Enabling Universal Authorization Models using Sentry

Hao Hao - hao.hao@cloudera.com
Anne Yu - anneyu@cloudera.com

Vancouver BC, Canada, May 9 - 12 2016

cloudera

Sentry

About us

- Software engineers at Cloudera
- Apache Sentry PMC and Committer
- Hao, used to work at Search Backend, eBay Inc.
- Anne, used to work at Search Backend, A9, an Amazon subsidiary.

cloudera

Sentry

Presentation Agenda

- Sentry Overview
  - Introduction
  - Architecture
- Sentry Generic Authorization Model
  - Motivation: easy integration with Apache data engines, even third-party data applications
  - Successfully integrated with Apache Solr, Kafka and Sqoop2
  - Integration examples
- Sentry Other Features and Future Work

cloudera

Sentry

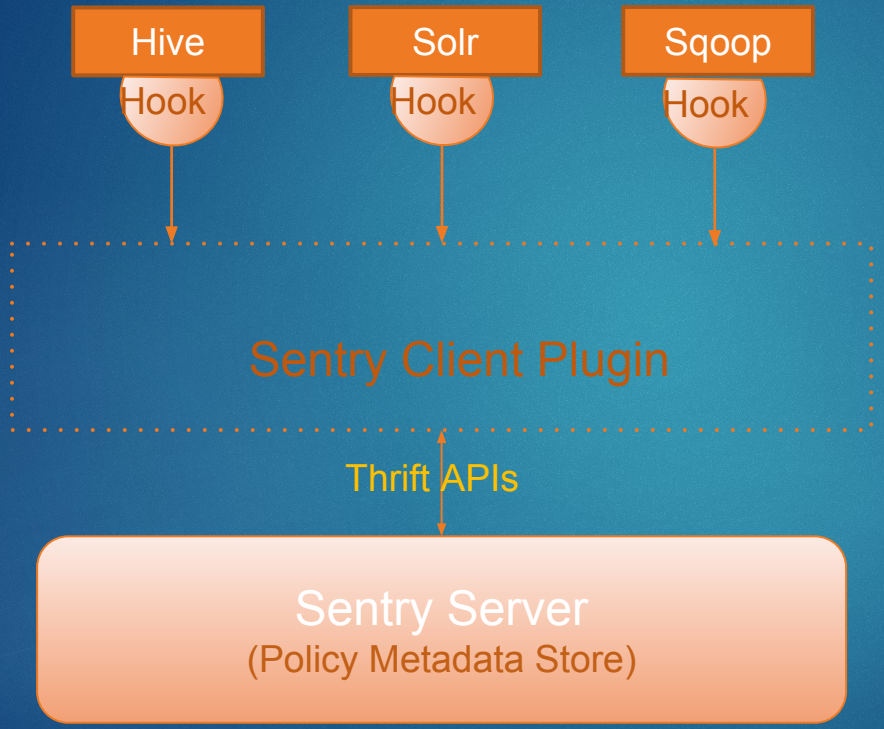
Sentry Overview

- **Authorization** Service
  - Sentry provides the ability to enforce role-based access control (RBAC) to data and/or metadata for authenticated users in a fine-grained manner.
  - Enterprise grade big data security.
  - Provides unified policy management.
  - Pluggable and highly modular.

- Work out of the box with Apache Hive, Hive metastore/HCataglog, Apache Solr, Apache Kafka, Apache Sqoop and Apache Impala.

