What's new in Apache Ranger 1.0.0

Dr. Colm O hEigeartaigh
Talend
@coheigea

APACHECON North America Sept. 24-27, 2018

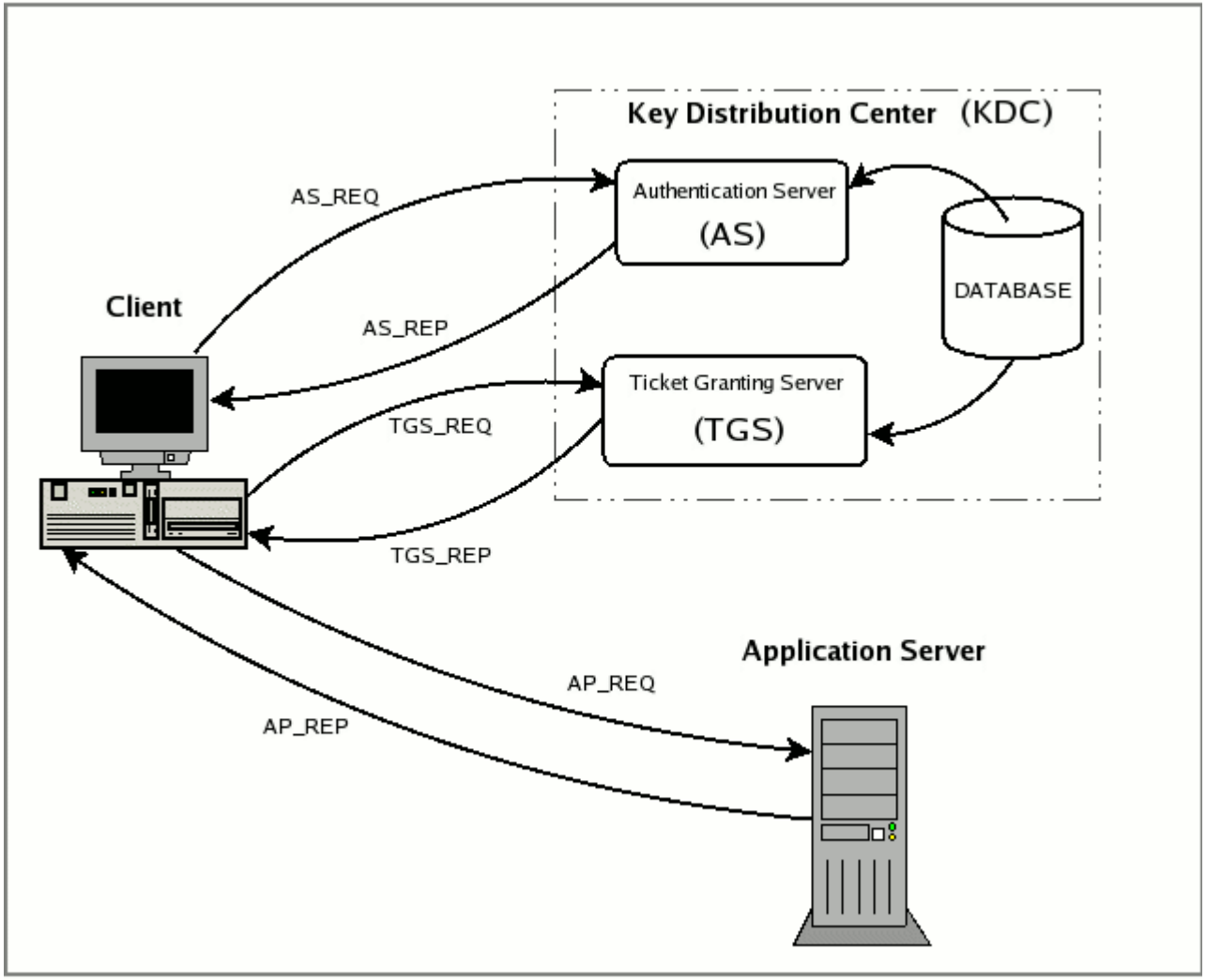
Speaker Introduction

Big Data ecosystem

APACHE
STORM™

Securing Big Data

Authentication



Credit: https://stackoverflow.com/questions/39850594/kerberos-how-does-application-server-decrypt-service-ticket

Authorization

- Each project has own solution, e.g. ACLs, custom authorization plugins, etc.
- For example, for HDFS:

  bin/hadoop fs -chmod g+w /data/*

  bin/hadoop fs -setfacl -m user:alice:r-x /data

  “dfs.namenode.inode.attributes.provider.class” property

Auditing

- How do we know when a user was authenticated?
- How do we know when an authentication failure occurred?
- How do we know when (and how many) authorization failures occurred?

Open Questions

- Security desirables:
  - Common way of applying authorization to different projects
  - Centralized security policy repository
  - REST API access to policies
  - ACLs to control who can create authorization policies
  - Single auditing framework that works with every project

