Journal of Theoretical and Applied Information Technology
15th April 2017. Vol.95. No 7

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

KEY EXCHANGE AUTHENTICATION PROTOCOL FOR NFS ENABLED HDFS CLIENT

[1]NAWAB MUHAMMAD FASEEH QURESHI, [2*]DONG RYEOL SHIN, [3]ISMA FARAH SIDDIQUI

[1,2] Department of Computer Science and Engineering, Sungkyunkwan University, Suwon, South Korea

[3] Department of Software Engineering, Mehran UET, Pakistan

[*]Corresponding Author

E-mail: [1]faseeh@skku.edu, [2*]drshin@skku.edu, [3]isma.farah@faculty.muet.edu.pk

ABSTRACT

By virtue of its built-in processing capabilities for large datasets, Hadoop ecosystem has been utilized to solve many critical problems. The ecosystem consists of three components; client, Namenode and Datanode, where client is a user component that requests cluster operations through Namenode and processes data blocks at Datanode enabled with Hadoop Distributed File System (HDFS). Recently, HDFS has launched an add-on to connect a client through Network File System (NFS) that can upload and download set of data blocks over Hadoop cluster. In this way, a client is not considered as part of the HDFS and could perform read and write operations through a contrast file system. This HDFS NFS gateway has raised many security concerns, most particularly; no reliable authentication support of upload and download of data blocks, no local and remote client efficient connectivity, and HDFS mounting durability issues through untrusted connectivity. To stabilize the NFS gateway strategy, we present in this paper a Key Exchange Authentication Protocol (KEAP) between NFS enabled client and HDFS NFS gateway. The proposed approach provides cryptographic assurance of authentication between clients and gateway. The protocol empowers local and remote client to reduce the problem of session lagging over server instances. Moreover, KEAP-NFS enabled client increases durability through stabilized session and increases ordered writes through HDFS trusted authorization. The experimental evaluation depicts that KEAP-NFS enabled client increases local and remote client I/O stability, increases durability of HDFS mount, and manages ordered and unordered writes over HDFS Hadoop cluster.

Keywords: *Hadoop, HDFS, NFS Gateway, Security, Reliability.*

1. INTRODUCTION

Big data analytics has strengthened the concept of large data processing in a functional manner [1]. For this purpose, we find multiple huge data processing systems i.e. Apache Hadoop [2], MapR [3], Cloudera [4]. Apache Hadoop is an open source ecosystem that process large scaled datasets through four components i.e. Hadoop commons, YARN, HDFS and MapReduce. Hadoop commons consists of functional library that support cluster environment processing. YARN is counted as brain of Hadoop that controls the functionality of data set processing [5]. HDFS is a file system that provides namespace to store datasets [6]. Whereas, MapReduce is a functional paradigm that processes largescale datasets in the distributed computing environment [7].

The HDFS is distributed over three-layer architecture consisting of, client, Namenode, and Datanode. The client connects to Namenode and processes authorized datasets over Datanode [8]. The authorization at this layer includes Namenode permission and location of data block processing over Datanode [9].

Recently, Hadoop has introduced an add-on functionality to connect a client having a different file system than HDFS [10]. The reason to provide such facility is to bypass a conditional limit of not allowing random writes over HDFS. Due to this, Hadoop extends client accessibility through Network file system (NFS) and security authorization protocols i.e., Kerberos and network-layer authorization protocols [11] over network layer. However, Hadoop ecosystem processes random application requests through multiple clients and most of them remain to be unprivileged

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

nodes [12]. Due to this, such authorization protocols at network layers would not be found useful. Therefore, the ecosystem enhanced client functionality to NFS client and HDFS NFS gateway as seen from Figure-1.

The current scenario of HDFS NFS gateway provides functional access to two types of clients i.e., privileged clients and unprivileged clients [13]. The privileged clients use system authorization only i.e., 'root', while unprivileged clients do not use any such authorization [14]. Thus, HDFS suffers due to non-durable connections, less ordered writes, and increase in unordered writes over cluster.

To solve mentioned issues, we propose Key Exchange Authorization Protocol (KEAP) NFS enabled client that provides a reliable and secure connectivity over HDFS. The KEAP-NFS enabled client reduces connectivity time at local and remote profiles. Moreover, the proposed approach increases durability in sessions over HDFS mount. To add with this, KEAP-NFS enabled client also reduced unordered writes and increased ordered writes as compared to privileged and unprivileged clients.

