IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 4, April 2017

Avoidance of Duplication of Encrypted Big-data in Cloud Storage

Dastagir Shaikh[1], Pratik Sen[2], Zubair Inamdar[3]

Sinhgad College of Engineering, Pune[1,2,3]

Abstract: Cloud computing deals with the practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer. The most important and popular cloud service is data storage. The data which is of primary importance is stored in an encrypted manner over the cloud so as to maintain its privacy. Hackers can access the data and can cause serious damages to the proprietary data and for its privacy concern data is stored in encrypted form. Now in cloud storage, deduplication technique is used by cloud service provider (CSP) to maintain the space complexity, privacy and to avoid redundancy. This is done by avoiding duplication of data and maintains a single copy of that particular file. However in the management of big data using encrypted manner for deduplication files is tedious task. Traditional deduplication scheme cannot work in encrypted manner because encrypted data are saved as different contents by applying different encryption keys. Whereas the present scenario of deduplication scheme lacks privacy or it is suffered from weakness in providing privacy. In this paper, we propose a scheme to deduplicate encrypted data stored in cloud based on ownership challenge and proxy re-encryption. It integrates cloud data deduplication with access control. We evaluate its performance based on extensive analysis and computer simulations. The results show the superior efficiency and effectiveness of the scheme for potential practical deployment, especially for big data deduplication in cloud storage.

Index Terms: Cloud Computing, Data Deduplication, System key-operations, Access Control, Big Data.

1. INTRODUCTION

Cloud computing is nothing but a specific style of computing where everything from computing infrastructure, business application are provided as a service. It is a computing service rather than product. Cloud computing provides numerous benefits in the form of scalability, elasticity, fault tolerance and pay per use etc. Data storage service is the most popular cloud service. In cloud environment data is stored in logical pools which span over different physical storage servers. These servers owned and managed by different hosting companies called as cloud service provider (CSP). The end users (data user) for their data storage requirements buy or lease storage capacity from providers. Cloud environment provides an efficient way for centralized data storage and anywhere, anytime to that data storage access. Thus a large number of users use cloud as their data storage preference. Now the same or different users may upload duplicated data in encrypted form to CSP, especially for scenarios where data are shared among many users. Although cloud storage space is huge, data duplication greatly wastes network resources, consumes a lot of energy, and complicates data management. A cloud storage provider uses different techniques to improve storage efficiency and one of leading technique employed by them is deduplication. Consequently, deduplication becomes critical for big data storage and processing in the cloud.

Deduplication has proved to achieve highcost savings, reducing up to 90-95% storage needs for backup applications and up to 68% in standard file systems. Obviously, the savings, which can be passed back directly or indirectly to cloud users, is essential for cloud business. How to manage encrypted data storage with deduplication in an efficient way is a practical issue. However, current industrial deduplication solutions cannot handle encrypted data. Existing solutions for deduplication suffer from brute-force attacks. In practice, it is hard to allow data holders to manage deduplication due to a number of reasons. First, data holders may not be always online or available for such a management, which could cause storage delay. Second, deduplication could become too complicated in terms of communications and computations to involve data holders into deduplication process. Third, it may intrude the privacy of data holders in the process of discovering duplicated data. Forth, a data holder may have no idea how to issue data access rights or deduplication keys to a user in some situations when it does not know other data holders due to data super-distribution. Therefore, CSP cannot cooperate with data holders on data storage deduplication in many situations.

In this paper, we propose a scheme based on data ownership challenge and manage encrypted data storage with deduplication. We aim to solve the issue of deduplication in the situation where the data holder is not available or difficult to get involved. Meanwhile, the performance of data deduplication in our scheme is not influenced by the size of data, thus applicable for big data. Specifically, the contributions of this paper can be summarized as below:

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 4, April 2017

• We motivate to save cloud storage and preserve the privacy of data holders by proposing a scheme to manage encrypted data storage with deduplication. Our scheme can flexibly support data sharing with deduplication even when the data holder is offline, and it does not intrude the privacy of data holders.
• We propose an effective approach to verify data ownership and check duplicate storage with secure challenge and big data support.
• We integrate cloud data deduplication with data access control in a simple way, thus reconciling data deduplication and encryption.
• We prove the security and assess the performance of the proposed scheme through analysis and simulation. The results show its efficiency, effectiveness and applicability.

