Insights to Hadoop Security Threats

Peipei Wang, Anwesha Das
Department of Computer Science
North Carolina State University
{adas4, pwang7}@ncsu.edu

Abstract

Hadoop is an important framework that allows distributed processing. It is now widely used for processing large amounts of data. While lot of work on Hadoop has focused on improving its efficiency, people are gradually paying attention to its security concerns and building security modules for Hadoop. However, there is no evaluation existing for these Hadoop security modules, currently. In this paper, we take initiative to look at Hadoop modules for security purposes in both open source Hadoop and commercial Hadoop versions. We choose Hadoop authentication, authorization, and congestion control modules, namely kerberos, oozie, and rate-limiting as the three aspects to evaluate Hadoop security modules.

We describe the setup and configuration of security enabled hadoop cluster and discuss tests and experiments conducted to try attacks. In addition to the cumbersome setup it is challenging to figure out the attacks and impersonation scope through the exploration of security implementation. Kerberos, oozie, and rate-limiting has been studied to get an idea of how complex it is, to enable security in hadoop and estimate the scope of threats. Based on our study we acknowledge kerberos's efficacy in correct user authentication, but its extremely cumbersome deployment coupled with performance degradation may make it potentially unusable for real-world clusters.

1 Introduction

Hadoop has been extensively used in the industry and academia for research purposes. There are startups coming up which focus on providing hadoop as a service. Moreover there are third party vendors like Cloudera, Hortonworks, Intel, EMC [5] which provide Apache hadoop software, support and services. In this paper we explore hadoop along the security perspective. Hadoop's *security* module has been comparatively recent. There exists no structured evaluation of security vulnerabilities in hadoop considering the security enabled module. Recent advances in distributed computing has brought in limelight hadoop infrastructure for voluminous computing. [39, 38] particularly discuss about the new security framework of map reduce implementations. Several other work such as [32, 27] throw some light on the security issues with Hadoop specific HDFS- hadoop distributed file system. Although a huge body of work exists in this space discussing about the security vulnerabilities there exists no structured evaluation or insights into the recent provision of security framework with hadoop. These didn't come bundled with the older versions of hadoop, since the first priority of hadoop is efficient voluminous computation and then comes security. Since, security seemed to be luxury rather than necessity, not much hindrance have been paid until recently. As a consequence more mental efforts have been spent in working on the HDFS functionality rather than the security breaches which can be a concern when multiple users using multiple nodes are accessing data in a common HDFS cluster.

In this context when we talk of security we mainly refer to the security of the nodes which are a part of a distributed system or a cluster. In other words protection of sensitive data from unauthorized access as well as user authentication when accessing the HDFS cluster are our key concerns. There are three primary areas which has been implemented in terms of the security framework, the *kerberos* authentication mechanism which is a network authentication protocol, the *token based protocol* for data access and other miscellaneous protocol like *RPC* and *data transfer* protocol. The primary challenge of enabling security in hadoop is to deal with access permissions of data blocks which goes via the name node and preventing impersonation where a user can misbehave and emulate another user to have access on other's data in the hdfs cluster. Another challenge is to deal with the trade-offs between hadoop cluster scalability

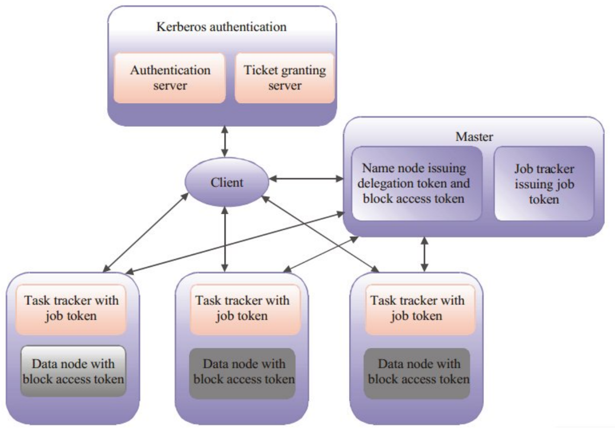

Kerberos authentication
Authentication server
Ticket granting server
Master
Name node issuing delegation token and block access token
Job tracker issuing job token
Client
Task tracker with job token
Data node with block access token
Task tracker with job token
Data node with block access token
Task tracker with job token
Data node with block access token

Figure 1: Security Framework

and prevention of performance degradation. Our study reveals that secure deployment of hadoop deteriorates performance particularly virtual machines in a cluster making it considerably slower.

The overall architecture of security mechanism is shown in Figure-1. Works like [6, 11, 10, 2] talk about the existing security status of hadoop and its possible future trends.