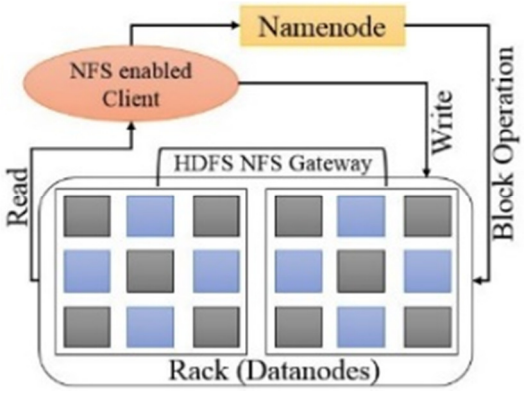

Figure 1: Default NFS Enabled HDFS Cluster

The main contributions of the proposed scheme are:

- A novel public key encryption strategy over NFS client.
- A novel private key decryption strategy over HDFS NFS gateway.
- An enhanced cryptographic key exchange strategy between NFS client and HDFS NFS gateway.
- KEAP enabled HDFS mount '/' directory session management.

The remaining paper is organized as follows. Section II discusses related work. Section III briefly explains proposed approach KEAP. Section IV depicts experimental environment and evaluation result for KEAP-NFS enabled client. Finally, section V shows conclusion and future research directions.

2. RELATED WORK

Researchers have presented contributions over HDFS security perspective. The prominent contributions could be divided into two categories i.e., Block Access Token (BAT) and Delegation Token (DT). Although, Kerberos authorization [15] could be used at HDFS NFS gateway but that arises result session lagging and latency issues [16]. Moreover, Kerberos eTicket authorization increases NFS mount timeout problem [17]. Therefore, we focus over the related contributions of BAT and DT approaches. HDFS NFS gateway is accessed by two types of clients i.e. privileged and unprivileged. In case of DT [18] that assigns authorization through Namenode, the gateway is unable to permit unprivileged clients. Moreover, Namenode assigns a specific session time to read / write data blocks which produce re-connect session problems in the HDFS NFS gateway environment [19]. BAT [20] strategy is specially use to pass data access authorization from Namenode to Datanode. In such a scenario, the NFS client is ignored to read / write a data block [21]. Moreover, BAT is limited to single 'owner' data block processing and could not facilitate multiple NFS client accessibility [22].

Considering such a limited scenario of related schemes for HDFS NFS gateway, we present KEAP-NFS enabled HDFS client that uses a novel Key Exchange Authorization protocol to authenticate any NFS client i.e. privileged or unprivileged. Moreover, our presented authorization scheme increases the durability of user's session through confirmation of certified user and increases ordered writes over trusted communication.

3. KEY EXCHANGE AUTHENTICATION PROTOCOL OVER NFS ENABLED HDFS

The proposed approach KEAP is distributed in five stages i.e., (i) Generation of public and private key certificates, (ii) Public key certificate for KEAP-NFS enabled client, (iii) KEAP-NFS HDFS private key certificate processing, and (iv) Exchange of Public and Private authorization keys. When a KEAP-NFS enabled client acquires public key, the Namenode receives certificate credentials and exchanges the authorization information with HDFS NFS gateway. The KEAP enabled gateway validates the NFS client certificate with private key certificate

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and authorizes access credentials to read / write data blocks over Datanode rack, as shown in Figure-2.

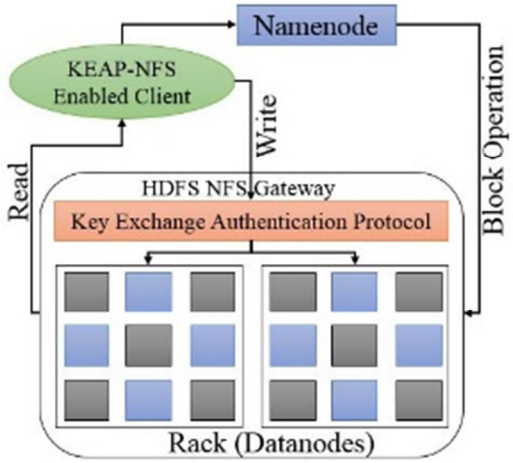

Figure 2: Key Exchange Authentication Protocol over NFS enabled HDFS

Table 1: KEAP Notations

Notations	Description
$user_{ACL_i}$	A user *i* defined in access control list
$Cert_{pub_i}$	Public key certificate
$Cert_{priv_i}$	Private key certificate
Pub_{key_i}	Public key
$Priv_{key_i}$	Private key
$Cert_{req_i}$	Certificate request
fun_{srf}	Pseudo random function
C_i	Client instance
HNG_i	HDFS NFS Gateway instance
$rights_{user_i}$	ACL users' rights

3.1 Generation of Public and Private key certificates

The Namenode is responsible to generate public and private key certificates as per access control list (ACL) of users. The $user_{ACL_i}$ is defined with $rights_{user_i}$ in Namenode.