Introducing

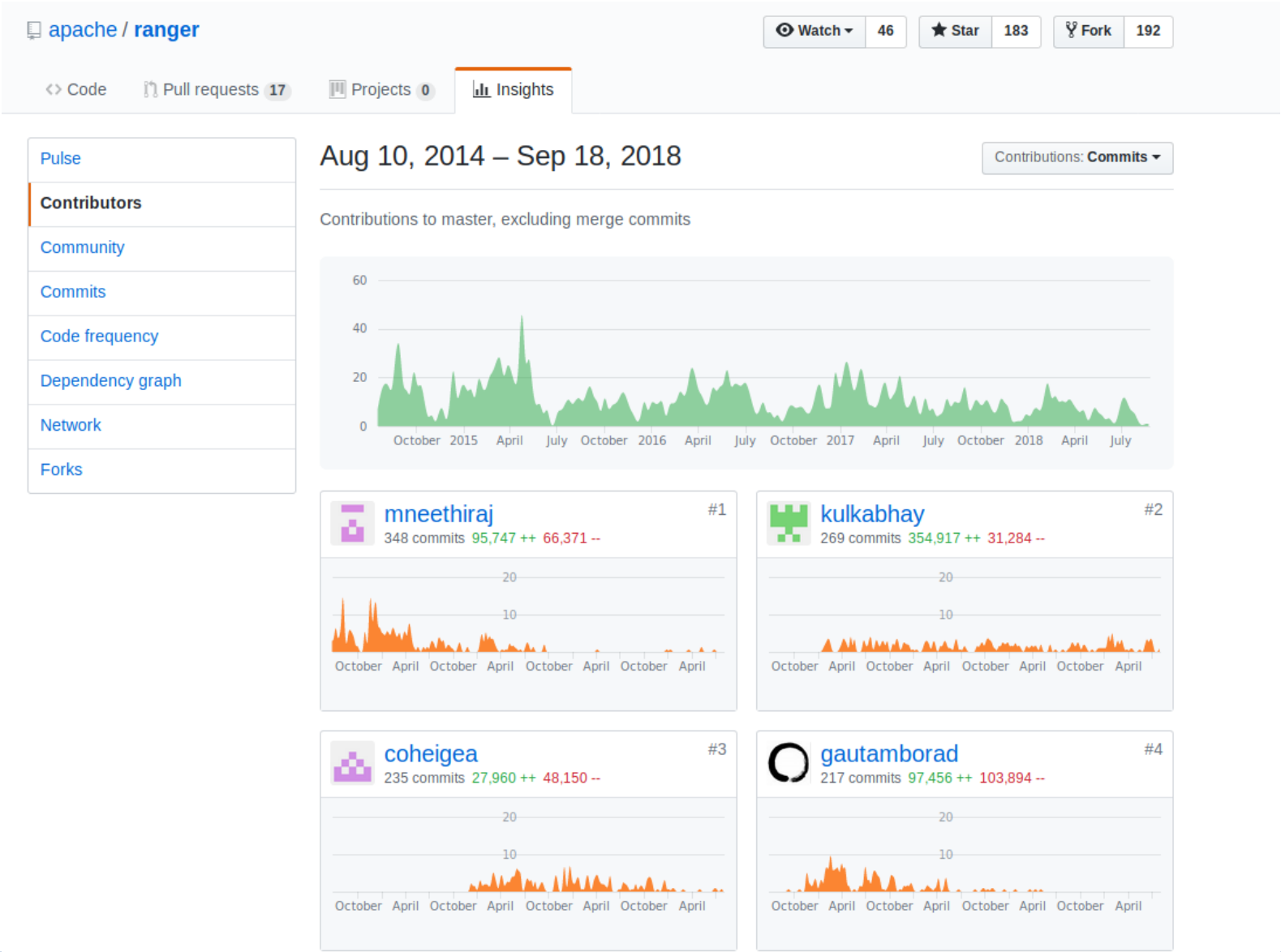
Apache Ranger

- Apache Ranger is a framework to enable, monitor and manage comprehensive data security across the Hadoop platform.

Apache Ranger information

- 2014-07: Enters Apache Incubator
- 2014-11: First Apache release (ranger-0.4-incubating)
- 2017-01: Graduates to a Top Level Project
- 2017-06: First TLP release (0.7.1)
- 2018-03: 1.0.0 release
- 2018-07: 1.1.0 release
- Currently 26 committers + 18 PMC

Apache Ranger information

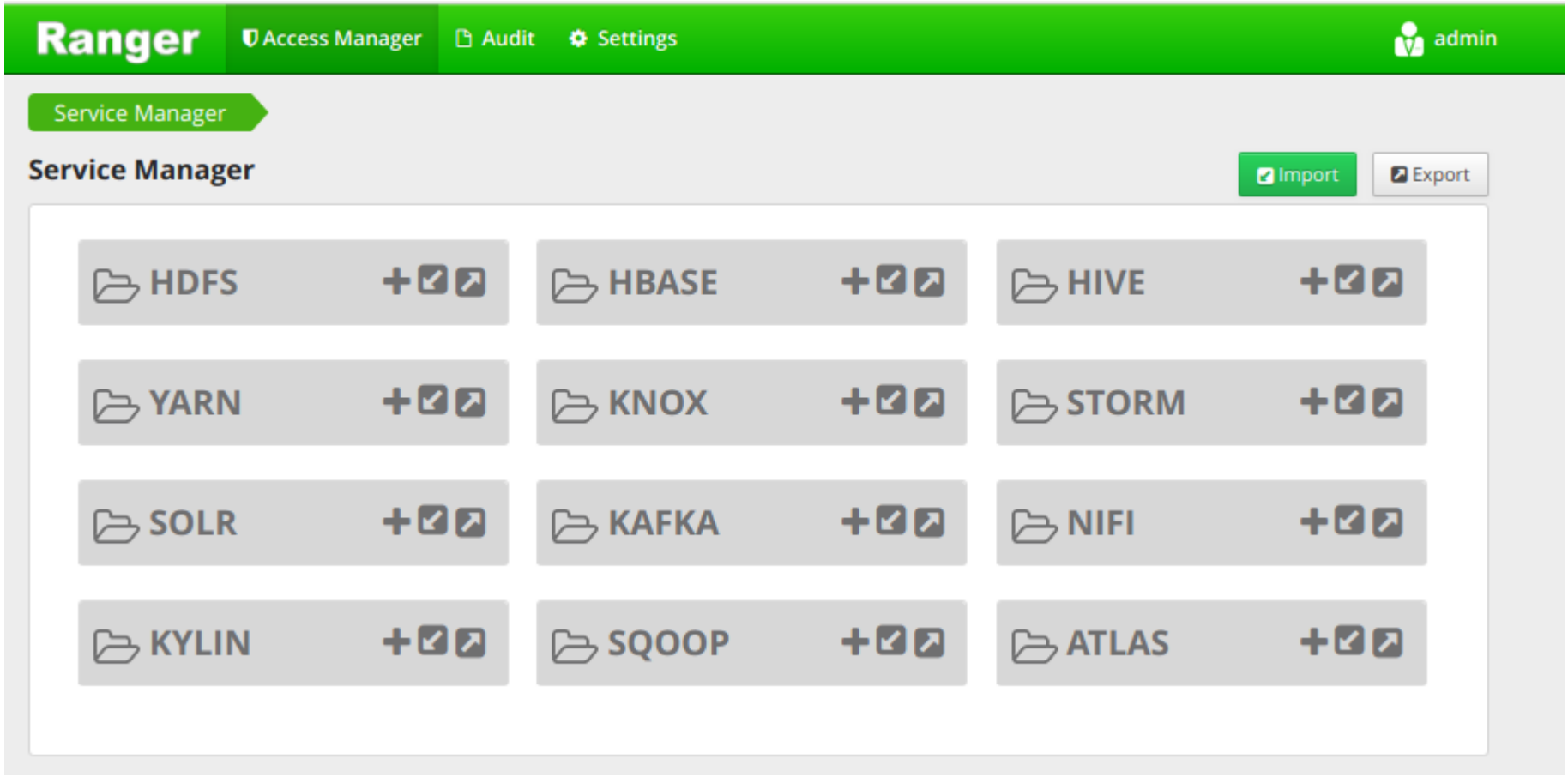


Apache Ranger Admin Service

- The Apache Ranger Admin Service is a centralized security admin service accessed via a UI or REST APIs.
- It allows us to manage authorization policies in a consistent manner for a wide range of big data projects.
- Policies are stored in a separate database.