As seen in Figure-1, we have the kerberos authentication module which has the authentication server and the ticket granting server. There exists three types of tokens for HDFS access namely: delegation token, block access token and job token. We will discuss them in details in the next section. There is communication happening between the client, the namenode and multiple datanodes which exposes the possibility of security breach in an unencrypted channel. Based on the security infrastructure following are some of the possibility of attack scenarios:

- Data nodes do not enforce any specific block access control. Block access tokens are issued by the Name node, however they are shared across all data nodes. Anyone with block_id can have access to data blocks. There can be a possibility that an attacker with a shared key can affect the data nodes and in turn affect integrity.

- Usually insecure channels are used for communication. Job tokens are issued by the job tracker. An intruder may get access to the job tokens in an unencrypted insecure channel and hence modify the results of map or reduce tasks.

2 Background

Here we discuss the basic protocols which are implemented in the current security framework. As mentioned earlier, the primary components are Kerberos authentication protocol, Token based access control and other security protocols. We talk about them briefly so that it can help understand the hadoop security attack context.

1. Kerberos [29, 31, 24] is a network based authentication mechanism which provides security across a hostile network. It mainly consists of the *KDC* key distribution center which is a centralized authentication service. It works in a *realm* which helps with the fully qualified domain name of the hosts. It is a popular third party commercial software being used with many other products which needs authentication. Usually kerberos works in a time synchronized manner. It contains a database which contains the principle identifiers with their secret keys. The main idea is as follows:

    - The host requests access at KDC. The authentication server at the KDC looks up for its identification in the local database and accordingly grants ticket.
    - The namenode checks for the details and if the client is valid accordingly issues the ticket to the client. The client and datanode exchange shared key for message exchange.
    - The client then sends the ticket along with the shared key to the data node for further communication.

2. Tokens - Tokens are used to control access to the jobs and data blocks inside HDFS. There are three primary types of tokens.

    - *Block Access Token* - It gives a read or write permission to a HDFS block. Data nodes donot have any ACL (access control list) mechanism to handle block access.
    - *Job Token* - It identifies a map reduce task its jobs.
    - *Delegation Token* - It identifies a user to a particular HDFS service. It addresses the problem of propagating authentication performed at job submission time to when the job is later executed.

    More details can be found in [17, 11].

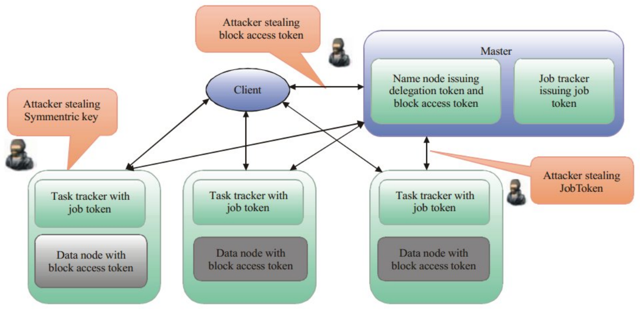
3. Other protocols - Other communication protocols used in HDFS are RPC and data transfer protocol. Hadoop clients use RPC which is a Hadoop's library to access most Hadoop services. In insecure versions of Hadoop, the users login name is determined from the client OS and sent across as part of the connection setup. For authenticated clusters,

all RPC's connect using simple authentication and secure layer(SASL). Data transfer Protocol is used when reading or writing data to HDFS, by clients using Hadoop API, by MapReduce jobs and among other Hadoop services. It is mainly used while data is in motion n HDFS storage. More details about these can be found in [25, 17].

4. Oozie [20] - It is a workflow scheduler to manage Hadoop jobs. They are directed acyclic graphs in actions. Oozie can be integrated with the rest of the Hadoop stack supporting several types of Hadoop jobs out of the box (such as Java map-reduce, Streaming map-reduce, Pig, Hive, Sqoop and Distcp) as well as system specific jobs. Oozie supports impersonation or *proxy user functionality*. For example, user David wants to submit a job and access hdfs on behalf of another user Oliver. David has kerberos credentials but user Oliver doesn't have any. Assume that the tasks are required to run as user Oliver and any file accesses on namenode are required to be done as user Oliver. It is required that user Oliver can connect to the namenode or job tracker on a connection authenticated with oozie's kerberos credentials. In other words oozie is impersonating the user Oliver. Hence, we have studied Oozie in the context of impersonation attacks. Oozie is a scalable, reliable and extensible system.