3.1.1 Public Key Certificate (PKC)

The generation of PKC involves a prime integer p, a long number generator g and $\emptyset(n) = (p \times g)$ into Pub_{key_i} of Certificate Authority (CA). The integer p belongs to Diffie-Hellman group [23] of Sophie Germain prime [24], long number generator g is primitive root modulo of integer p and $\emptyset(n)$ is Euler's totient function [25]. Therefore, Pub_{key_i} can be generated as:

$$Pub_{key_i} = \left(p_{int_i}, g_{mod_p}, \emptyset(n)\right) \quad (1)$$

The certificate contains public key Pub_{key_i} and digital signature DS_{Cert_i}[26]. Therefore, the public key certificate $Cert_{pub_i}$ can be generated as:

$$Cert_{pub_i} = \left(DS_{Cert_i}, Pub_{key_i}\right) \quad (2)$$

As we know that the HDFS client's ACL [27], contains $user_{ACL_i}$ information. The KEAP encrypts $user_{ACL_i}$ with $Cert_{pub_i}$. Therefore, the encrypted $Message_E$ can be generated over public key certificate as:

$$Message_E = \left(user_{ACL_i}, Cert_{pub_i}\right) \quad (3)$$

3.1.2 PRivate Key Certificate (PRKC)

The generation of PRKC involves a prime integer p, a long number generator g, modular multiplicative inverse $w = b\ (mod\ \emptyset(n))$ having b as coprime to $\phi(n)$ $w_p = w\ mod\ (p-1)$, $w_g = w\ mod\ (q-1)$ and $g_{inv} = g^{-1} mod\ p$. Therefore, $Priv_{key_i}$ can be generated as:

$$Priv_{key_i} = \left(p_{int_i}, g_{mod_p}, w, w_p, w_g, g_{inv}\right) \quad (4)$$

The certificate consists of private key $Priv_{key_i}$ and digital signature DS_{Cert_i}. Therefore, the private key certificate $Cert_{priv_i}$ can be generated as,

$$Cert_{priv_i} = \left(DS_{Cert_i}, Priv_{key_i}\right) \quad (5)$$

The HDFS NFS gateway decrypts $Message_E$ through validating $Cert_{priv_i}$ and match $user_{ACL_i}$ information. Therefore, the decryption of $Message_E$ can be represented as,

$$Message_D = \left(user_{ACL_i}, Cert_{priv_i}\right) \quad (6)$$

3.2 Public key certificate for KEAP-NFS enabled client

When a client contacts Namenode, HDFS classifies this node into two categories i.e. Local client and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

remote client [28]. The NFS gateway validates HOST/IP address of client and sends an interface to submit authentication credentials $Auth_{C_i}$. The client instance C_i receives public key certificate $Cert_{pub_i}$ and generates a request session over HDFS NFS gateway as:

$$Req_{C_i} = (Auth_{C_i}, Val_{ID_i}, Message_E) \quad (7)$$

The Req_{C_i} security header SH_{C_i} differs with node classification. The local client request session can be represented as:

$$Req_{Local_i} = (Req_{C_i}, SH_{Local_i}) \quad (8)$$

Similarly, the remote client request session can be represented as:

$$Req_{Remote_i} = (Req_{C_i}, SH_{Remote_i}) \quad (9)$$

3.3 KEAP-NFS HDFS private key certificate processing

At this stage, the Req_{C_i} connects NFS gateway HNG_i through *portmap* configuration [29]. The NFS gateway facilitates Req_{Local_i} over $Local_{Gateway}$ and Req_{Remote_i} over $Remote_{Gateway}$. The $Message_E$ is decrypted using $Cert_{priv_i}$ and $client_{Session_i}$ receives $rights_{user_i}$ through *keytab*. The *keytab* is a set of principles to allocate $HDFS_{Namespace}$. After this, $client_{Session_i}$ receives mount point '/' through HDFS proxy user and establishes a connection with $Rack_{Datanodes}$ as illustrated in Figure-3.

