ranger service. Audit filters are defined using a JSON string that is added to each service configuration. The audit filter JSON string is a simplified form of the Ranger policy JSON. Audit filters appear as rows in the Audit Filter section of the Edit Service view for each service. The set of audit filter rows defines the audit log policy for the service. For example, the default audit log policy for the Hadoop SQL service appears in Ranger Admin web UI Service Manager Edit Service when you scroll down to Audit Filter. Audit Filter is checked (enabled) by default. In this example, the top row defines an audit filter that causes all instances of "access denied" to appear in audit logs. The lower row defines a filter that causes no metadata operations to appear in audit logs. These two filters comprise the default audit filter policy for Hadoop SQL service.

Figure 3: Default audit filter policy for the Hadoop SQL service

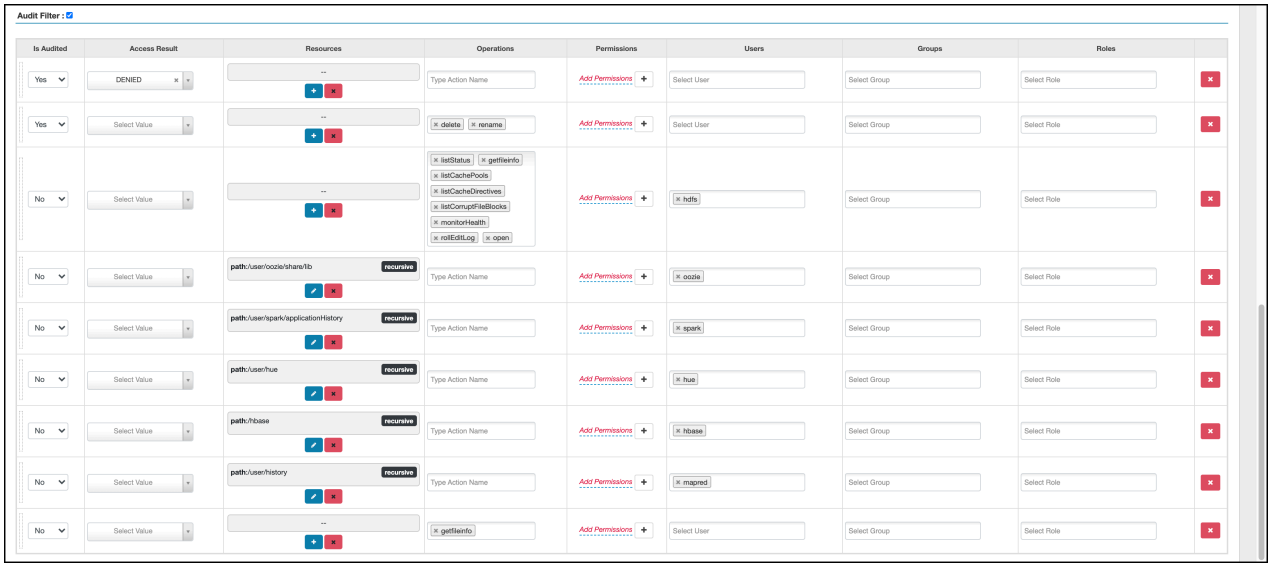


Default Ranger audit filters

Default audit filters for the following Ranger service appear in Edit Services and may be modified as necessary by Ranger Admin users.