5. *Sandbox HDP-2.1*- Sandbox [6] is a third party vendor software provided by Hortonworks which provides a portable Hadoop environment. This platform contains a management software called Apache *Ambari* which manages and provisions the Hadoop cluster. As an enterprise Hadoop product, our study investigates HDP Sandbox [1, 9] to see the overall architecture and security implementation. Kerberos implementations have been on hadoop cluster and Apache Ambari framework provisions, manages and monitors the Hadoop clusters. Integrated with it are many third party softwares like Nagios, Ganglia, Oozie, Zookeeper and other database stores like HBase, Hive which can help in an end to end management of a distributed hadoop cluster. The monitoring data can be used for dynamic provisioning and resource management purposes.

Figure-2 shows possible attacks in HDFS considering unsecure communication channel with the tokens. We discussed the basic security aspects of the current security implementation. [4, 12, 14, 41, 28] can be referenced for further context.


Attacker stealing block access token
Master
Name node issuing delegation token and block access token
Job tracker issuing job token
Client
Attacker stealing Symmentric key
Task tracker with job token
Data node with block access token
Task tracker with job token
Data node with block access token
Task tracker with job token
Data node with block access token
Attacker stealing JobToken
Figure 2: HDFS Attack

3 Related Work

Prior work has discussed about security implication in Hadoop infrastructure, now that hadoop has been increasingly used standalone as well as with many other third party vendors. As per our knowledge, we have known several such works which discuss about the potential security problems with Hadoop.

[39, 19, 35] talk of solutions for hadoop based setups. [27] focuses on access control within HDFS. It discusses a dependable security model and explains how it differs from existing federated systems. Further, it discusses the difference between securities in federated systems and general distributed systems.

[32, 40, 21, 34, 30] does a survey of the security in Map Reduce Implementation Systems and big data security in general. Data security and privacy have also been discussed in [36, 37, 23]. These give a slightly higher level picture of the danger in the clouds in terms of data privacy. [13] speaks of a security management framework which can facilitate enforcement of security policies. It basically attempts to prevent a DOS attack.

[26, 8] talks of HDFS encryption in particular. It talks about the affordable low computation overhead of their proposed encrypted HDFS which basically has file protection in data blocks. It does AES data encryption/decryption for the same.

SpongeFiles [18] talk of data skew in mapreduce. It proposes a novel distributed-memory abstraction tailored to data processing environments like MapReduce. Although our work is not related to it, it gives an idea of the mitigation of disk spilling problem improving performance.

While [25] discuss the hadoop security design in details [22, 33, 15, 16] discuss about the attacks and threats on HDFS. They address the common security concern of Hadoop and propose an encryption scheme crypto framework for Hadoop.

Zheng et.al. [31, 24] come up with kerberos details and integration of Kerberos with Apache hadoop. This has been the focus of our work and we have referred to it for details. Islam et.al [20] describes Oozie a hadoop workflow management system. They discuss oozie architecture used in production environment for scalable parallel computation.

[29] proposes a novel authentication mechanism. They suggest three approaches for security enhancement in hadoop environment based on triangle properties. An analysis on the security level and complexity of these approaches has been presented in it.

Existing work like [3, 5, 7, 2, 11, 6, 10] also give an idea of big data security, gaps in existing hadoop, third party vendors like Cloudera, HortonWorks using hadoop and the potential security issues which are still popular and are a concern.

Although we have enlisted a considerable amount of existing related work, we strongly feel that security enhancement has been done very recently lacking a evaluation scheme currently. User reviews like missing data files in HDFS in the bug repository still leaves us with a grain of salt even though security modules exist. Not only are security vulnerabilities existing, which have not been rigorously tested, quality of service and performance using security enhancements also remains a potential concern.

4 Motivation & Problem Statement

Inspite of the huge volume of hadoop related research work and big data computations there is no well structured study on Hadoop vulnerabilities available with the latest security implementation. This is mainly because security implementations are comparatively new and are expected to be less stable. Moreover, hadoop providers' primary concerns have been and will always be efficient data intensive computations in large clusters. Nevertheless, since hadoop usage is on its rise, and huge amount of sensitive user data is in HDFS, security is a legitimate concern now. Following are some of the compelling reasons to take up this study:

- Extensive usage of hadoop for big data computations
- Less emphasis on security aspects, hence lower community support for figuring out breaches
- Absence of existing security model evaluation
- Bug repository reviews about security related issues and new security model comparatively less deployed and exhaustively used

Following are some of the users reviews in bug repository which can provide some hints about the security vulnerabilities in hadoop:

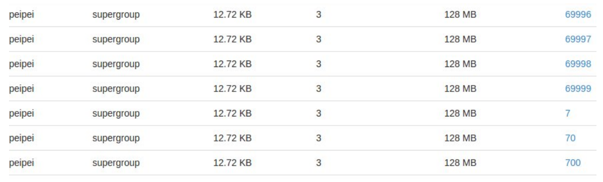
- Hadoop DoS attack

```
Make NameNode resilient to DoS attacks
(malicious or otherwise) defective
applications cause havoc on the NameNode,
for e.g. by doing 100k+ 'listStatus' on
very large directories (60k files) etc.
It seems there are a number of DoS scenarios
to worry about:

 RPC flooding (as you noted above)
 Malformed packets
 (it's probably not too hard to find a spot
 where you can make the NN allocate way too
 much memory and crash some important thread)
 Open socket limit exhaustion - what if a
 client just opened thousands of connections
 to the NN's RPC ports without actually
 sending commands? At some point
 you'll hit the ulimit -n lots of others.

 focus on detection of such attacks and
 counter acts with, say, iptables
 filtering to cut off an intruder
 or an honest fool.detection
```

- Rate Limiting

```
   RPC Support for QoS
 ----HDFS has a separate port for
 "service" IPC, allows you do to
 port-based QOS (see HDFS-599)
 Dos attack on server communication
 ports: hdfs, name node, datanode
 (what is a secure port??)
 HADOOP-9640-------Enable optional
 RPC-level priority to combat
 congestion and make request
 latencies more consistent. RPC
 Congestion Control with FairCallQueue
 replacing the FIFO call queue
 with a Fair Call Queue
```

The above reasons formed our main motivation to study kerberos and hadoop security. However because of



peipei	supergroup	12.72 KB	3	128 MB	69996
peipei	supergroup	12.72 KB	3	128 MB	69997
peipei	supergroup	12.72 KB	3	128 MB	69998
peipei	supergroup	12.72 KB	3	128 MB	69999
peipei	supergroup	12.72 KB	3	128 MB	7
peipei	supergroup	12.72 KB	3	128 MB	70
peipei	supergroup	12.72 KB	3	128 MB	700

Figure 3: RPC Congestion Creation

unanticipated complicacy in setups and time constraints we could not cover all aspects of security module like Token based data access mechanism. Our evaluation has been predominantly kerberos with rate limiting tests.

Following are our problem questions which we seek to answer:

- Difficulty level in implementation and use of kerberos deployment
- Efficacy of Kerberos in preventing impersonation attacks
- Rate Limiting Study
- Risk in investments in Hadoop service providers like Cloudera Vendors

4.1 Attacks

We have tried to address the following two kinds of attacks:

- *DOS attack* - Resource becomes unavailable for some users because another user is overwhelming the network by consuming too much of resources. Rate limiting study pertains to this attack. Here rate of traffic is controlled so that one user cannot starve another user.
- *Impersonation attack* - One user behaves like another user and tries to access another user's data or access to HDFS cluster.

These were the problems we faced while implementing the attacks:

- *Rate Limiting* - We need to create RPC congestion before evaluating rate limiting. That implies too many RPC calls trying to flood the queue. However after repeated attempts we could not create congestion for which we couldn't go ahead and perform rate limiting attack evaluation.

  Figure 3 shows that we tried creating huge number of files to create RPC congestion. However even after creating 70000 files, RPC congestion, flooding queues could not be created which prevented us from running rate limiting experiments.

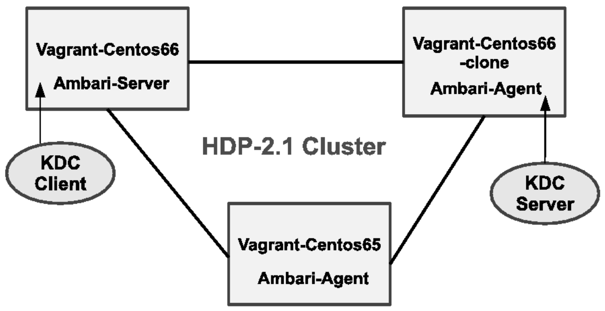


Figure 4: Sandbox Ambari HDP Setup

- *Impersonation* - It is a challenge in itself to figure out how to do impersonation. No prior work or web material clearly mentions how to perform impersonation. We took a while to figure out how to do it. In this context, we did try an attack on kerberos using the same named clients.

  We tried to use two clients with the same name trying to access HDFS datanode. However, Kerberos correctly checked for authentication and did not allow any of the clients to access datanode since that datanode is not identified with the corresponding clients in kerberos configuration. This is a very naive way of testing kerberos and it is imperative that kerberos will be successful.

  We did figure out a way to perform impersonation using oozie. Oozie can act as a super user and supports proxy functionality, it can be leveraged to check for malicious access.

