Access Control Mechanisms in Big Data Processing

Yenumula B Reddy
Department of Computer Science
Grambling State University, Grambling, LA 71245, USA
ybreddy@gram.edu

ABSTRACT
Hadoop distributed file system (HDFS) must provide a distributed file system and MapReduce framework. The core components of HDFS are fault tolerant, high throughput, and files of arbitrary size. These components include the shared nothing architecture, a massive parallelism of tasks, and basic data structure is key/value pair. HDFS has shared multi-talent service and is used to store sensitive data. Currently, HDFS is used on private clusters behind firewalls and requires strong authentication and authorization (access control) to protect the sensitive private and public data. The HDFS job is partitioned and distributed on nodes for execution different from the node that the client authenticated and submitted the job. Further, job tasks from various users are executed on the same computer node and the system scales thousands of servers and performs many concurrent tasks. Therefore, the total performance path of the system requires authentication checks at multiple points. Kerberos authentication mechanism helps to meet the security requirements as a supplement to trusted users. Special access control mechanisms may require for high sensitive data to keep the hackers away. The proposed model helps the access control mechanism for high sensitive data in Big Data processing.

KEY WORDS
Hadoop, Big Data, security, MapReduce, Kerberos, authentication;

1. Introduction

Hadoop distributed file system (HDFS) is a Java-based file system that provides reliable and scalable data storage. The HDFS is designed to span large clusters of commodity servers. Since Hadoop systems are an ever-growing volume of enterprise data, security platform enables the enterprise customers to secure their data through encrypting or access control mechanisms. In Hadoop systems, centralizing key management enforces the access control policies and gathering security intelligence on data access. Excess access control mechanisms may slow down the response.

Hackers are increasing on networking from internal and external parts of organizations. It takes days or months to detect the problem and those affected are paying the price. The organizations ignoring or not properly controlled the access to their data sets are facing lawsuits, regulatory fines, and negative publicity. If the organization ignores the proper security checks, Big Data become a Big Price Tag.

Currently, the organizations required to make sure that the data analysts have permissions to see the sensitive data that they are analyzing. If an organization does not place proper controls, there is a chance of releasing unauthorized information that is related to personal records. Currently, Hadoop systems need additional security when using Hadoop. Due to the demand for Hadoop, security requirements are the ongoing problem and the new products require additional access controls to meet the customer satisfaction.

Security was not an issue during the initial development of Hadoop to manage a large amount of public data. The developers assumed that the clusters consisted of trusted machines managed by trusted users in an environment of confidence. Therefore, there was no privacy or security model. Although earlier distributions have authentication controls, they were easily circumvented through command line switch. Due to this reason, the checks were limited effective. The organization then added restricted access to authorized users. That is all authorized users (programmers and users) have the same level of access. Due to limited controls in Hadoop and MapReduce have no authorization, the mischievous users use to lower the priorities of other Hadoop jobs.

In initial Hadoop distribution, DataNodes do not enforce control. Due to this reason, the malicious user can read arbitrary DataNode blocks from DataNodes and bypass the access control restrictions or write the garbage data in dataNodes. Therefore, any user can submit a job to Job Tracker and could arbitrarily execute the job. The Hadoop community in Yahoo realized the security problem and enforced robust security controls through Kerberos as the authentication mechanism for Hadoop. The recent release of Hadoop accomplished the goal using following security goals.

- Kerberos remote procedure call (RPC) for mutual authentication of users, their processes, and services
- Web users implement their pluggable authentication called SPNEGO (Simple and Protected GSSAPI (Generic Security Service Application Program Interface) Negotiation Mechanism)

- Access control lists of users and groups on Hadoop files
- Delegation tokens and subsequent authentication checks
- NameNode takes access control decisions for Data blocks access
- Job tokens enforce task authentication to ensure that tasks could do only assigned tasks
- Enforce network level encryption to ensure quality of protection

Protection of data in Hadoop-based systems processing increased after the redesign of security in Hadoop models. Some of the later implemented Hadoop models have security as a layer over Hadoop. In spite of these additions to security, there are many security challenges to Hadoop Distributed File Systems (HDFS). The following are the common questions related HDFS security.