cloudera

Sentry Architecture

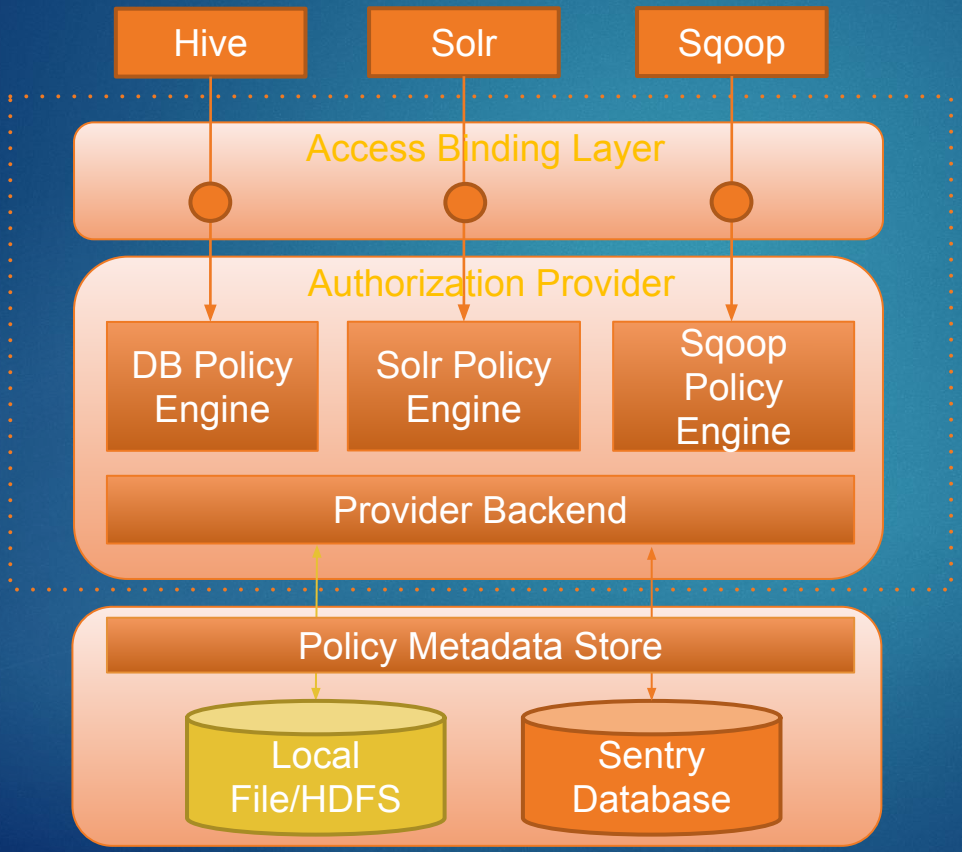


- Server Client Model
- Thrift client APIs
  - Get privileges;
  - Grant/Revoke role;
  - Grant/Revoke privileges;
  - List roles