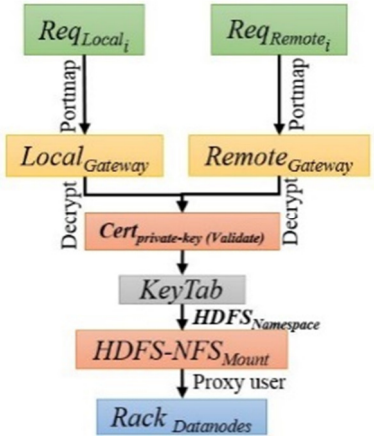

Figure 3: Workflow of Private key certificate processing over KEAP-NFS HDFS

3.4 Exchange of Public and Private authorization keys

We consider that client instance C_i shows encrypted $Message_E$ over KEAP-NFS HDFS gateway. The *validator* isolates $Cert_{pub_i}$ and $m_{Encrypt}$. Furthermore, $Cert_{pub_i}$ is extracted between public key Pub_{key_i} and digital signature DS_{cert_i}. The DS_{cert_i} depicts the reliable source ownership of KEAP-NFS gateway and Pub_{key_i} refers to the encryption key [30]. The NFS gateway instance HNG_i calculates all encryption credentials through calculating Pub_{key_i} as summarized in Figure-4.

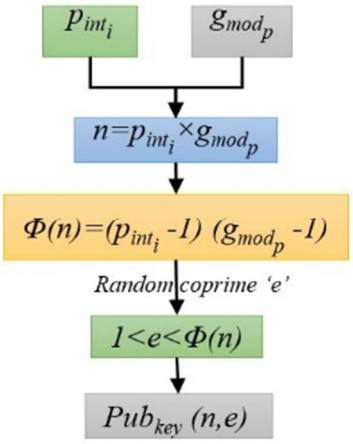

Figure 4: Public key encryption procedure

The decryption work-flow includes parameters of $Message_D$ i.e. repository of $user_{ACL_i}$ and processing of $Cert_{priv_i}$ credentials. Furthermore, digital signature DS_{cert_i} cross checks the validity of KEAP-NFS client and decrypt $C(Message_E)$ through private key $Priv_{key_i}$ as illustrated in Figure-5.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

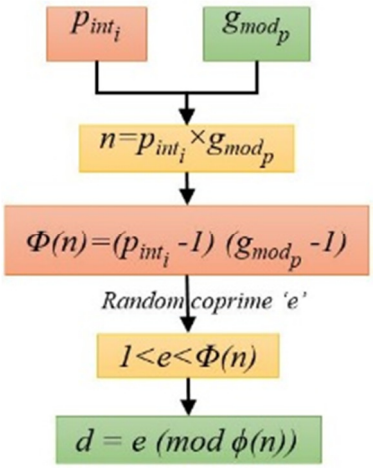

Figure 5: Private key decryption procedure

3.4.1 Message Encryption

The KEAP-NFS enabled client formats real authorization M into integer m. The corresponding value of m remain in between $0<m<n$ and $gcd(m,n) = 1$. The m and n are coprime integers obtained using padding strategy. The client computes ciphered text $C(Message_E)$ using public key Pub_{key_i}. This message transformation can be represented as:

$$C(Message_E) \equiv m^e (mod\ n) \qquad (10)$$

3.4.2 Message Decryption

The NFS gateway instance HNG_i decrypts $C(Message_E)$ through the $Priv_{key_i}$ d as:

$$Message\ (m) \equiv\ C(Message_E)^d \qquad (11)$$

4. EXPERIMENTAL EVALUATION

In this section, we evaluate KEAP approach over cluster configuration as observed from Table-1.

Table 1: Hadoop Cluster.

Machine	Specifications	No. of VM	
Intel Xeon E5-2600 v2	8 CPUs, 32GB memory, 1T Disk and 128 GB SSD	3	1 Master Node, 2 Datanodes
Intel core i5	4 Core, 16GB memory, 1T Disk and 128 GB SSD	2	2 Datanodes
Hadoop	Hadoop-2.7.2 (stable)		
Virtual Machine Management	VirtualBox 5.0.16		

4.1 Environment

The Hadoop cluster contains Intel Xeon with 8 CPUs, 32GB memory and storage devices i.e. 1TB Hard disk drive and 128GB Samsung SSD. Furthermore, we also used Intel core i5 with 4 Core, 16GB memory and storage devices i.e. 1TB Hard disk drive and 128 GB Samsung SSD. We used virtualbox 5.0.16 for installing 5 virtual machines on depicted cluster configurations as observed from Table- 2.