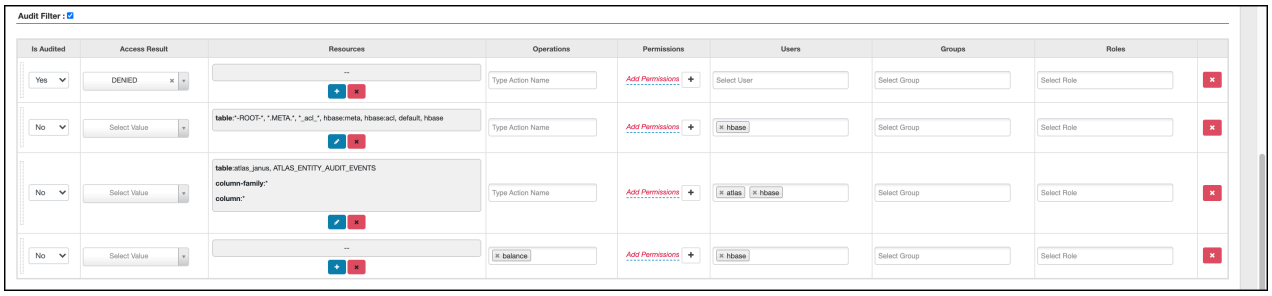
HDFS

Figure 4: Default audit filters for HDFS service



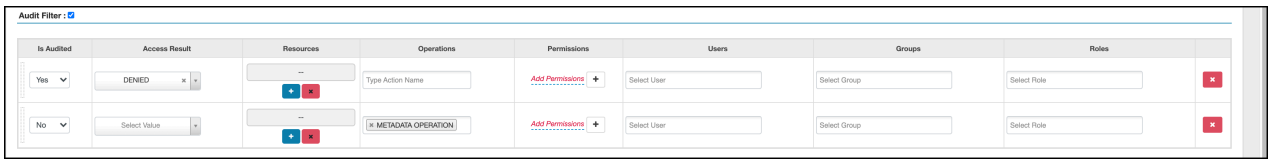
Hbase

Figure 5: Default audit filters for the Hbase service



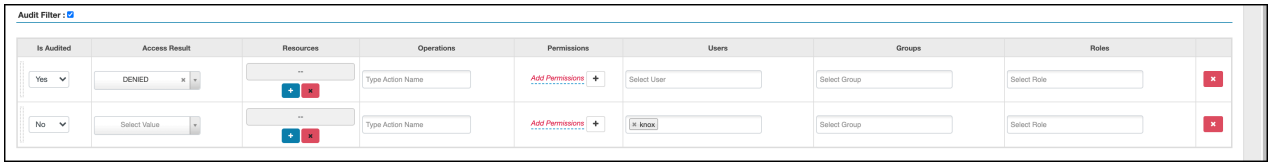
Hadoop SQL

Figure 6: Default audit filters for the Hadoop SQL service



Knox

Figure 7: Default audit filters for the Knox service

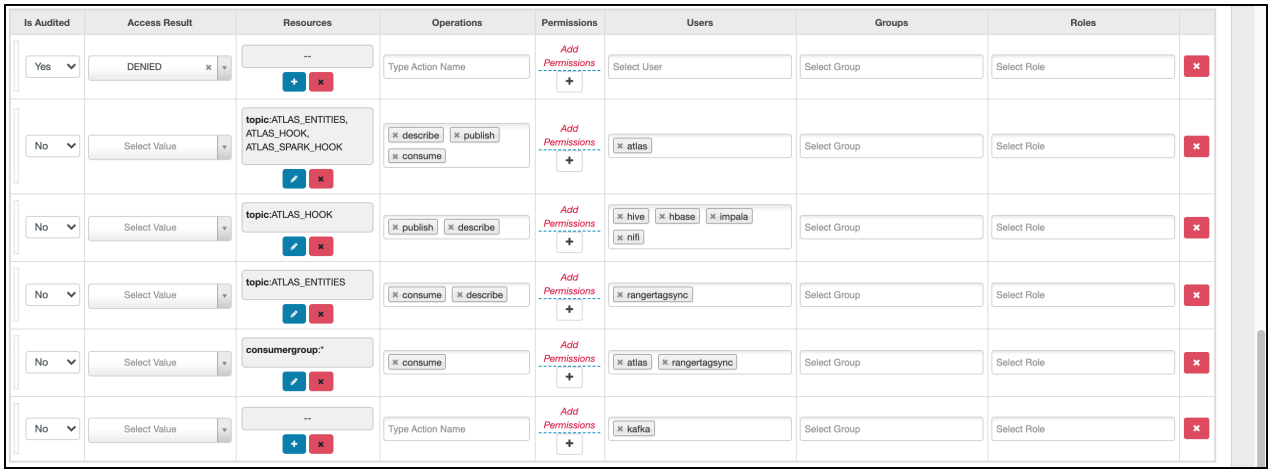


Solr

Figure 8: Default audit filters for the Solr service

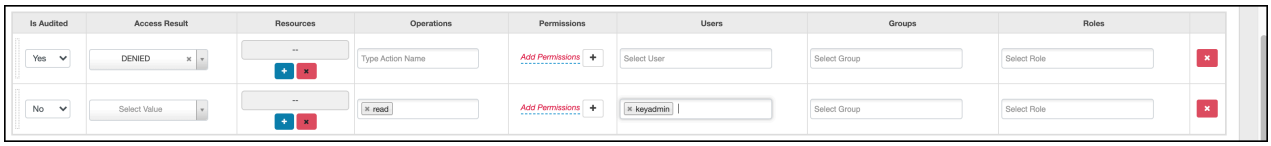
Kafka

Figure 9: Default audit filters for the Kafka service



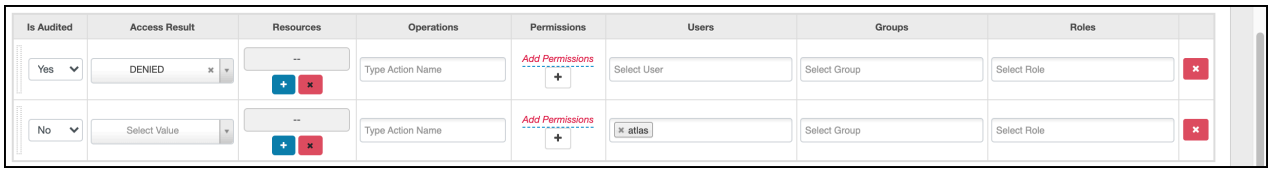
Ranger KMS

Figure 10: Default audit filters for the Ranger KMS service



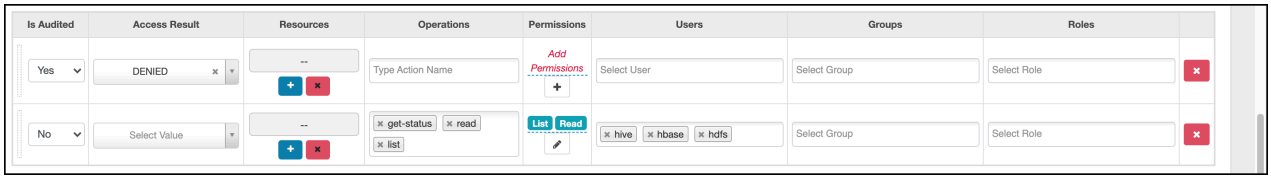
Atlas

Figure 11: Default audit filters for the Atlas service



ADLS

Figure 12: Default audit filters for the ADLS service

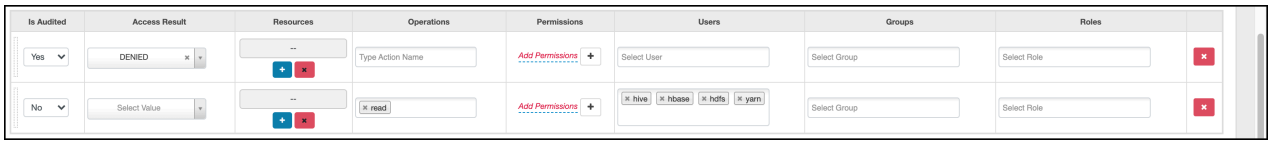


Ozone

Figure 13: Default audit filters for the Ozone service

S3

Figure 14: Default audit filters for the S3 service



Tag-based services

Figure 15: Default audit filters for a tag-based service

Note:

Default audit filter policies do not exist for Yarn, NiFi, NiFi Registry, Kudu, or schema registry services.

Configuring a Ranger audit filter policy

You can configure an audit filter as you add or edit a resource- or tag-based service.

To configure an audit filter policy:

1. In Ranger Admin web UI Service Manager clcik Add or Edit for either a resource-, or tag-based service.
2. Scroll down to Audit Filter.
3. Click Audit Filter flag.

   You configure a Ranger audit filter policy by adding (+), deleting (X), or modifying each audit filter row for the service.

4. Use the controls in the filter row to edit filter properties. For example, you can configure:
   Is Audited: choose Yes or No

   to include or not include a filter in the audit logs for a service

   Access Result: choose DENIED, ALLOWED, or NOT_DETERMINED

   to include that access result in the audit log filter

   Resources: Add or Delete a resource item

   to include or remove the resource from the audit log filter

   Operations: Add or Remove an action name

   to include the action/operation in the audit log filter

   (click x to remove an existing operation)

   Permissions: Add or Remove permissions

   a. Click + in Permissions to open the Add dialog.
   b. Select/Unselect required permissions.

   For example, in HDFS service select read, write, execute, or All permissions.

   Users: click Select User to see a list of defined users

   to include one or multiple users in the audit log filter

   Groups: click Select Group to see a list of defined groups

   to include one or multiple groups in the audit log filter

   Roles: click Select Role to see a list of defined roles

   to include one or multiple roles in the audit log filter

Audit filter details

- When you save the UI selections described in the preceding list, audit filters are defined as a JSON list. Each service references a unique list.
- For example, ranger.plugin.audit.filters for the HDFS service includes:

```
[

                {
                "accessResult":"DENIED",
                "isAudited":true
                },
                {
                "users":[
                "unaudited-user1"
                ],
                "groups":[
                "unaudited-group1"
                ],
                "roles":[
                "unaudited-role1"
                ],
                "isAudited":false
                },
                {
                "actions":[
                "listStatus",
                "getfileinfo"
                ],
                "accessTypes":[
                "execute"
                ],
```

```
                "isAudited":false
                },
                {
                "resources":{
                "path":{
                "values":[
                "/audited"
                ],
                "isRecursive":true
                }
                },
                "isAudited":true
                },
                {
                "resources":{
                "path":{
                "values":[
                "/unaudited"
                ],
                "isRecursive":true
                }
                },
                "isAudited":false
                }
                ]
```

- Each value in the list is an audit filter, which takes the format of a simplified Ranger policy, along with access results fields.
- Audit filters are defined with rules on Ranger policy attributes and access result attributes.
  - Policy attributes: resources, users, groups, roles, accessTypes
  - Access result attributes: isAudited, actions, accessResult
- The following audit filter specifies that accessResult=DENIED will be audited.

  The isAudited flag specifies whether or not to audit.

  ```
  {"accessResult":"DENIED","isAudited":true}
  ```
- The following audit filter specifies that “resource => /unaudited” will not be audited.

  ```
  {"resources":{"path":{"values":["/
  unaudited"],"isRecursive":true}},"isAudited":false}
  ```
- The following audit filter specifies that access to resource database=> sys table=> dump by user “use2” will not be audited.

  ```
  {"resources":{"database":{"values":["sys"]},"table":{"values":
  ["dump"]}},"users":["user2"],"isAudited":false}
  ```
- The following audit filter specifies that access result in actions => listStatus, getfileInfo and accessType => execute will not be audited.

  ```
  {"actions":["listStatus","getfileinfo"],"accessTypes":
  ["execute"],"isAudited":false}
  ```
- The following audit filter specifies that access by user "superuser1" and group "supergroup1" will not be audited.

  ```
  {"users":["superuser1"],"groups":["supergroup1"],"isAudited":false}
  ```
- The following audit filter specifies that access to any resource tagged as NO_AUDIT will not be audited.

  ```
  {"resources":{"tag":{"values":["NO_AUDIT"]}},"isAudited":false}
  ```

How to set audit filters in Ranger Admin Web UI

You can set specific audit filter conditions for each service, using Create/Edit Service .