Projects supported by Ranger

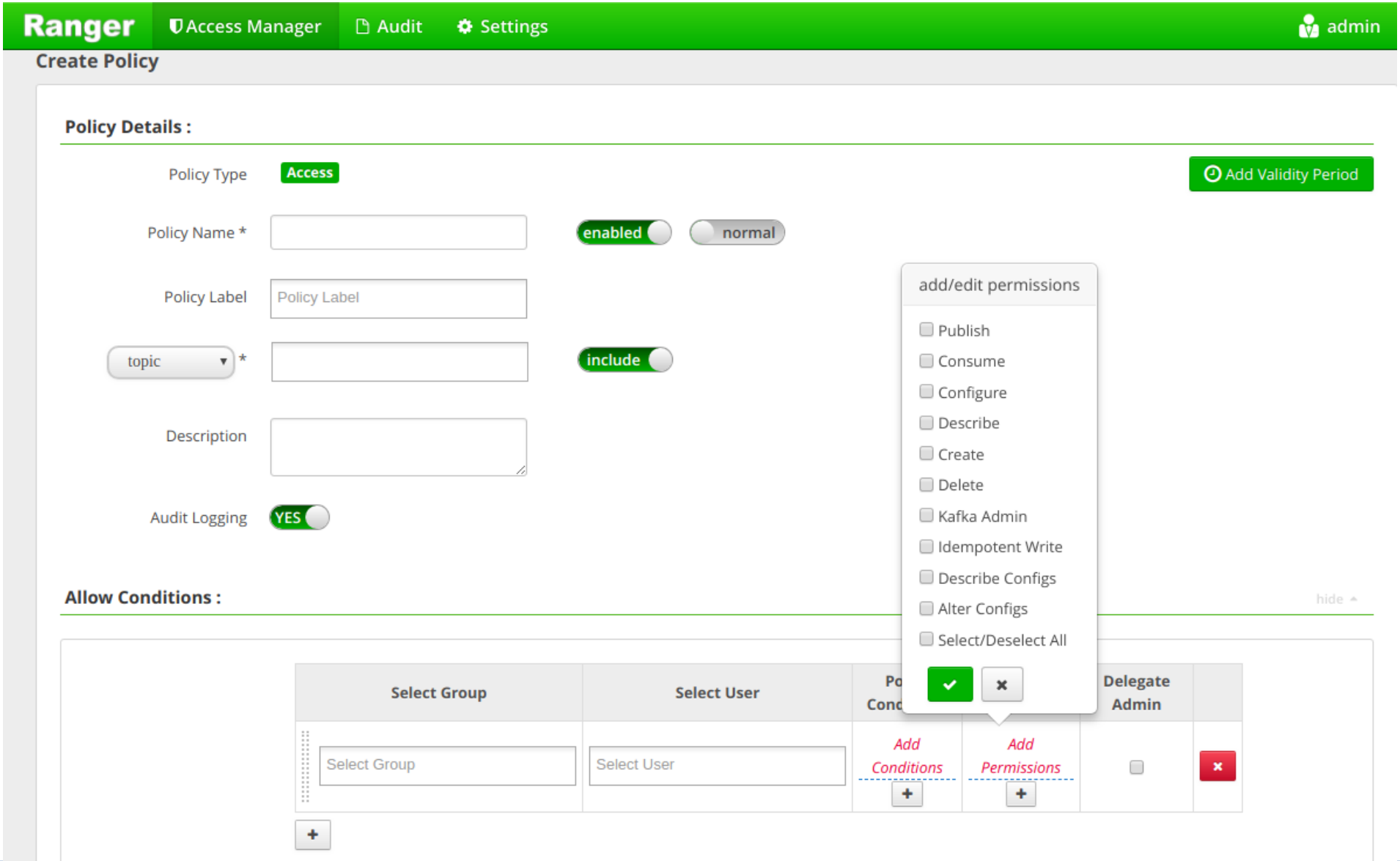
Projects supported by Ranger



Authorization Policies

- Apache Ranger has two types of authorization policies: resource and tag.
- Resource Authorization Policies are created per-project + associate a “resource” with allow/deny conditions.
- Allow/deny conditions associate a subject (user/group) with some permissions specific to that plugin.
- Some plugins have additional conditions: e.g. IP address range (Kafka)

