

Cloudera Runtime 7.1.6

Ranger Auditing

Date published: 2019-11-01

Date modified:

CLOUdera

<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
View audit details.....	4
Create a read-only Admin user (Auditor).....	7
Ranger Audit Filters (Technical Preview).....	8
Changing Ranger audit storage location and migrating data.....	13

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.

Managing Auditing with Ranger

To explore options for auditing policies in Ranger, click Audit in the top menu.

Policy ID	Policy Version	Event Time	Application	User	Service Name / Type	Resource Name / Type	Access Type	Result	Access Enforcer	Agent Host Name	Client IP	C
3	1	07/21/2019 12:21:35 PM	hbaseMaster	hbase	cm_hbase	hbase	--	balance	Allowed	ranger-acl	dhoyle-7-1-1.vpc.cloudera.com	C
3	1	07/21/2019 12:16:30 PM	hbaseMaster	hbase	cm_hbase	hbase	--	balance	Allowed	ranger-acl	dhoyle-7-1-1.vpc.cloudera.com	C
3	1	07/21/2019 12:11:30 PM	hbaseMaster	hbase	cm_hbase	hbase	--	balance	Allowed	ranger-acl	dhoyle-7-1-1.vpc.cloudera.com	C
3	1	07/21/2019 12:06:30 PM	hbaseMaster	hbase	cm_hbase	hbase	--	balance	Allowed	ranger-acl	dhoyle-7-1-1.vpc.cloudera.com	C

There are six tabs on the Audit page:

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

View audit details

How to view operation details in Ranger audits.

Procedure

To view details for a particular operation, click any tab, then Policy ID, Operation name, or Session ID.

Audit > Admin: Update

The screenshot displays the 'Access Manager' application. At the top, there are navigation tabs: 'Access Manager', 'Audit', 'Security Zone', and 'Settings'. Below these is a header bar with 'ACCESS', 'Admin', 'Login Sessions', 'Plugins', 'Plugin Status', and 'User Sync'. A search bar is present with the text 'Search for your access logs...'. The main content area shows a table of operations with columns: Operation, Audit Type, User, Date (Eastern Daylight Time), Actions, and Session Id. The table lists various operations like 'Service updated tag_service2', 'Group created temp_employees', etc. A blue arrow points to the 'Update' button in the 'Actions' column for the first row. A modal window titled 'Operation : update' is open, showing details for the operation 'tag_service2' and a table of service details.

Operation : update

Name : tag_service2
Date : 07/21/2019 01:09:34 PM Eastern Daylight Time
Updated By : admin

Service Details :

Fields	Old Value	New Value
Service Description	--	--
Service Name	tag_tag	tag_service2

OK

Ranger

Access Manager

Audit

Security Zone

Settings

admin

Access

Admin

Login Sessions

Plugins

Plugin Status

User Sync

Search for your access logs...

Entries : 1 to 25 of 70

Last Updated Time : 07/21/2019 01:09:40 PM

Operation	Audit Type	User	Date (Eastern Daylight Time)	Actions	Session Id
Service updated tag_service2	Ranger Service	admin	07/21/2019 01:09:34 PM	<button>Update</button>	40
Group created temp_employees	Ranger Group	admin	07/20/2019 02:15:05 PM	<button>Create</button>	38
Group created audit	Ranger Group	admin	07/18/2019 04:18:42 PM	<button>Create</button>	35
Exported policies	Ranger Policy	admin	07/17/2019 03:06:22 PM	<button>Export Json</button>	32
Service updated tag_service1	Ranger Service		07/15/2019 04:11:25 PM	<button>Update</button>	
Policy created EXPIRES_ON	Ranger Policy		07/15/2019 04:11:25 PM	<button>Create</button>	
Service created tag_tag	Ranger Service		07/15/2019 04:11:25 PM	<button>Create</button>	
Policy created EXPIRES_ON	Ranger Policy	admin	07/15/2019 04:11:24 PM	<button>Create</button>	29
Service created tag_service1	Ranger Service	admin	07/15/2019 04:11:24 PM	<button>Create</button>	29
Security Zone created security-zone2	Ranger Security Zone	admin	07/14/2019 05:24:36 PM	<button>Create</button>	27
Policy created all - database, udf	Ranger Policy	admin	07/14/2019 05:24:36 PM	<button>Create</button>	27
Policy created all - database, table, column	Ranger Policy	admin	07/14/2019 05:24:36 PM	<button>Create</button>	27
Policy created all - url	Ranger Policy	admin	07/14/2019 05:24:36 PM	<button>Create</button>	27
Policy created all - label	Ranger Policy	admin	07/14/2019 05:24:36 PM	<button>Create</button>	27
				<button>Create</button>	27
				<button>Create</button>	27
				<button>Create</button>	27
				<button>Create</button>	27
				<button>Create</button>	27
				<button>Create</button>	27
				<button>Create</button>	27
				<button>Create</button>	27
				<button>Update</button>	
				<button>Create</button>	
				<button>Create</button>	