- How to enforce authentication for all types of users?
- How to enforce access control to ordinary and sensitive data over the existing access controlling policies?
- How to integrate Hadoop with enterprise security?
- How to encrypt data at the node, transfer, and network level?
- How to keep track of user access to data and control the unauthorized access?
- How to catch the malicious activities and punish such users?
- How the Kerberos servers handle the more than 100,000 concurrent jobs on large clusters?
- Can the user authentication at job submission be propagated for later use?

The technique like compression/decompression and encryption/deception at software level degrades the performance. Intel suggested using a hardware-based approach for such critical policies. The Hadoop security suggests that all HDFS clients required the authentication. Further, each task must run on submitting user and user's privileges. The DataNodes and TaskTrackers must ensure as part of the grid. These authentications are in addition to Kerberos authentication system.

The above discussion explains that Kerberos mechanism is good enough for most of the cases. However, to maintain the level of access to access high sensitive data, there is a need to add additional security called access control mechanism. The mechanism helps to control the access at every process level. The current paper motivates the need for access control mechanism for sensitive data in addition to Kerberos.

The rest of the paper discusses the recent developments, HDFS current status, Kerberos authentication mechanism, Proposed Security Model for HDFS sensitive data, and conclusions.

2. Review of Literature and recent Developments

The design of HDFS has the capability of storing a large amount of data to process parallel (does not mean it uses graphics processing units). Yahoo Hadoop stores personally identifiable information (sensitive data) [1, 2]. Therefore, it requires strong authentication and authorization. Further, the presentations [1, 2] discuss the privileges and identity of users. Similarly, Data Nodes and Task Trackers must authenticate themselves to ensure that they are part of the grid. Kerberos authentication services help the authentication for user services. Since Kerberos authentication requires additional access controls, Yahoo likes to introduce additional controls for their customer data.

Google executes hundreds of thousands of MapReduce jobs every day [3]. Users specify the computations in terms of map and reduce function. The MapReduce package automatically paralyzes the jobs and executes by using network facilities and produce results faster. The document did not discuss Kerberos security or any other security incorporated into their MapReduce jobs. Smith [4] discussed the Hadoop accomplishments and security challenges. The document explains the history of security, Kerberos-centric approach, and Big Data security challenges. O'Malley [5] explains the prevention of unauthorized HDFS access, integrated Kerberos in Hadoop, and authentication. Rapidminer presented the Big Data security on Hadoop [6]. The document describes the security implementation model in four steps: Perimeter security (authentication), data access security (authorization), accountability (auditing and data lineage), and data protection (encryption). To provide security RapidMiner Hadoop interfaces with Kerberos authentication server and Hadoop server. The combination provides security through request authentication – ticket granting and request service ticket – service session ticket. The document does not discuss the access level depending on the users accessing the data.

Sharma discusses a brief review of securing Big Data, issues, threats, and solutions [7]. The authors presented three varieties of Big Data that include data velocity, data variety, and data volume. The paper includes related threat areas and analysis of security solution through a table. Thuraisingham discussed the security issues and policies in federated database systems [8]. The paper describes the federated database as autonomous, cooperative, distributed, and heterogeneous. The security policies to enforce include local, generic, component, export and federated.

Access controls and their applications as part of security mechanisms are important. Jonscher and Dittrich [9] explained the access control mechanisms as part of global layer and the way of mapping these mechanisms in component databases. Tari and Fernadez proposed a unified security model incorporated in object kernel [10].

They proposed that the model can integrate into existing control models. To work with better existing procedures in Hadoop systems, the authors introduced task agents within database agents. The proposed model expected to provide security to Hadoop users. However, none of these authors discussed the access control to data depending upon the type of user accessing the data.

Ebmayr et al. examined the survey of database security and taxonomy of major security design [11]. The design issues include authorization, access control, and security mechanisms in federated databases. Neuman described the security in federated systems and reasons for failure [12]. The authors highlighted that the future technology depends upon federated systems and security boundaries must be identified to such environment. Gamble [13] discussed the data recovery from failure and the policies and Oleskeer proposed the data policies and law enforcement in Hadoop systems [14].

Roy et al. presented "Airavat", a MapReduce-based prototype system that provides strong security and privacy for sensitive data [15]. The proposed method uses complicated process in specific parameters. The parametric incorporation for security may not be enough to provide strong security for sensitive data. Ravi [16] used encryption and authorization. Ravi suggested that Kerberos security mechanism provides better security in federated systems. Chary et al. [17] presented security implications in HDFS. The research reviews the security in distributed systems and Kerberos security mechanisms.

