
ranger-cli Documentation

Release 0.0.1.dev0

Deric Degagne

Sep 06, 2021

OVERVIEW

1	Installation	3
2	Ranger connection profiles	5
3	Getting Started	7
3.1	Apache Ranger Overview	7
3.2	Configure CLI	10
3.3	Policy CLI	12
3.4	Service CLI	15
3.5	ServiceDef CLI	18
3.6	Plugins CLI	20
3.7	Plugin Templates	22
3.8	LICENSE	24
4	Links	25

The RangerCLI (command-line interface) provides a straightforward interface for managing resources within the Apache Ranger framework. The open-source project is hosted on [GitHub](#) and is built on top of the Apache Ranger public REST APIs.

INSTALLATION

Before installing the RangerCLI, the following requirements must be met.

- Python 3.6

Furthermore, the following Python libraries are also required and will be installed with the Ranger CLI.

- click
- six
- confuse
- PyYAML
- requests
- urllib3
- xmldict
- simplejson

Using the appropriate version of `pip` for your Python installation, invoke the following command to install the Ranger-CLI:

Listing 1: Bash

```
$ pip install ranger-cli
```

You can upgrade the RangerCLI with:

Listing 2: Bash

```
$ pip install ranger-cli --upgrade
```

Once you've installed or updated the RangerCLI, you can verify the installation with:

Listing 3: Bash

```
$ ranger-cli --version
```


RANGER CONNECTION PROFILES

The RangerCLI supports multiple Apache Ranger connection profiles. These profiles are stored locally `~/.config/ranger/config.yaml` and are used to make API calls to multiple Apache Ranger environments.

To create a new profile, invoke the following command:

Listing 1: Bash

```
$ ranger-cli configure --policy <profile-name>
```

Important: If the profile name already exists in the configuration file, it will be overwritten with the new properties.

Example configuration file:

Listing 2: YAML

```
1 default:
2   endpoint: https://ranger1.example.com:6182
3   authentication:
4     - jsmith
5     - supersecurepassword
6   verification: /home/jsmith/.config/ranger/ca-bundle.crt
```

To use the connection profile

Listing 3: Bash

```
$ ranger-cli policy --policy <profile-name>
```

If `--profile` option is not invoked, the default profile is used. If the default profile is not found, the first profile in the configuration file is used.

GETTING STARTED

To start using the RangerCLI, use the `--help` option to view the latest available group commands.

Listing 1: Bash

```
$ ranger-cli --help
```

To view the latest available sub-commands, the `--help` option can be used on the group command:

Listing 2: Bash

```
$ ranger-cli <group> --help
```

3.1 Apache Ranger Overview

The Apache Ranger framework, provides a centralized platform which administers and regulates security policies across Hadoop components. Apache Ranger also provides a framework for collecting access audit history and reporting data.

3.1.1 Hadoop components

Hadoop components supported by Apache Ranger, include, but not limited to the following:

- Apache Hadoop HDFS
- Apache Hive
- Apache HBase
- Apache Storm
- Apache Knox
- Apache Solr
- Apache Kafka
- Apache NiFi
- Apache Atlas
- Apache Sqoop
- Apache Spark
- Apache Kudu

- [Apache Kylin](#)
- [Apache Hadoop Ozone](#)
- [Apache Hadoop Yarn](#)

3.1.2 Policies overview

Apache Ranger supports 2 types of policies:

1. resource-based
2. tag-based

Warning: The RangerCLI currently does not support tag-based policies.

Resource-based policies

Resource-based policies allows users to configure access policies on Hadoop services.

Tag-based policies

Tag-based policies allows users to configure access policies across multiple Hadoop components without creating separate services and policies in each component.

3.1.3 Row-filter and masking policies in Hive

Apache Ranger supports row-level filters and column masking to set access policies for rows in Hive tables.