cloudera

Sentry

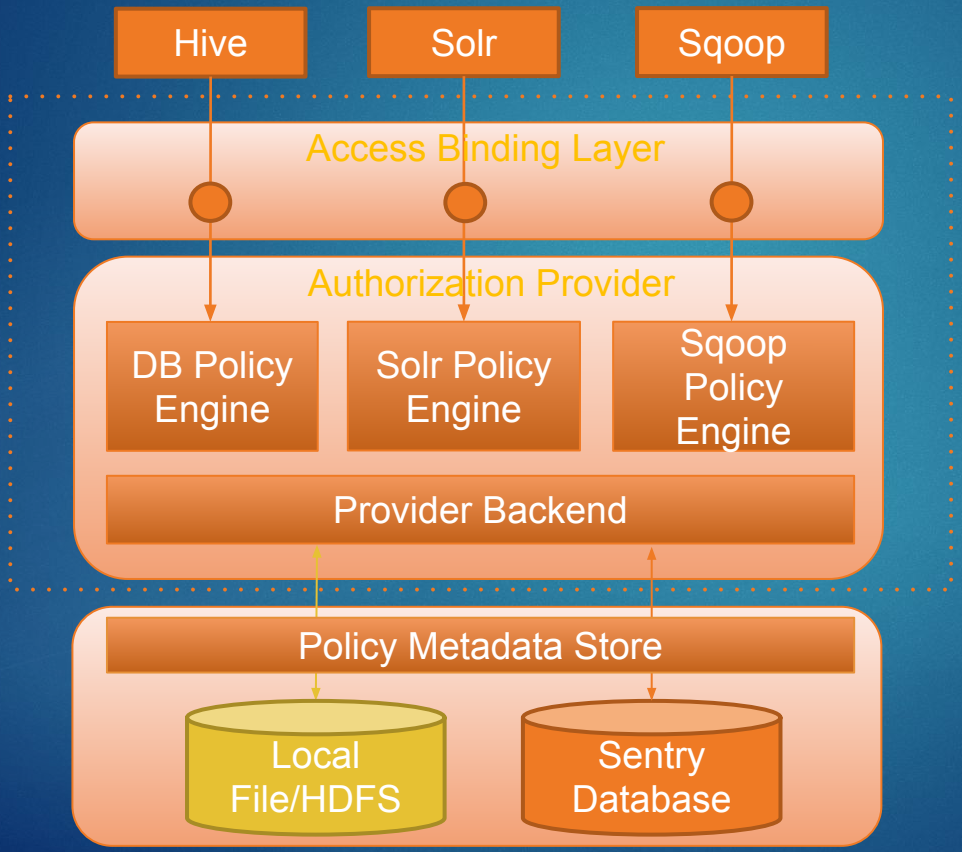
Sentry Architecture



cloudera

Sentry

Sentry Architecture

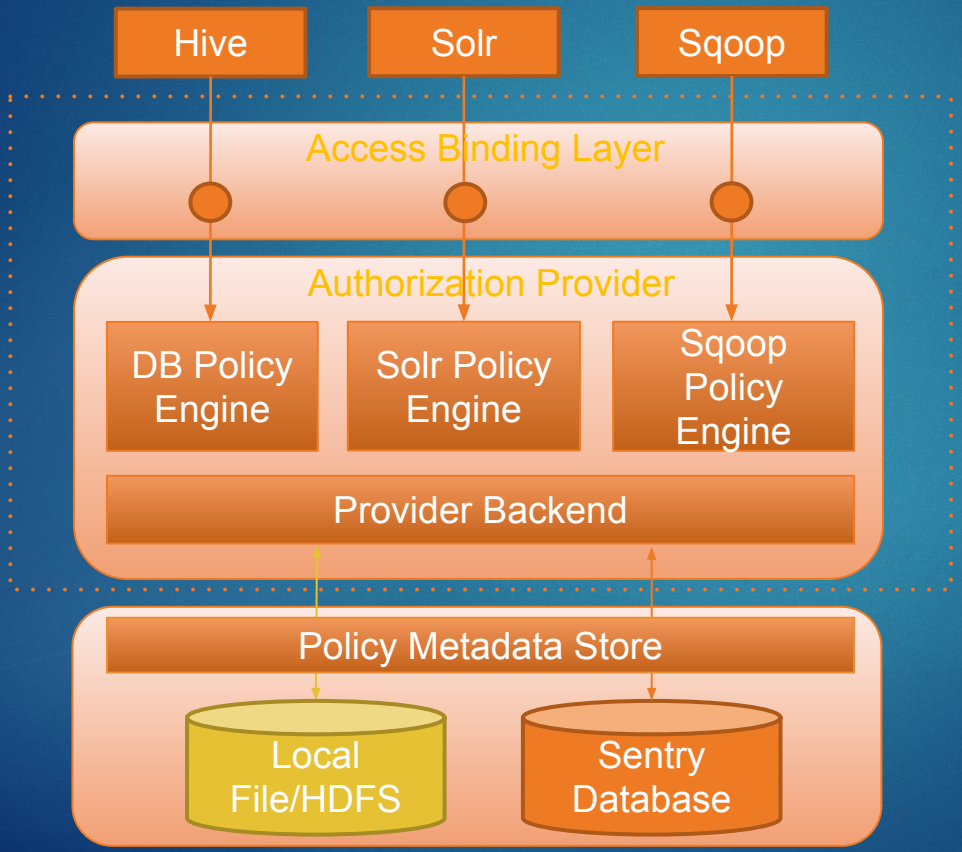


- **Binding Layer**: takes the authorization requests in the native format of requestors and converts that into a authz request based on the **authorization data model** that can be handled by Sentry authorization provider.