Reddy [18] discussed the access control mechanisms currently available in HDFS. The research further discusses the requirements of access control mechanisms and change of access controls depending upon the user's access to data. The proposed model extends the Kerberos mechanism. Sameer et al. [19] proposed the snapshots enable system administrator's data recovery from failure. They proposed the snapshot solution for HDFS. They claimed that the proposed snapshot data structure addresses the inherent challenges related to data replication and distribution in Hadoop systems. Hediyeh et al. discussed the comparison study of inverted indexed in Hadoop [20]. The paper discussed three different indexes (Indexer, IndexerCombiner, and IndexerMap) in Hadoop environment and evaluated their impact on data format and output file format in MapReduce.

Study of about literature shows that there is a need for additional access control on authenticated users. The users may be inside an organization or outside the organization. Hackers can be inside or outside of the organization.

3. HDFS – Current Status and Process User Quarries

The firms are growing in using Hadoop and related technologies including Hive, Pig, and Hbase. These organizations using Hadoop to analyze their data in other ways cannot be done using traditional methods. Large credit card banks including Chase, J. P. Morgan, Bank of America, American Express, and other banks are using Hadoop to improve fraud detection and comprehensive view of the customers. Analysts of IT operations advised to be aware of potential security risks. Further, aggregating and storing the data from multiple sources creates more problems related to management and access control as well as ownership.

Aggregating data into one environment increases the data theft and accidental disclosure. The Hadoop environment must include security sensitivities, appropriate security controls, and role-based data access. The most effective approach is encrypting the data at the individual record level during the transit in Hadoop environment. Today most of the Hadoop-based systems include access control lists of Kerberos. Since Kerberos access control mechanisms are not sufficient, the organizations are trying to incorporate additional controls to secure the data.

The Hadoop design includes master-slave shared nothing architecture. The execution engine name is MapReduce. Figure 1 shows the simple architecture of Hadoop. The HDFS configuration consists of thousands of server machines and each storing part of the file system data. Each data block replicates many times. HDFS design includes the automatic recovery from failure or fault occurs. HDFS analysis takes three phases called mapper, shuffle, and sort, and reduces. Figure 2 shows the analysis process. The MapReduce engine receives the user job, decides on how many tasks runs (number of mappers), and where to run. Figure 3 shows the example of MapReduce function to analyze and output the number of times each word occurs. Figure 4 displays the shuffle sort of MapReduce function.

Authorization, authentication, encryption of data at the network node, encryption at transit, and audit trials are important in databases transactions. MapReduce does not have encryption at the node. The remaining functionalities are available in MapReduce. However, HDFS has all functionalities. It is necessary to identify these functionalities before starting the analysis of Big data using HBase, Hive, Oozie, and Zookeeper. Different models use their security solutions. The track of application to MapReduce involves the NameNode and DataNode. Files and directories reside on NameNode. The NameNode maintains the namespace tree and mapping of blocks to DataNodes. In DataNode, each block represents two files. The first file contains the data and the second file contains records of block's metadata.

User application accesses the file system using the HDFS client. The application reads the file, and HDFS asks the NameNode for the list of DataNodes. The list is sorted by network topology. Now, client contacts DataNode and request the needed block. The client maintains the read block and writes block through the pipeline. The HDFS client requests a new block after filling the current block. The hacker may modify these file blocks, without appropriate security. Kerberos security token does not check at each step. If the data is sensitive, an extra level of security must be maintained.