About this task

Creating audit filters for a service using the Ranger Admin Web UI can prevent audit logs from being sent to destinations like SOLR and HDFS.

Procedure

1. In the Ranger Admin Web UI Service Manager , click Add New Service or Edit (existing service).
2. On Create/Edit Service, scroll down to Audit Filters.
   a) Verify that Audit Filter is checked.

   Optionally, define any of the following to include in the filter definition:

   Is Audited

   Defines whether audit logs are stored or not.

   Is Audited=Yes: stores audit records in the defined audit destination.

   Is Audited=No: do not store audit records.

   Access Results

   Denied, Allowed, or Not Determined

   select to filter access=denied, access=allowed or access=Not determined

   Resource

   use Resource Details to include or exclude specific resources such as databases, tables, or columns.

   Operations

   select specific operations to filter

   Note: For Operations field, you can refer to the Ranger access audit page and find out the available access types.

   Permissions

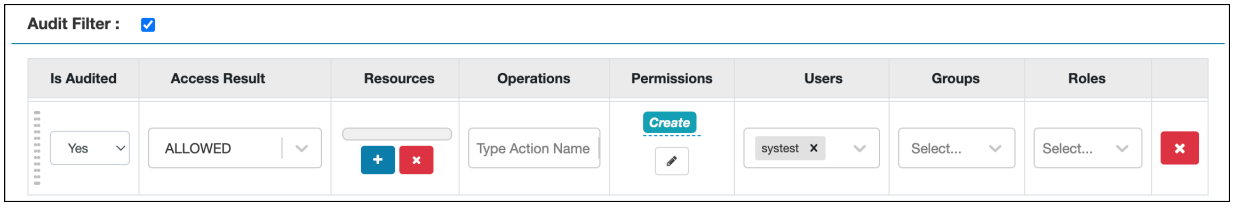
   select specific permissions

   Users, Groups, Roles

   select specific users, groups, and roles

   b) Click Save.

Figure 16: Adding an audit filter that stores user systest, access=Allowed logs for Hive service



3. Test your filters to verify that defined audit filters perform as expected.

Results

Defining specific filtering properties can prevent access logs for service users from being stored in the configured audit destination, if Is Audited = No.

Filter service access logs from Ranger UI

You can limit display of system access/audit log records generated by service users in each service.

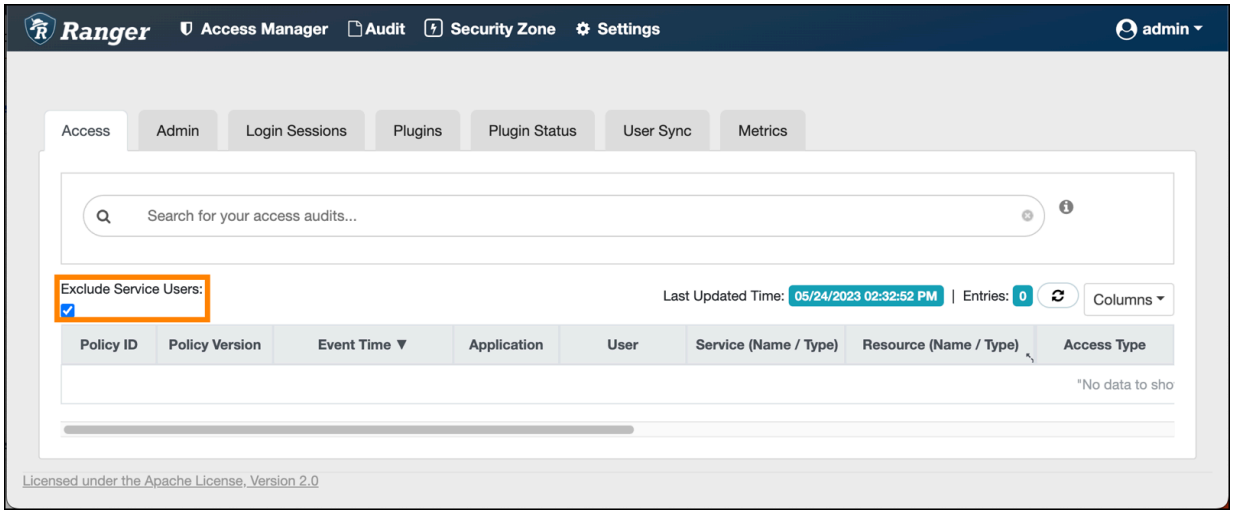
About this task

This topic describes how to limit the display of access log records on the Access tab in the Ranger Admin Web UI.

Procedure

1. Go to Ranger Admin Web UI Audit Access .
2. Check the Exclude Service Users box, as shown in:

Figure 17: Setting the Exclude Service Users flag to true



3. Define specific component services and users for access logs to filter out, in ranger-admin-site.xml.
   a) Go to Cloudera Manager Ranger Configuration
   b) In Search, type ranger-admin-site.
   c) Define the following properties:

   Name

   ranger.plugins.<service_name>.serviceuser

   Value

   <service_name>

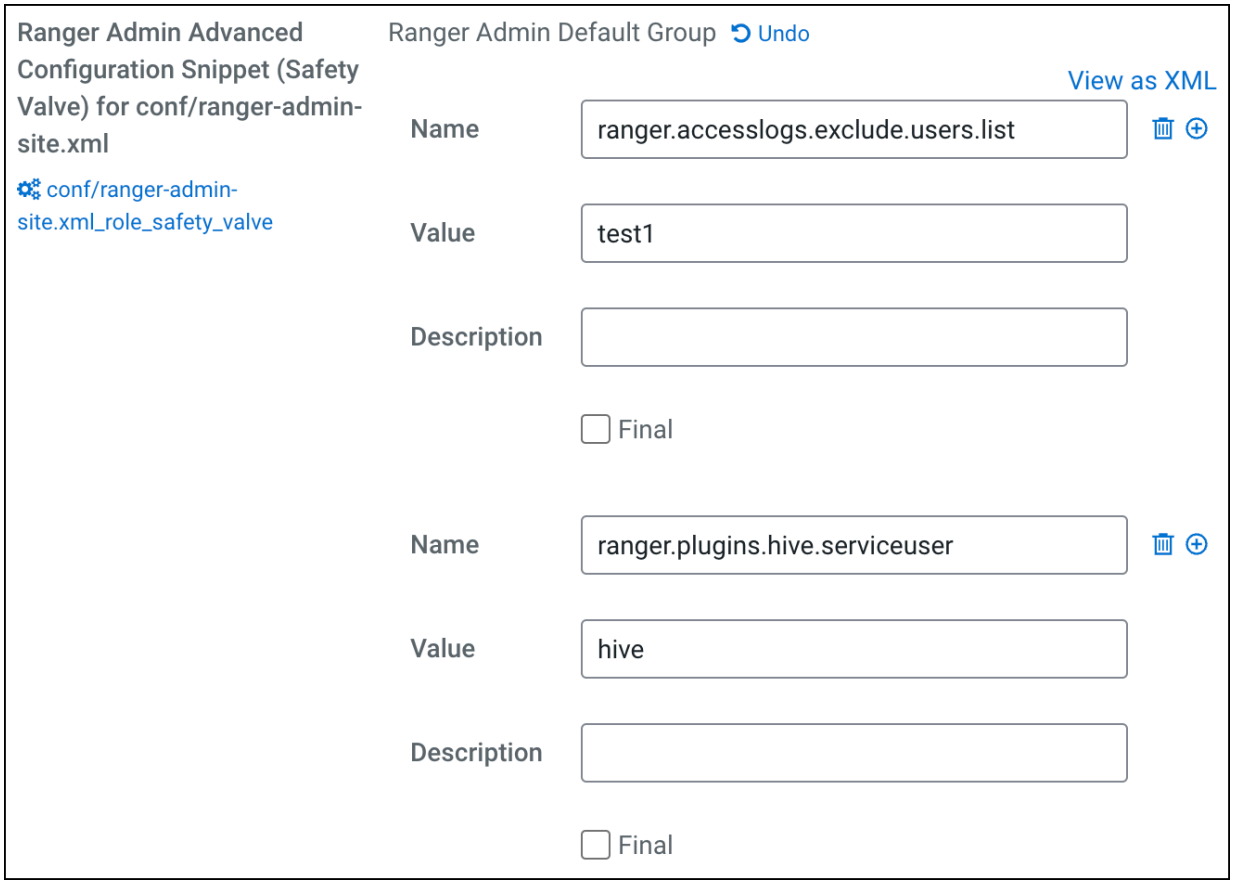
   Name

   ranger.accesslogs.exclude.users.list

   Value

   user1, user2

Figure 18: Filtering out service and user logs for Hive service



4. Click Save Changes (CTRL+S).
5. Restart the Ranger service.

Results

Setting Exclude Service Users to true and defining specific filtering properties prevents audit logs from service users from appearing on Ranger Admin Web UI Audit Access , but does NOT prevent access logs for service users from being generated in Solr.

Configuring audit spool alert notifications

You can enable and configure alerts for Ranger plugin supported services on Cloudera Manager that notify when audit spool files are accumulated.

About this task

Ranger stores plugin access audit events (audit logs) in Solr and in HDFS. Typically, you store audit logs in Solr for short-term auditing purposes and in HDFS for longer-term purposes. When the Solr server is down, Ranger plugin audit logs are stored in the spool directory as spool files. You configure the spool directory location using the following properties:

For Solr

xasecure.audit.destination.solr.batch.filespool.dir = /var/log/<service_name>/audit/solr/spool

For HDFS

xasecure.audit.destination.hdfs.batch.filespool.dir = /var/log/<service_name>/audit/hdfs/spool

If the Solr server goes down for a long period of time or, if a large number of audit events occur while the Solr server is down; then spool files accumulate in the spool directory. Spool file accumulation consumes system memory. Sometimes, audit records in spool files become corrupted, and may not be restored when the Solr server returns to a running state. Corrupted, un-restored records also cause spool file accumulation. This requires manual cleanup of corrupted spool files. An unnoticed large accumulation or "piling up" of spool files may fill the local filesystem and result in service failure.

After you enable spool directory metric usage for a service, an alert appears on the Cloudera Manager UI which notifies the user when spool files have piled up in the spool directory. The Cloudera Manager agent measures the disk usage of the spool directory and registers it as a metric value. This metric value is compared against a threshold value. The spool alert appears on the Cloudera Manager UI if the metric value is greater than the threshold value.

Single-level metrics for collecting the disk usage of Solr and Hdfs Ranger plugin spool directories are registered in Cloudera Manager, using the following disk usage metric names:

Solr

ranger_plugin_solr_spool_directory_size

HDFS

ranger_plugin_hdfs_spool_directory_size

The following table lists Ranger plugin supported service names and roles that support spool alerts.

Table 4: Ranger plugin supported services and their roles supporting audit spool alerts

Services	Roles
HDFS	NAMENODE
HIVE	HIVEMETASTORE
HIVE_ON_TEZ	HIVESERVER2
HBASE	MASTER, REGIONSERVER
YARN	RESOURCEMANAGER
IMPALA	IMPALAD, CATALOGSERVER
ATLAS	ATLAS_SERVER
KAFKA	KAFKA_BROKER
KNOX	KNOX_GATEWAY
KUDU	KUDU_MASTER

Services	Roles
RANGER_KMS	RANGER_KMS_SERVER
RANGER_RAZ	RANGER_RAZ_SERVER
SCHEMAREGISTRY	SCHEMA_REGISTRY_SERVER
STREAMS_MESSAGING_MANAGER	STREAMS_MESSAGING_MANAGER_SERVER

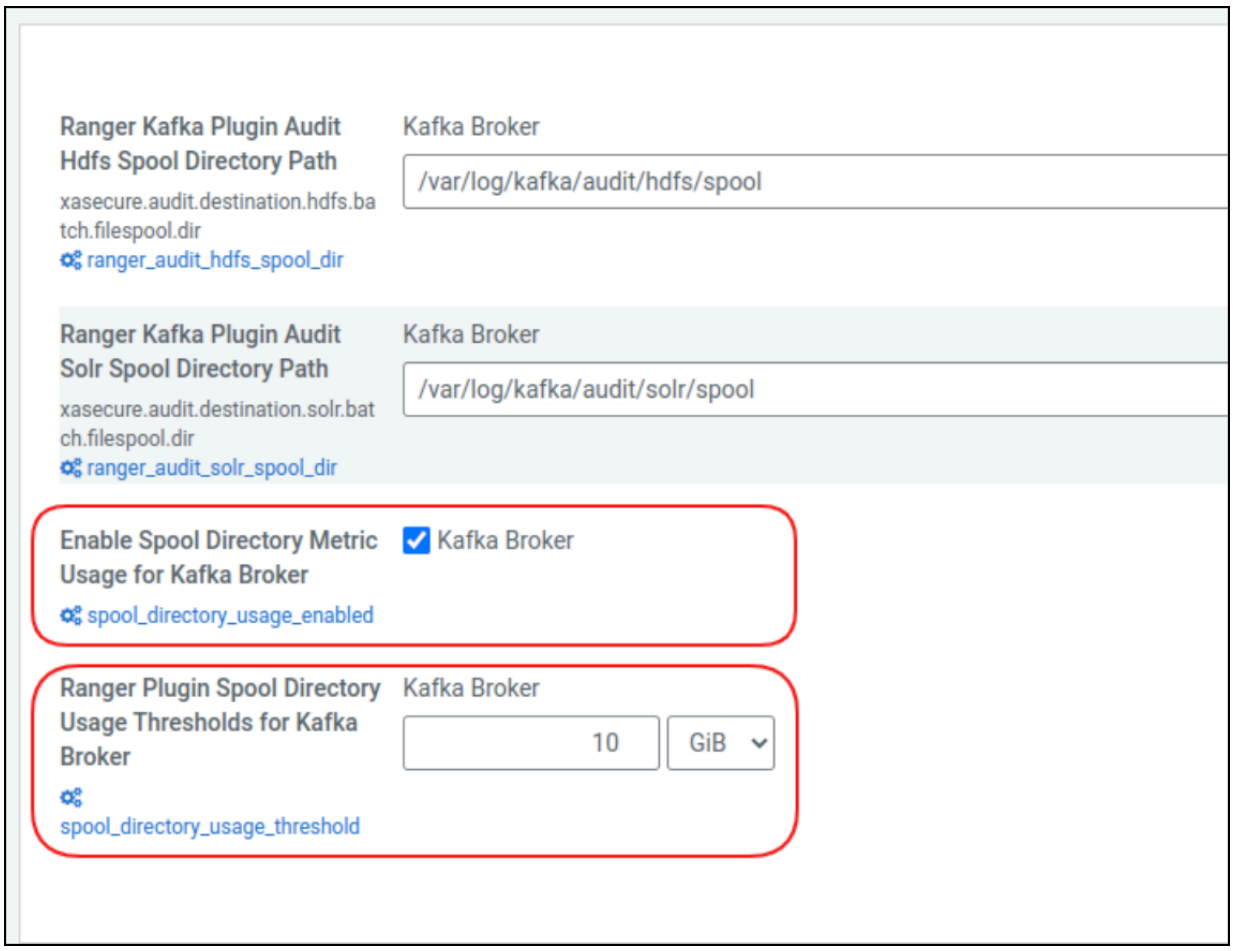
To enable or disable spool directory alerts:

Procedure

1. Go to Cloudera Manager Ranger Plugin Supported Services page Configuration .
2. In Search, type spool directory.

   The following configuration properties display (this example uses the Kafka Broker role from Kafka service)

   Figure 19: Audit Spool Alert Configurations for Kafka Broker



3. In Enable Spool Directory Metric Usage for <service-name>, check the box.

   (un-checking the Enable box disables spool directory metric usage)

   a) Refresh the role.

