ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 6, Issue 3, pp: (13-17), Month: July - September 2018, Available at: www.researchpublish.com

POSTER: ACCESS CONTROL MODEL FOR THE HADOOP ECOSYSTEM

J. Pavithra

Abstract: **Apache Hadoop is an important framework for fault-tolerant and distributed storage and processing of Big Data. Hadoop 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. Apache Ranger and Apache Sentry provide access control capabilities to several ecosystem components by offering centralized policy administration and enforcement through plug-in. 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. We further outline some preliminary approaches to extend the HeAC model consistent with widely accepted access control models.**

Keywords: **Access Control; Hadoop Ecosystem; Big Data; Data Lake; Role Based; Attributes; Groups Hierarchy; Object Tags.**

1. INTRODUCTION

It is highly anticipated that Big Data will significantly impact and transform human society in 21st century. Enormous and varied data sets are generated and collected from smart home devices, genomes, pace makers, satellites, social blogs, etc., to garner valuable insights useful in business and commerce. Such Big Data requires distributed, scalable and robust system with fast computational capabilities, wherein Apache Hadoop is a dominant open-source system. Hadoop 2.x core components (Hadoop Common, Hadoop Distributed File System (HDFS), Map Reduce and YARN) along with several Apache projects such as Hive, HBase, Storm, and Kafka, enable non-expert users to harness the potential hidden in the valuable data assets.

Hadoop data lake can jeopardize the confidentiality and integrity of data and cluster resources if they are not protected from nefarious actors. Distributed and massive scale of the platform results in broader attack surface making it vulnerable to security threats. For example, Hadoop service daemons such as HDFS NameNode, DataNode, and YARN Resource Manager, can be impersonated by malicious applications to communicate within the Hadoop cluster. Once these malicious services are registered, an attacker can easily access data blocks or modify other user applications running inside the cluster. An attacker can also execute denial of service attacks on the ecosystem by running highly intensive applications, exhausting cluster resources. Unencrypted RPC communication between DataNode and across daemon services can also open transit data to attacks. These threats can be manifested from external or inside user of an enterprise.

The basic requirements in securing the Hadoop cluster include protecting machines, services and network. It is can be done using network segmentation (physical or logical), firewalls, gateways, enabling SSL/TLS for secure communication between client-node and inter-nodes, intrusion detection and prevention systems, users or system validation, authorization rules etc. LDAP and Kerberos can be used for user and identity management to provide authentication. Users are authorized to access the service using ACLs (access control lists) via a single API gateway for all ecosystem services. Some strategies to secure data residing in HDFS are to use € le or OS encryption, tokenization, data masking, ACLs, and POSIX € le permissions. Data in motion can be secured by encrypted RPC or HTTP. Multi-tenant cluster also requires resources to be shared among several users. Hadoop 2.x includes YARN execution framework which schedules and

manages cluster resources. YARN capacity (or fair) scheduler uses queues to allow authorized users to submit, modify or view applications in cluster using ACLs. Node labels are associated with different queues to restrict applications to run on specific compute nodes. Delegation and block access tokens are used to pass user authentication or authorization information among Hadoop components. Also, proper log auditing mechanisms are required to identify possible attacks.

In this paper we primarily focus on access control and describe the multi-layer authorization capabilities currently offered by Hadoop ecosystem specific to Apache Ranger, Apache Sentry and Hadoop 2.x native access control features. The paper is organized as follows. Section 2 reviews some previous research work related to Hadoop and Big Data security. Section 3 presents the authorization model for Hadoop ecosystem (referred as HeAC) followed by some proposed ne grained HeAC extensions in Section 4. Section 5 summarizes our work.

2. RELATED WORKS

Hu et al proposed a generalized access control model for Big Data processing frameworks, which can be extended to Hadoop environment. The paper discusses trust chain among different entities to allow user access requests and to ingest data into the cluster. Security requirements and design in Hadoop system are discussed in several papers including. Fine-grained access policies for key-value pair in Big Data environment are proposed. Ulusoy et al proposed fine-grained access control for Map Reduce systems. Colombo et al presented a road map for achieving Big Data privacy.

3. ACCESS CONTROL MODEL

In the following subsection we summarize the authorization capabilities in Hadoop ecosystem provided by Hadoop 2.x, Apache Ranger and Sentry. We present the conceptual model for multilayer access control model for Hadoop ecosystem (HeAC) in the next subsection.