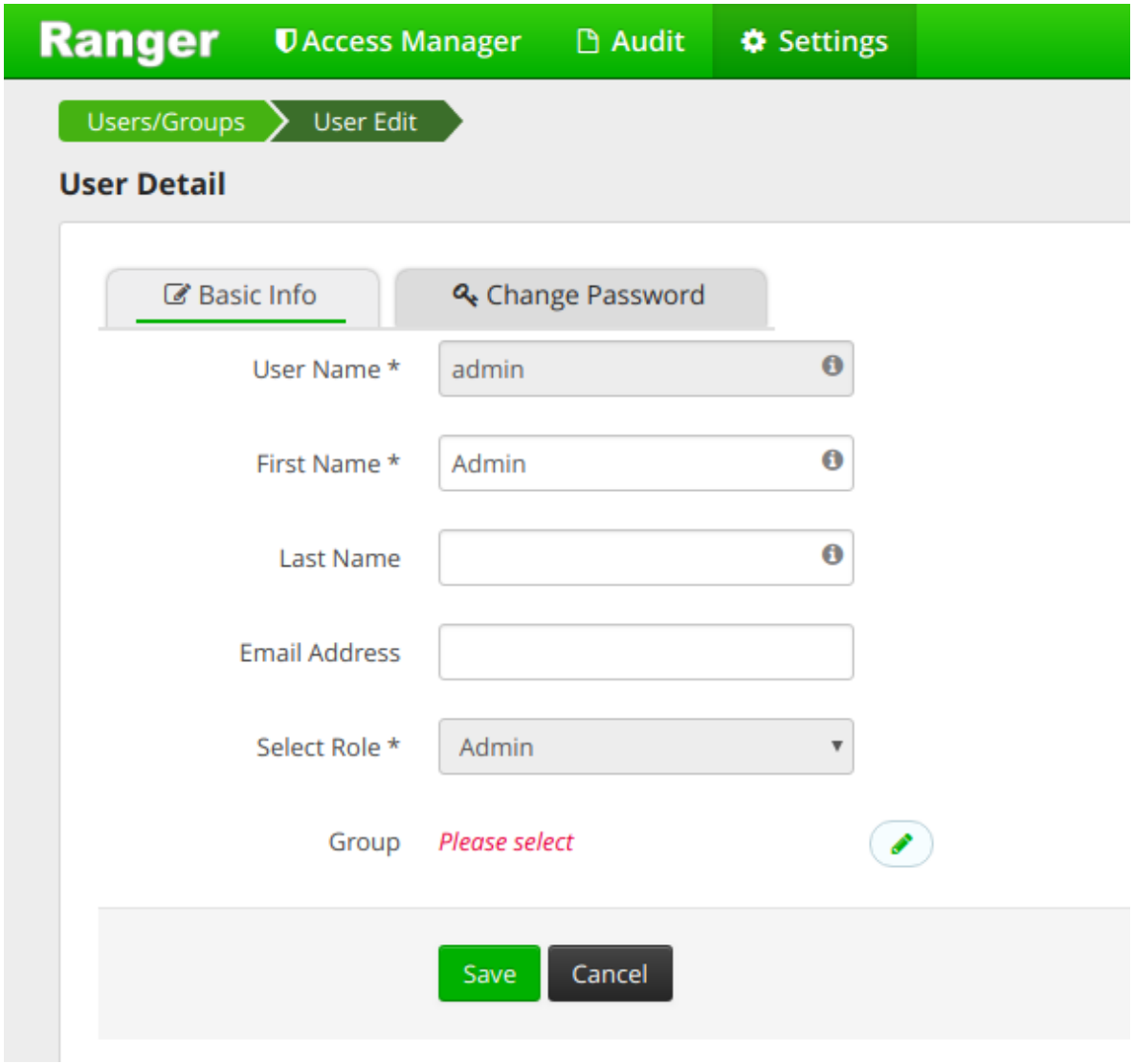
Apache Kafka example



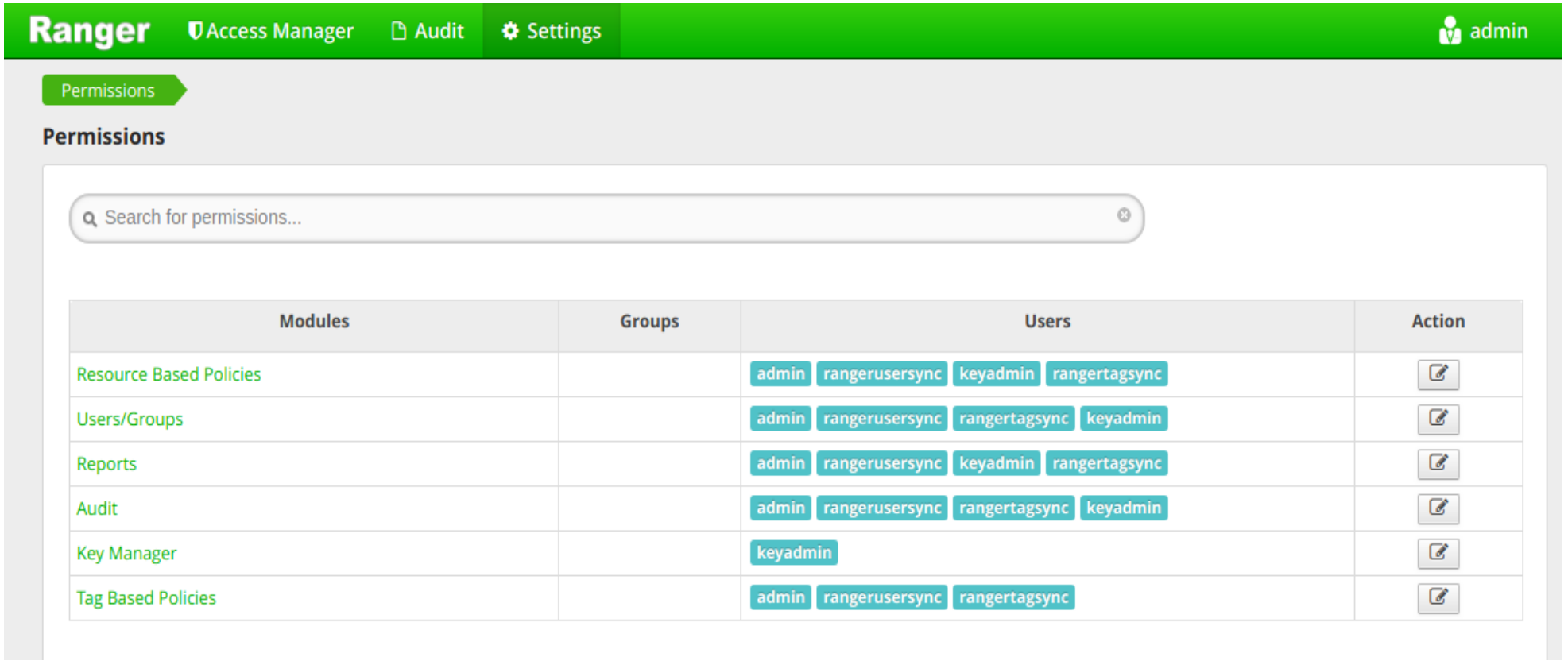
User/Group Management

- To create authorization policies, we need to have a subject, i.e. a user or group.
- The admin service provides a way of managing users and groups.
- We can also associate permissions with users in the context of the admin service.
- Apache Ranger supports syncing users into the Admin service from both unix and ldap via the usersync service.

