B.AmarNadh Reddy, P.Raja Sekhar Reddy / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 5, September- October 2012, pp.1130-1134

Effective Data Distribution Techniques for Multi-Cloud Storage in Cloud Computing

B.AmarNadh Reddy[*1], P.Raja Sekhar Reddy[2]
*[1]CSE, Anurag Group of Institutions, Hyderabad, A.P, India
[2]CSE, Anurag Group of Institutions, Hyderabad, A.P, India

Abstract

Cloud data storage redefines the issues targeted on customer's out-sourced data (data that is not stored/retrieved from the costumers own servers). In this work we observed that, from a customer's point of view, relying upon a solo SP for his outsourced data is not very promising. In addition, providing better privacy as well as ensure data availability and reliability can be achieved by dividing the user's data block into data pieces and distributing them among the available SP s in such a way that no less than a threshold number of SP s can take part in successful retrieval of the whole data block. In this paper, we propose a traditional technique followed to distribute data to multi-cloud storage model in cloud computing which holds an economical distribution of data among the available SPs in the market, to provide customers with data availability as well as reliability . Data fragmentation plays an important role in data distribution

Keywords Cloud computing, storage, Cloud service provider, customer .Fragmentation

I. INTRODUCTION

The end of this decade is marked by a paradigm shift of the industrial information technology towards a subscription based or pay-per-use service business model known as *cloud computing*. This paradigm provides users with a long list of advantages, such as provision computing capabilities; broad, heterogeneous network access; resource pooling and rapid elas-ticity with measured services .Huge amounts of data being retrieved from geographically distributed data sources, and non-localized data-handling requirements, creates such a change in technological as well as business model. One of the prominent services offered in *cloud computing* is the *cloud data storage*, in which, subscribers do not have to store their own data on their servers, where instead their data will be stored on the cloud service provider's servers. In cloud computing, subscribers have to pay the provides for this storage service. This service does not only provides flexibility and scalability data storage, it also provides customers with the benefit of paying only for the amount of data they needs to store for a particular period of time, without any concerns of efficient storage mechanisms and maintainability issues with large amounts of data storage. In addition to these benefits, customers can easily access their data from any geographical region where the Cloud Service Provider's network or Internet can be accessed [1]. An example of the cloud computing is shown in Fig. 1. Since *cloud service providers* (*SP*) are separate market entities, data integrity and privacy are the most critical issues that need to be addressed in *cloud computing*. Even though the cloud service providers have standard regulations and powerful infrastructure to ensure customer's data privacy and provide a better availability, the reports of privacy breach and service outage have been apparent in last few years

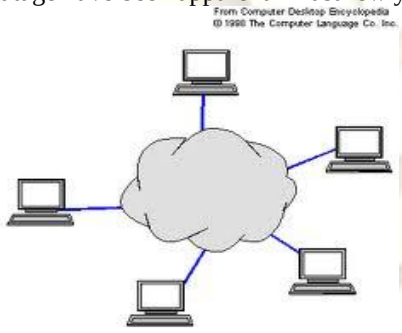


Fig. 1. Cloud computing architecture example

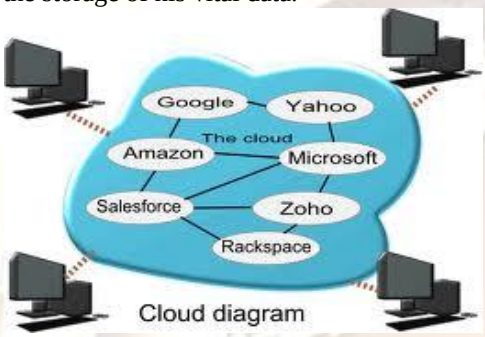
In this work we observed that, from a customer's point of view, relying upon a solo SP for his outsourced data is not very promising. In addition, providing reliability as well as ensure data availability, can be achieved by dividing the user's data block into data pieces and distributing them among the available sp's.

To address these issues in this paper, we proposed the techniques for distribution of data among the available SP s in the market, to provide customers with data availability as well as reliability [1]. In our model, the customer divides his data among several SP s available with respect to data access quality of service offered by the SP s at the location of data retrieval. This not only rules out the possibility of a SP misusing the customers' data, breaching the privacy of data, but can easily ensure the data availability with a better quality of service.