Operation : create

Name : security-zone2

Date : 07/14/2019 05:24:36 PM Eastern Daylight Time

Created By : admin

Zone Details :

Fields :	New Value
Zone Description	--
Zone Audit User Groups	--
Zone Audit Users	auditor1
Zone Admin User Groups	--
Zone Admin Users	admin
Zone Tag Services	cm_tag
Zone Name	security-zone2

Zone Service Details :

Service Name	Zone Service Resources

OK

Audit > User Sync: Sync details

The screenshot shows the Ranger web interface with the 'User Sync' tab selected. A search bar at the top indicates 'START DATE: 07/21/2019'. Below the search bar, a status bar shows 'Entries: 1 to 25 of 803' and 'Last Updated Time: 07/21/2019 01:23:45 PM'. The main table displays sync events for 'rangerusersync' from a 'Unix' source. The table columns are: User Name, Sync Source, Number Of New (Users, Groups), Number Of Modified (Users, Groups), Event Time, and Sync Details. A blue box highlights the 'Sync Details' icon in the second row, which is linked by a blue arrow to a modal window titled 'Sync Details'.

User Name	Sync Source	Number Of New		Number Of Modified		Event Time	Sync Details
		Users	Groups	Users	Groups		
rangerusersync	Unix	0	0	0	0	07/21/2019 01:22:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:21:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:20:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:19:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:18:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:17:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:16:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:15:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:14:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:13:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:12:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:11:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:10:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:09:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:08:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:07:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:06:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:05:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:04:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:03:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:02:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:01:48 PM	[Icon]
rangerusersync	Unix	0	0	0	0	07/21/2019 01:00:47 PM	[Icon]

Sync Details

Name	Value
Unix	nss
File Name	/etc/passwd
Sync time	07/21/2019 10:21:48 AM
Last modified time	12/31/1969 04:00:00 PM
Minimum user id	500
Minimum group id	0
Total number of users synced	35
Total number of groups synced	39

OK

Create 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. Select Settings > Users/Groups/Roles.
2. Click Add New User.

3. Complete the **User Detail** section, selecting Auditor as the role:

The screenshot shows the Ranger web interface for creating a user. The top navigation bar includes 'Ranger', 'Access Manager', 'Audit', 'Security Zone', and 'Settings'. The user 'admin' is logged in. The breadcrumb trail is 'Users/Groups/Roles > User Create'. The 'User Detail' section contains the following fields:

- User Name *: auditor1
- New Password *: [masked]
- Password Confirm *: [masked]
- First Name *: Audrey
- Last Name: [empty]
- Email Address: [empty]
- Select Role *: Auditor (selected from a dropdown)
- Group: audit (with a '+' button to add more groups)

At the bottom of the form are 'Save' and 'Cancel' buttons.

4. Click Save.

Ranger Audit Filters (Technical Preview)

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.