      For example, the Kafka service which supports Ranger Plugin in Kafka Broker role, go to Cloudera Manager Kafka Instances Kafka Broker Actions Refresh Kafka Broker .

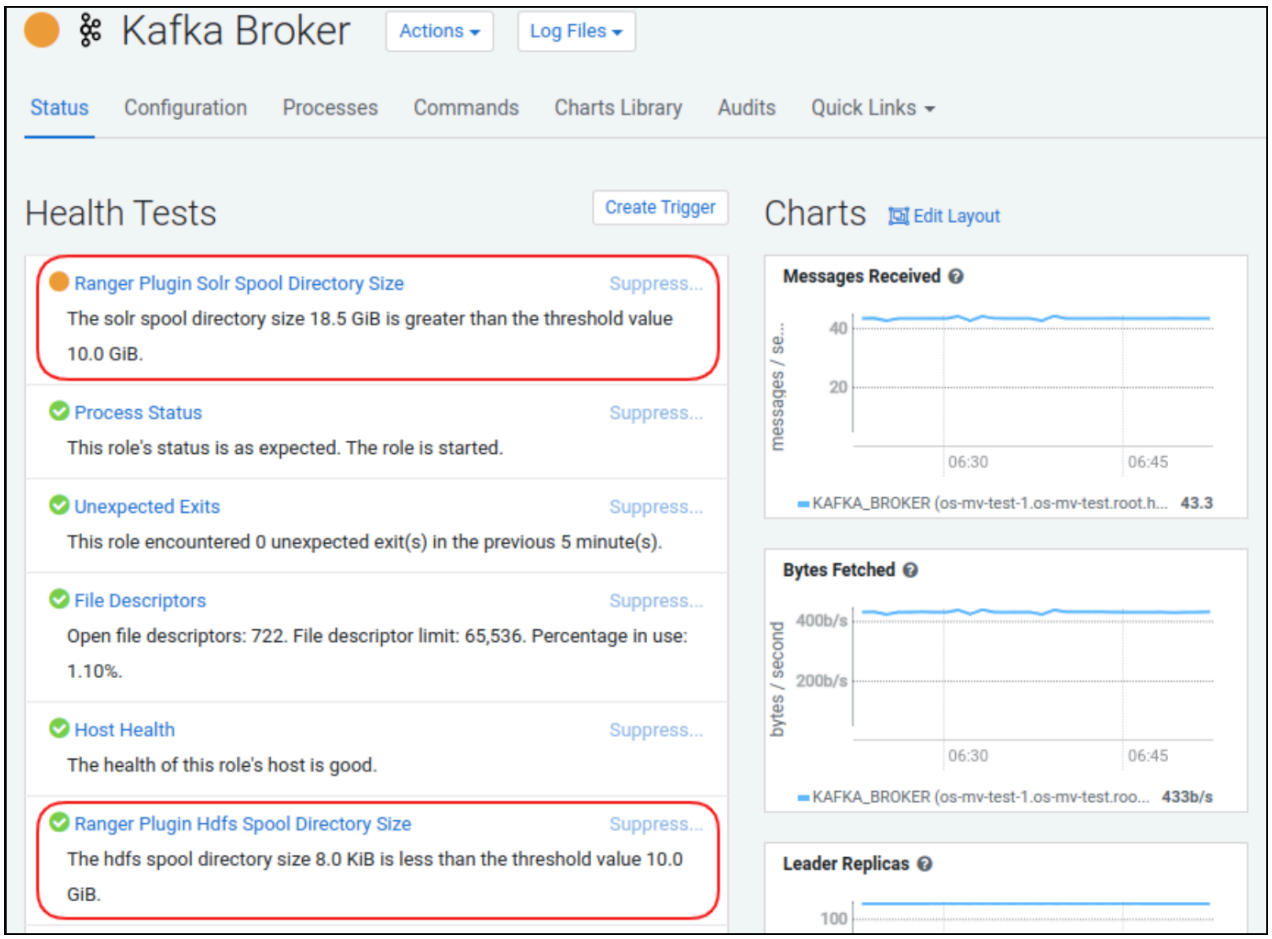
4. In Ranger Plugin Spool Directory Usage Thresholds for <service-name>, type values and select units.

   By default the threshold is set to 1 GB. If the disk usage of the spool directory exceeds this threshold, an alert will be shown.

Results

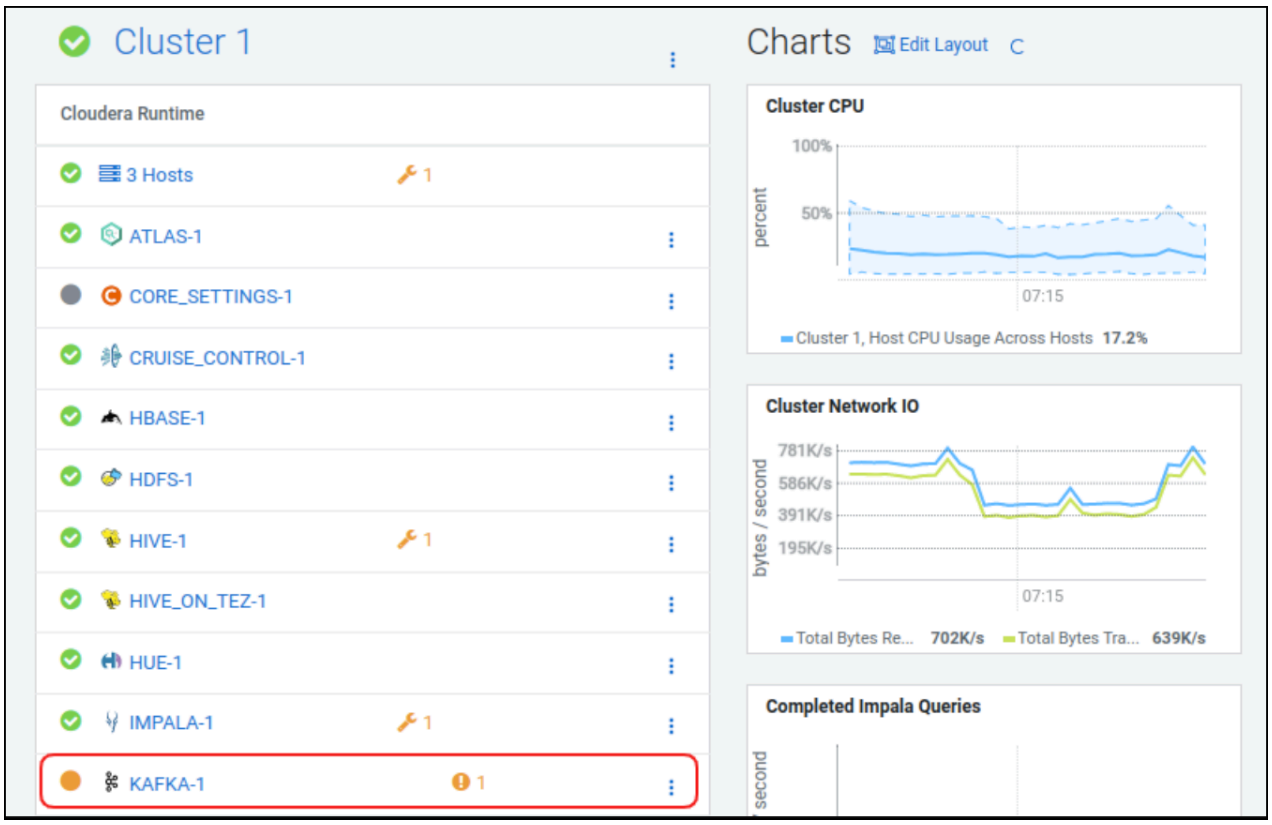
These spool alert details appear on the Ranger plugin service Role status page in the Cloudera Manager UI, as shown for Kafka Broker role in the following example:

Figure 20: Audit spool details for the Kafka Broker role



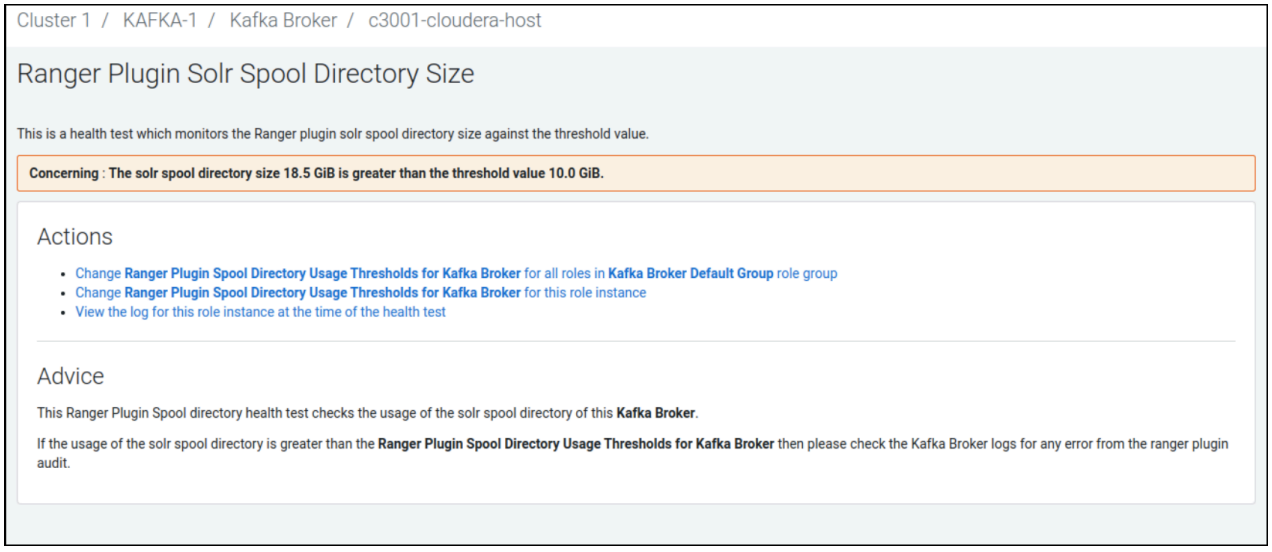
Also, an alert appears on the Cloudera Manager Home page next to the service name, as shown for Kafka service in the following example.

Figure 21: Audit file spool notification for Kafka service plugin on Cloudera Manager home page.



Also, Cloudera Manager shows Cluster Heath Actions and Advice for the role, as shown in the following example:

Figure 22: Health Test For Ranger Plugin Spool Alert



What to do next

Optionally, you can create a graph for each spool alert metric.

Related Information

Charting spool alert metrics

Charting spool alert metrics

Cloudera Manager supports chart visualization of spool directory disk usage metrics.

Single-level metrics for collecting the disk usage of Solr and Hdfs Ranger plugin spool directories are registered in Cloudera Manager, using the following disk usage metric names:

Solr

ranger_plugin_solr_spool_directory_size

HDFS

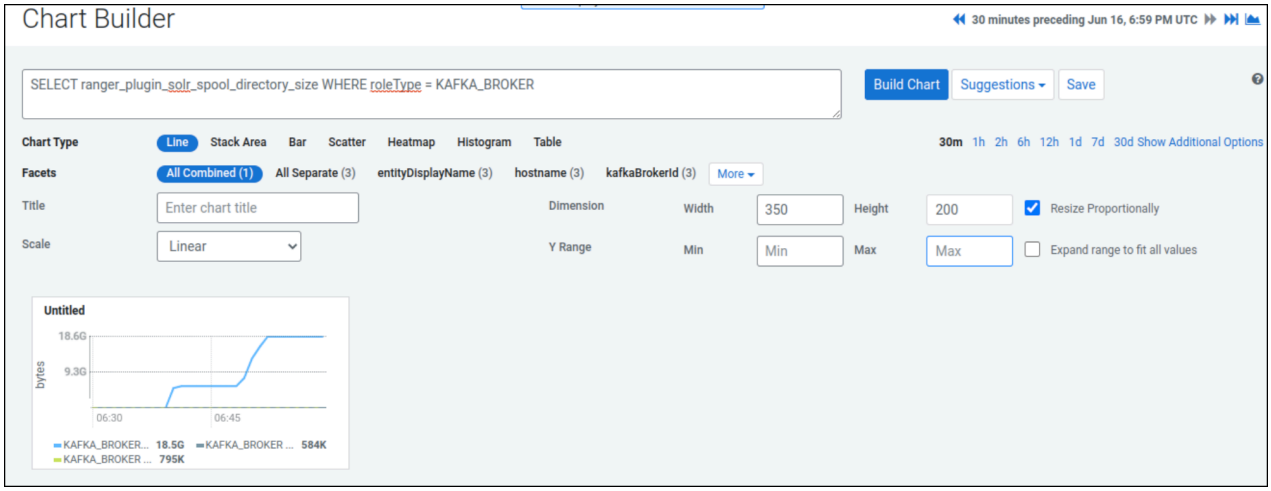
ranger_plugin_hdfs_spool_directory_size

You can chart disk usage of Solr and HDFS spool directory size as a time-series, using Cloudera Manager Chart Builder .

For example, to chart the disk usage for the Solr spool directory defined for Kafka Broker, use the folllowing query syntax:

SELECT ranger_plugin_solr_spool_directory_size WHERE roleType = KAFKA_BROKER, as shown in

Figure 23: Charting the disk usage of the Ranger Solr plugin spool directory for Kafka Broker



For more specific information about charting time-series data, see:

Related Information

Building a Chart with Time-Series Data

Excluding audits for specific users, groups, and roles

You can exclude audit records for specific users, groups, and roles from each service from appearing in the Ranger UI.

About this task

Ranger default log functionality creates audit log records for access and authorization requests, specifically around service accounts such as hbase, atlas and solr. Writing so much data to solr can limit the availability of Solr for further usage. This topic describes how to exclude audit records for specific users, groups, and roles from each service from appearing in the Ranger UI. Excluding specific users, groups or roles is also known as creating a blacklist for Ranger audits.