cloudera

Sentry

Sentry Architecture

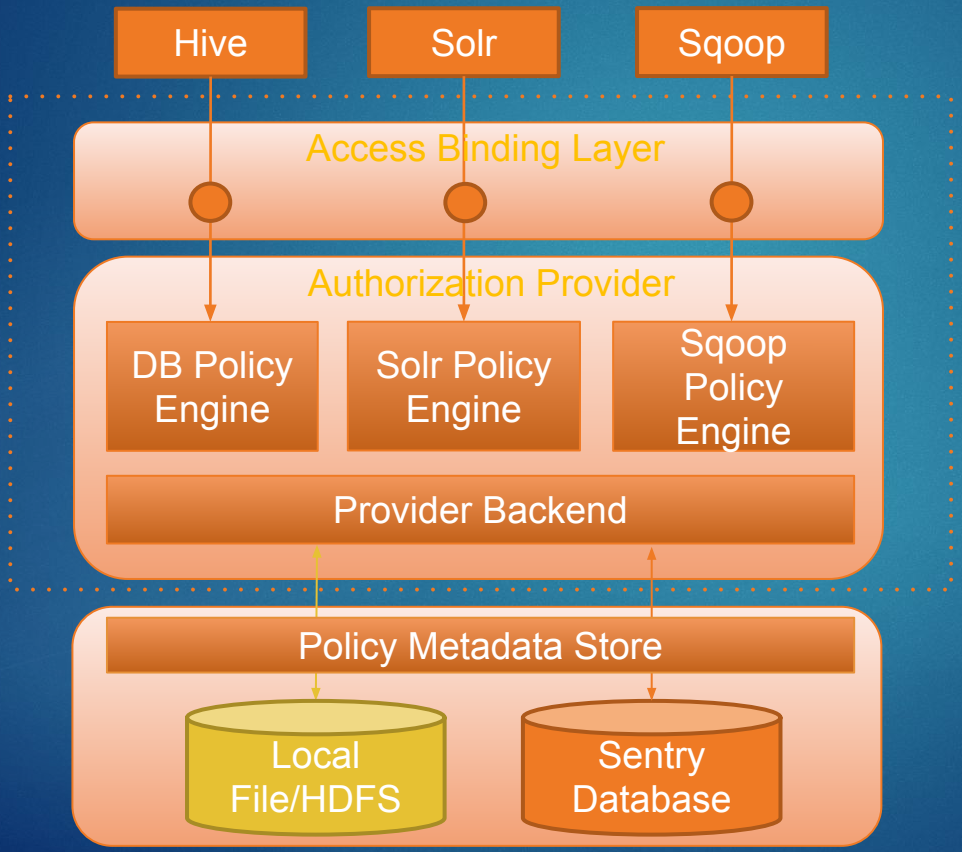


- **Authorization provider**: an abstraction for making the authorization decision for the authz request from binding layer. Currently, supplies a RBAC authorization model implementation.