Table 2: Hadoop Cluster Virtual Machines Configuration.

Node	CPU	Memory	Disk	Configuration
Master Node	6	16 GB	HDD & SSD	Intel Xeon
Slave1	2	4GB	HDD & SSD	Intel Xeon
Slave2	2	4GB	HDD & SSD	Intel Core i5
Slave3	2	4GB	HDD & SSD	Intel Core i5
Slave4	2	4GB	HDD & SSD	Intel Core i5

4.2 Experimental Dataset

The dataset used to process experimental work includes: (i) 640 SSD wordcount data blocks of 64MB (40GB size), (ii) 640 HDD wordcount data blocks (40GB size), (iii) 640 SSD grep data blocks (40GB size), and (iv) 640 HDD grep data blocks (40GB size) [31].

4.3 Experimental Results

The experiments performed to evaluate KEAP scheme are: (i) Local client access, (ii) Remote client access, (iii) NFS mount durability, (iv) Ordered writes, and (v) Unordered writes.

4.3.1 Local client access

To evaluate the efficiency of Local client connectivity, we performed lookup analysis [32] over three type of connections i.e. Local KEAP-NFS enabled client, Local Privileged client, and Local unprivileged client. We evaluated above mentioned clients over '500' HDFS NFS gateway instances and found that Local KEAP-NFS enabled client consumes '13' seconds averagely over connecting to NFS gateway. Similarly, we evaluate that Local privileged client consumes '19' seconds averagely while connecting to NFS gateway. In the same way, we evaluate that Local unprivileged client consumes '25' seconds averagely at connecting to NFS Gateway. The KEAP-enabled client is 46.15% efficient than Local privileged and 92.3% efficient than Local unprivileged client over connecting to NFS gateway, as shown in Figure-6. The reason of this robust efficiency is unattended bypass of session through NFS authorization due to KEAP scheme while Local privileged need to authorize connection at mount point and local

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

unprivileged client consumes huge time due to open proxy connectivity lagging.

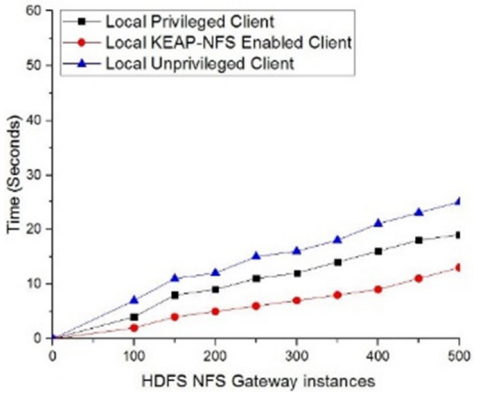

Figure 6: Local client connectivity over HDFS NFS Gateway

4.3.2 Remote client access

We evaluate the effectiveness of Remote client connectivity through remote lookup analysis over three type of connections i.e. Remote KEAP-NFS enabled client, Remote privileged client, and Remote unprivileged client. The threshold HDFS NFS gateway connection is set to '500' instances. We evaluate that Remote KEAP-NFS enabled client consumes '19' seconds averagely for connecting to NFS gateway. In the same way, we find that Remote privileged client consumes '28' seconds averagely over connecting to NFS gateway. Similarly, we evaluate that Remote unprivileged client consumes '32' seconds averagely while connecting to NFS Gateway. The Remote KEAP-enabled client is 47.36% effective than Remote privileged and 68.42% efficient than Remote unprivileged client while connecting to NFS gateway as shown in Figure-7. The proposed scheme is efficient due to unattended bypass of session through NFS authorization and remote authorization. The Remote privilege produces latency due to mount point authorization and remote network delay, while Remote unprivileged client secures network delay and opens proxy connectivity lagging.