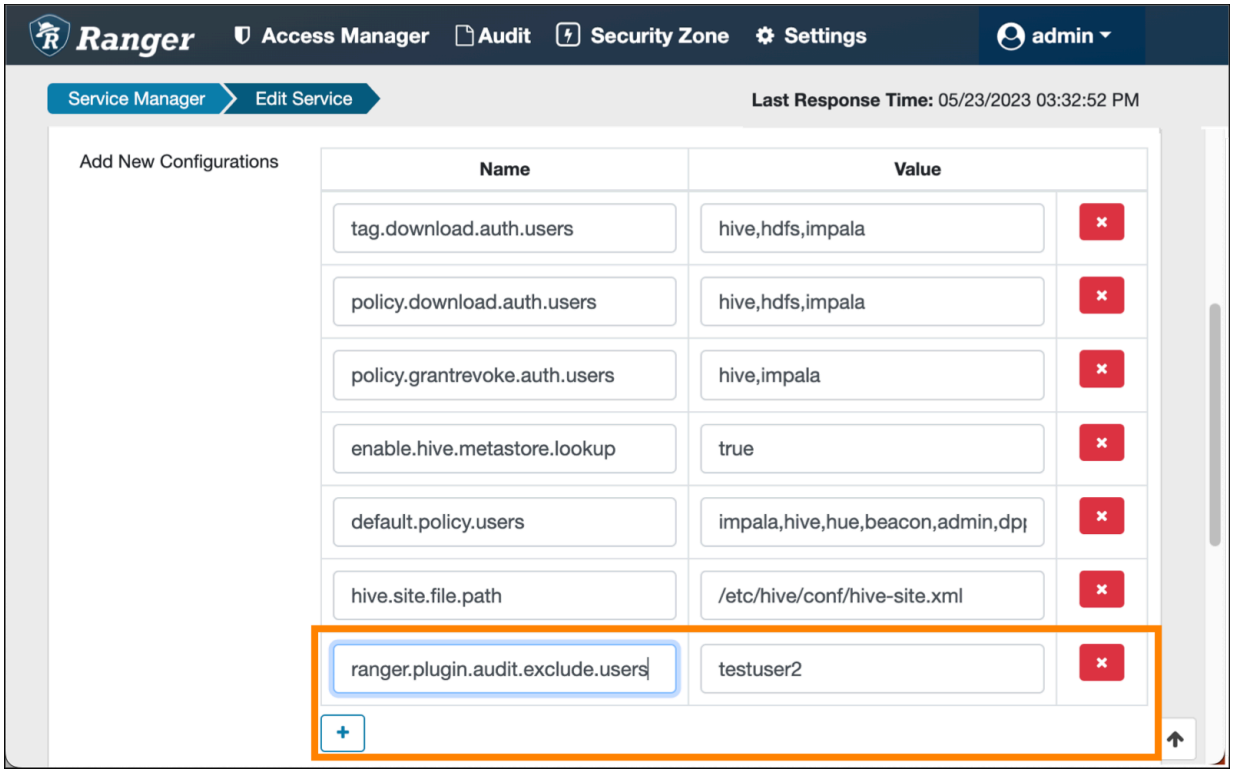
Procedure

1. In the Ranger Admin Web UI Service Manager , click Add New Service or Edit (existing service).
2. On Create/Edit Service, scroll down to Config Properties Add New Configurations .
3. Remove all audit filters from the existing service.
4. Click +, then type one of the following property names:
   - ranger.plugin.audit.exclude.users
   - ranger.plugin.audit.exclude.groups
   - ranger.plugin.audit.exclude.roles

   followed by one or more values.

   Note: You can include multiple values for each exclude property using a comma-separated list.

Figure 24: Adding an exclude users property to the HadoopSQL service



After adding the above configuration; if testuser2 user performs any actions for HadoopSQL service, Audit Access logs will not appear in the Ranger UI, but are still sent to Solr.

Similarly, you can exclude (or blacklist) users belonging to a particular group or role by adding a user-specific or role-specific configuration.

Changing Ranger audit storage location and migrating data

How to change the location of existing and future Ranger audit data collected by Solr from HDFS to a local file system or from a local file system to HDFS.

Before you begin

- Stop Atlas from Cloudera Manager.
- If using Kerberos, set the SOLR_PROCESS_DIR environment variable.

```
# export SOLR_PROCESS_DIR=$(ls -1dtr /var/run/cloudera-scm-agent/process/
*SOLR_SERVER | tail -1)
```

About this task

Starting with Cloudera Runtime version 7.1.4 / 7.2.2, the storage location for ranger audit data collected by Solr changed to local file system from HDFS, as was true for previous versions. The default storage location Ranger audit data storage location for Cloudera Runtine-7.1.4+ and Cloudera Runtine-7.2.2+ installations is local file system. After upgrading from an earlier Cloudera platform version, follow these steps to backup and migrate your Ranger audit data and change the location where Solr stores your future Ranger audit records.

- The default value of the index storage in the local file system is /var/lib/solr-infra. You can configure this, using Cloudera Manager Solr Configuration "Solr Data Directory" .
- The default value of the index storage in HDFS is /solr-infra. You can configure this, using Cloudera Manager Solr Configuration "HDFS Data Directory" .

Procedure

1. Create HDFS Directory to store the collection backups.

   As an HDFS super user, run the following commands to create the backup directory:

   ```
   # hdfs dfs -mkdir /solr-backups
   # hdfs dfs -chown solr:solr /solr-backups
   ```
2. Obtain valid kerberos ticket for Solr user.

   ```
   # kinit -kt solr.keytab solr/$(hostname -f)
   ```
3. Download the configs for the collection.

   ```
   # solrctl instancedir --get ranger_audits /tmp/ranger_audits
   # solrctl instancedir --get atlas_configs /tmp/atlas_configs
   ```
4. Modify the solrconfig.xml for each of the configs for which data needs to be stored in HDFS.

   In /tmp/<config_name>/conf created during Step 3., edit properties in the solrconfig.xml file as follows:

   - When migrating your data storage location from a local file system to HDFS, replace these two lines:

     ```
     <directoryFactory name="DirectoryFactory"
      class="${solr.directoryFactory:solr.NRTCachingDirectoryFactory}">
     <lockType>${solr.lock.type:native}</lockType>
     ```

     with

     ```
     <directoryFactory name="DirectoryFactory"
      class="${solr.directoryFactory:org.apache.solr.core.HdfsDirectoryFactory}">
     <lockType>${solr.lock.type:hdfs}</lockType>
     ```
   - When migrating your data storage location from HDFS to a local file system, replace these two lines:

     ```
     <directoryFactory name="DirectoryFactory"
      class="${solr.directoryFactory:org.apache.solr.core.HdfsDirectoryFactory}">
     <lockType>${solr.lock.type:hdfs}</lockType>
     ```

     with

     ```
     <directoryFactory name="DirectoryFactory"
      class="${solr.directoryFactory:solr.NRTCachingDirectoryFactory}">
     ```

```
<lockType>${solr.lock.type:native}</lockType>
```

5. Backup the Solr collections.

   - When migrating your data storage location from a local file system to HDFS, run:

     ```
     # curl -k --negotiate -u : "https://$(hostname
      -f):8995/solr/admin/collections?action=BACKUP&name=vertex_backup&col
     lection=vertex_index&
     location=hdfs://<Namenode_Hostname>:8020/solr-backups"
     ```

     In the preceding command, the important points are name, collection, and location:

     name

     specifies the name of the backup. It should be unique per collection

     collection

     specifies the collection name for which the backup will be performed

     location

     specifies the HDFS path, where the backup will be stored

     Repeat the curl command for different collections, modifying the parameters as necessary for each collection.