5 Implementation and Evaluation

We have three setup implemented for evaluation. Following are the same:

- Hadoop 2.5.0 cluster containing three nodes
- Hadoop 1.2.1 cluster containing three nodes
- Sandbox HDP 2.1 cluster containing three nodes

We have three hgcc07, hgcc11, hgcc13 machines in the cluster. In each of these nodes we have two virtual machines belonging to different Hadoop clusters namely 2.5.0 and 1.2.1 respectively. These virtual machines have Ubuntu 10.04 installed in them. This is the setup used for Hadoop-2.5.0, for Hadoop-1.2.1 we do use a similar setup.

We have setup Sandbox HDP version 2.1 on a cluster of 3 virtual machines on virtualbox. Apache Ambari server and agents had to be deployed to form a HDP cluster. The ambari server ran on one of the VMs and the agents ran on the other two. One of the VMs had the KDC server installed. Unlike standalone Hadoop cluster which had a dedicated KDC out of the cluster, HDP has the option of enabling kerberos security within the cluster nodes.

5.1 Sandbox HDP

Figure 4 shows the Sandbox HDP-2.1 setup structure. We have setup a cluster of three VMs using ambari provisioning framework. We enabled Kerberos in the cluster by deploying the libraries and creating principal database. We did check that kerberos client is authenticated against the KDC (Kerberos Key Distribution Center), however we have not performed any attacks since we took a while as to how to do impersonation. Oozie configuration needs to be done in order to enable impersonation and check for malicious threats. Overall experience has been comparatively better with Sandbox HDP than Hadoop native versions in terms of enabling kerberos on the cluster. Moreover KDC is a part of the cluster unlike in the standalone case where it had to be outside the cluster. We feel that commercial products like Sandbox which has hadoop bundled with many other products are a better option than enabling kerberos as a third party software on hadoop. The later is much more painful compared to the former. We did observe slightly slower performance of the VMs after enabling security, but it was not worser than the standalone Hadoop versions study.

5.2 Kerberos

Figure 5 is the architecture for kerberos-enabled HDFS in Hadoop 2.5.0. hgccvm110, hgccvm111, and hgccvm113 are the virtual machines in the Hadoop cluster. `alaska.csc.ncsu.edu` is a host machine outside the Hadoop cluster which works as the Kerberos KDC. We shall discuss the pain in deploying kerberos in Hadoop cluster in the next section.

Following were some problems faced while installing kerberos in the Hadoop Cluster:

- The first problem we met is in creating kerberos database in virtual machine. Our first attempt is installing kerberos KDC in Hadoop cluster. However, when using *kdb5_util* to create principal database, the program hangs in "Loading random data". Getting sufficient random data in virtual machine is tricky. That's why finally we installed KDC in the physical host `alaska.csc.ncsu.edu`.

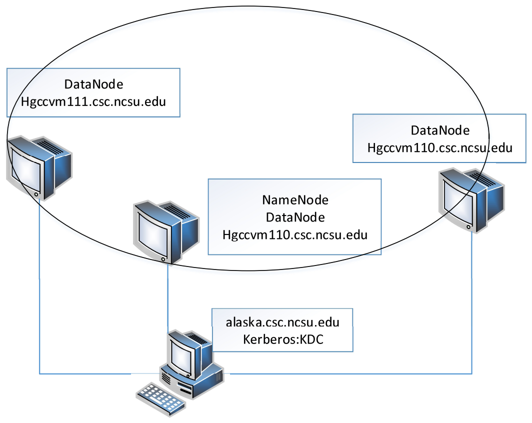


Figure 5: Hadoop with Kerberos

- The second problem is that Hadoop cluster is identified with fully-qualified domain names (FQDNs) to do kerberos authentication. This means each node should have its own FQDN. In the figure 5, the listed names, namely `hgccvm110.csc.ncsu.edu`, `hgccvm111.csc.ncsu.edu`, `hgccvm113.csc.ncsu.edu`, and `alaska.csc.ncsu.edu`, are all FQDNs.

- Another problem is the way to start DataNodes. After kerberos is enabled, the way to start NameNode does not change. However, secure DataNodes must use privileged port less than 1024 to assure that the server was started securely. This means that secure DataNodes must be started by root user via *jsvc* instead of running ordinary java program. Otherwise, it will throw *RuntimeException* with message that it cannot start secure cluster without privileged resources.

- Finally, we notice that our Hadoop cluster runs slowly in this deployment. alaska is running in subnet `152.14.xx.xx`, while hgccvm110, hgccvm111, and hgcc113 are running in subnet `10.76.2.xx`. Since KDC server and Hadoop cluster are not running on the same subnet, it could be one factor to cause Hadoop slow down. Besides, kerberos authentication itself is lumpy, which is also the reason that it is not widely adopted in distributed computing. Based on this observation, we removed kerberos when doing rate limiting experiment.