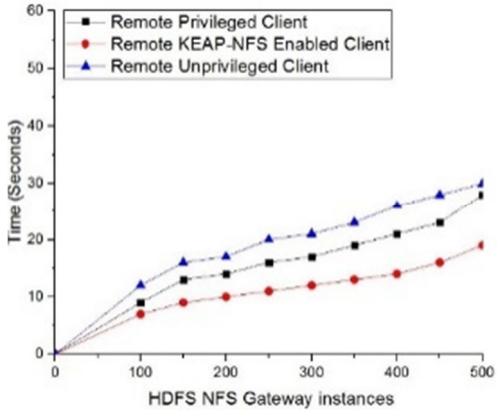

Figure 7: Remote client connectivity over HDFS NFS Gateway

4.3.3 NFS mount durability

When a client session is granted access to NFS mount '/', HDFS https generate a session timeout due to passage of huge data block processing. Thus, privileged and unprivileged clients re-connects to get a new session and resume the I/O operations. The NFS gateway logs resuming connections and calculates a session duration [33]. Furthermore, NFS computes a session durability percentile over HDFS cluster. We evaluate 6 type of connections over 300 instances equally dividing into 50 sessions i.e., (i) Remote KEAP-NFS enabled client, (ii) Remote privileged client, (iii) Remote unprivileged client, (iv) Local KEAP-NFS enabled client, (v) Local privileged client, and (vi) Local unprivileged client. We performed rigorous analytics and evaluated that Remote KEAP-NFS enabled client lengthen a session up to '60' seconds averagely. Moreover, Remote privileged client maintains a session up to '52' seconds averagely. Furthermore, Remote unprivileged client sustains a session up to '42' seconds averagely. Similarly, we found that Local KEAP-NFS enabled client lengthen a session up to '56' seconds averagely, and Local privileged client maintains a session up to '48' seconds averagely. Furthermore, Local unprivileged client sustains a session up to '38' seconds averagely, as shown in Figure-8. In this way, the Remote KEAP-NFS enabled client is 15.38% efficient than Remote privileged and 42.85% efficient than Remote unprivileged client. Moreover, Local KEAP-NFS enabled client is 16.67% effective than Local privileged and 47.36% efficient than Local unprivileged client in terms of session duration.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

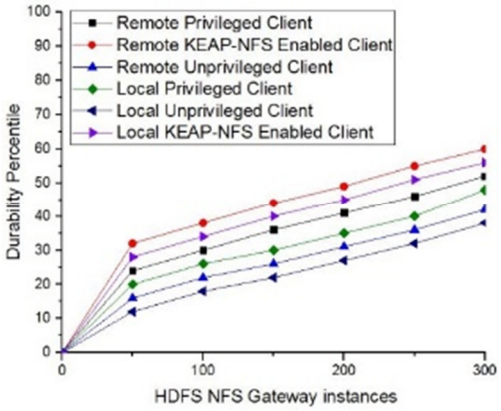

Figure 8: NFS client durability percentile over HDFS NFS Gateway

4.3.4 HDFS NFS Gateway Operations

The NFS gateway provides a medium to place dataset over mount directory of HDFS. As we know that, HDFS do not support random writes storage, whereas NFS client offers two types of random writes i.e. ordered writes and unordered writes [34]. Due to this, NFS client cannot write dataset operations over HDFS directly but upload a local file system storage dataset to HDFS. This untrusted authorization strategy decreases the number of ordered writes and increases number of unorders writes over NFS HDFS.

To observe the performance of NFS gateway operations, we evaluated connection types over 300 instances, equally divided into 50 sessions. We found that Remote KEAP-NFS enabled client processed '27342' ordered writes averagely. The Remote privileged client processed '24729' ordered writes averagely and Remote unprivileged client processed '21721' ordered writes averagely. Furthermore, we found that Local KEAP-NFS enabled client produced '26016' ordered writes averagely. The Local privileged client produced '23428' ordered writes averagely and Local unprivileged client produced '19624' ordered writes averagely. Thus, with this evaluation we found that, the Remote KEAP-NFS enabled client is 10.56% efficient than Remote privileged and 25.87% effective than Remote unprivileged client ordered writes processing. The Local KEAP-NFS enabled client is 11.04% efficient than Local privileged and 32.57% effective than Local unprivileged client ordered writes processing, as shown in Figure-9.

