Volume 3, Issue 1, January-2016, pp. 23-29 ISSN (O): 2349-7084

International Journal of Computer Engineering In Research Trends

Available online at: www.ijcert.org

Efficient Technique to Optimize Cloud Storage in Multi-Tenant Environment

*P Jyothi **

Assistant Professor, CSE Dept., P.V.K.K Institute of Technology, Anantapuramu

Abstract- With the advent of cloud computing technologies, a paradigm of migrating on-premise cloud storage management to multitenant cloud storage management is been an active challenge for the researchers. Storage management in cloud environment aims to enhance performance of the data center by efficiently optimizing the underlying storage resources. The vital concept behind the centralized multitenant storage management is defining an integrated platform for data analytics, collection and governance which in turn increases the efficiency and effectiveness of the multitenant storage environments. This paper introduces an optimized multitenant cloud storage framework that incorporates the storage data analytics that suggest the service recommendation for the storage administrator. In addition, secured cloud storage architecture is presented to assure the security of the data stored in the multitenant cloud storage system.

Key words— Cloud Storage, multitenant cloud, Storage management, security, data analytics.

1. Introduction

Cloud computing[1] is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. In this Information age, several organizations posses huge amount of data which needs to be kept secured. These data includes personal information, health information and financial data. Local maintenance of such huge amount of data will be cost effective and problematic. Hence Cloud Service Provider offered Storage as a Service to alleviate the burden of huge local data storage and to reduce the cost by means of outsourcing data storage to the cloud. Since the data owner outsources their sensitive data to the cloud, they want their data to be guaranteed with some security concerns like confidentiality, integrity and proper access control. In some practical applications data confidentiality is not only a security concern but also a juristic issue. For example in e-Health applications in USA the usage and exposure of data should satisfy the policies confessed by Health Insurance Portability and Accountability Act (HIPAA) [2], thus keeping the privacy of the outsourced data on the cloud is not an option, but it is a demand. Confidentiality can be guaranteed by encrypting the data before outsourcing it to the remote server. Also the outsourced data should not be modified by unauthorized users. Traditional access control techniques assume that the data owner and the storage servers in the same trust domain. However this assumption no longer holds when the data is outsourced to the cloud storage, which takes full maintenance of the outsourced data, and it, is untrusted by the data owner.

1.1 Principles of Cloud Storage Services

In this section, we shall explore the main feature of the clouted storage services. These features are copy, backup, synchronization, and file sharing. Any storage service must include at least one of these features, and may include multiple features at the same time. Figure.1 shows the features of cloud storage services.

COPY: The copy feature creates an image of the user's local data into cloud. The user uses this copy to make sure that his data is always available even if there is a local hardware failure (e.g. a hard disk crash).Moreover, this copy will help the user to access his data from any place (e.g. through web browser) even if his local hardware is not available.

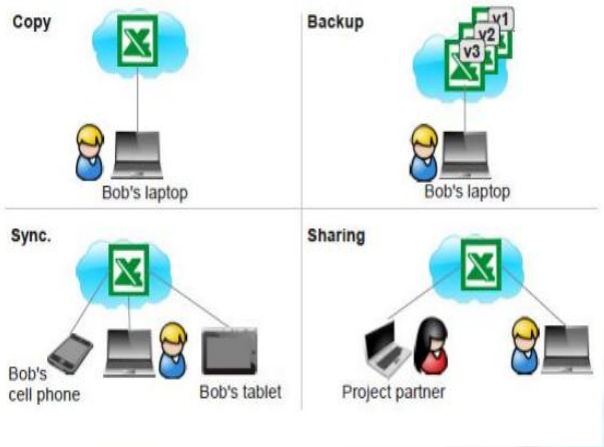


Figure .1 Features of Cloud Storage Services

This gives the user two ways to store his data in the cloud. In the first way, the user manually uploads his files and folders into the cloud through the web browser. while in the second way, the user make use of a client software, that is locally installed in user's machine ,to automatically upload his files and folders from a given folder belonging to the client to the cloud storage. It is also important to note that the copy feature is different from backup feature.

BACKUP: The backup feature allows users to restore any previously stored version of a file or a folder over a long period of time (usually years).To create backups, the cloud storage services usually make use of an automatic process. In this process, data is copied, transmitted, and stored periodically in the cloud so as to be recovered in case of original data loss. To perform this process, the cloud storage service providers have to offer client software that is installed locally in the user's machine. This software enables the users to select the data to be backed up, to configure the retention period, and schedule for backups. In addition, this software can either run continuously in background or is configured to perform the backup on regular basis so as to backup the newly created or changed files

SYNCHRONIZATION: Synchronization means having consistency among data stored in different sources. For example, a user can own a set of devices, e.g. a pc, laptop, tablets and a Smartphone, Page 73 and wants to have the same data available on all devices and whenever data is changed in one device the others are modified with these changes. Therefore, the cloud storage service providers have to provide users with client software that is able to detect any changes in any file in any device and reflect these changes to other devices.