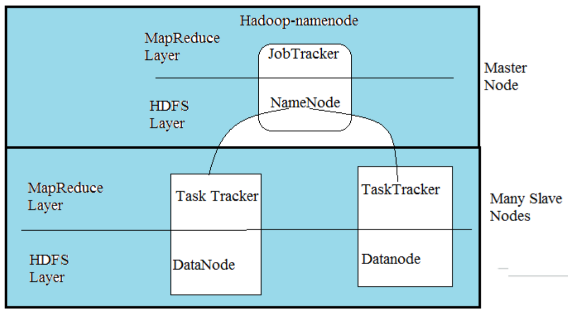

Figure 1: Hadoop Architecture

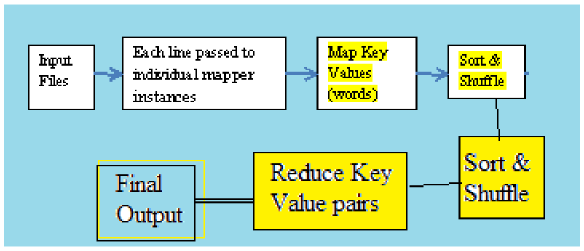

Figure 2: Hadoop Analysis Process

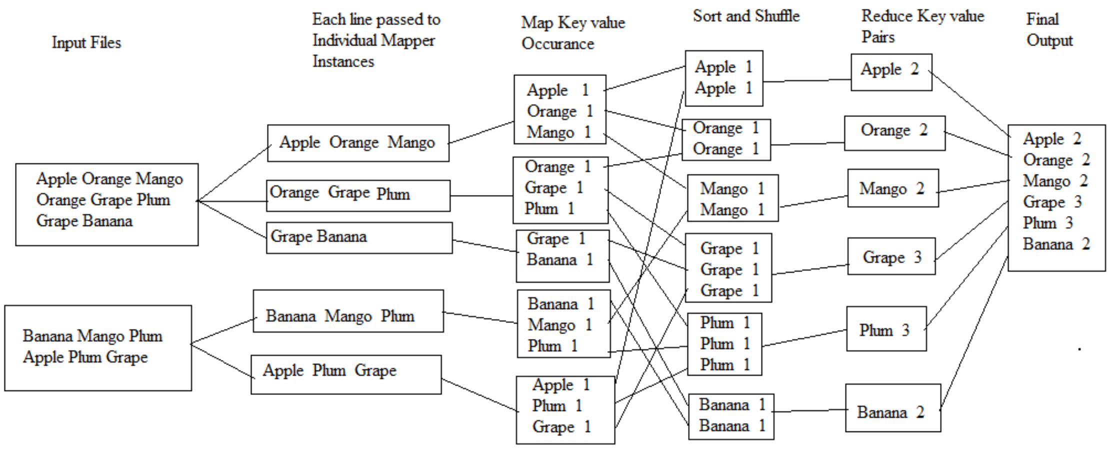

Figure 3: Number of times each word occurs in MapReduce

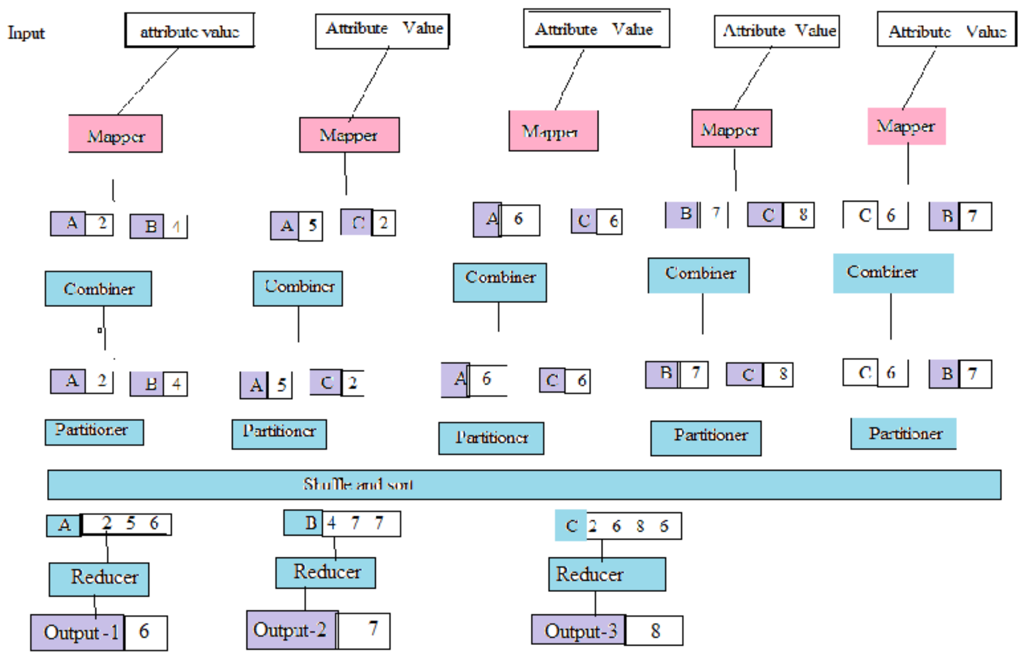

Figure 4: Shuffle sort Example

4. Kerberos authentication mechanism

Kerberos protocol is developed at MIT in the mid-80s. It is a network authentication protocol. It is available as open source or in supported commercial software. Kerberos remote procedure call (RPC) authentication mechanism is used for users' applications, authentication, and Hadoop services. In Kerberos, the user requests for a ticket from authentication server and server sends a ticket and encrypted request to the application server. The tickets are limited duration. The other widely used mechanism is Transport Layer Security protocol (TLS) and Secure Socket Layer (SSL) protocol.