2. RELATED WORK

Deduplication techniques can be categorized into two different approaches: deduplication over unencrypted data and deduplication over encrypted data. Cloud storage service providers such as Dropbox, Google Drive, Mozy, and others perform deduplication to save space by only storing one copy of each file uploaded. However, if clients conventionally encrypt their data, storage savings by deduplication are totally lost. This is because the encrypted data are saved as different contents by applying different encryption keys. Existing industrial solutions fail in encrypted data deduplication.
In the former approach, most of the existing schemes have been proposed in order to perform a PoW process in an efficient and robust manner, since the hash of the file, which is treated as a "proof" for the entire file, is vulnerable to being leaked to outside adversaries because of its relatively small size. Thus, most of the schemes have been proposed to provide data encryption, while still benefiting from a deduplication technique, by enabling data owners to share the encryption keys in the presence of the inside and outside adversaries. Since encrypted data are given to a user, data access control can be additionally implemented by selective key distribution after the PoW process. However, not much work has yet been done to address dynamic ownership management and its related security problem.

3. PROBLEM STATEMENTS

3.1 System and Security Model:

We offering an idea to deduplicate encrypted data at Cloud Storage Provider by applying PRE to issue keys to different authorized data holders based on data ownership challenge. It is applicable in scenarios where data holders are not available for deduplication control.

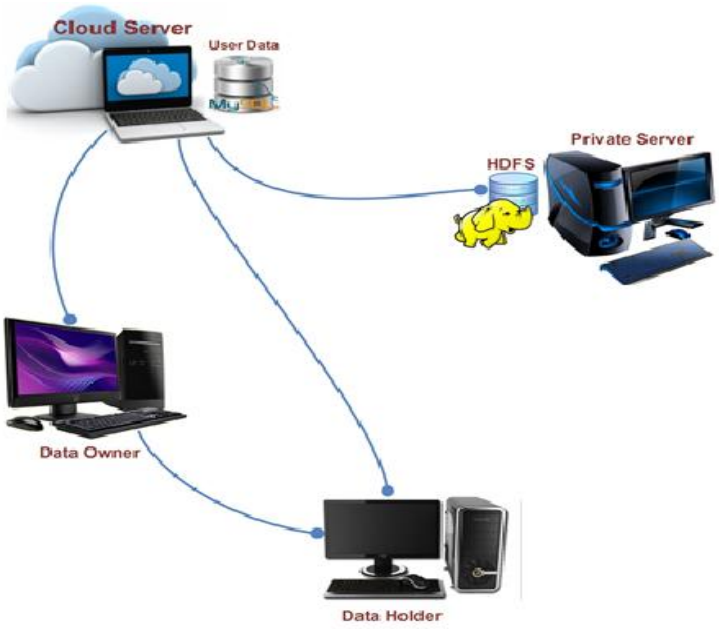

Figure Error! No sequence specified.**. System model**

As shown in Fig. 1, the system contains three types of entities: 1) Cloud server i.e. cloud service provider (CSP) that offers storage services and cannot be fully trusted since it is curious about the contents of stored data, but should perform honestly on data storage in order to gain commercial profits 2) Data user (data user and data owner) that

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 4, April 2017

uploads and saves its data at CSP. In the system, it is possible to have a number of eligible data holders (ui, i = 1, ---, n) that can save the same encrypted data in CSP. The data holder that produces or creates the file is treated as data owner. It has higher priority than other normal data holders. 3) A private server let`s say authorized party (AP) that does not collaborate with CSP and is fully trusted by the data users to verify data ownership and handle data deduplication. In this case, AP cannot know the data stored in CSP and CSP should not know the plain user data in its storage.

4. KEY OPERATIONS

Our proposed system contains the following key operations:

- **Encrypted Data Upload:** If data duplication check is negative, the data user encrypts its data using a randomly selected symmetric key DEK in order to ensure the security and privacy of data, and stores the encrypted data at CSP together with the hash used for data duplication check. The data holder encrypts DEK with private server and passes the encrypted key to CSP.
- **Data Deduplication**: Data duplication occurs at the time when data holder u tries to store the same data that has been stored already at CSP. This is checked by CSP through hash comparison. If the comparison is positive, CSP contacts private server for deduplication by providing the token and the data user's public key. The private server challenges data ownership, checks the eligibility of the data holder, and then issues a re-encryption key that can convert the encrypted DEK to a form that can only be decrypted by the eligible data holder.
- **Data Deletion:** When the data holder deletes data from CSP, CSP firstly manages the records of duplicated data holders by removing the duplication record of this user. If the rest records are not empty, the CSP will not delete the stored encrypted data, but block data access from the holder that requests data deletion. If the rest records are empty, the encrypted data should be removed at CSP.
- **Encrypted Data Update:** In case that DEK is updated by a data owner with DEK' and the new encrypted raw data is provided to CSP to replace old storage for the reason of achieving better security, CSP issues the new re-encrypted DEK' to all data holders with the support of private server.
- **Data Owner Management:** In case that a real data owner uploads the data later than the data holder, the CSP can manage to save the data encrypted by the real data owner at the cloud with the owner generated DEK and later on, AP supports re-encryption of DEK at CSP for eligible data holders.

4.1 Procedures:

- **Data Deduplication**

Figure.2 is referred from [1] which illustrates the procedure of data deduplication at CSP with the support of private server based on the proposed scheme. We suppose that user u saves its sensitive data M at CSP with protection using DEK, while user u is a data holder who tries to save the same data at CSP. The detailed procedure of data deduplication is presented below.

Step 1 - System setup as described in Section 3.

Step 2 - Data token generation: User u generates data token of M, $x1 = H\ (H\ (M) \times P)$ and sends $\{x1, pk1, (pk1)\}$ to CSP.

Step 3 - Duplication check: CSP verifies $Cert\ pk1$ and checks if the duplicated data is stored by finding whether $x1$ exists. If the check is negative, it requests data upload.
User $u1$ encrypts data M with $DEK1$ to get $CT1$ and encrypted $DEK1$ with $pk1$ to get $CK1$. $u1$ sends $CT1$ and $CK1$ to CSP, which saves them together with $x1$ and $pk1$. If the check is positive and the pre-stored data is from the same user, it informs the user about this situation. If the same data is from a different user, refer to Step 6 for deduplication.

Step 4 - Duplicated data upload and check: User $u2$ later on tries to save the same data M at CSP following the same procedure of Step 2 and 3. That is, $u2$ sends the data package $\{x2, pk2, (\ 2)\}$ to CSP. Duplication happens because $x2$ exists, so CSP forwards $\{x2, pk2, (\ 2)\}$ to AP.

Step 5 – Ownership challenge: AP challenges the data ownership of $u2$ by randomly choosing $c \in$r (0, ---, 2r – 1) and sending it to $u2$. $u2$ checks c to make sure that $0 \leq c \leq 2r - 1$, computes $y = (M) + (s2 \times c)$ and sends $(pkap, y)$ to private server. Private server gets y, computes $H\ (yP + cV2)$ and compares it with $x2$. If $H\ (yP + cV2) = x2$, i.e., the ownership challenge is successful, private server generates re-encryption key $rkap \rightarrow u2$ by calling $(pkap; skap; pk2)$ if it has not been generated and issued to CSP.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 4, April 2017

Step 6 - Deduplication: CSP re-encrypts (pkap, $DEK1$) by calling (rkap→u2; (pkap; $DEK1$) = ($pk2$; $DEK1$) and provides the re-encrypted key ($pk2$, $DEK1$) to $u2$. Then $u2$ can get $DEK1$ with its secret key $sk2$. $u2$ confirms the success of data deduplication to CSP that records corresponding deduplication information in the system after getting this notification.

At this moment, both $u1$ and $u2$ can access the same data M saved at CSP. User $u1$ uses $DEK1$ directly, while $u2$ gets to know $DEK1$ by calling {$sk2$; ($pk2$; $DEK1$)}.