Customers' stored data at cloud service providers is vulnerable

to various threats. Previous studies in [9], [11] discussed in detail that a cloud service provider can be a victim to Denial
of service attacks or its variants.

we consider two types of threat models. First is the single point of failure [5], [6], which will affect the data availability, that could occur if a server at the cloud service provider failed or crashed, which make it hard for the costumer to retrieve his stored data from the server. Availability of data is also an important issue which could be affected, if the cloud service provider (SP) runs out of business. Such worries are no more hypothetical issues, therefore, a cloud service customer can not entirely rely upon a solo cloud service provider to
ensure the storage of his vital data.



Cloud diagram

To illustrate this threat we use an example in Fig. 2. Let us assume that three customers (C1, C2 and C3) stored their data on three different service providers (CSP1, CSP2 and CSP3) respectively. Each customer can retrieve his own data from the cloud service provider who it has a contract with. If a failure
occur at CSP1, due to internal problem with the server or some issues with the cloud service provider, all C1's data which was stored on CSP1's servers will be lost and cannot be retrieved. One solution for this threat is that, the user will seek to store his data at multiple service providers to ensure better availability of his data.
Our second threat discussed in this paper is the *colluding service providers*, in which the cloud service providers might collude together to reconstruct and access the user stored
data.
In [9] the authors provide the idea for distributing the data among two storage clouds such that, an adversary cannot retrieve the contents of the data without having access toboth the storage clouds. Relaying entirely upon a couple of service providers for the storage and retrieval of data might not be secured against colluding service providers. Such an attack scenario is entirely passive, because the cloud user cannot detect that his information has been collectively retrieved from the service providers without his consent. We illustrate the colluding service providers' threat .Let us assume that two cloud service providers are available for customer (C1), who want to store his own data securely. In here he will divide his data into two parts (D1 and D2) and distribute these parts on the two available *CSPs* (CSP1 and CSP2) respectively. The two cloud service providers might collude with each other, and Exchange the parts of data that the customer has stored on their server and reconstruct the whole data without being detected
by the user

Our proposed approach will provide the cloud computing users a decision model, that provides a better reliability and availability by distributing the data over multiple cloud service providers in such a way that, none of the SP can successfully retrieve and use it.

II. DATA DISTRIBUTION

Data preservation and data integrity are two of the most critical security issues related to user data[2],[11]. In conventional paradigm, the organizations had the physical possession of their data, and thus have an ease of implementing better data availability policies. But in case of cloud computing, the data is stored on an autonomous business party, that provides data storage as a subscription service. The users have to trust the *cloud service provider* (*SP*) with security of their data. In the author discussed the criticality of the privacy issues in cloud computing, and pointed out that obtaining an information from a third party is much more easier than from the creator himself.

One more bigger concern that arises in such schemes of cloud storage services, is that, there is no full-proof way to be certain that the service provider doe not retains the user data, even after the user opts out of the subscription. With enormous amount of time, such data can be decrypted and meaningful information can be retrieved and user privacy can easily be breached. In order to stop the SP to observe the data ,data can be fragmented and distributed to several SP's .

Why Fragment?

- Usage

- Applications work with views rather than entire relations.
- Efficiency
- Data is stored close to where it is most frequently used.
- Data that is not needed by local applications is not stored

- Parallelism

- With fragments as unit of distribution, transaction can be divided into several

sub-queries that operate on fragments. [11]

- Security

Data not required by local applications is not stored and so not available to unauthorized users

Types of Fragmentation

- Four types of fragmentation:
- Horizontal, [11]
- Vertical,
- Mixed,
- Derived.
- Other possibility is no fragmentation:

If relation is small and not updated frequently, may be better not to fragment relation

Horizontal Fragmentation

- Each fragment consists of a subset of the tuples of a relation R. [11]
- Defined using Selection operation of relational algebra:
  - $\sigma_p(R)$
- Example:
  - Relation: Sells(pub, address,price,type)
  - Fragments:
    - » SellsBitter= $\sigma_{type = \text{“bitter”}}$(Sells)
    - » SellsLager= $\sigma_{type = \text{“lager”}}$(Sells)

This strategy is determined by looking at predicates used by transactions.

- Involves finding set of *minimal* (*complete* and *relevant*) predicates.
- Set of predicates is *complete*, if and only if, any two tuples in same fragment are referenced with same probability by any application.
- Predicate is *relevant* if there is at least one application that accesses fragments differently.