However for Sandbox HDP-2.1 we did not face major problems in setting up the principal database and it was comparatively easier in getting the Kerberos module running although we did observe that Kerberos enabled VMs performed comparatively slowly.

5.3 Takeaway from Kerberos Study

Our observation shows that without kerberos, Hadoop is vulnerable to malicious user operation because it does not do any authentication. Any file in HDFS can be easily accessed by its owner and other Hadoop users. However, we did not find the same observation when kerberos is enabled. We made sure to perform this ground truth verification although it is intuitive.

We verified kerberos with *two experiments*. The *first* experiment is using a Hadoop client without kerberos to access HDFS. The following text shows that HDFS recognized this illegal operation in its output messages.

```
peipei@ubuntu10045: /hadoop-2.5.0/bin$ ./hadoop dfs -ls hdfs://10.76.2.127:9000/ DEPRECATED: Use of this script to execute hdfs command is deprecated. Instead use the hdfs command for it.
14/12/01 16:25:31 WARN util.KerberosName: Kerberos krb5 configuration not found, setting default realm to empty ls: Authorization (hadoop.security.authorization) is enabled but authentication (hadoop.security.authentication) is configured as simple. Please configure another method like kerberos or digest.
```

The *second* experiment is joining Hadoop cluster with a kerberos enabled Datanode which is not predefined in Hadoop cluster. The following text shows that Hadoop does not allow nodes to join Hadoop cluster randomly. It requires the user principals to be added before joining it.

```
2014-12-01 18:40:29,887 INFO org.apache.hadoop.ipc.Server: IPC Server listener on 9000: readAndProcess threw exception org.apache.hadoop.security.AccessControlException: Connection from 152.14.92.184:53152 for protocol org.apache.hadoop.hdfs.server.protocol.DatanodeProtocol is unauthorized for user peipei/peipei-optiplex-9010@HADOOP.DOMAIN (auth:KERBEROS) from client 152.14.92.184. Count of bytes read: 0
```

This validates the fact that kerberos authentication successfully worked for the kerberos enabled case unlike the case without kerberos.

5.4 Oozie

We did configure and setup oozie. How to perform impersonation with oozie was figured out however we could not methodically do any attack to impersonate a user and check for malicious attacks. For example, suppose user David wants to access Oliver's data. Now, Oozie can behave like a super user and have access to both David and Oliver, since it is the workflow manager in hadoop. Impersonation is within the HDFS cluster where it behaves as a superuser. In that case, there remains a chance of a malicious node to access other user's data.

5.5 Rate limiting

We tried to create a RPC congestion case without rate limiting configurations with two hadoop commands. `hadoop dfs -lsr` will list file information in a specified directory. In order to get file information, the hadoop client has to create RPC connections to Hadoop NameNode. Another command is `hadoop fsck` to check the health of HDFS filesystem in order to find missing files or blocks. To diagnose HDFS files, Hadoop client also need to communicate with Hadoop NameNode. Before running hadoop commands, we created 70000 files and uploaded those files into a specific HDFS directory which is also the directory for `dfs -lsr`. We run `dfs -lsr` and `fsck` commands in two Hadoop DataNodes at the same time and measured their execution time. The average execution time in 10 rounds for `fsck` is 13 seconds, for `dfs -lsr`, it is 12 seconds. In this experiment, we could not create RPC congestion successfully, either in Hadoop 1.2.1 or in Hadoop 2.5.0.

Because of this failure, we could not have RPC congestion environment, and therefore did not go further to the evaluation of rate limiting configuration.

6 Discussion

Considerable amount of time has been spent in setting up and configuring hadoop cluster. The next challenge was to figure out how to test attacks. It is not at all straight forward to figure out how to imitate and implement an attack scenario. It is not hard to have a large hadoop cluster but enabling security in hadoop is hard. It is extremely painful particularly for the standalone hadoop clusters like we evaluated Hadoop-2.5.0 and Hadoop-1.2.1 compared to HDP Sandbox-2.1. Enabling kerberos in Sandbox is much easier than in the manual way.