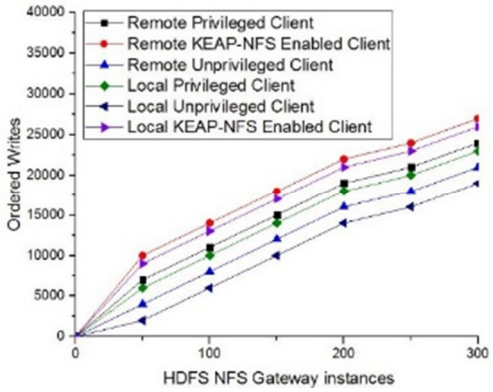

Figure 9: NFS client ordered writes over HDFS NFS Gateway

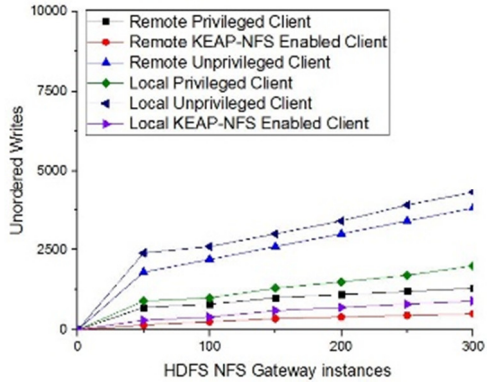

Figure 10: NFS client Unordered writes over HDFS NFS Gateway

Moreover, we evaluated that Remote KEAP-NFS enabled client produced '527' unordered writes averagely. The Remote privileged client produced '1341' unordered writes averagely and Remote unprivileged client produced '3891' unordered writes averagely. Furthermore, we found that Local KEAP-NFS enabled client processed '904' unordered writes averagely. The Local privileged client processed '2067' unordered writes averagely and Local unprivileged client produced '4383' unordered writes averagely. Therefore, the Remote KEAP-NFS enabled client is 60.7% efficient than Remote privileged and 86.45% effective than Remote unprivileged client unordered writes processing. The Local KEAP-NFS enabled client is 56.26% efficient than Local privileged and 79.37% effective than Local unprivileged client unordered writes processing, as presented in Figure-10.

5. CONCLUSION

This paper proposes a novel Key Exchange Authorization Protocol (KEAP) over HDFS NFS gateway. The proposed approach

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

secures NFS client and HDFS NFS gateway at user session environment. With extensive evaluations, the KEAP-NFS enabled client is found to be effectively stable than local privileged and unprivileged clients. Moreover, the KEAP-NFS enabled client performs efficiently than privileged and unprivileged remotely. The HDFS mount point remains much stable than default approaches. Finally, KEAP reduces unordered writes and increases ordered writes over HDFS NFS gateway.

In future, we would focus to extend NFS gateway integrity over interoperability aware multihoming inter-networks.

ACKNOWLEDGEMENT

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) (No. R0113-15-0002, Automotive ICT based e-Call standardization and after-market device development)

REFRENCES:

[1] LaValle, Steve, et al. "Big data, analytics and the path from insights to value." MIT sloan management review 52.2, 2011, pp. 21.

[2] "Welcome to Apache™ Hadoop®!" 2014. [Online]. Available: http://hadoop.apache.org/. Accessed: Dec. 20, 2016.

[3] M. Technologies, "Featured customers", 2016. [Online]. Available: https://www.mapr.com/. Accessed: Dec. 20, 2016.

[4] Cloudera, "The modern platform for data management and analytics," Cloudera, 2016. [Online]. Available: http://www.cloudera.com/. Accessed: Dec. 20, 2016.

[5] "Apache Hadoop 2.7.2 – Apache Hadoop YARN," 2016. [Online]. Available: https://hadoop.apache.org/docs/r2.7.2/hadoop-yarn/hadoop-yarn-site/YARN.html. Accessed: Dec. 20, 2016.

[6] "Apache Hadoop 2.7.2 – HDFS users guide," 2016. [Online]. Available: https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html. Accessed: Dec. 20, 2016.

[7] "Apache Hadoop 2.7.2 – MapReduce Tutorial," 2016. [Online]. Available: https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html. Accessed: Dec. 20, 2016.

[8] A. Kala Karun and K. Chitharanjan, "A review on hadoop — HDFS infrastructure extensions," 2013 IEEE CONFERENCE ON INFORMATION AND COMMUNICATION TECHNOLOGIES, Apr. 2013.

[9] H. Chen and Z. Yan, "Security and privacy in big data lifetime: A review," in Security, Privacy and Anonymity in Computation, Communication and Storage. Springer Nature, 2016, pp. 3-15.

[10] "Apache Hadoop 2.7.2 – HDFS NFS gateway," 2016. [Online]. Available: https://hadoop.apache.org/docs/r2.7.2/hadoop-project-dist/hadoop-hdfs/HdfsNfsGateway.html. Accessed: Dec. 20, 2016.

[11] S. Shaw, A. F. Vermeulen, A. Gupta, and D. Kjerrumgaard, "Loading data into hive," in Practical Hive. Springer Nature, 2016, pp. 99-114.