DATA SHARING: Data sharing is the process of sharing data files or folders with others. Cloud storage service providers offer users different forms of sharing. Users can share data with other subscribers of the same service, with a closed group of people from other services, or with everybody. For example, users can collaborate with colleagues, project partners, or friends.

2. Related Work

Data security in untrusted third parties in general and in the cloud in particular is often provided through access control policies and cryptographic methods [4]. As typical examples, Yu et al. introduced a fine-grained access control for data in untrusted cloud storage [5] based on a combination of attribute-based encryption, proxy re-encryption, and lazy re-encryption whereas Yarlagadda et al. proposed an encryption scheme [6] that integrates Playfair and Vigenere ciphers with the structural aspects of DES and S-DES.

To further support data sharing, Kang et al. proposed an IdentityBased Authentication scheme by which the owner can share his encrypted data stored in the cloud [7] while Zhao at el. [8] presented a progressive encryption system based on Elliptic Curve Cryptography that allows the owner to share his encrypted data with other consumers without revealing the plaintext data. Additionally, to address the issue of key loss, Huang et al. presented a key recovery scheme in YiCloud [9].

On the other hand, considering public audit-ability, Wang et al. presented a model on cloud storage for data integrity verification based on Merkle hash tree [10] while Almualla et al. designed a new security architecture with sharing keys [11] that fulfils all security requirements in an environment that supports lawful interception. Finally, Vimercati et al. proposed an approach to address the integrity of join queries in unreliable computational services [12]. More recently the concept of DEaaS has been introduced in many cloud platforms, from commercial ones such as Amazon and Google to open-source ones such as Swift in Open Stack and Cumulus in Nimbus [1], [2], [3]. Besides DEaaS, the authors of [4] introduced a Privacy as a Service, which was implemented by a set of protocols to ensure the security of data in cloud architecture.

On the other hand, authors of [5] proposed a Secret Storage as a Service that is designed to securely storing, tracking, and controlling access to digital secrets such as cryptographic keys and hashed passwords. Our proposed framework is different from these services in the sense that we provide a full life-cycle of data security management and maximize the flexibility of users in using the data encryption service. The closest works to ours are [6] and [7]. Specifically, with respect to the work in [6], our service is similar because both services target multicloud environments and provide several options for users to customize the services. The

difference, however, is that the security service in that work focuses on protecting the VM with firewalls and security tools while our focus is specifically on data encryption.

On the other hand, while [7] shares the ideas with our work in providing data encryption as a service, the architecture and design principles of the two works are very different found in the areas of integrity verification of remotely stored data and file encryption schemes in distributed systems and access control mechanisms over outsourced data. Ateniese et al. [4] designed a model based on PDP (Provable Data Possession) protocol which allows a client to verify the server's data possession. In this scheme the client preprocesses the file and generates meta-data, stores it locally, and then outsource the file to the server. The server stores the file and starts respond to challenges issued by the client. Integrity verification is done through batch verification of homomorphic hash functions.

Curtmola et al. [5] designed a model based on MRPDP which uses replication in order to improve data availability and reliability. By storing multiple copies, if some copies are destroyed still the data can be recovered from the remaining copies. But challenges incur relatively more cost in MR-PDP.

Dodis et al. [6] presented a model based on POR (Proofs of Retrievability) in which the client stores a file F on a server and keeps only a short private verification string locally. Later, the client can run an audit protocol to verify the server's data possession, in which the client acts as a verifier and the server proves that it possesses the data. POR is a complementary approach to PDP, and is stronger than PDP in the way that it can be reconstructed from the portions of the data which are reliably stored on the remote server. Kallahalla et al. [7] presented a cryptographic based file system called Plutus: Scalable secure file sharing on untrusted storage, which enforces access control over outsourced data. In which a file is divided into blocks and each block is encrypted with File-block key and each File-block key is encrypted with File- lockbox key. If the data owner wants to share the file with his clients he just distributes the File- lockbox key to them.

3. Storage Resilience Optimization Technique

This paper presents the Storage Resiliency License Optimization, where the goal is to recommend a storage volume consolidation plan to improve the efficiency of storage resiliency license usage. The challenge arises from the unique characteristics of storage resiliency sessions.