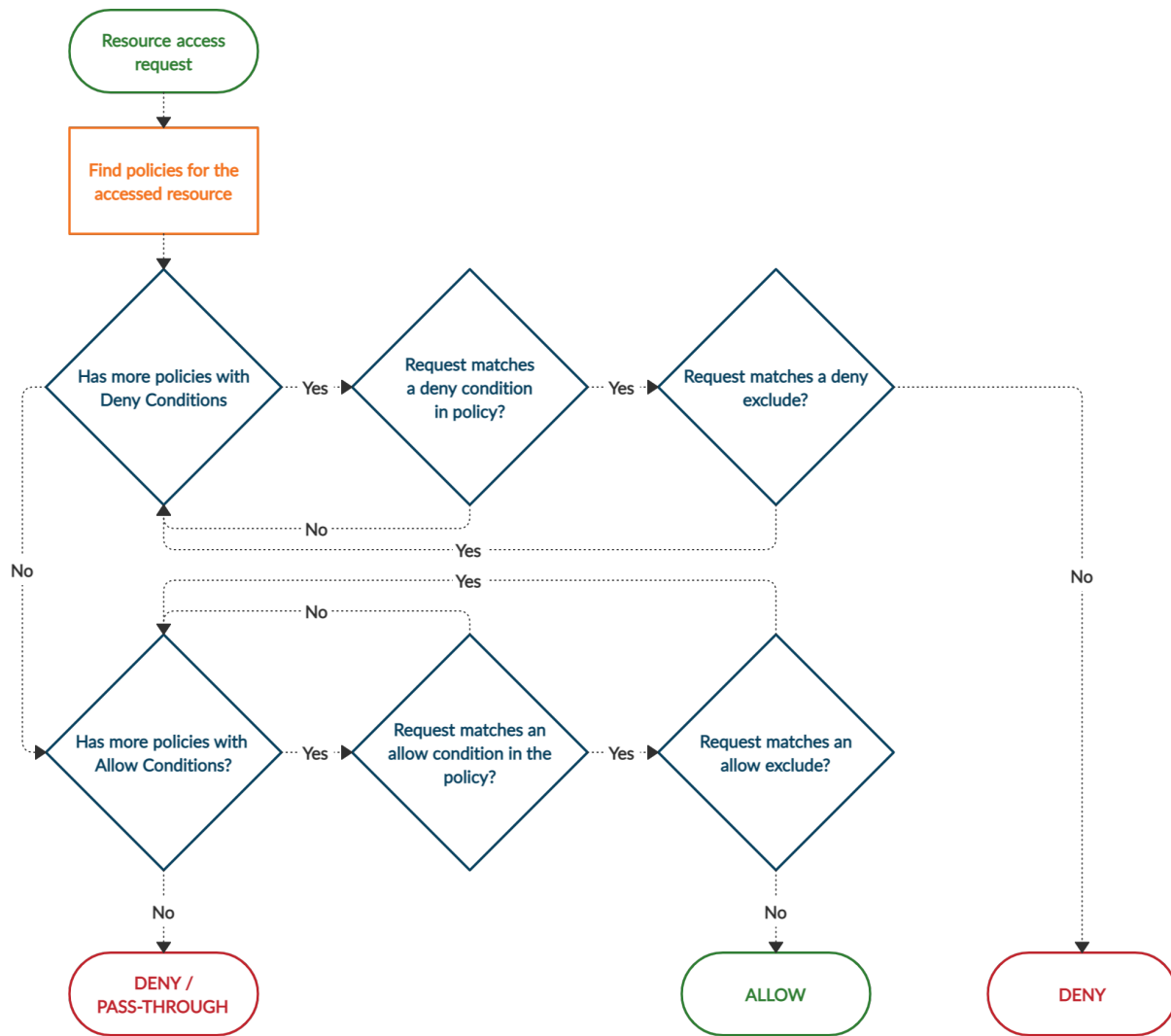
Row-level filtering in Hive

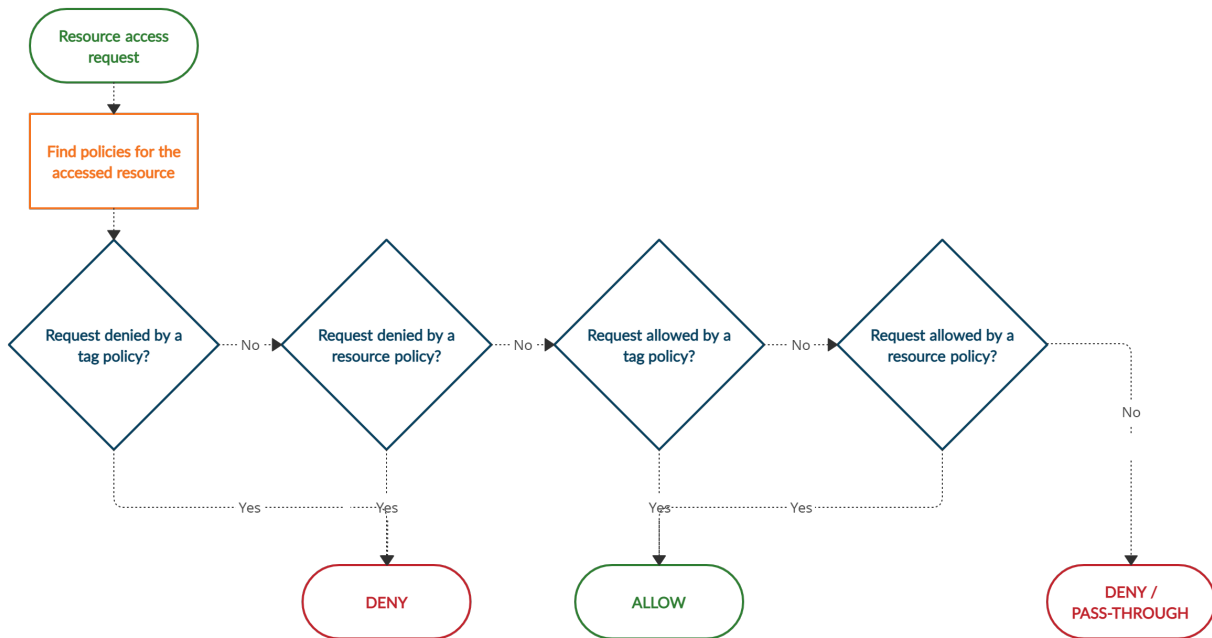
Row-level filtering helps simplify authoring of the Hive query, and provides seamless behind-the-scenes enforcement of row-level segmentation without having to add this logic to the predicate of the query.

Data masking in Hive

Resource-based column masking

Column masking capabilities protect sensitive data in Hive in near real-time. You can set policies that mask or anonymize sensitive data columns from Hive query output.





Tag-based column masking

Tag-based masking policy anonymizes Hive column data based on tags and tag attribute values associated with Hive column.

Apache Ranger

3.2 Configure CLI

The purpose of the `configure` command, is to provide users the ability to create, update or delete connection profiles. Connection profiles are use to establish connections to Apache Ranger REST API environments.

To display the usage documentation, run:

Listing 3: Bash

```
$ ranger-cli configure --help
```

Command examples

- *Create a connection profile*
- *Delete a connection profile*

3.2.1 Create a connection profile

The following command will create the “default” profile. If you wish to create a different profile, then you need to include the `--profile` option.

Listing 4: Bash

```
$ ranger-cli configure
```

3.2.2 Delete a connection profile

To delete a connection profile, execute the `configure` with the `--delete` flag.

Listing 5: Bash

```
$ ranger-cli configure --delete --profile <profile-name>
```

Warning: Please be advised that if you do not include `--policy` option, the default profile will be deleted if it exists.

3.3 Policy CLI

Administration for Apache Ranger's policy public REST API. Users can view, create, update or delete Apache Ranger policies.