In the TLS/SSL authentication process, the client sends request and server responds after authentication by itself. Before dialog ends secure communication begins, the client and server exchange dialog of session keys. In TLS/SSL client confirm the validity of a server's credentials with a trusted root certification authority. In TLS/SSL, users do not need to establish accounts before they create a secure connection. TLS/SSL provides strong authentication, message privacy, and integrity. It has interoperability, algorithm flexibility, ease of deployment and ease of use. These are general purpose protocols and can be used wherever authentication and data protection are needed. They depend on the operating system, TCP/IP network connectivity, active directory domain, and trusted certificate authorities.

Kerberos has benefits over TLS/SSL. It has better performance due to symmetric key operations and simpler user management. Kerberos primary authentication in Hadoop has Delegation token, block access token, and job token. The delegation token identifies the user at submission and later execution time to a particular HDFS or MapReduce services. Block access token provides read/write permissions to a particular HDFS block. Job token identifies for MapReduce job.

Currently, Kerberos uses delegation token, block access token, and job token. Delegation token authenticate a particular HDFS or MapReduce service at the time of job submission. Block access token provides read/write access to a particular HDFS block. That means DataNodes do not have any access control list. The job token identifies MapReduce job. Therefore, DataNode is the weak point for access control. The delegation token has the following information.

tokenID = {ownerID, renewerID, issueDate, maxDate, sequenceNumber}

tokenAuthenticator = HMAC-SHAI (masterkey, tokenID)
Token = {tokenID, tokenAuthenticator}

The NameNode has the list of a user to access a particular block. The access capabilities include read, write, copy, or replace. Besides, sensitive information in a block requires the level of access (read or write, copy, or replace or the combination). Since the NameNode shares the key information from DataNode, the user has following identification information at the block.

tokenID = {expiration, keygen, owner, block, access}
tokenAuthenticator = HMAC (masterkey, tokenID)
Token = {tokenID, tokenAuthenticator}

The block location information includes the NameNode capabilities. The DataNode verifies the capabilities to read data block before it goes to DataNode. In HDFS files, the capabilities can be reused by clients at any time after issue. The file permissions may change at any time. In such cases, it is required to revocation the capabilities. The capabilities issued by NameNode have expiration data. Integrating Kerberos protocol is a difficult task and requires more efficient design.

5. Proposed Model

Most of the Big Data processing projects can work with Kerberos security mechanism or TLS/SSL authentication process. Incorporating of these access controls into working model is a difficult task. Large companies like Google, Yahoo, IBM, Amazon.com and Best Buy can afford to implement these models. For these businesses, most of the data is precious and sensitive. They cannot afford to lose the data or unauthorized activity. Sensitive data requires extra access control at MapReduce and DataNode level.

Hadoop users are four types. The first types of users are simple. They store, access, process and transfer the data. The second type acquires the information (Google and Yahoo) for business improvement. The third type uses the information for research purposes. The fourth type is dangerous. They try to access the authorized and unauthorized data and misuse the data. Keeping in view of the fourth type of users, we propose a new security control on accessing the sensitive data.

Further, the hackers can be internal or external to the organization. Internal hackers are more dangerous than external hackers. The external hackers can be identified at various stages. For internal hackers, an extra security is required to protect the sensitive data to avoid the unexpected damage or harm to innocent customers (users).

The NameNode has the list of a user to access a particular block. The access capabilities include read, write, copy, or replace. Besides, sensitive information in a block requires the level of access (read or write, copy, or replace or the combination). Since the NameNode shares the key information from DataNode, the equations (1 – 3) provides the identification information to Hadoop user.

Let us assume the Kerberos delegation token is active for 8 hours and can be renewed for a week. In general, the token is active till the job is complete. The execution process allows the user to access a complete block of information. It cannot restrict limited access to information block. If we incorporate the access limits to each user, the user has a copy or read or write or append or modify or a combination of these accesses. Also, the history must be stored for each access to the data block, and inform the malicious activity to the owner.