cloudera

Sentry

Sentry Architecture

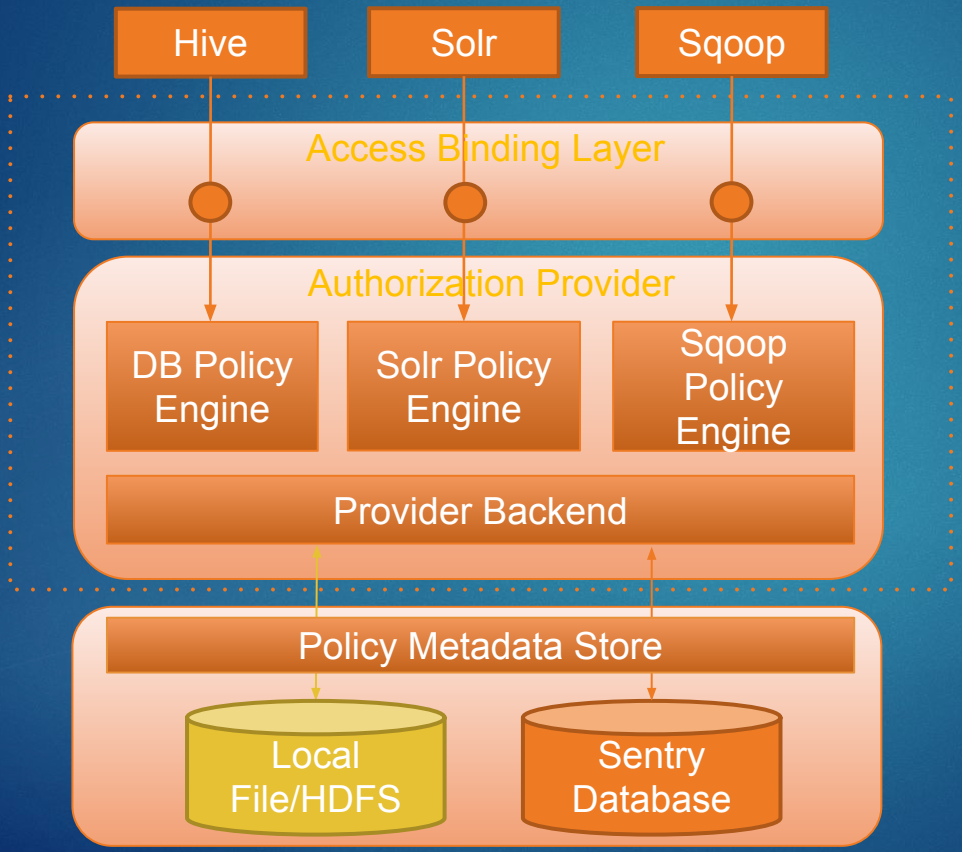


- **Policy Engine**: gets the requested privileges from the binding layer and the required privileges from the provider layer. It looks at the requested and required privileges and makes the decision whether the action should be allowed.

cloudera

Sentry

Sentry Architecture

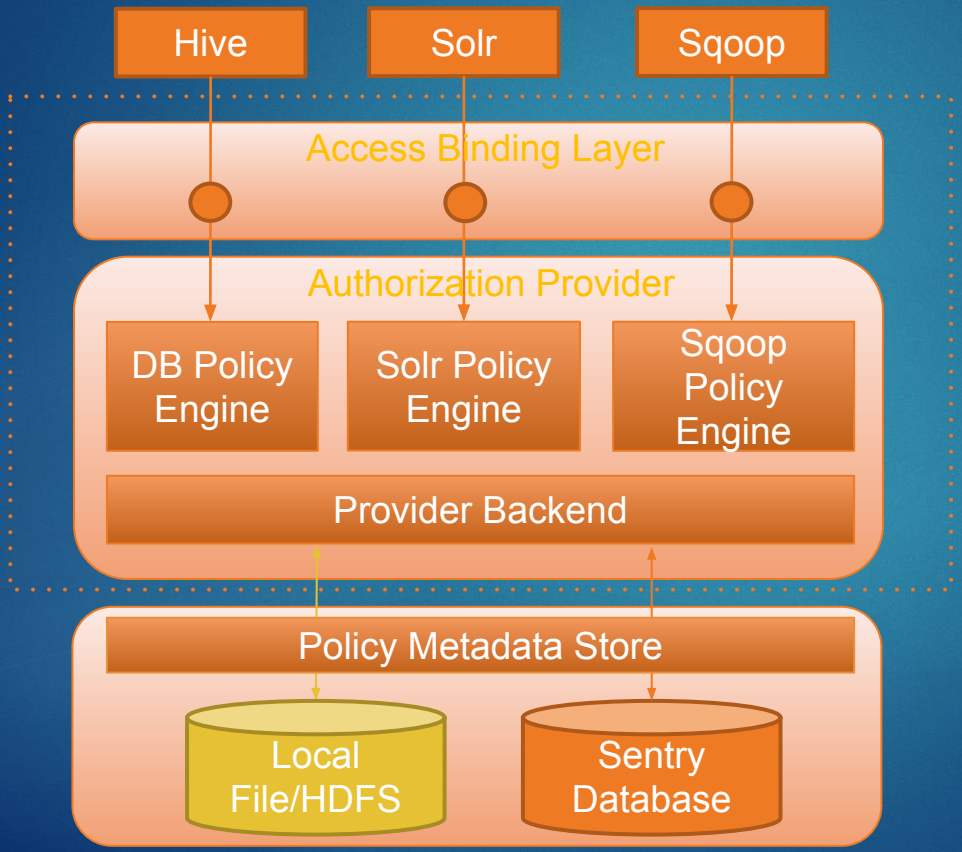


- **Policy Backend**: making the authorization metadata available for the policy engine. It allows the metadata to be pulled out of the underlying repository independent of the way that metadata is stored.