[12] Y. Tsuruoka, "Cloud computing - current status and future directions," Journal of Information Processing, vol. 24, no. 2, 2016, pp. 183–194.

[13] J. Raja, et al., "A comprehensive study on big data security and integrity over cloud storage," Indian Journal of Science and Technology, vol. 9, no. 40, Nov. 2016.

[14] C. Rodríguez-Quintana, A. F. Díaz, J. Ortega, R. H. Palacios, and A. Ortiz, "A new Scalable approach for distributed Metadata in HPC," in Algorithms and Architectures for Parallel Processing. Springer Nature, 2016, pp. 106-117.

[15] T. White, Hadoop: The definitive guide, "O'Reilly Media, Inc.", 2012.

[16] D. S. Terzi, R. Terzi, and S. Sagiroglu, "A survey on security and privacy issues in big data," 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST), Dec. 2015.

[17] A. F. Díaz, I. Blokhin, J. Ortega, R. H. Palacios, C. Rodríguez-Quintana, and J. Díaz-García, "Secure data access in Hadoop using Elliptic curve Cryptography," in Algorithms and Architectures for Parallel Processing. Springer Nature, 2016, pp. 136-145.

[18] J. Cohen, and S. Acharya, "Towards a more secure Apache Hadoop HDFS infrastructure," in Lecture Notes in Computer Science. Springer Nature, 2013, pp. 735-741.

[19] X. Yu, P. Ning, and M. A. Vouk, "Enhancing security of Hadoop in a public cloud," 2015 6th International Conference on Information and Communication Systems (ICICS), Apr. 2015.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[20] O. O'Malley, et al. "Hadoop security design." Yahoo, Inc., Tech. Rep, 2009.

[21] M. Kanyeba, and L. Yu, "Securing Authentication Within Hadoop.", 2016.

[22] I. Khalil, I. Hababeh, and A. Khreishah, "Secure inter cloud data migration," 2016 7th International Conference on Information and Communication Systems (ICICS), Apr. 2016.

[23] A. Boldyreva, "Threshold signatures, Multisignatures and blind signatures based on the Gap-Diffie-Hellman-Group signature scheme," in Public Key Cryptography — PKC 2003. Springer Nature, 2002, pp. 31-46.

[24] H. Dubner, "Large Sophie Germain primes." Mathematics of Computation of the American Mathematical Society 65.213, 1996, pp. 393-396.

[25] D. H. Lehmer, "On Euler's totient function." Bulletin of the American Mathematical Society 38.10, 1932, pp. 745-751.

[26] M. Bellare, and S. K. Miner. "A forward-secure digital signature scheme." Annual International Cryptology Conference. Springer Berlin Heidelberg, 1999.

[27] T. V. Hu, D. F. Grance, and D. Kuhn, "An access control scheme for big data processing," Proceedings of the 10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2014.

[28] A. Rasheed, and M. Mohamed. "Fedora Commons With Apache Hadoop: A Research Study", 2013.

[29] D. Eadline, Hadoop 2 Quick-Start Guide: Learn the Essentials of Big Data Computing in the Apache Hadoop 2 Ecosystem. Addison-Wesley Professional, 2015. [Online]. Available: http://www.networkrevolution.co.uk/project-library/dataset-tc1a-basic-profiling-domestic-smart-meter-customers/. Accessed: Dec. 20, 2016.

[30] "RSA (cryptosystem)," in Wikipedia, Wikimedia Foundation, 2016. [Online]. Available: https://en.wikipedia.org/wiki/RSA_(cryptosystem). Accessed: Dec. 20, 2016.

[31] N. M. F. Qureshi, and D. R. Shin, "RDP: A storage-tier-aware Robust Data Placement strategy for Hadoop in a Cloud-based Heterogeneous Environment", KSII Transactions on Internet and Information Systems, vol. 10, no. 9, 2016, pp. 4063-4086.

[32] S. Huang, et al. "The HiBench benchmark suite: Characterization of the MapReduce-based data analysis." New Frontiers in Information and Software as Services. Springer Berlin Heidelberg, 2011, pp. 209-228.

[33] S. Mazumder, "Big Data Tools and Platforms." Big Data Concepts, Theories, and Applications. Springer International Publishing, 2016, pp. 29-128.

[34] S. Magana-zook, et al. "Large-scale seismic waveform quality metric calculation using Hadoop." Computers & Geosciences, 2016.