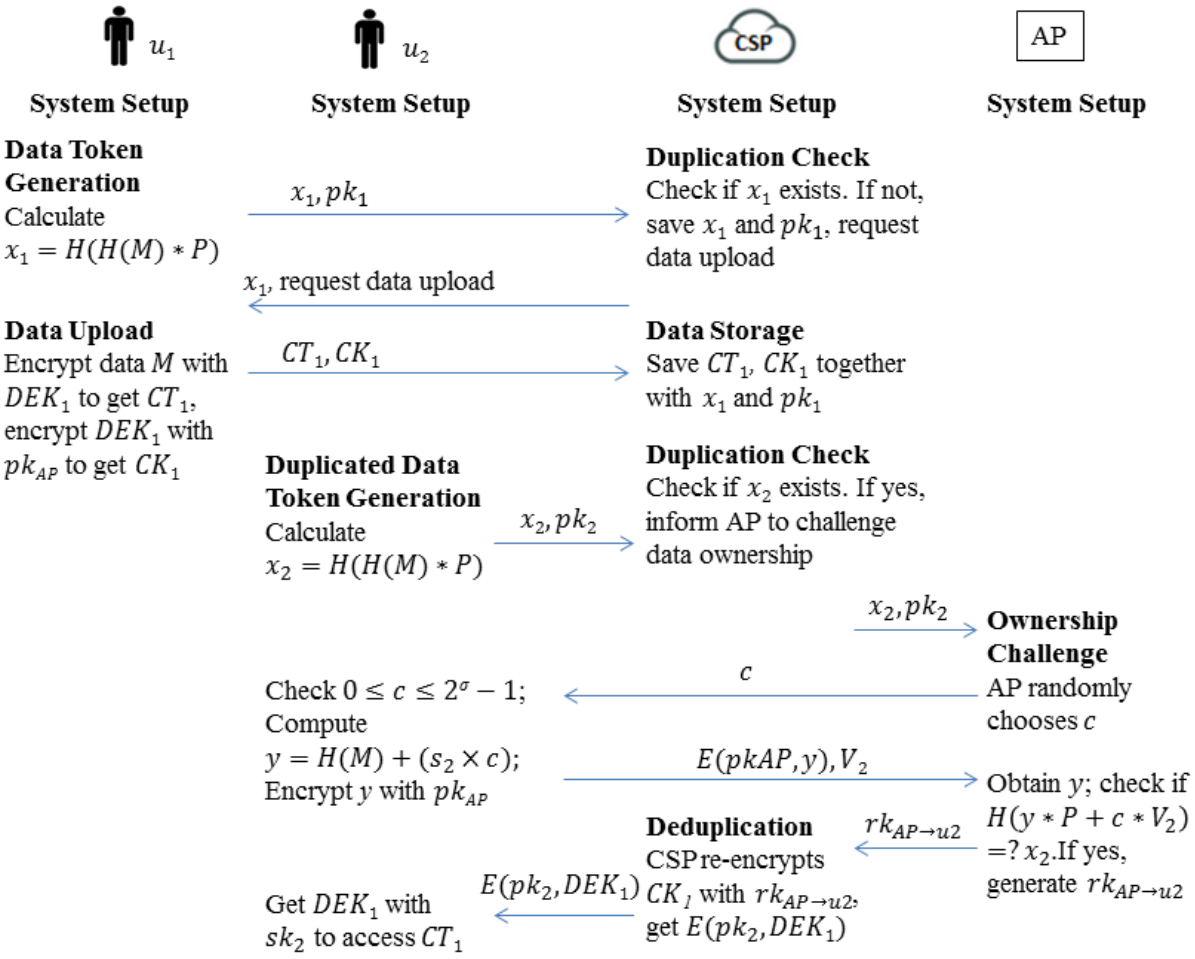

Figure 2. procedure_of_data_deduplication, referred from [1]

- **Data Deletion at CSP**

When data holder $u2$ wants to delete the data from CSP, it sends deletion request to CSP: ($pk2$), $x2$. CSP checks the validity of the request, and then removes deduplication record of $u2$, and block $u2$'s later access to M. CSP further checks if the deduplication record is empty. If yes, it deletes encrypted data CT and related records.

- **Encrypted Data Update**

In some cases, a data holder could update encrypted data stored at CSP by generating a new DEK` and upload the newly encrypted data with $DEK$` to CSP. As illustrated in Figure.4 which is referred from [1], $u1$ wants to update encrypted data stored at CSP with new symmetric key DEK`1. User $u1$ sends an update request: {$x1$, $CT$`1, CK`1, $update$ $CT1$}. CSP saves $CT$`1, CK`1 together with $x1$ and $pk1$. CSP contacts private server for deduplication for other data holders if their re-encryption keys are not known. Private server checks its policy for generating and sending corresponding re-encryption keys (e.g., rkap→u2), which are used by CSP to perform re-encryption on CK`1 for generating re-encrypted keys that can be decrypted by all eligible data holders (e.g., $E(pk2, DEK`1)$). The re-encrypted keys are then sent to the eligible data holders for future access on data M. Any data holder can perform the encrypted data update. Based on storage policy and service agreement between the data holder and CSP, CSP decides if such an update can be performed.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 4, April 2017