3.1 Authorization in Hadoop Ecosystem ‘

The first layer which acts as perimeter security, checks if a user is allowed to access ecosystem services and core Hadoop daemons inside the cluster before access to data and objects is layer, called service-level authorization, provides ACLs to enforce authorization for access to HDFS NameNode, DataNode, YARN Resource Manager etc. Besides user to service interaction, cross-service communication (for example, between HDFS DataNode and NameNode) is also controlled by this layer. Apache Knox offers a single point API gateway to control access to several Hadoop ecosystem services such as Apache Hive, HDFS, YARN Resource Manager etc., to authorized external users using ACLs.

Data level authorization involves extended ACLs and POSIX style permissions and directories in HDFS. Apache Ranger and Sentry also provide access control (using plug-in) for several data and service objects across different services. For example, Apache HBase supports HBase Table column family supporting read and write operations, while Apache Storm has topology object with read and write operations. Attribute values (called Tags) can be associated with objects and are used to create Tag-based policies. The policy created on tags will then control access to all objects associated with the object tags.

In Hadoop 2.x, Apache YARN offers capacity (or fair) scheduler queues, which restrict cluster resources to authorized users. These queues are allowed set of resources which can be used only by users submitting applications to queues. Each queue has associated ACLs which determine the set of users allowed to submit or modify applications inside the cluster. Applications can be also protected from unauthorized modification using configuration attached to each application. Cluster nodes can be assigned node labels which are associated with different capacity (or fair) scheduler queues to restrict users to run applications on set of nodes with particular node labels.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 6, Issue 3, pp: (13-17), Month: July - September 2018, Available at: www.researchpublish.com

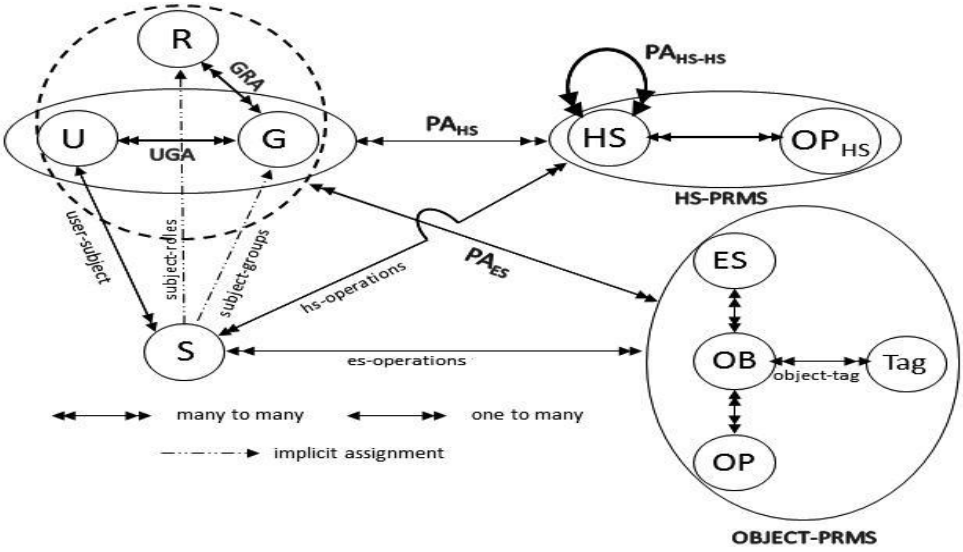

Figure 1: A Conceptual HeAC Model

3.2 HeAC Model Components

In this subsection we discuss the components of HeAC model as shown in Figure 1. Users (U), Subjects (S) and Roles (R) are the conventional access control components. Groups (G) are collections of users which have similar functional requirements. 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. Other components defined as sets in HeAC model are as follows.

• Ecosystem Services (ES): These are 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. Access to ecosystem service is required before the underlying objects are accessed.

• Objects (OB): Objects are resources secured from unauthorized users. Different services supports multiple objects with a many to many relation between them. For example, Apache Hive supports objects tables and databases whereas YARN has queue objects.

• Operations (OP): OP is the set of actions which can be performed on different objects by users. For example, Hive tables support select and create operations whereas YARN queue objects support submit-application and administer operations by authorized users.

• Object Tags (Tag): Tag is the set of attribute values which can be associated to objects. These values can define classification, content or any other attribute of objects. An object can be associated with multiple tags and vice-versa.

• Hadoop Service (HS): These are set of Hadoop 2.x daemon services used by users to submit jobs or to get status of submitted applications. Services such as HDFS NameNode, YARN Resource Manager also interact with each other for cluster resource or task updates. There are no objects associated with these services.

• Hadoop Operation (OP_{HS}): The set of operations which can be performed on Hadoop services. Most commonly such actions include access operation by user or by other Hadoop services.