     The expected output would be -

     ```
     "responseHeader":{
      "status":0,
      "QTime":10567},
     "success":{
      "Solr_Server_Hostname:8995_solr":{
        "responseHeader":{
        "status":0,
        "QTime":8959}}}
     ```

   - When migrating your data storage location from HDFS to a local file system:

     Refer to Back up a Solr collection for specific steps, and make the following adjustments:

     - If TLS is enabled for the Solr service, specify the trust store and password by using the ZKCLI_JVM_FLAGS environment variable before you begin the procedure.

       ```
       # export ZKCLI_JVM_FLAGS="-Djavax.net.ssl.trustStore=/path/to/
       truststore.jks -Djavax.net.ssl.trustStorePassword="
       ```

     - Create Snapshot

       ```
       # solrctl --jaas $SOLR_PROCESS_DIR/jaas.conf collection --create-
       snapshot <snapshot_name> -c <collection_name>
       ```

     - or use the Solr API to take the backup:

       ```
       curl -i -k --negotiate -u : "https://(hostname -f):8995/solr/admin/
       collections?
       action=BACKUP&name=ranger_audits_bkp&collection=ranger_audits&location=/
       path/to/solr-backups"
       ```

     - Export Snapshot

       ```
       # solrctl --jaas $SOLR_PROCESS_DIR/jaas.conf collection
        --export-snapshot <snapshot_name> -c <collection_name> -d
        <destination_directory>
       ```

       Note: The <destination_directory> is a HDFS path. The ownership of this directory should be solr:solr.

6. Update the modified configs in Zookeeper.

```
# solrctl --jaas $SOLR_PROCESS_DIR/jaas.conf instancedir --update
 atlas_configs /tmp/atlas_configs
# solrctl --jaas $SOLR_PROCESS_DIR/jaas.conf instancedir --update
 ranger_audits /tmp/ranger_audits
```

7. Delete the collections from the original location.

   All instances of Solr service should be up, running, and healthy before deleting the collections. Use Cloudera Manager to check for any alerts or warnings for any of the instances. If alerts or warnings exist, fix those before deleting the collection.

```
# solrctl collection --delete edge_index
# solrctl collection --delete vertex_index
# solrctl collection --delete fulltext_index
# solrctl collection --delete ranger_audits
```

8. Verify that the collections are deleted from the original location.

```
# solrctl collection --list
```

   This will give an empty result.

9. Verify that no leftover directories for any of the collections have been deleted.

   - When migrating your data storage location from a local file system to HDFS:

```
# cd /var/lib/solr-infra
```

     Get the value of "Solr Data Directory, using Cloudera Manager Solr Configuration .

```
# ls -ltr
```

   - When migrating your data storage location from HDFS to a local file system, replace these two lines:

```
# hdfs dfs -ls /solr/<collection_name>
```

     Note: If any directory name which starts with the collection name deleted in Step 7. exists, delete/move the directory to another path.

10. Restore the collection from backup to the new location.

    Refer to Restore a Solr collection, for more specific steps.

```
# curl -k --negotiate -u : "https://$(hostname
 -f):8995/solr/admin/collections?
action=RESTORE&name=<Name_of_backup>&location=hdfs:/
<<Namenode_Hostname>:8020/solr-backups&collection=<Collection_Name>"
```

```
# solrctl collection --restore ranger_audits
 -l hdfs://<Namenode_Hostname>:8020/solr-backups
 -b ranger_backup -i ranger1
```

    The request id must be unique for each restore operation, as well as for each retry.

    To check the status of restore operation:

```
# solrctl collection --request-status <requestId>
```

    Note: If the Atlas Collections (vertex_index, fulltext_index and edge_index) restore operations fail, restart the solr service and rerun the restore command. Now, the restart operations should complete successfully.

11. Verify the Atlas & Ranger functionality.

Verify that both Atlas and Ranger audits functions properly, and that you can see the latest audits in Ranger Web UI and latest lineage in Atlas.

- To verify Atlas audits, create a test table in Hive, and then query the collections to see if you are able to view the data.
- You can also query the collections every 20-30 seconds (depending on how other services utilize Atlas/Ranger), and verify if the "numDocs" value increases at every query.

```
# curl -k --negotiate -u : "https://$(hostname -f):8995/solr/edge_index/
select?q=*%3A*&wt=json&ident=true&rows=0"
# curl -k --negotiate -u : "https://$(hostname -f):8995/solr/vertex_index/
select?q=*%3A*&wt=json&ident=true&rows=0"
# curl -k --negotiate -u : "https://$(hostname -f):8995/solr/
fulltext_index/select?q=*%3A*&wt=json&ident=true&rows=0"
# curl -k --negotiate -u : "https://$(hostname -f):8995/solr/
ranger_audits/select?q=*%3A*&wt=json&ident=true&rows=0"
```

Configuring Ranger audits to show actual client IP address

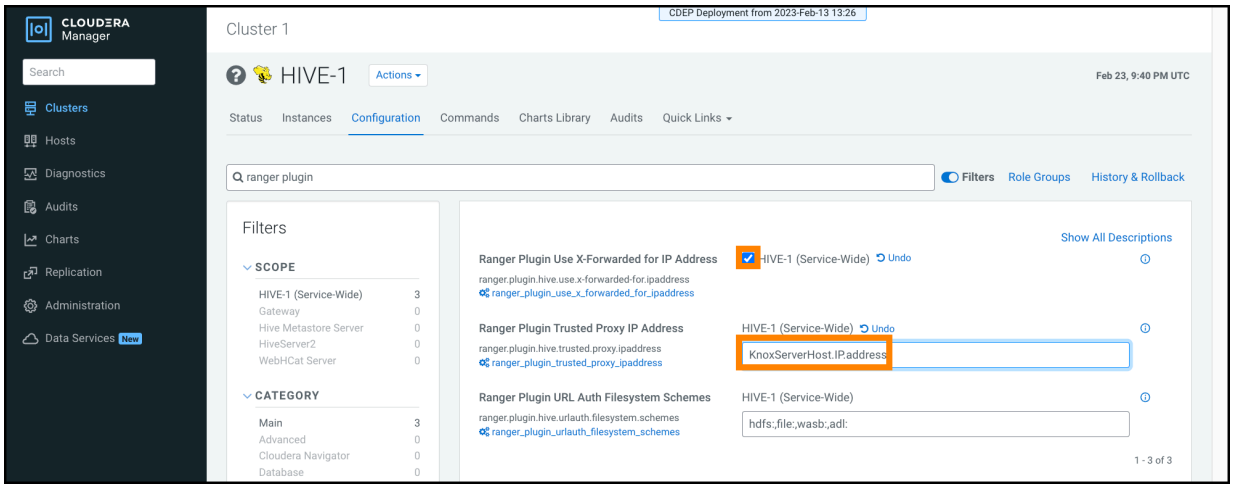
How to forward the actual client IP address to audit logs generated from a Ranger plugin.

About this task

Ranger audit logs record the IP address through which Ranger policies grant/authorize access. When Ranger is set up behind a Knox proxy server, the proxy server IP address appears in the audit logs generated for each Ranger plugin. You can configure each plugin to forward the actual client IP address on which that service runs, so that the audit logs for that service more specifically reflect access/authorization activity. You must configure each plugin individually. This topic uses the Hive (Hadoop SQL) service as an example.

Procedure

1. From Cloudera Manager choose <service_name> Configuration .
2. In <service_name> Configuration Search , type ranger-plugin, then press Return.
3. In Ranger Plugin Use X-Forwarded for IP Address, check the box.
4. In Ranger Plugin Trusted Proxy IP Address, type the IP address of the Knox proxy server host.



Results

Hive audit logs will now show the IP address of the host on which Hive service runs.