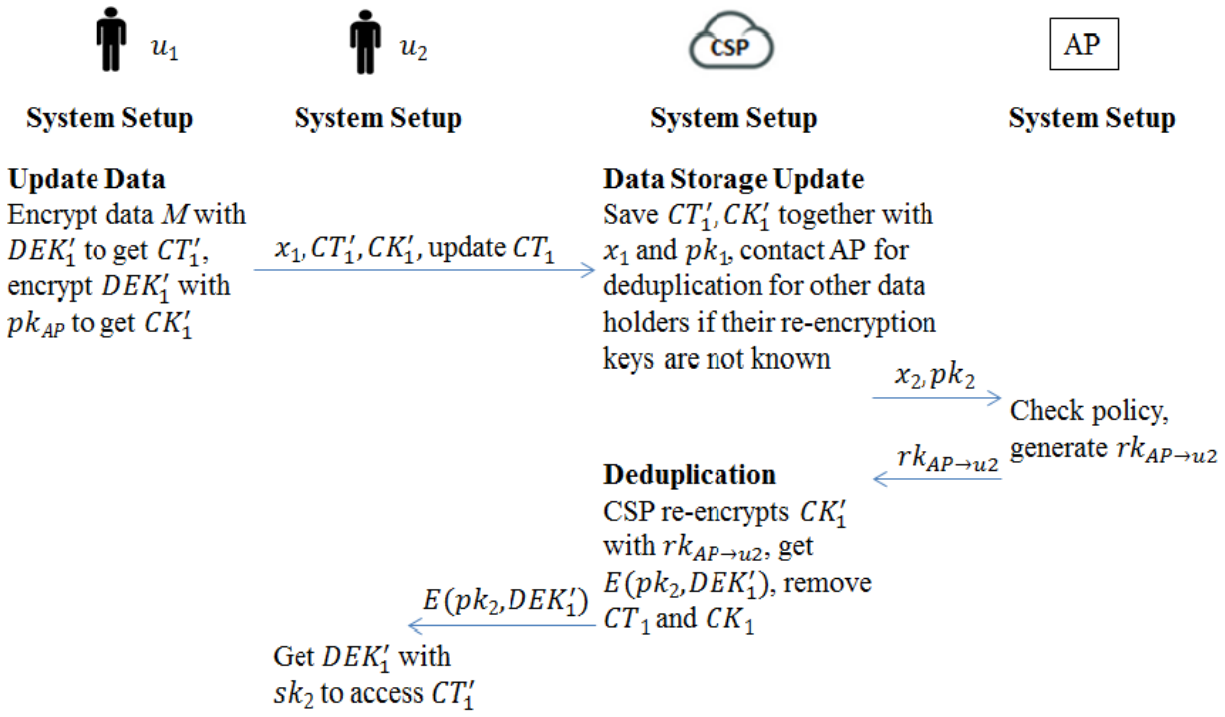

Figure 4. procedure_of_data_update, referred from [1]

5. FURTHER DISCUSSIONS

5.4 Performance Evaluation:

We implemented the proposed scheme and tested its performance. Table 1 describes our implementation and testing environments. We applied a MySQL database to store metadata files and related information. In our test, we did take into account the time of data uploading and downloading. We focused on testing the performance of the deduplication procedure and algorithms designed in our scheme.

Table Error! No sequence specified.**. Test Environments**

Hardware Environment	For both server (private server, Cloud service provider)- CPU: Intel Core 2 Quad CPU Q9400 2.66GHZ (minimum) Memory: 4GB SDRAM (minimum)
Software Environment	Operating System: Ubuntu v14.04, Windows Programming Environment: Netbeans IDE, Java Server – Glassfish server Secure Protocol: SOAP, XML Library: Pairing Based Cryptography (http://crypto.stanford.edu/pbc/) Database: MySQL v5.5.41, Hadoop

In this experiment, we tested the operation time of data encryption and decryption with AES by applying different AES key sizes (196 bits) and different data size (from 10 kb to 600 kb). The testing environment was Intel Core i5-3337U CPU 2.6 GHz 6.00GB RAM, Ubuntu v13.10 6.0GB RAM, Quad- Core processor, 1TB Hard disk. We observed that even when the data is as 600KB, the encryption/decryption time is less than 13 seconds if applying 196-bit AES key. Applying symmetric encryption for data protection is a reasonable and practical choice. The time spent on AES encryption and decryption is increased with the size of data. This is inevitable in any encryption schemes. Since AES is very efficient on data encryption and decryption, thus it is practical to be applied for big data as prospective their software and hardware requirements.