Storage resilience solutions are designed for data protection usually in the process of replication pair relationship. To enable the replication relationship the devices hosting the primary and the secondary volume must have the specific resilience. I

The consolidation example is depicted in figure 2.

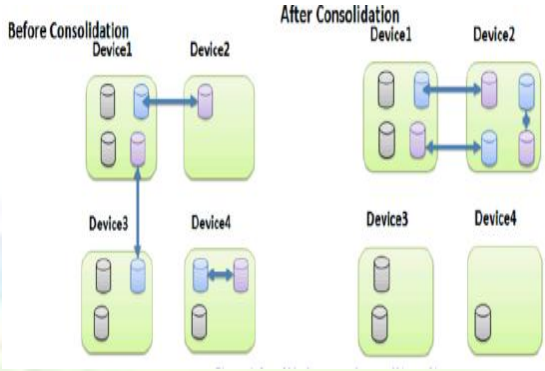


Figure.2 Consolidation can reduce the Resilience

The consolidation outcome is illustrated in the figure 2. For example, there are three replication sessions as shown in the "Before Consolidation" diagram. We can observe that all four storage devices are part of a replication session and thus four software licenses are needed. However, if we meticulously migrate the storage volumes to new locations as specified n "After Consolidation" diagram, we can observe that only two storage devices are sufficient to support all replication sessions. Although the licenses fees on Device 3 and Device 4 might have already been expensed, the process of consolidation planning would significantly benefit the storage administrator and service provider for future budget planning and costaware optimization, where the exact saving depends on the license pricing model and charging policy. While the example shown above is simplistic, the objective is to demonstrate that by carefully consolidating storage volumes involved in replication sessions, we can significantly reduce the license cost without affecting the replication functionalities.

3.1 Resilience optimization algorithm:

Step 1: *Identify session and Device properties .Each volume in the session is represented by set of volume specific attributes*

In storage environments, identify all the volumes that are in any resiliency relationship denote the set of volumes as 'V'. For each volume $i \in V$ it is denoted as follows:

- ***l_i : indicates the type of relationship***
- ***I_i : The estimated average I/O demand of this volume i.***
- ***C_i : The volume size***

Similarly in the storage environments identify all the storage devices as D. For each device that support the resilience sessions denote the set of devices as J∈D.

- ***L_j: The type of relationship that device can support.***
- ***N_i: The maximum number of resiliency session that the device can support***

Step 2: *After obtaining the information of V and D calculate the constraint constants as*

- *Compatibility constraints: For each volume i ∈ V and device J∈D calculate $S_{ij} = 1 \leftrightarrow L_i \in L_j$ else $S_{ij} = 0$*
- *Anti-affinity constraints: For each pair of volume $P_{ij} = 1$ if volume i and k are on the same device else $P_{ij} = 0$*

Step 3:

$$\min \quad \sum_j y_j \tag{1}$$

$$\text{s.t.} \quad \sum_j x_{i,j} = 1 \quad \forall i \in V \tag{2}$$

$$\sum_i I_i x_{i,j} \le RI_j y_j \quad \forall j \in D \tag{3}$$

$$\sum_i C_i x_{i,j} \le RC_j y_j \quad \forall j \in D \tag{4}$$

$$x_{i,j} \le s_{i,j} \quad \forall i,j \tag{5}$$

$$x_{i,j} x_{k,j} p_{i,k} = 0 \quad \forall i,k \in V \& j \in D \tag{6}$$

$$x_{i,j} q_{i,k} = x_{k,j} q_{i,j} \quad \forall i,k \in V \& j \in D \tag{7}$$

$$\sum_i x_{i,j} \le N_j \quad \forall j \in D \tag{8}$$

$$x_{i,j} \in \{0,1\} \quad \forall i \in V, j \in D \tag{9}$$

$$y_j \in \{0,1\} \quad \forall j \in D. \tag{10}$$

In the formulation, the objective function (1) is the number of total storage devices we use to consolidate all volumes in a replication relationship (the objective function to be minimized). The constraint (2) ensures that the same volume can be only placed at one storage device. The constraints (3) and (4) are applied to impose limits on the maximum IO load and capacity that a storage device can support. The constraint (5) specifies that a volume can only be allocated on a storage device that has the specific license and storage type (e.g., SSD pool) that the volume requires. The constraints (6) and (7) require that the anti-affinity and affinity constraints must be satisfied. The constraint (8) implies that the number of replication sessions cannot exceed a storage device specific bound.

Therefore, we can formulate the cost-aware replication license consolidation problem in an integer programming framework, where the variables are placement variables and on-off variables. The obtained solution of on-off variables will indicate among all eligible storage devices, which set of devices, should be used to host the consolidated replication sessions. The obtained solutions of placement variables will indicate the placement of volumes on this set of storage devices.