In a way it makes sense to delegate tasks to a third party vendor when it comes to complex security enabled configurations across multiple data nodes and users accessing HDFS. Moroever we get measurement and monitoring of resources which can help in provisioning and maintenance. Based on our experience we feel that even though kerberos successfully performs simple authentication, its complicated deployment and affect on performance may be a concern particularly for large clusters in a virtualized production environment. To summarize we have the following major remarks to make based on our hands on experience:

- Kerberos is effective for simple authentication. But that does not demean its complicated deployment.
- Sandbox Security deployment is fairly simpler and less painful than the Hadoop standalone case.
- Unlike before, after our study we feel that oozie can be leveraged for impersonation, and can have potential security concerns. However, it is yet to be evaluated correctly.
- Enabling security degrades performance and this is valid for both the Sandbox HDP version and the naive versions of hadoop cluster. Hence its practical ease of usage is definately a concern when it comes to performance.

Sandbox performance even when used with three virtual machines running on the same host physical machine was better than the hadoop cluster when virtual machines were running on different physical hosts. However, since we could not get sufficient time to emulate an attack in Sandbox, we cannot comment on its security breaches yet.

7 Conclusion

We configured and setup hadoop cluster both native and thirty party vendor boxes. Through Hadoop deployment and simple attack tests, we find that kerberos is effective for user authentication. However, kerberos is extremely difficult to deploy in Hadoop cluster, especially when the Hadoop cluster is deployed in virtual machines. Besides, enabling kerberos makes Hadoop run much slower than without kerberos. Another insight is that *oozie* has potential security concerns since it supports impersonation to use proxy user to impersonate a set of configured hosts or groups. Because new features of Hadoop could expose potential threats, Hadoop products which provide additional functionalities to Hadoop with security guarantees will be a good choice when adopting these functionalities. Since kerberos is a third-party software, when deployed it degrades Hadoop performance, we believe that a new security module that is integrated in Hadoop itself can make its deployment easier with less performance impact.

References

[1] Apache ambari. http://hortonworks.com/hadoop/ambari/.

[2] Big data security gap: Protecting the hadoop cluster. http://www.zettaset.com/wp-content/uploads/2014/04/zettaset_wp_security_0413.pdf.

[3] Big data security: The evolution of hadoop security model. http://www.infoq.com/articles/HadoopSecurityModel.

[4] Cloud security in mapreduce. http://hackedexistence.com/downloads/Cloud_Security_in_Map_Reduce.pdf.

[5] Hadoop cdh cloudera. http://www.cloudera.com/content/cloudera/en/products-and-services/cdh.html.

[6] Hadoop security today and tomorrow. http://hortonworks.com/blog/hadoop-security-today-and-tomorrow/.

[7] Hadoop user guide. https://hadoop.apache.org/docs/r2.4.1/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html.

[8] Hdfs encryption. http://blog.cloudera.com/blog/2014/06/project-rhino-goal-at-rest-encryption/.

[9] Hdp sandbox. http://hortonworks.com/products/hortonworks-sandbox/.

[10] Loopholes in hadoop. http://readwrite.com/2014/08/13/hadoop-slow-security-issues-still-popular.

[11] Security implementation in hadoop. http://search.iiit.ac.in/cloud/presentations/28.pdf.

[12] Taking hadoop security to the next level. http://www.securityweek.com/bigger-data-smaller-problems-taking-hadoop-security-next-level.

[13] C. Basescu, A. Carpen-Amarie, C. Leordeanu, A. Costan, and G. Antoniu. Managing data access on clouds: A generic framework for enforcing security policies. In *Advanced Information Networking and Applications (AINA), 2011 IEEE International Conference on*, pages 459–466. IEEE, 2011.

[14] B. R. Chang, H. F. Tsai, Z.-Y. Lin, and C.-M. Chen. Access security on cloud computing implemented in hadoop system. In *Genetic and Evolutionary Computing (ICGEC), 2011 Fifth International Conference on*, pages 77–80. IEEE, 2011.

[15] J. Cohen and S. Acharya. Towards a more secure apache hadoop hdfs infrastructure. In *Network and System Security*, pages 735–741. Springer, 2013.

[16] J. C. Cohen and S. Acharya. Towards a trusted hdfs storage platform: Mitigating threats to hadoop infrastructures using hardware-accelerated encryption with tpm-rooted key protection. *Journal of Information Security and Applications*, 19(3):224–244, 2014.

[17] D. Das, O. O'Malley, S. Radia, and K. Zhang. Adding security to apache hadoop. *hortonworks report, http://www. Hortonworks. com [accessed March 2013]*.

[18] K. Elmeleegy, C. Olston, and B. Reed. Spongefiles: mitigating data skew in mapreduce using distributed memory. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 551–562. ACM, 2014.

[19] V. N. Inukollu, S. Arsi, and S. R. Ravuri. High level view of cloud security: Issues and solutions. *International Journal of Computer Science & Information Technology*, 6(2), 2014.