Important:

Ranger Audit Filters is available for technical preview and considered to be under development. Do not use this feature in your production systems. If you have questions regarding Ranger Audit Filters, contact support by logging a case on the [Cloudera Support Portal](#).

Ranger audit filters format

Audit filters consist of a subset of Ranger policy attributes, along with access results attributes. These attributes define the audit filter rules.

```
{ 'resources':
  {
    'database': { 'values': ['data', 'data2'] },
    'table': { 'values': ['*'] },
    'column': { 'values': ['*'] }
  },
  'accessTypes': ['select', 'insert'],
  'users': ['user1', 'user2'],
  'groups': ['no_audit_group'],
  'roles': ['role1', 'role2'],
  'actions': ['METADATA OPERATION', 'SHOW_ROLES'],
  'accessResult': 'DENIED' or 'ALLOWED' or 'NOT_DETERMINED' or empty
```



```
'isAudited' : false
}
```

Ranger audit filters configuration

Ranger audit filters are configured by adding the audit filter JSON string as the value of the `ranger.plugin.audit.filters` configuration property in each Ranger service.

To configure resource-based audit filters, click the Edit icon for the applicable service in the Ranger Admin web UI. On the Edit Service page, click the Add (+) icon under Add New Configurations, then enter `ranger.plugin.audit.filters` in the Name box, and enter the audit filter JSON string in the Value box.

The screenshot shows the 'Edit Service' page in the Ranger Admin web UI. The page has a dark blue header with the 'Ranger' logo and navigation tabs: 'Access Manager', 'Audit', 'Security Zone', and 'Settings'. The user 'admin' is logged in. The 'Service Manager' tab is active, and the 'Edit Service' sub-tab is selected.

The configuration fields include:

- Authorization Enabled: Yes
- Authentication Type *: Kerberos
- hadoop.security.auth_to_local: (empty)
- dfs.datanode.kerberos.principal: (empty)
- dfs.namenode.kerberos.principal: (empty)
- dfs.secondary.namenode.kerberos.principal: (empty)
- RPC Protection Type: Authentication
- Common Name for Certificate: (empty)

Under 'Add New Configurations', there is a table:

Name	Value	
tag.download.auth.users	hdfs	✕
policy.download.auth.use	hdfs	✕
ranger.plugin.audit.filters	[{'accessResult': 'DENIED'}]	✕

The third row of the table is highlighted with a red box. Below the table is a '+' icon and a 'Test Connection' button. At the bottom of the page are 'Save', 'Cancel', and 'Delete' buttons.

To configure tag-based audit filters, click the Edit icon for the applicable tag-based service in the Ranger Admin web UI. On the Edit Service page, click the Add (+) icon under Add New Configurations, then enter `ranger.plugin.audit.filters` in the Name box, and enter the audit filter JSON string in the Value box.

Ranger Access Manager Audit Security Zone Settings admin

Service Manager Edit Service

Edit Service

Service Details :

Service Name *

Display Name

Description

Active Status ☒ Enabled ☐ Disabled

Config Properties :

Add New Configurations

Name	Value
ranger.plugin.audit.filters	[{"resources":{"tag":{"value":...

+

Test Connection

Save Cancel Delete

HDFS service audit filters example:

```
"serviceConfig": {
  "ranger.plugin.audit.filters":
    "[
      { '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 }
    ]"
```

Hive service audit filters example:

```
"serviceConfig": {
  "ranger.plugin.audit.filters":
    "[ { 'accessResult': 'DENIED', 'isAudited': true },
      { 'resources': { 'database': { 'values': [ 'temp' ] }, 'table': { 'values': [ 'tempdata' ] },
        'column': { 'values': [ '*' ] }, 'isAudited': false },
      { 'resources': { 'database': { 'values': [ 'sys' ] }, 'table': { 'values': [ 'dump' ] },
        'users': [ 'user2' ], 'isAudited': false },
      { 'actions': [ 'METADATA OPERATION' ], 'isAudited': false },
```

```
{ 'users': ['superuser1'], 'groups': ['supergroup1'], 'isAudited': false} ]"
```