Vertical Fragmentation

- Each fragment consists of a subset of attributes of a relation R. [11]
- Defined using projection operation of relational algebra:
  - $\Pi_{a_1,\ldots a_n}(R)$
- Determined by establishing *affinity* of one attribute to another.
- Example:
  - Relation: Bars(name,address,licence,employees,owner)
  - Fragments:
    - » $\Pi_{name,address,licence}$ (Bars)
    - » $\Pi_{name,address,employees,owner}$(Bars)

Mixed Fragmentation

- We can also mix horizontal and vertical fragmentation.
- We obtain a fragment that consist of an horizontal fragment that is vertically fragmented, or a vertical fragment that is horizontally fragmented.
- Defined using Selection and Projection operations of relational algebra.

$\sigma_p(\Pi_{a_1,\ldots a_n})\ \Pi_{a_1,\ldots a_n}(\sigma_p)$

Derived Horizontal Fragmentation

- A horizontal fragment that is based on horizontal fragmentation of a parent relation.[11]
- Ensures that fragments that are frequently joined together are at same site.
- Defined using *Semijoin* operation of relational algebra:

$R_i = R >_F S_i, \quad 1 \le i \le w$

- If relation contains more than one foreign key, need to select one as parent.
- Choice can be based on fragmentation used most frequently or fragmentation with better join characteristics.

How can we define fragments correctly
In defining fragments we have to be very careful.

Three correctness rules:
Completeness:

If relation R is decomposed into fragments r_1, r_2, …r_n, each data item that

can be found in R must appear in at least one fragment. This ensures no loss of data during fragmentation[11]

Reconstruction:
we must be able to reconstruct the entire R from fragments.For horizontal fragmentation is union operation.[11]

$$R = r_1 \cup r_2 \cup \ldots \cup r_n,$$

For vertical fragmentation is natural join operation. $R = r_1 >< r_2 >< \ldots >< r_n$,

To ensure reconstruction we have to include primary key attributes in all fragments

Disjointness

If data item x appears in fragment r_i, then it should not appear in any other fragment.

Exception: vertical fragmentation, where primary key attributes must be repeated to allow reconstruction. For horizontal fragmentation, data item is a tuple For vertical fragmentation, data item is an attribute

Correctness of Horizontal Fragmentation

Relation:Sells(pub,address,price,type)
type={Bitter, Lager} Fragments:

- SellsBitter= $\sigma_{type = \text{“bitter”}}$(Sells)
- SellsLager= $\sigma_{type = \text{“lager”}}$(Sells)

Correctness rules

Completeness: Each tuple in the relation appears either in SellsBitter, or in SellsLager

Reconstruction: The Sells relation can be reconstructed from the fragments

Sells = SellsBitter X SellsLager

Disjointness: The two fragments are disjoint, there can be no beer that is both “Lager” and “Bitter” Correctness of Vertical Fragmentation
Relation:
Bars(name,address,licence,employees,owner)

Fragments:
$r1 = \Pi_{name,address,licence}^{(Bars)}$
$r2 = \Pi_{name,address,employees,owner}^{(Bars)}$

Correctness rules

Completeness: Each attribute in the Bars relation appears either in r_1 or in r_2

Reconstruction: The Bars relation can be reconstructed from the fragments

Disjointness: The two fragments are disjoint, except for the primary key, name, which is necessary for reconstruction

Our model distributes the data pieces among more than one *service providers*, in such a way that no one of the *SP* s can retrieve any meaningful information from the pieces of data stored on its servers, without getting some more pieces of data from other service providers. Therefore, the conventional single service provider based techniques does not seem too much promising.

Distributing the data over multiple clouds or networks in such a way that if an adversary is able to intrude in one network, still he cannot retrieve any meaningful data, because its complementary pieces are stored in the other network. Our approach is similar to this approach, because both aim to remove the centralized distribution of cloud data. Although, in their approach, if the adversary causes a service outage even in one of the data networks, the user data cannot be retrieved at all. This is why in our model; we propose to use a redundant distribution scheme in which at least a threshold number of pieces of the data are required out of the entire distribution range, for successful retrieval.