Note: Supports resource-based policies **ONLY**.

Resource-based policy: grants permissions to users and/or groups on a set of resource objects.

To display usage documentation, run:

Listing 6: Bash

```
$ ranger-cli policy --help
```

Command examples

- *Get resource-based policy by id*
- *Get resource-based policy by name*
- *Get resource-based policy by service name*
- *Get all resource-based policies*
- *Create a resource-based policy*
- *Update an existing resource-based policy*
- *Delete an existing resource-based policy*

3.3.1 Get resource-based policy by id

Gets policy data for an Apache Ranger resource-based policy using the policy id.

To display usage documentation, run:

Listing 7: Bash

```
$ ranger-cli policy get --help
```

Listing 8: Bash

```
$ ranger-cli policy get --policy-id 1234567
```

On success, this command returns a JSON object with the resource-based policy found.

3.3.2 Get resource-based policy by name

Gets policy data for an Apache Ranger resource-based policy using the policy name.

To display usage documentation, run:

Listing 9: Bash

```
$ ranger-cli policy get --help
```

Listing 10: Bash

```
$ ranger-cli policy get --policy-name 'Demo resource-based HDFS policy'
```

On success, this command returns a JSON object with the resource-based policy found.

3.3.3 Get resource-based policy by service name

Gets policy data for Apache Ranger resource-based policy (or policies) using the service repository name.

Listing 11: Bash

```
$ ranger-cli policy get --help
```

Listing 12: Bash

```
$ ranger-cli policy get --service-name 'hadoop.example.com_hadoop'
```

On success, this command returns a JSON object with the resource-based policy found.

3.3.4 Get all resource-based policies

Gets all policy data for every Apache Ranger resource-based policies.

Listing 13: Bash

```
$ ranger-cli policy get --help
```

Listing 14: Bash

```
$ ranger-cli policy get
```

On success, this command returns a JSON object with the resource-based policy (or policies) found.

3.3.5 Create a resource-based policy

Creates a new Apache Ranger resource-based policy.

To display usage documentation, run:

Listing 15: Bash

```
$ ranger-cli policy create --help
```

Listing 16: Bash

```
$ ranger-cli policy create --config /home/jsmith/hive-policy.json
```

On success, this command returns a JSON object with the resource-based policy created.

3.3.6 Update an existing resource-based policy

Updates an existing Apache Ranger resource-based policy.

To display usage documentation, run:

Listing 17: Bash

```
$ ranger-cli policy update --help
```

Listing 18: Bash

```
$ ranger-cli policy update --policy-id 12345 --config /home/jsmith/hive-policy.json
```

On success, this command returns a JSON object with the resource-based policy updated.

3.3.7 Delete an existing resource-based policy

Deletes an existing Apache Ranger resource-based policy.

To display usage documentation, run:

Listing 19: Bash

```
$ ranger-cli policy delete --help
```

Listing 20: Bash

```
$ ranger-cli policy delete --policy-id 12345
```

On success, this command returns nothing, otherwise HTTP status code/reason

3.4 Service CLI

Administration for Apache Ranger's policy service REST API. Users can view, create, update or delete Apache Ranger service repositories.

To display usage documentation, run:

Listing 21: Bash

```
$ ranger-cli service --help
```

Command examples

- *Get service repository by id*
- *Get service repository by name*
- *Get all service repositories*
- *Create a service repository*
- *Update an existing service repository*
- *Delete an existing service repository*

3.4.1 Get service repository by id

Gets repository data for an Apache Ranger service repository using the service id.

To display usage documentation, run:

Listing 22: Bash

```
$ ranger-cli service get --help
```

Listing 23: Bash

```
$ ranger-cli service get --service-id 1234567
```

On success, this command returns a JSON object with the service repository found.

3.4.2 Get service repository by name

Gets repository data for an Apache Ranger service repository using the service name.