As shown in Figure 1, there are two sets of permissions— Hadoop service permissions (HS-PRMS) and service object permissions (OBJECT-PRMS). HS-PRMS comprises operations on Hadoop services such as DataNode, NameNode, and can be assigned to both users and groups (reflected by PA_{HS}). On the other hand, OBJECT-PRMS comprises operations on objects in ecosystem services, and can be assigned to users, groups or roles as shown by PA_{ES}. Dotted circle encompassing U, G and R reflects PAES permission assignment. It should be noted that OBJECT-PRMS can be set on object or object tags associated with objects (shown by object-tag). Also, OBJECT-PRMS has ES service as a component, which specifies the need of service access before service objects are accessed. Interaction among different Hadoop

services, for example, DataNode accessing NameNode or ApplicationMaster communication to Resource Manager, is reflected by self loop shown by $PA_{HS\text{-}HS}$. These permissions are generally set by Hadoop 2.x native service-level authorization ACLs.

With group membership, effective HS-PRMS of a user will be the union of direct HS-PRMS and permissions of member groups. For OBJECT-PRMS, a user will have direct permissions, permissions of member groups and the permissions of roles to which its member groups are assigned. Based on operations, a user may need both types of permissions. A subject Screated by user will get all OBJECT-PRMS and HS-PRMS permissions of its creator user. It should be noted that we assume data is already ingested in the cluster and authorization model take effect when authenticated users try to access objects and services in the ecosystem.

4. PROPOSED HeAC EXTENSIONS

In this section, we outline strategies to reformulate HeAC model to more acceptable and generalized access models including roles and attributes, as follows.

• 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 to achieve € ner grained permissions can be also incorporated in HeAC. These extensions require introduction of attributes for users, groups, services and objects. Users can also get attributes from group membership. Below we discuss NIST strategies for combining roles and attributes, and corresponding HeAC extensions.

• Dynamic Roles: It is involves attributes of users and environment for user to role assignment. Policy rules are defined using policy language which includes attributes and corresponding roles. Similar permissions to roles assignment can be done based on object and service attributes present in the permission.

• Attribute Centric: It is a pure attribute based approach where authorization policies comprising attributes are de- fined and access decision is made based on attributes of ecosystem services or objects and users. Environment or contextual attributes can be added to access request to achieve finer grained permissions.

• 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.

5. SUMMARY AND CONCLUSION

In this paper we present a conceptual Hadoop ecosystem access control model HeAC based on Hadoop native authorization capabilities and model used by Apache Ranger and Sentry. ‘ is model assigns different sets of permissions to users either directly or via groups and roles. ‘ e model also discusses the concept of object tags which are used for defining object permissions besides using objects. We proposed some extensions to HeAC model in line with conventional access models and NIST defined strategies. We further plan to introduce RBAC model for Hadoop Ecosystem and present trust models for data ingestion and processing.

ACKNOWLEDGMENTS

Sincere gratitude is extended to James Benson, Technology Research Analyst at Institute for Cyber Security, UTSA, for his useful comments. This research is partially supported by NSF Grants CNS1111925, CNS-1423481, CNS-1538418, DoD ARL Grant W911NF-15- 1-0518 and by the Texas Sustainable Energy Research Institute at University of Texas at San Antonio.

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] Devaraj Das, Owen O'Malley, Sanjay Radia, and Kan Zhang. 2011. Adding security to Apache Hadoop. Hortonworks, IBM (2011).
[6] Maanak Gupta, Farhan Patwa, and Ravi Sandhu. 2017. Object-Tagged RBAC Model for the Hadoop Ecosystem. In Proc. of IFIP DBSec (To appear). Springer, 18 Pages.
[7] 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.
[8] Vincent C Hu, D Richard Kuhn, and David F Ferraiolo. 2015. Attribute-based access control. Computer 48, 2 (2015), 85–88.
[9] Xin Jin, Ravi Sandhu, and Ram Krishnan. 2012. RABAC: role-centric attribute based access control. In Proc. of MMM-ACNS. Springer, 84–96.
[10] D Richard Kuhn, Edward J Coyne, and Timothy R Weil. 2010. Adding attributes to role-based access control. IEEE Computer 43, 6 (2010), 79–81.
[11] Devda‹ a Kulkarni. 2013. A fine-grained access control model for key-value systems. In Proc. of ACM CODASPY. ACM, 161–164.
[12] Owen O'Malley, Kan Zhang, Sanjay Radia, Ram Marti, and Christopher Harrell. 2009. Hadoop security design. Yahoo, Inc., Tech. Rep (2009).
[13] Ravi S Sandhu, Edward J Coyne, Hal L Feinstein, and Charles E Youman. 1996. Role-based access control models. IEEE Computer 29, 2 (1996), 38–47.