Access rights to sensitive information help limit the unauthorized access (type 4 users). Further, verification requires at shuffling and reducing phases. Rewrite the equations (1) with access rights.

tokenID = {ownerID-Ac, renewerID, issueDate, maxDate, sequenceNumber} (4)

The parameter ownerID-Ac includes the user access rights to data blocks. The ownerID-Ac formatted as below.

ownerID-Ac = {ownerID, accessLimit, accessTime}

AccessLimit is limited read/write/execute, modify. AccessTime is the time of Access to the database. Much sensitive information limits to terminals. Let us take university BannerWeb. Students grades and student information is sensitive. Let us assume the grades for discussion. Each faculty member has access to submit final grades and no one modifies (including registrar or President) without faculty member permission. The faculty member can enter only particular dates provided by the university and no modifications allowed after the scheduled dates. If anyone modifies the grade (s) the system automatically tag and alarm (inform) the faculty. Faculty grade access limits and controls are the typical examples of access control.

Access controls and key management, auditing, policy management, and distributed authentication are critical in federated databases. The design of access control depends upon the environment and sensitivity of the information. Additional controls to Kerberos helps for sensitive information. If the access controls are incorporated in bannerWeb database, the bannerWeb automatically sends the message related to the modification to the respective faculty. Table 1 shows the modification note to the respective persons.

Detecting the internal hacker is easy with access controls. If an authorized internal hacker modifies the grade during office time or after office time (scheduled time slot or not) the access control mechanism detects the hacker. The simple modification in access controls helps the university. The alarm helps the faculty member about the unscheduled and unauthorized modification.

Table 1: Grade change note to faculty

Modifier	Current Status	Grade Changed to	Action
Registrar office	C	B	Alarm to faculty, Dean, Vice president for Academic Affairs
Hacker	F	A	Alarm to faculty,

			Registrar, Dean, Vice president for Academic Affairs
Internal Hacker	C	A	Alarm to faculty, Registrar, Dean, Vice president for Academic Affairs

6. Controlling the Hackers Using Object Function

We define the objective function G with four parameters N, A, D, and U. The hacker (internal or external) alarmed to the respective office member for all malicious activities. We first define the function and then present an algorithm to catch the hacker. The objective function is defined as below.

$$G = \{N, A, D, U\} \qquad (5)$$

where

$n_i \in N$ (Set of users);

$a_i \in A$ (Set of access rights);

$d_i \in D$ (Set of allowed resources in the database);

$u_i \in U$ (The return result of the user query)

Each user accessing the bannerWeb grade system has defined access rights. The timings of scheduled dates are clearly defined.

If the user $n_i\,(n_i \in N)$ with access right $a_i\ (a_i \in A)$ authorize to modify student grade, the system generates the correspondence as in Table 1. The senior officials and related faculty know the change. The process helps the unauthorized modification. The grade modification is a quarry matching process. Algorithm 1 contributes to detect the intruder.

Algorithm 1

Step 1: Let $Q(n_i, a_i, d_i, u_i)$ be the query to modify the grade. The query token matches the user access rights and allows the user to change the grade with appropriate login.

Step 2: If the user is an internal user, and grade change official, the system allows the grade change and informs all officials as in Table 1.

- If the user is an internal hacker with appropriate access rights, grade change accepted and informed all officials as in Table 1. The internal hacker will be detected. If the internal hacker attempts more than once, the history will be recorded and informs the officials. After three attempts, the system locks for the user.
- If the user is an external hacker, each attempt is recorded and alarmed to officials and security.
- If the external hacker is successful, the history is stored, and officials are informed of the action.

Step 3: Hacker attempts are from outside terminal environment. The system alarms the security and misleads the hacker.

The process controls the malicious activities in the grading system. Similarly, the algorithm can be used in business, hospitals, and any Web-based information systems.

7. Conclusions and Future Research

The research establishes the current status of security mechanisms in Hadoop. The security mechanisms used in most of the Hadoop systems are Kerberos and TLS/SSL. The protocols provide most of the security controls and authorizations. Additions level of access controls helps the organization to minimize and avoid unauthorized modifications and access to sensitive information. The Kerberos is more recommended due to its additional advantage in Hadoop. Many organizations prefer TLS/SSL security mechanism. Incorporation of the proposed model in Kerberos or TLS/SSL helps better to control the hackers. An example of bannerWeb is provided to show the need for access control. The bannerWeb example is not a Hadoop-based system. However, the example helps to understand the access control and to hack the database. The Algorithm provided in section 6 is for access control model provided in section 6 to detect and alarm the intruder activity.