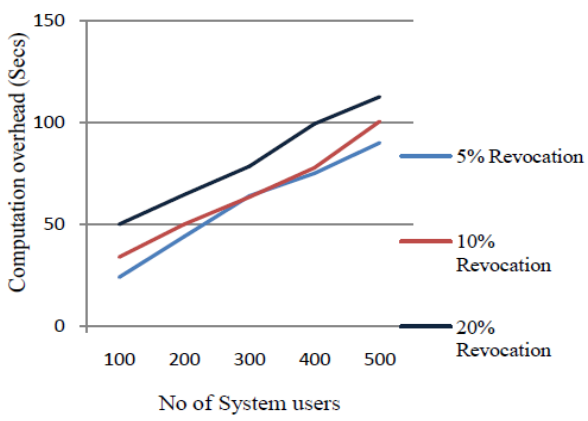
As a result, the outcome of our formulation is a volume to storage device mapping. Finally, the mapping results can be associated with a storage volume migration plan which will perform the consolidation work by moving the volumes to their designated devices. Another option is that the obtained allocation result can be used as one of the inputs for other license optimization frameworks such as server license optimization with other performance objective functions, e.g., reducing the overall licenses used by the data center while maintaining a tolerable application level response time.

4. Performance Analysis

We evaluate the performance of the proposed scheme by analyzing storage and computation overhead. The data file we have used for our experiments is of size 10GB with block size of 100MB.

Storage overhead. This is the additional storage space required to store necessary information other than the outsourced file F. An entry of BST at the owner side is of 8bytes, and the no of entries will be equal to number of blocks q of the file F. Likewise, at the CSP side the additional storage of BST requires 8q bytes, where q is the number of blocks. Each may require 800 MB storage.

Computation overhead The computation cost for encrypting the data before outsourcing, and the dynamic operations require hash function, encryption, BrdEnc and it may require FR forward rotation if there is a revocation. To reflect latest version of the outsourced data, the TTPA updates the combined hash values for the file F and BST. Therefore the computation overhead on the TTPA side is 4h. On accessing data from CSP the authorized user has to verify the two signatures generated by CSP on F and BST, and verifies the data file and entries. The computation overhead is calculated as follows.



5. Conclusion

Centralized storage management analytics can offload the overhead and cost for each individual tenet storage environment. Aggregation of the metadata across multiple tenets can also enable the population based data analytics that specifies the storage resiliency optimization technique that produces the specific framework to capture the constraints. The data owner enforces access control for the outsourced data by combining three cryptographic techniques: broadcast encryption, lazy revocation, and key rotation. The experimental results show that the proposed scheme is a robust model in terms of security

6. References

[1] "Amazon Encryption Service," http://docs.aws.amazon.com/amazonS3/latest/dev/serv-side-encryption.html.

[2] "Google Cloud Storage," http://googlecloudplatform.blogspot.co.uk/ 2013/08/google-cloud-storage-now-provides.html.

[3]"ESCUDO-CLOUD," http://www.escudocloud.eu/index.php.

[4] S. D. C. D. Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Encryption policies for regulating access to outsourced data," ACM Trans. Database Syst., vol. 35, no. 2, May 2010.

[5] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing," in Proceedings of the IEEE INFOCOM, March 2010.

[6] V. K. Yarlagadda and S. Ramanujam, "Data Security in Cloud Computing," Journal of Computer and Mathematical Sciences, no. 1, pp. 15–23, 2011.

[7] L. Kang and X. Zhang, "Identity-Based Authentication in Cloud Storage Sharing," in Proceedings of the International Conference on Multimedia Information Networking and Security (MINES), November 2010.

[8] G. Zhao, C. Rong, J. Li, F. Zhang, and Y. Tang, "Trusted Data Sharing over Untrusted Cloud Storage Providers," in Proceedings of the 2nd International Conference on Cloud Computing Technology and Science (CloudCom), November 2010.

[9] Z. Huang, Q. Li, D. Zheng, K. Chen, and X. Li, "YI Cloud: Improving user privacy with secret key recovery in cloud storage," in Proceedings of the 6th International Symposium on Service Oriented System Engineering (SOSE), December 2011.

[10] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," IEEE Transactions on Parallel and Distributed System, 2011.

[11] S. A. Almulla and C. Y. Yeun, "New secure storage architecture for
cloud computing," in Proceedings of the International Conference on Future Information Technology (FutureTech), 2011.

[12] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Integrity for join queries in the cloud," Cloud Computing, IEEE Transactions on, vol. 1, no. 2, July 2013.