User/Group Management



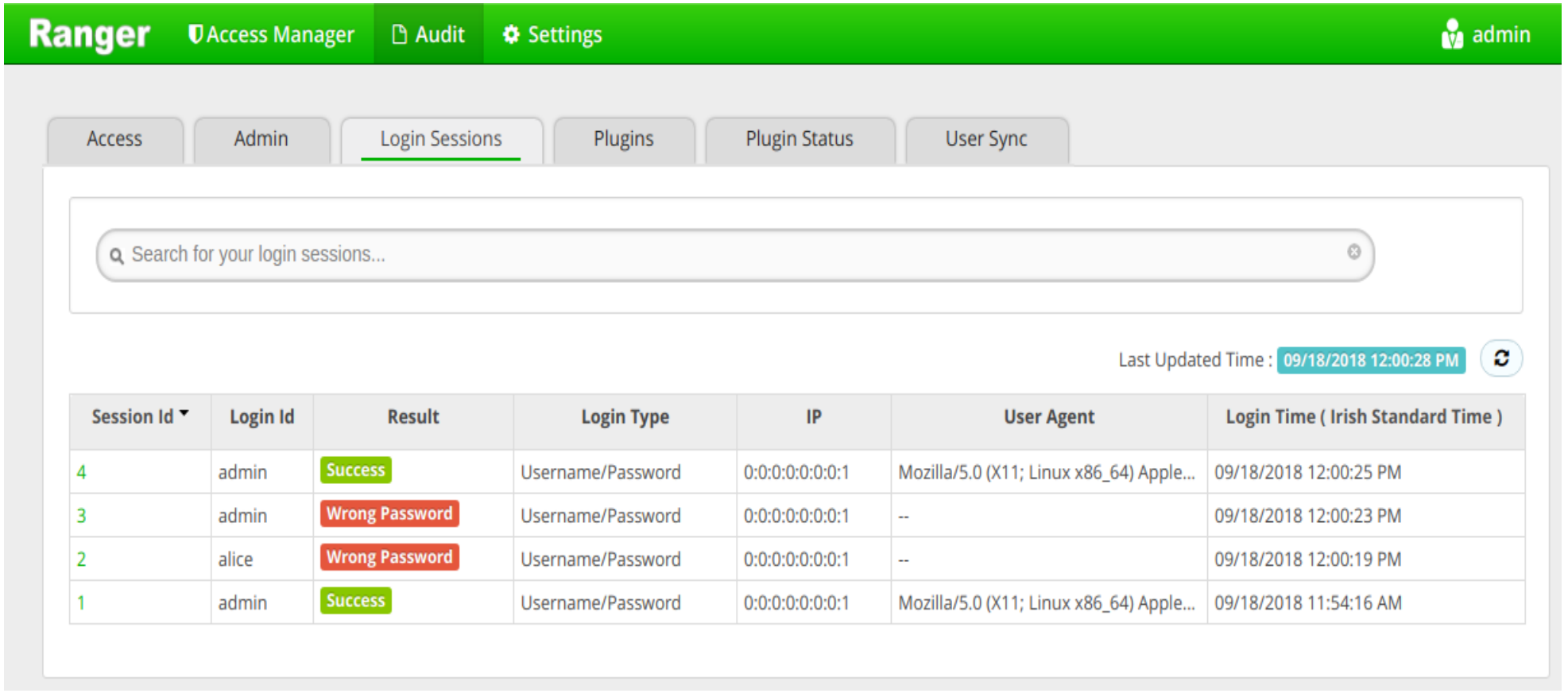
User/Group Management



Auditing

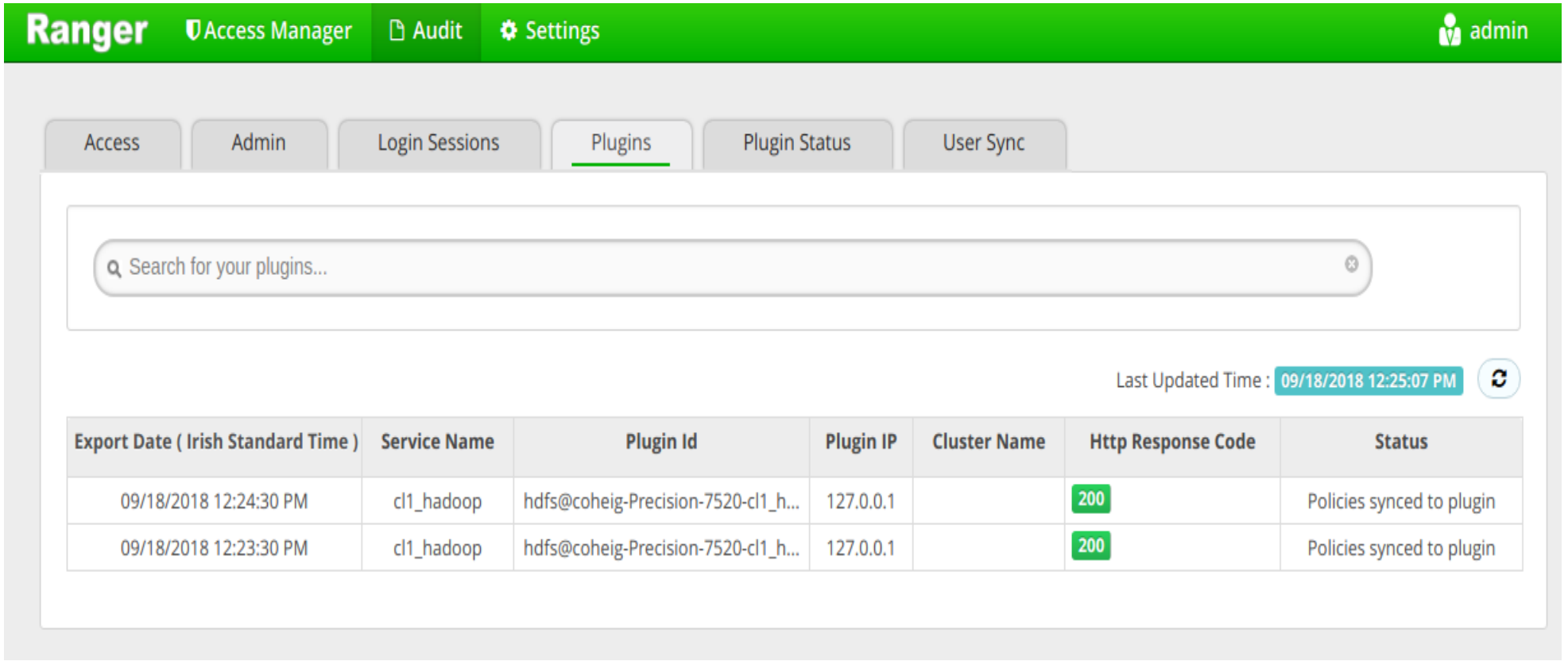
- Apache Ranger supports storing audit information in Apache Solr.
- Ranger can store audit information relating to the admin service itself, e.g. login sessions, user sync.
- Ranger can also collect audit information from the authorization plugins.

Auditing



Session Id	Login Id	Result	Login Type	IP	User Agent	Login Time (Irish Standard Time)
4	admin	Success	Username/Password	0:0:0:0:0:0:0:1	Mozilla/5.0 (X11; Linux x86_64) Apple...	09/18/2018 12:00:25 PM
3	admin	Wrong Password	Username/Password	0:0:0:0:0:0:0:1	--	09/18/2018 12:00:23 PM
2	alice	Wrong Password	Username/Password	0:0:0:0:0:0:0:1	--	09/18/2018 12:00:19 PM
1	admin	Success	Username/Password	0:0:0:0:0:0:0:1	Mozilla/5.0 (X11; Linux x86_64) Apple...	09/18/2018 11:54:16 AM

Auditing


Ranger
Access Manager
Audit
Settings
admin
Access
Admin
Login Sessions
Plugins
Plugin Status
User Sync
Search for your plugins...
Last Updated Time : 09/18/2018 12:25:07 PM
Export Date (Irish Standard Time)
Service Name
Plugin Id
Plugin IP
Cluster Name
Http Response Code
Status
09/18/2018 12:24:30 PM
cl1_hadoop
hdfs@coheig-Precision-7520-cl1_h...
127.0.0.1
200
Policies synced to plugin
09/18/2018 12:23:30 PM
cl1_hadoop
hdfs@coheig-Precision-7520-cl1_h...
127.0.0.1
200
Policies synced to plugin

Authorization Plugins

- The admin service allows us to manage authorization policies, but how are they enforced?
- Ranger ships separate plugins for each of the big data projects that are supported
- The plugins use custom authorization hooks.
- The plugins pull policies from the admin service REST API.

Ranger Admin Service Demo I

- Build Ranger source tarball via Maven:

  mvn clean package assembly:assembly -DskipTests

- Extract target/ranger-${version}-admin.tar.gz
- Start the database:

  docker run -p 5432:5432 postgres

  psql -h localhost -U postgres -c 'CREATE DATABASE ranger;'