Acknowledgements

References

[1] D. Das, O. O'Malley, S. Radia & K. Zang., "Adding Security to Apache Hadoop", *Horttonworks Technical Report I,* www.hortonworks.com.

[2] O. O'Malley., "*Hadoop Security Architecture",* Slide share, Published on Jan 27, 2014.

[3] R. Lammel., "Google's MapReduce Programming Model – Revisited", Science of Computer Programming 70: 1. doi:10.1016/j.scico.2007.07.001

[4] K. T. Smith., "Big Data Security: The Evolution of Hadoop's Security Model", http://www.infoq.com/articles/HadoopSecurityModel , Aug 14, 2013.

[5] O. O'Malley., "Integrating Kerberos into Apache Hadoop", Kerberos Conference, 2010

[6] T. Malbrecht and Z. Prekopcsak., "Big Data Security on Hadoop", RapidMiner Orange Paper, https://rapidminer.com/wp-content/uploads/2015/02/Big-Data-Security-on-Hadoop.pdf, February, 2015

[7] P. Sharma and C. Navdeti., "Securing Big Data Hadoop: A Review of Security Issues, Threats, and Solution", International Journal of computer science and information technologies, Vol. 5(2), 2014, 2126-2131.

[8] B. Thuraisingham., "Security issues for federated database systems", Computers & Security, 13 (1994), pp. 509-525.

[9] D. Jonscher. and K. R. Dittrich., An Approach for Building Secure Database Federations, VLDB '94 Proceedings of the 20th International Conference on Very Large Data Base, Morgan Kaufmann Publishers Inc. (1994), pp. 24-35.

[10] Z. Tari. and G. Fernandez., "Security Enforcement in the DOK Federated Database System", Pierangela Samarati, Ravi S. Sandhu (Eds.): Database Security Volume X, Status and Prospects, IFIP TC11 / WG11.3 Tenth International Conference on Database Security, 22-24 July 1996, Como, Italy. IFIP Conference Proceedings 79 Chapman & Hall 1997, ISBN 0-412-80820-X, pp. 23-42W.

[11] Ebmayr., F. Kastner., G. Pernul., S. Preishuber., and A. M. Tjoa, (1995) Access Controls for Federated Database Environments. Proc. Joint IFIP TC 6 and TC 11 Working Conf. on Comm. and Multimedia Security, Graz, Austria.

[12] C Neuman., "Why Security fails in Federated Systems", Univ. of Southern CA, 7 March 2012, http://csse.usc.edu/csse/event/2012/ARR/presentations/20120307-neuman-csse.pdf .

[13] G. Gamble., "Data Recovery Solutions", R3 Data Recovery, [accessed: October 2012].

[14] A. Olesker., "White paper: Big Data Solutions For Law Enforcement", June 2012, http://ctolabs.com/wp-content/uploads/2012/06/120627HadoopForLawEnforcement.pdf [accessed: January 2013].

[15] I. Roy Srinath, T.V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel, "Airavat: Security and Privacy for MapReduce", 7th USENIX conference on Networked systems design and implementation (NSDI'10), 2010, Berkeley, CA.

[16] P. Ravi, "Security for Big Data in Hadoop", http://ravistechblog.wordpress.com/tag/Hadoop-security/, April 15, 2013 [Retrieved - April 2013].

[17] N. Chary., Siddalinga K M., and Rahman., "Security Implementation in Hadoop", http://search.iiit.ac.in/cloud/presentations/28.pdf [accessed: January 2013].

[18] Y. B. Reddy., "Access Control for Sensitive Data in Hadoop Distributed File Systems", Third International Conference on Advanced Communications and Computation, INFOCOMP 2013, November 17 - 22, 2013 - Lisbon, Portugal.

[19] S. Agarwal., D. Borthakur., and I. Stoica., "Snapshots in Hadoop Distributed File System", UC Berkeley Technical Report, UCB/EECS, 2011.

[20] H. Baban, S. Makki, and S. Andrei: "Comparison of Different Implementation of Inverted Indexes in Hadoop", The 2nd Int. Conf. on E-Technologies and Business on the Web (EBW2014), March 2014, pp. 52-58.