Tag-based service audit filters example:

```
"serviceConfig": {
  "ranger.plugin.audit.filters": "[ { 'resources': { 'tag': { 'values': [ 'NO_AUDIT' ] } }, 'isAudited': false },
  { 'resources': { 'tag': { 'values': [ 'SYS_DATA' ] } }, 'users': [ 'user1' ], 'isAudited': false },
  { 'resources': { 'tag': { 'values': [ 'HIPPA' ] } }, 'users': [ 'user1' ], 'isAudited': true } ]"
```

Audit filters details

- As shown in the examples above, audit filters are defined as a JSON list.
- When adding the JSON string to the Value box in each service configuration, you should only enter the text within the top-level brackets, as `ranger.plugin.audit.filters` is already set in the Name box in the UI. For the HDFS example above, you would enter:

```
[ { '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}
```

Default audit filters

You can create default audit filters for each Ranger service, which can then be modified as needed by Admin users.

HDFS service:

```
"ranger.plugin.audit.filters":
"[ { 'accessResult': 'DENIED', 'isAudited': true},
{ 'resources':{'path':{'values':['*/hive-staging','*/staging', '*/sparkStaging',
'*/_impala_insert_staging', '/user/history/done_intermediate','/user/spark/s
park2ApplicationHistory',
'/user/spark/ApplicationHistory', '/user/hue/.cloudera_manager_hive_metast
ore_canary',
'/user/hue/.Trash/Current/user/hue/.cloudera_manager_hive_metastore_canary',
'/tmp', '/user/oozie/share/lib', '/hbase/archive', '/hbase/oldWALs', '/hba
se/MasterProcWALs']},
'isRecursive':true}}, 'isAudited': false}, { 'actions':['delete','rename*'],
'isAudited':true},
{ 'users':['cloudera-scm','dr.who','hbase','hive','impala','mapred','solr','s
park', 'hue'],
'isAudited':false}, { 'users':['hdfs'], 'actions': ['listStatus', 'getFileinf
o', 'listCachePools',
'listCacheDirectives'], 'isAudited': false}, { 'actions': ['getFileinfo'],
'isAudited':true} ]"
```

HBase service:

```
"ranger.plugin.audit.filters": "[ { 'resources':{'table':{'values':['*-ROOT-*',
'*.META.*', '*_acl_*',
'hbase:meta', 'hbase:acl']}}, 'isAudited': false }, { 'resources':{'table':{'
values':['atlas_janus',
'ATLAS_ENTITY_AUDIT_EVENTS']}, 'column-family':{'values':['*']}, 'column':{'va
lues':['*']}},
'isAudited':false}, { 'users':['hbase'], 'actions':['balance'], 'isAudited':
false} ]"
```

Hive service:

```
"ranger.plugin.audit.filters": "[ { 'accessResult': 'DENIED', 'isAudited': tr
ue}, { 'accessTypes':['_any'],
'isAudited':false}, { 'actions':['METADATA OPERATION'], 'isAudited': false},
{ 'users':['hive','hue'],
'actions':['SHOW_ROLES'], 'isAudited':false} ]"
```

Kafka service:

```
"ranger.plugin.audit.filters": "[{'resources': {'consumergroup': {'values': ['atlas', 'ranger_entities_consumer']}}, 'isAudited': false}, {'resources': {'topic': {'values': ['ATLAS_*']}}, 'isAudited': false}, {'users': ['rangertagsync', 'kafka', 'atlas'], 'isAudited': false}]"
```

ADLS service:

```
"ranger.plugin.audit.filters": "[ {'accessResult': 'DENIED', 'isAudited': true}, {'users': ['hive', 'hdfs', 'zeppelin', 'hbase', 'solr', 'kafka'], 'isAudited': false} ]"
```

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 local or from local 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 -ldtr /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 Runtime-7.1.4+ and Cloudera Runtime-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 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 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. 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
```

6. Backup the Solr collections.

- When migrating your data storage location from 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 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.

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 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 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"
```