cloudera

Sentry

Sentry Architecture



- **Sentry policy store and Sentry Service**: persist the role to privilege and group to role mappings in an RDBMS and provide programmatic APIs to create, query, update and delete it. This enables various Sentry clients to retrieve and modify the privileges concurrently and securely.

cloudera

Sentry

Authorization Model for SQL Engines

- Provides out of box integration with Apache SQL engines
  - Apache Hive, Impala (incubating)
- Grant permissions to allow DDLs and DMLs on these objects
- A fixed metadata model designed to policies and objects suitable for SQL data
  - databases, tables, views and columns
- Specific Client plugin for SQL engines
  - DB policy engine
- If support other data applications, will need extensive development

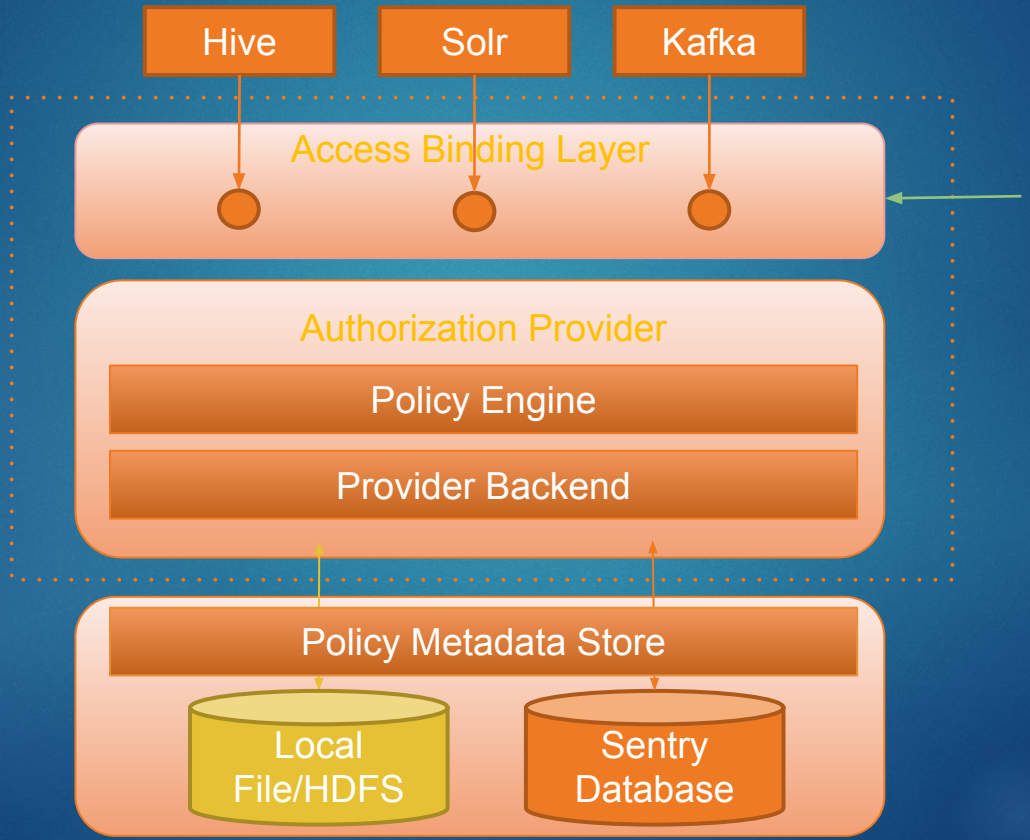
Generic Authorization Model

- Motivation
  - Supports various data applications out of box
  - Easy integration with any new components
    - Very few implementation
- A more **flexible** design
  - Generic authorization data model for defining sensitive resources
  - Generic policy engine that includes: access actions and privileges abstraction could be interpreted by various engines

