# Access Control Model for the Hadoop Ecosystem

Maanak Gupta, Farhan Patwa and Ravi Sandhu

Institute for Cyber Security and Department of Computer Science, University of Texas at San Antonio

## INTRODUCTION

- Apache **Hadoop** is an important framework for fault-tolerant and distributed storage and processing of Big Data. **Hadoop 2.x** core platform along with other open-source tools such as Apache **Hive**, **Storm**, **HBase** offer an ecosystem to enable users to fully harness Big Data potential.
- **Multi-Tenant Hadoop Data Lake** can jeopardize the confidentiality and integrity of data and cluster resources if they are not protected from nefarious actors.
- Apache **Ranger** and Apache **Sentry** provide access control capabilities to several ecosystem components by offering centralized policy administration and enforcement through plugins.
- In this work we discuss the access control model for **Hadoop ecosystem** (referred as **HeAC**) used by Apache Ranger (release 0.6) and Sentry (release 1.7.0) along with Hadoop 2.x native authorization capabilities. This multi-layer model provides several access enforcement points to restrict unauthorized users to cluster resources.

## AUTHORIZATION IN HADOOP ECOSYSTEM

- **Multi-layer** Authorization in Hadoop Ecosystem offers **Defense in Depth** approach.

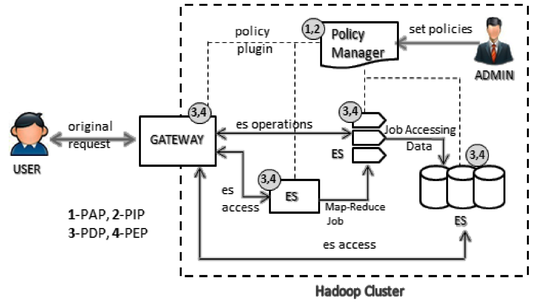


**Figure 1: Authorization Architecture for Hadoop Ecosystem**

- **Service Access:** The first layer of defense is provided by service level authorization which checks if a user or application is allowed to access the Hadoop ecosystem services and Hadoop core daemons.
- **Data and Objects Access:** Hadoop Distributed File System (HDFS) enforces POSIX style model and ACLs for setting permissions on files and directories holding data. Apache Hive requires columns, whereas Apache Kafka secures topic objects. Attribute values (called Tags) can be associated with objects to create Tag-based policies.
- **Cluster Resource and Applications Access:** In Hadoop 2.x, Apache YARN offers capacity (or fair) scheduler queues, which restrict cluster resources to authorized users. Each queue has associated ACLs which determine the set of users allowed to submit or modify applications inside the cluster.

## HADOOP ECOSYSTEM ACCESS CONTROL MODEL (HeAC)

- The model covers capabilities offered by Hadoop 2.x core and Apache Ranger and Sentry. Apache Ranger allows object permissions to user and groups, whereas Apache Sentry assigns permissions to roles, which are assigned to groups and through groups to member users.

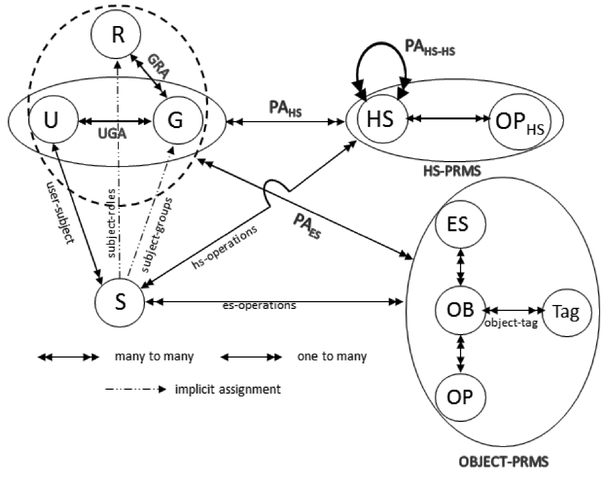


**Figure 2: The Conceptual HeAC Model**

- **Ecosystem Services (ES):** Set of services such as HDFS, Apache Hive, Apache HBase, Apache Kafka etc., which are used by users and applications to access the ecosystem objects.
- **Objects (OB):** Resources secured from unauthorized users. Apache Hive supports objects tables and databases whereas YARN has queue objects.
- **Operations (OP):** Set of actions which can be performed on objects by users. Hive tables support select operation, YARN queue support submit-application.
- **Object Tags (Tag):** Set of attribute values which can be associated to objects.
- **Hadoop Service (HS):** Set of daemon services such as HDFS NameNode, YARN ResourceManager.
- **Hadoop Operation ($OP_{HS}$):** Set of operations which can be performed on Hadoop services.

- As shown in Figure 2, there are two sets of permissions—Hadoop service permissions (HS-PRMS) and service object permissions (OBJECT-PRMS).
- OBJECT-PRMS can be set on object or object tags associated with objects (shown by object-tag).
- A subject S created by user will get all OBJECT-PRMS and HS-PRMS permissions of its creator user.

## PROPOSED EXTENSIONS

- We outline strategies to reformulate HeAC model to more acceptable and generalized access models:
  - **Role Based Model:** A pure RBAC can be implemented where permissions are assigned only to roles and user and groups are assigned to roles. This approach also presents a novel way to combine RBAC and object attributes (Tags) beyond NIST strategies.
- NIST proposed strategies for adding attributes to RBAC:
  - **Dynamic Roles:** This involves attributes of users and environment for user to role assignment. Policy rules are defined using policy language which includes attributes and corresponding roles.
  - **Attribute Centric:** This a pure attribute based approach where authorization policies comprising attributes are defined and access decision is made based on attributes of ecosystem services or objects and users.
  - **Role Centric:** In this approach, a user is assigned initial set of permissions through roles but these permissions are reduced based on attributes of entities. Filtering functions are defined using attributes based policies, which are checked to determine the final set of permissions of a user.

## REFERENCES

[1] Apache Hadoop. http://hadoop.apache.org/.
[2] Apache Knox. http://knox.apache.org/.
[3] Apache Ranger. http://ranger.apache.org/.
[4] Apache Sentry. http://sentry.apache.org/.
[5] Pietro Colombo and Elena Ferrari. 2015. Privacy aware access control for Big Data: a research roadmap. Big Data Research 2, 4 (2015), 145–154.
[6] Devaraj Das, Owen O'Malley, Sanjay Radia, and Kan Zhang. 2011. Adding security to Apache Hadoop. Hortonworks, IBM (2011).
[7] Maanak Gupta, Farhan Patwa, James Benson, and Ravi Sandhu. 2017. Multi-Layer Authorization Framework for a Representative Hadoop Ecosystem Deployment. In Proc. of ACM SACMAT (To appear). 8 Pages.
[8] Vincent C Hu, Tim Grance, David F Ferraiolo, and D Rick Kuhn. 2014. An access control scheme for Big Data processing. In Proc. of IEEE CollaborateCom. 1–7.
[9] Huseyin Ulusoy, Murat Kantarcioglu, Erman Pa.uk, and Kevin Hamlen. 2014. Vigiles: Fine-grained access control for MapReduce systems. In Proc. of IEEE Congress on Big Data. IEEE, 40–47.