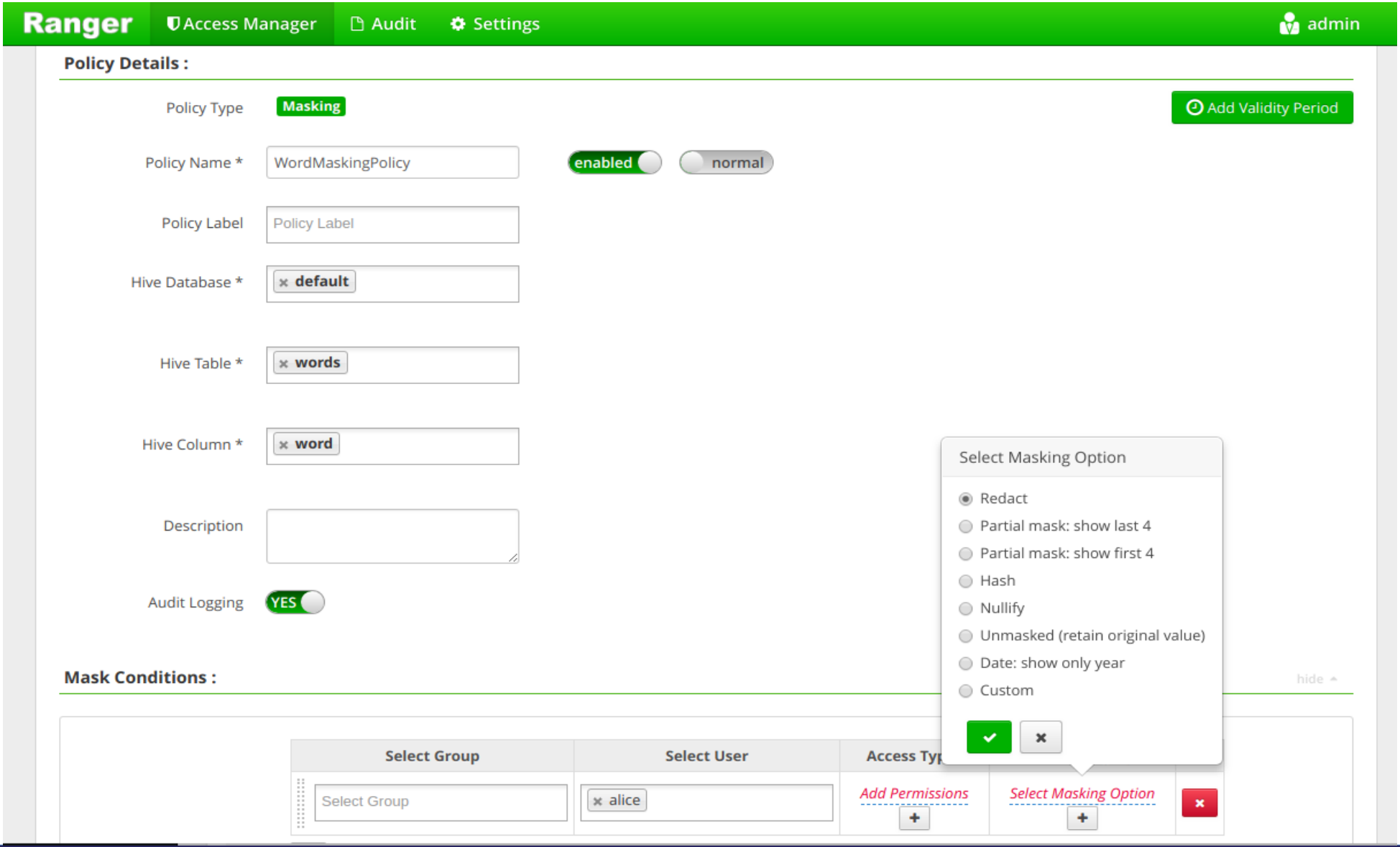
Ranger Admin Service Demo II

- Configure ${ranger_admin}/install.properties with DB connection information
- Install: sudo -E ${ranger_admin}/setup.sh
- Start: sudo ranger-admin start
- Navigate to: http://localhost:6080 (creds: admin/admin)

Some advanced features

- Hive Data-Masking and Row-level filtering
- Tag based authorization policies
- KMS support

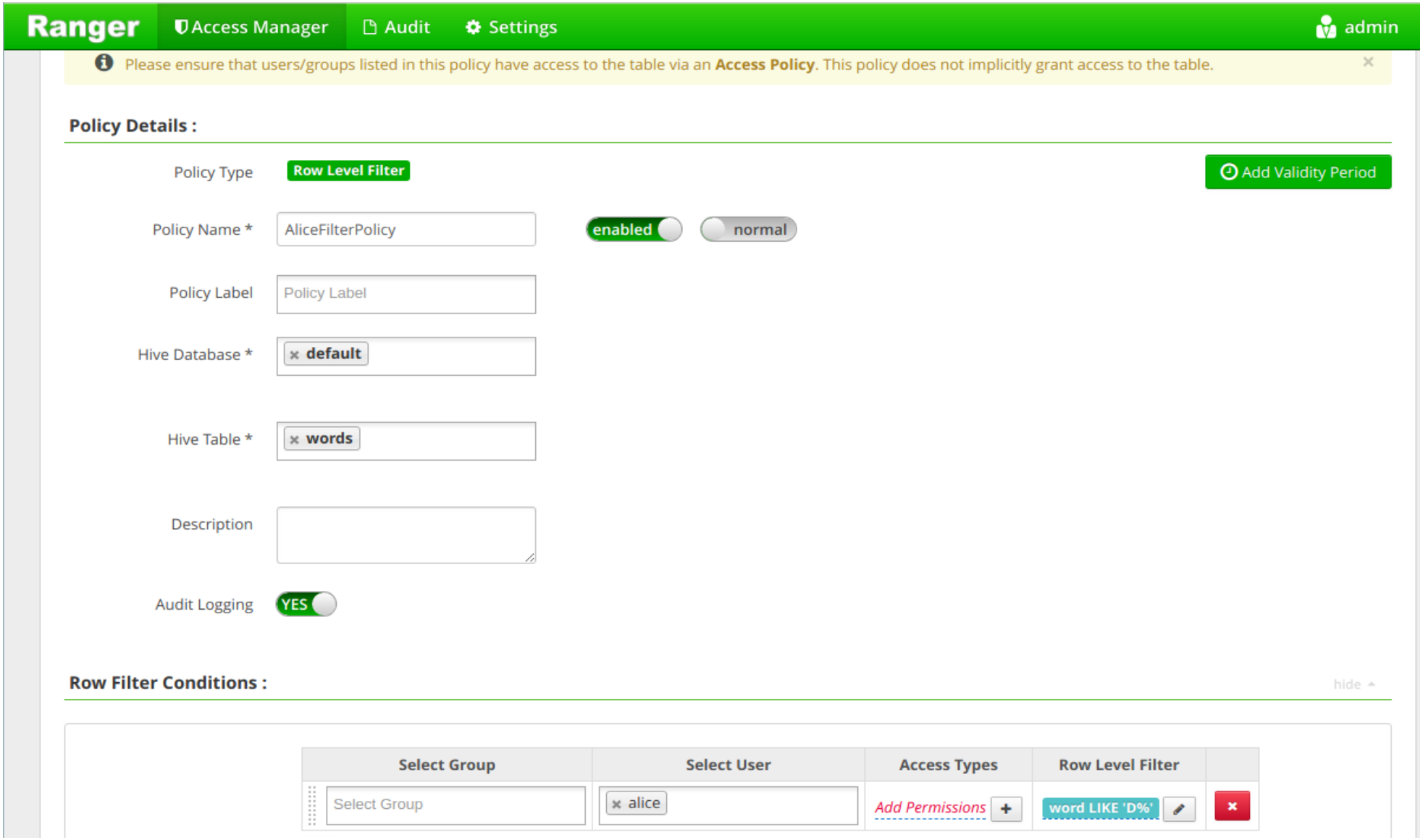
Hive Data-masking policy



Hive Data-masking output

```
0: jdbc:hive2://localhost:10000> select * from words LIMIT 5;
+-------------+--------------+--+
| words.word  | words.count  |
+-------------+--------------+--+
| "'Xxx       | 1            |
| "'X'x       | 1            |
| "'X         | 1            |
| "'Xxxxx'    | 1            |
| "'Xxxxxx    | 1            |
+-------------+--------------+--+
```