cloudera

Sentry

Generic Authorization Model



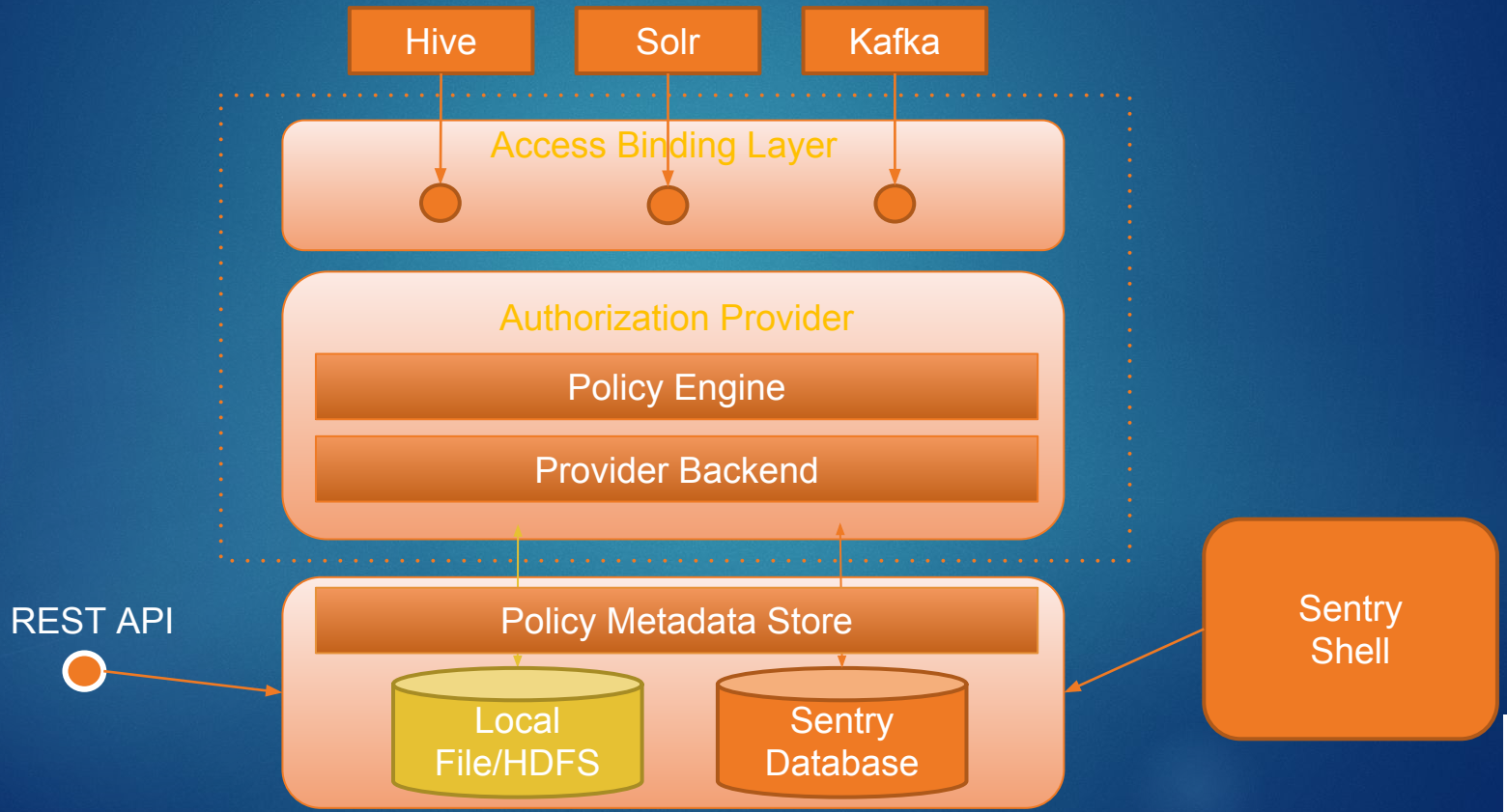
cloudera

Sentry

Server Client Model

- Thrift client APIs
  - Get privilege; Grant/Revoke role; Grant/Revoke privilege; List roles
- Provide SentryCLI to manage policies and metadata
  - sentryShell --grant_role_privilege --role analyst --privilege server=server1->db=db2->table=tab1->action=select --conf sentry-site.xml
- RESTful client APIs (Ongoing)
  - Use HTTP requests to manage roles and privileges

Generic Authorization Model



Authorization Data Model

- Authorization data model defines the objects to be subject to authorization rules, the granularity of actions to be allowed and privilege rules that allow access to objects.
  - In generic model, different data application objects are represented as resources. Actions and privileges can be transformed into read, write or all operations based on data application definition.
  - For example, SQL authorization data model: objects are databases, tables, views, uris or functions; privileges are create database, insert into table, create function using ‘func.jar’ and so on.