The proposed scheme has the following additional advantages.

- **Flexibility:** The proposed scheme can flexibly support access control on encrypted data with deduplication. One data holder can flexibly update DEK. The new key can be easily issued to other data holders or eligible data users

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 4, April 2017

by CSP with a low cost, especially when private server has issued the re-encryption key already. Data revocation can be realized by blocking data access at CSP and rejecting key re-encryption on a newly applied key DEK'.

- **Low cost of storage:** The scheme can obviously save the storage space of CSP since it only stores one copy of the same data that is shared by data owner and data holders. Storing deduplication records occupies some storage or memory for saving token pki and xi (only 1024+160 bits). But comparing with the big volume of duplicated data, this storage cost can be ignored.
- **Big data support:** The proposed scheme can efficiently perform big data deduplication. First, duplicated big data upload is efficient because only xi and pki are sent to CSP. CSP performs hash comparison and then contacts private server to challenge ownership for issuing a re-encryption key. The computation and communication cost of this process (involving ownership challenge, re-encryption key generation, CK re-encryption and re-encrypted key decryption) is not influenced by the size of big data. Second, uploading ciphertext CT is unavoidable in almost all schemes for deduplication. The proposed scheme only introduces a bit extra communication load (i.e., CK) and a little bit additional communication cost for ownership challenge. Compared with big data upload cost and storage cost, they are very minor and efficient.

6. CONCLUSION

Managing encrypted data with deduplication is important and significant in practice for achieving a successful cloud storage service, especially for big data storage. In this paper, we proposed a practical scheme to manage the encrypted big data in cloud with deduplication based on ownership challenge. Our scheme can flexibly support data update and sharing with deduplication even when the data holders are offline. Encrypted data can be securely accessed because only authorized data holders can obtain the symmetric keys used for data decryption. Extensive performance analysis and test showed that our scheme is secure and efficient under the described security model and very suitable for big data deduplication. The results of our computer simulations further showed the practicability of our scheme.

REFERENCES

[1] Zheng Yan, Senior Member, IEEE, Wenxiu Ding, Xixun Yu, Haiqi Zhu, and Robert H. Deng, Fellow, IEEE, "Deduplication on Encrypted Big Data in Cloud" IEEE TRANSACTIONS ON JOURNAL NAME, MANUSCRIPT ID.
[2] M. Bellare, S. Keelveedhi, and T. Ristenpart, "DupLESS: Server Aided Encryption for Deduplicated Storage," Proceedings of the 22nd USENIX Conference on Security, 2013, pp. 179-194.
[3] Dropbox, "A File-Storage and Sharing Service," http://www.dropbox.com/.
[4] Google Drive, http://drive.google.com.
[5] Mozy, "Mozy: A File-storage and Sharing Service," http://mozy.com/.
[6] Ajay Jangra, Vandna Bhatia, Upasana Lakhinazand, Niharika Singhx, "An Efficient Storage Framework design for Cloud Computing: Deploying Compression on De-duplicated No-SQL DB using HDFS" 2015 1st International Conference on Next Generation Computing Technologies (NGCT-2015) Dehradun, India, 4-5 September 2015.
[7] Vivek Waghmare, Smita Kapse, Purnima Selokar, "Implementation of Authorized Deduplication: An Approach for Secure Cloud En" International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3297: 2007 Certified Organization) Vol. 4, Issue 4, April 2016.
[8] J.R. Douceur, A. Adya, W.J. Bolosky, D. Simon, and M. Theimer, "Reclaiming Space from Duplicate Files in a Serverless Distributed File System," Proceedings of IEEE International Conference on Distributed Computing Systems, 2002, pp. 617 -624, doi:10.1109/ICDCS.2002.1022312.
[9] G. Wallace, F. Douglis, H. Qian, P. Shilane , S. Smaldone, M. Chamness, and W. Hsu, "Characteristics of Backup Workloads in Production Systems," Proceedings of USENIX Conference on File and Storage Technologies, 2012, pp. 1-16.