[20] M. Islam, A. K. Huang, M. Battisha, M. Chiang, S. Srinivasan, C. Peters, A. Neumann, and A. Abdelnur. Oozie: towards a scalable workflow management system for hadoop. In *Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies*, page 4. ACM, 2012.

[21] R. C. Jose and S. Paul. Privacy in map reduce based systems: A review. 2014.

[22] H.-Y. Lin, S.-T. Shen, W.-G. Tzeng, and B.-S. Lin. Toward data confidentiality via integrating hybrid encryption schemes and hadoop distributed file system. In *Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on*, pages 740–747. IEEE, 2012.

[23] S. Madaan and R. K. Agrawal. Implementation of identity based distributed cloud storage encryption scheme using php and c for hadoop file system. In *Tier 2 Federation Grid, Cloud & High Performance Computing Science (RO-LCG), 2012 5th Romania*, pages 74–77. IEEE, 2012.

[24] O. OMalley. Integrating kerberos into apache hadoop. In *Kerberos Conference*, pages 26–27, 2010.

[25] O. OMalley, K. Zhang, S. Radia, R. Marti, and C. Harrell. Hadoop security design. *Yahoo, Inc., Tech. Rep*, 2009.

[26] S. Park and Y. Lee. Secure hadoop with encrypted hdfs. In *Grid and Pervasive Computing*, pages 134–141. Springer, 2013.

[27] Y. Reddy. Access control for sensitive data in hadoop distributed file systems. In *INFOCOMP 2013, The Third International Conference on Advanced Communications and Computation*, pages 72–78, 2013.

[28] I. Roy, S. T. Setty, A. Kilzer, V. Shmatikov, and E. Witchel. Airavat: Security and privacy for mapreduce. In *NSDI*, volume 10, pages 297–312, 2010.

[29] G. Sadasivam, K. Kumari, and S. Rubika. A novel authentication service for hadoop in cloud environment. In *Cloud Computing in Emerging Markets (CCEM), 2012 IEEE International Conference on*, pages 1–6. IEEE, 2012.

[30] Z. Shen and Q. Tong. The security of cloud computing system enabled by trusted computing technology. In *Signal Processing Systems (ICSPS), 2010 2nd International Conference on*, volume 2, pages V2–11. IEEE, 2010.

[31] J. G. Steiner, B. C. Neuman, and J. I. Schiller. Kerberos: An authentication service for open network systems. In *USENIX Winter*, pages 191–202, 1988.

[32] S. Subashini and V. Kavitha. A survey on security issues in service delivery models of cloud computing. *Journal of Network and Computer Applications*, 34(1):1–11, 2011.

[33] G. Sujitha, M. Varadharajan, Y. V. Rao, R. Sridev, M. Gauthaum, S. Narayanan, R. S. Raja, and S. M. Shalinie. Improving security of parallel algorithm using key encryption technique. *Information Technology Journal*, 12(12), 2013.

[34] C. Tankard. Big data security. *Network security*, 2012(7):5–8, 2012.

[35] S. Tripathi, B. Gupta, A. Almomani, A. Mishra, and S. Veluru. Hadoop based defense solution to handle distributed denial of service (ddos) attacks. 2013.

[36] F.-H. Tseng, C.-Y. Chen, L.-D. Chou, and H.-C. Chao. Implement a reliable and secure cloud distributed file system. In *Intelligent Signal Processing and Communications Systems (ISPACS), 2012 International Symposium on*, pages 227–232. IEEE, 2012.

[37] L. Wei, H. Zhu, Z. Cao, X. Dong, W. Jia, Y. Chen, and A. V. Vasilakos. Security and privacy for storage and computation in cloud computing. *Information Sciences*, 258:371–386, 2014.

[38] J. Zhao, L. Wang, J. Tao, J. Chen, W. Sun, R. Ranjan, J. Kołodziej, A. Streit, and D. Georgakopoulos. A security framework in g-hadoop for big data computing across distributed cloud data centres. *Journal of Computer and System Sciences*, 80(5):994–1007, 2014.

[39] H. Zhou and Q. Wen. A new solution of data security accessing for hadoop based on cp-abe. In *Software Engineering and Service Science (ICSESS), 2014 5th IEEE International Conference on*, pages 525–528. IEEE, 2014.

[40] M. Zhou, R. Zhang, W. Xie, W. Qian, and A. Zhou. Security and privacy in cloud computing: A survey. In *Semantics Knowledge and Grid (SKG), 2010 Sixth International Conference on*, pages 105–112. IEEE, 2010.

[41] D. Zissis and D. Lekkas. Addressing cloud computing security issues. *Future Generation Computer Systems*, 28(3):583–592, 2012.