To display usage documentation, run:

Listing 24: Bash

```
$ ranger-cli service get --help
```

Listing 25: Bash

```
$ ranger-cli service get --service-name 'HDFS repository'
```

On success, this command returns a JSON object with the service repository found.

3.4.3 Get all service repositories

Gets all repository data for every Apache Ranger service repositories.

Listing 26: Bash

```
$ ranger-cli service get --help
```

Listing 27: Bash

```
$ ranger-cli service get
```

On success, this command returns a JSON object with the service repository (or repositories) found.

3.4.4 Create a service repository

Creates a new Apache Ranger service repository.

To display usage documentation, run:

Listing 28: Bash

```
$ ranger-cli service create --help
```

Listing 29: Bash

```
$ ranger-cli service create --config hive-service.json
```

On success, this command returns a JSON object with the service repository created.

See also:

For additional details on JSON configuration files, please see [Plugin Templates](#)

3.4.5 Update an existing service repository

Updates an existing Apache Ranger service repository.

To display usage documentation, run:

Listing 30: Bash

```
$ ranger-cli service update --help
```

Listing 31: Bash

```
$ ranger-cli service update --service-id 12345 --config hive-service.json
```

On success, this command returns a JSON object with the service repository updated.

See also:

For additional details on JSON configuration files, please see *Plugin Templates*

3.4.6 Delete an existing service repository

Deletes an existing Apache Ranger service repository.

To display usage documentation, run:

Listing 32: Bash

```
$ ranger-cli service delete --help
```

Listing 33: Bash

```
$ ranger-cli service delete --service-id 12345
```

On success, this command returns nothing, otherwise HTTP status code/reason

3.5 ServiceDef CLI

Administration for Apache Ranger's servicedef service REST API. Users can view, create, update or delete Apache Ranger service definitions. Service definitions are use to create service repositories and users must be aware of certain requirements within the definitions.

3.5.1 Service Definition configurations

The service definitions contains

To display usage documentation, run:

Listing 34: Bash

```
$ ranger-cli servicedef --help
```

Command examples

- *Get service definitions*
- *Get service definitions by type*
- *List configuration properties*
- *List resource properties*

Get service definitions

Gets service definition data for all Apache Ranger service definitions.

To display usage documentation, run:

Listing 35: Bash

```
$ ranger-cli servicedef get --help
```

Listing 36: Bash

```
$ ranger-cli servicedef get
```

On success, this command returns a JSON object with the service definitions found.

Note: To view the latest service types available, please use `--help` option.

Get service definitions by type

Gets service definition data for all Apache Ranger service definitions by type.

To display usage documentation, run:

Listing 37: Bash

```
$ ranger-cli servicedef get --help
```

Listing 38: Bash

```
$ ranger-cli servicedef get --service-type HDFS
```

On success, this command returns a JSON object with the service definitions found.

List configuration properties

Gets service definition configuration properties which are used to create new service repository (or repositories).

To display usage documentation, run:

Listing 39: Bash

```
$ ranger-cli servicedef list-configs --help
```

Below is an example of listing service definition configuration properties for the Hive service.

Listing 40: Bash

```
$ ranger-cli servicedef list-configs --service-type HIVE
```

Listing 41: Console

HIVE Plugin Configs	
PROPERTY	VALUE
name mandatory	username True
name mandatory	password True
name mandatory defaultValue	jdbc.driverClassName True org.apache.hive.jdbc.HiveDriver
name mandatory defaultValue	jdbc.url True
name mandatory	commonNameForCertificate False

The table above provides the configuration properties required to be included in your `<service>-service.json` file.

See also:

For available templates, please check out [Plugin Templates](#)

List resource properties

Returns a table with the name, type and default value of the service definition configuration properties. These properties are used to create new service repositories.

3.6 Plugins CLI

Administration for Apache Ranger's plugins public REST API. Users can view Apache Ranger plugins.