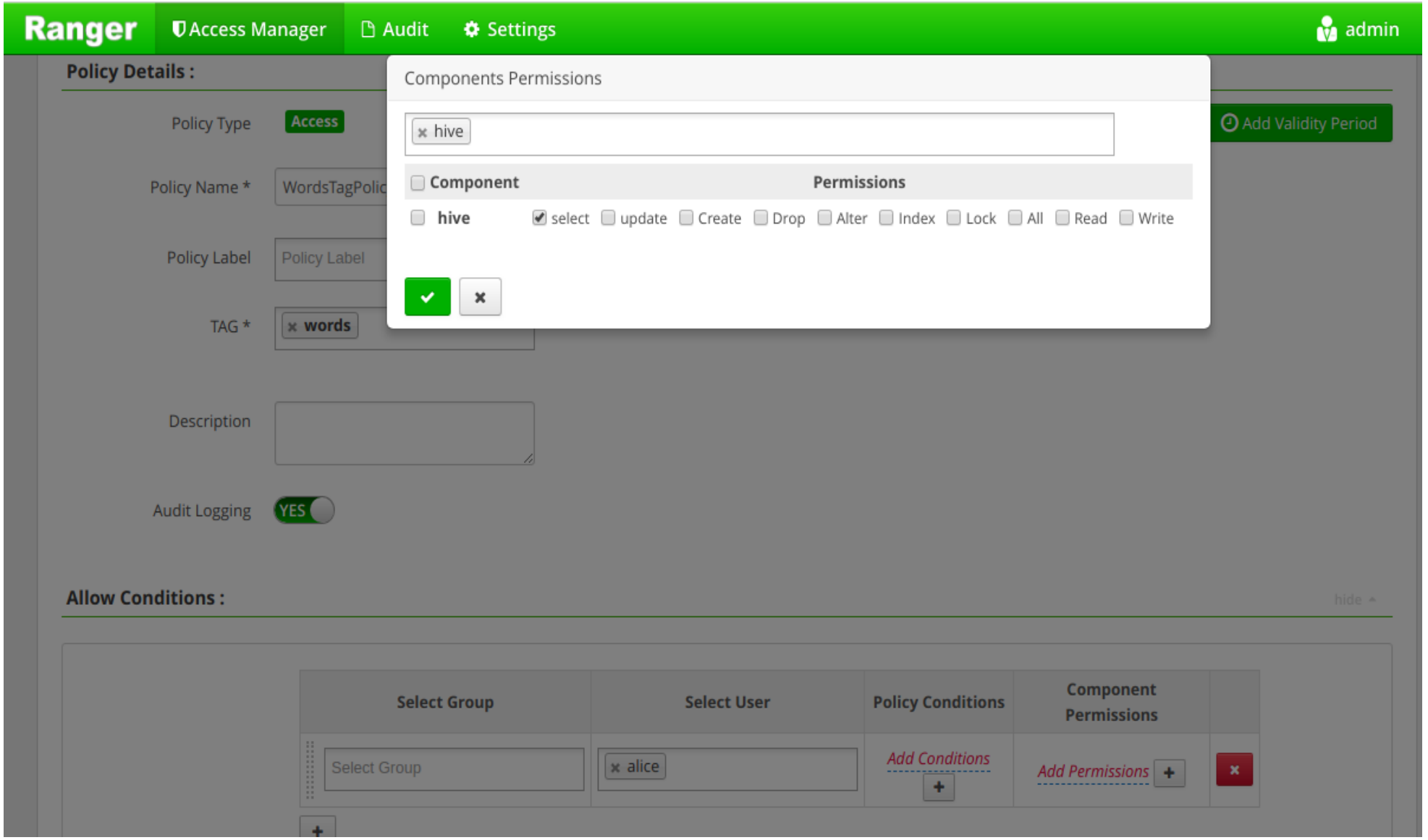
Hive Row-Level Filtering



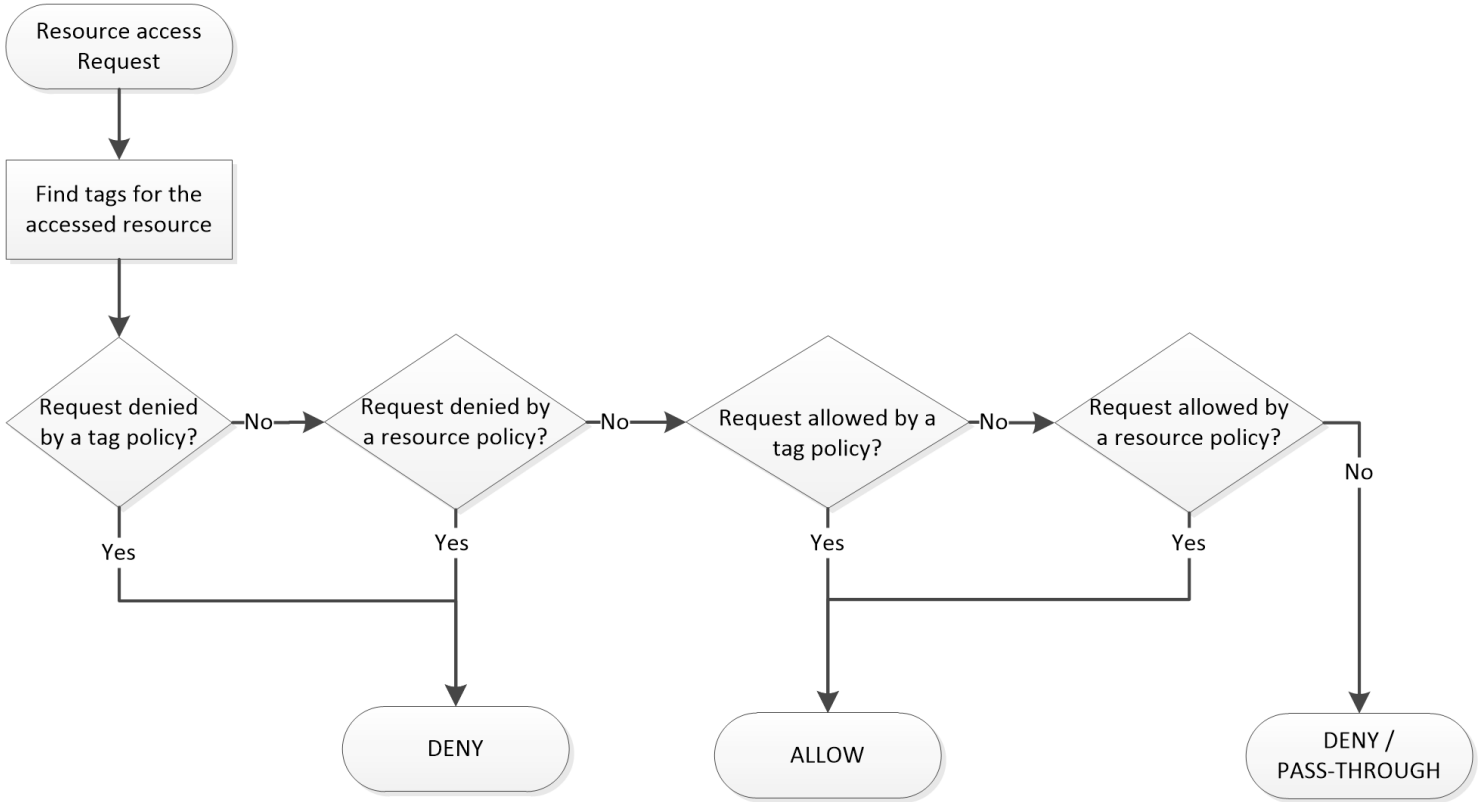
Tag-based authorization

- Instead of associating an allow/deny condition with a resource, we can associate it with a “tag”.
- This means we don’t have to manage resources in Ranger.
- Apache Atlas allows us to associate tags with metadata.
- Ranger TagSync service syncs tags into Ranger from Atlas.

Tag-based authorization



Ranger Policy evaluation



Apache Ranger Policy Evaluation Flow with Tags

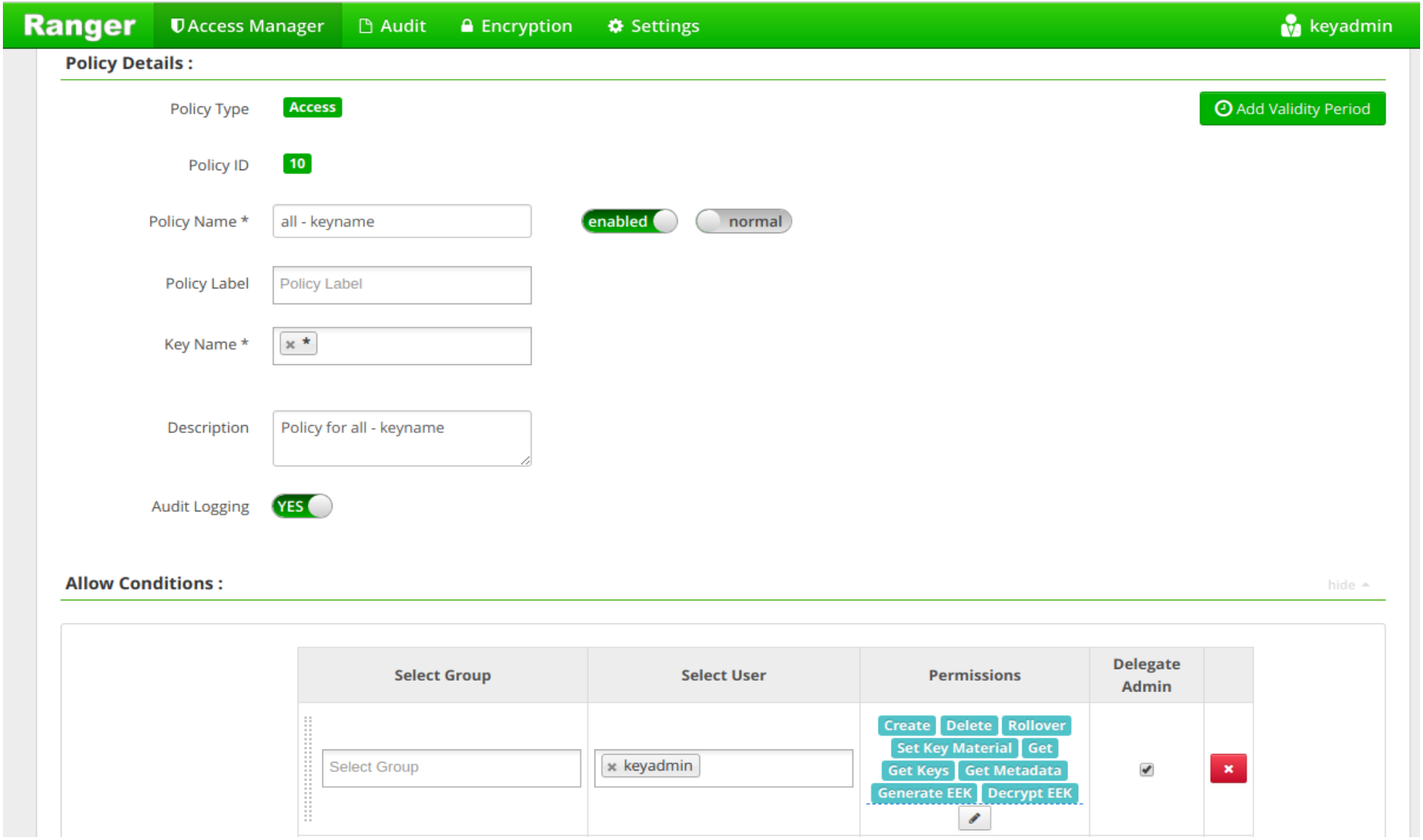
Hadoop Transparent Encryption

- HDFS supports transparent encryption via Encryption Zones.
- Each encryption zone associated with an encryption zone key.
- Each file in the encryption zone associated with a DEK (data encryption key).
- Hadoop KMS stores encryption zone keys, generates/decrypts encrypted DEKs.

Apache Ranger KMS

- Ranger ships with a KMS service which stores encryption keys in a database.
- Authorization policies can be created to authorize each operation associated with KMS.
- Authorization policies map to key names.

Apache Ranger KMS



Ranger KMS Service Demo