Authorization Data Model (cont…)

- Solr authorization data model: objects are indexes, collections, or documents; actions and privileges can be search through collections or documents, create index, list status of cores and so on.
- Sqoop authorization data model: objects are resources, servers, connectors, links or jobs; actions and privileges can be interpreted as connect to server, execute job.
- Kafka authorization data model: objects are clusters, topics or consumer groups; actions and privileges could be write on topic for producer, create on cluster for auto topic etc.

Integrating with Universal Authorization Model

- Define Authorization data:
  - Define authorizables as resources. For example, Solr:///collection=c1/field=f1; Hive:///database=db/table=tbl/column=cl.
  - Define actions and privilege rules. For example, Solr read on collection means user can search through collection. Hive read on database means user can show databases.
- Extend generic Plugin Engine:
  - Query Sentry to retrieve privileges, do the transformation, return rules which can be used to authorize access actions.
- Policy enforcement on data application side:
  - Implement hook and binding to enforce authorization by using Plugin Engine.

Integration Use Cases

- Authorizing Hive
- Authorizing Solr
- Authorizing Sqoop2

Authorizing Hive

- Define authorization data model:
  - Authorization objects: server, database, table, view, column and so on
  - Access actions: create, insert, select etc.
  - Privileges: create database, insert into table, select column
- Extend binding and hook to retrieve privileges from Sentry
  - HS2 integrate hook into build stage
  - HMS integrate hook into a pre metadata change event
- Extend metadata in the backend Db: privilege (including resources and actions). Resource could be defined as: database=db1->table=tbl1->column=c1
- Extend HS2 or HMS clients to manage Sentry metadata such as roles and privileges

cloudera

Sentry

Authorizing Solr

- Extend authorization data model:
  - Authorization objects: collections, documents, indexes or admin operations.
  - Privileges: query on collection, list status of cores, create index
- Extend hook and binding to retrieve privileges from Sentry:
  - Implement hooks to override request handler implementation
  - Hooks enforce collection or document level authorization
- Extend metadata in the backend Db: privilege (including resources and actions). Resource could be defined as collection=c1->field=f1
- SolrCli wraps around SentryCLI to manage metadata

Authorizing Sqoop2

- Extend authorization data model:
  - Authorize objects: server, connector, link and job
  - Access actions: show, read, write etc.
  - Privileges: connection read, link write, job read
- Extend binding and hook to retrieve privileges from Sentry
  - Client -> JobRequestHandler -> Authorize through Sentry -> JDBC Repo
- Extend metadata: privileges (including resources and actions). Resource could be defined as: server=server1->connector=conn
- Sqoop Cli is extended to manage Sentry metadata such as roles and privileges

Sentry Releases

- Successfully graduated from the Incubator in March of 2016 and now is a Top-Level Apache project
- Sentry 1.6.0 released on Feb 24, 2016
  - Add capability to export/import to dump or load Sentry metadata
  - Integrate Sqoop with Sentry by using generic authorization model
- About to release 1.7.0
  - Integrate Hive v2 into Sentry
  - Integrate Kafka with Sentry by using generic authorization model
  - Integrate Solr with Sentry by using generic authorization model

cloudera

Sentry

Future Work

- New design of Sentry HA
  - Compatible with HMS HA and HDFS ACLs Sync Up
- RESTful client APIs to define authorization data model
  - Continue work for generic authorization model
- Attribute Based Access Control

Reference

- Apache Sentry: https://cwiki.apache.org/confluence/display/SENTRY/Home
- Integrating with Sentry New Universal Authorization Model: https://cwiki.apache.org/confluence/display/SENTRY/Integrating+with+Sentry+New+Universal+Authorization+Model

cloudera

Sentry

Questions?