To display usage documentation, run:

Listing 42: Bash

```
$ ranger-cli plugins --help
```

Command examples

- *Get plugins info*
- *Get plugins info by apptype*
- *Get plugins info by hostname*
- *Get plugins info by service name*

3.6.1 Get plugins info

Gets all Apache Ranger plugins info.

To display usage documentation, run:

Listing 43: Bash

```
$ ranger-cli plugins info --help
```

3.6.2 Get plugins info by apptype

Gets all Apache Ranger plugins info by app type.

To display usage documentation, run:

Listing 44: Bash

```
$ ranger-cli plugins info --help
```

Listing 45: Bash

```
$ ranger-cli plugins info --apptype yarn
```

On success, this command returns a JSON object with the service definitions found.

3.6.3 Get plugins info by hostname

Gets all Apache Ranger plugins info by hostname.

To display usage documentation, run:

Listing 46: Bash

```
$ ranger-cli plugins info --help
```

Listing 47: Bash

```
$ ranger-cli plugins info --hostname nn.example.com
```

On success, this command returns a JSON object with the service definitions found.

3.6.4 Get plugins info by service name

Gets all Apache Ranger plugins info by service repository name.

To display usage documentation, run:

Listing 48: Bash

```
$ ranger-cli plugins info --help
```

Listing 49: Bash

```
$ ranger-cli plugins info --service-name cl1_hive
```

On success, this command returns a JSON object with the service definitions found.

3.7 Plugin Templates

Plugin templates are used to create or update service repositories. These templates can be edited then invoked with either:

Warning: This interface is still a “work-in-progress”.

Create service repository

Listing 50: Bash

```
$ ranger-cli service create --config <service-name>-service.json
```

Update existing service repository

Listing 51: Bash

```
$ ranger-cli service update --config <service-name>-service.json --service-id 12345
```

3.7.1 HDFS

HDFS service JSON template file.

Listing 52: hdfs-service.json

```
{
  "name": "cl1_hdfs",
  "description": "hdfs service client",
  "isEnabled": true,
  "type": "hdfs",
  "configs": {
    "username": "hdfs",
    "password": "hdfs",
    "fs.default.name": "hdfs://hadoop.example.com",
    "hadoop.security.authorization": true,
    "hadoop.security.authentication": "kerberos",
    "hadoop.security.auth_to_local": "RULE:[1:$1@$0](ambari-qa-hadoop.example.
    ↪com@EXAMPLE.COM)s/.*\/ambari-qa/",
    "dfs.datanode.kerberos.principal": "dn/m1.hadoop.example.com@EXAMPLE.COM",
    "dfs.namenode.kerberos.principal": "nn/m1.hadoop.example.com@EXAMPLE.COM",
    "dfs.secondary.namenode.kerberos.principal": "nn/m1.hadoop.example.com@EXAMPLE.
    ↪COM",
    "hadoop.rpc.protection": "authentication",
    "ambari.service.check.user": "ambari-qa",
    "tag.download.auth.users": "hdfs",
    "policy.download.auth.users": "hdfs"
  }
}
```

3.7.2 Hive

Hive service JSON template file.

Listing 53: hive-service.json

```
{
  "name": "cl1_hive",
  "description": "hive service client",
  "isEnabled": true,
  "type": "hive",
  "configs": {
    "username": "hive",
    "password": "hive",
    "jdbc.driverClassName": "org.apache.hive.jdbc.HiveDriver",
    "jdbc.url": "jdbc:hive2://localhost:10500",
    "ambari.service.check.user": "ambari-qa",

```

(continues on next page)

(continued from previous page)

```
"tag.download.auth.users": "hive",  
"policy.download.auth.users": "hive",  
"policy.grantrevoke.auth.users": "hive"  
}  
}
```

3.8 LICENSE

Copyright (c) 2021 Deric Degagne

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

CHAPTER FOUR

LINKS

- [Apache Ranger REST APIs](#)