- Extract target/ranger-${version}-kms.tar.gz
- Create a database: psql -h localhost -U postgres -c 'CREATE DATABASE rangerkms;'
- Configure ${ranger_kms}/install.properties with DB connection information
- Install: sudo -E ${ranger_kms}/setup.sh
- Start: sudo ranger-kms start
- Navigate to: http://localhost:6080 (creds: keyadmin/keyadmin)

1.0.0 release (2018/03)

- First major release as a TLP.
- 468 JIRA issues completed!
- JDK 1.8 minimum requirement.
- Support for Apache Sqoop added
- Kafka TransactionID resource.
- Tag Data Masking policies.

1.0.0 release (2018/03)

- Plugin updates:
  - Apache Hive 2.3.x
  - Apache Hbase 1.2.x
  - Apache Atlas 0.8.0
  - Apache Kafka 1.0.0
  - Apache Knox 1.0.0
  - Apache Storm 1.2.0

1.1.0 release (2018/07)

- 127 JIRA issues completed!
- Support for Apache Kylin added.
- Support for time-bound and temporary authorization
- Support for Prioritized Policies
- Plugin updates:
  - Apache Atlas
  - Apache Hbase

2.0.0 plans

- Plugin updates:
  - Hadoop 3.0.x
  - Hive 3.0.x
  - Kafka 2.0.x
  - Hbase 2.0.x
  - Atlas 2.0.x
- Potential new Elasticsearch plugin.

THANK YOU

Dr. Colm O hEigeartaigh

@coheigea

http://coheigea.blogspot.com/