Meaningful information from the data pieces allocated at their servers. Also, in addition, we provide the user with better assurance of availability of data, by maintaining redundancy in data distribution. In this case, if a service provider suffers service outage [1] [12] or goes bankrupt, the user still can access his data by retrieving it from other service providers.

From the business point of view, since *cloud data storage* is a subscription service, the higher the data redundancy, the higher will be the cost to be paid by the user.

III. MODELS

we will describe system model and the distribution model. The terms cloud storage and cloud data storage are interchangeable, also the terms user and customer are interchangeable.

System Overview

We consider the storage services for cloud data storage be-tween two entities, *cloud users* (*U*) and *cloud service providers*

(*SP*). The *cloud storage service* is generally priced on two factors, how much data is to be stored on the cloud servers and for how long the data is to be stored. In our model, we assume that all the data is to be stored for same period of time.

We consider *p* number of *cloud service providers* (*SP*), each available cloud service provider is associated with a *QoS* factor, along with its cost of providing storage service per unit of stored data (*C*). Every *SP* has a different level of *quality of service* (*QoS*) offered as well as a different cost associated with it. Hence, the *cloud user* can store his data on more than one *SP* s according to the required level of security and their affordable budgets.

IV. Distribution Model

Customers' stored data at cloud service providers is vulner-able to various threats. Previous studies in a cloud service provider can be a victim to Denial of service attacks or its variants[3],[4]. The idea for distributing the data among two storage clouds such that, an adversary cannot retrieve the contents of the data without having access to both the storage clouds. Relaying entirely upon a couple of service providers for the storage Such an attack scenario is entirely passive, because the cloud user cannot detect that his information has been collectively retrieved from the service providers without his consent.. Let us assume that two cloud service providers are available for customer who want to store his own data securely. seeks a distribution of customer's data pieces among the available SP s in such a way that, at least q number of SP s must take part in data retrieval, while minimizing the total cost of storing the data on SP s as well as maximizing the quality of service and availability of data provided by the SP s.

V. CONCLUSION

In this paper, we proposed a different data fragmentation schemes for multi cloud storage in cloud computing, which seeks to provide each customer with reliability, availability and better cloud data storage decisions.

ACKNOWLEDGEMENT

We are very much thankful to Prof. G. Vishnu Murthy, Head of the CSE Dept who had given valuable suggestions in carrying out this paper.

REFERENCES

[1] Amazon.com, "Amazon s3 availablity event: July 20, 2008", Online at http://status.aws.amazon.com/s3-20080720.html, 2008.

[2] P. S. Browne, "Data privacy and integrity: an overview", *In Proceeding of SIGFIDET '71 Proceedings of the ACM SIGFIDET (now SIGMOD)*, 1971.

[3] A. Cavoukian, "Privacy in clouds", *Identity in the Information Society*, Dec 2008.

[4] R. Gellman, "Privacy in the clouds: Risks to privacy and confidentiality from cloud computing", Prepared for *the World Privacy Forum*, online at http://www.worldprivacyforum.org/pdf/WPF Cloud Privacy Report.pdf,Feb 2009.

[5] N. Gruschka, M. Jensen, "Attack surfaces: A taxonomy for attacks on cloud services", *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, 5-10 July 2010.

[6] W. Itani, A. Kayssi, A. Chehab, "Privacy as a Service: Privacy-Aware Data Storage and Processing in Cloud Computing Architectures," *Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, Dec 2009

[7] M. Jensen, J. Schwenk, N. Gruschka, L.L. Iacono, "On Technical Security Issues in Cloud Computing", *IEEE International Conference on Cloud Computing, (CLOUD II 2009)*, Banglore, India, September 2009, 109-116

[8] J. Kincaid, "MediaMax/TheLinkup Closes Its Doors", Online at http://www.techcrunch.com/2008/7/10/mediamaxthelinkup-closes-itsdorrs/, July 2008.

[9] P. F. Oliveira, L. Lima, T. T. V. Vinhoza, J. Barros, M. M´edard,"Trusted storage over untrusted networks", *IEEE GLOBECOM 2010*, Miami, FL. USA.

[10] S. H. Shin, K. Kobara, "Towards secure cloud storage", *Demo forCloudCom2010*, Dec 2010.

[11] Stefano ceri Giuseppe pelagati "Distributed Databases-Principles and systems"Tata MC Graw Hill 2008

[12] J. Du, W. Wei, X. Gu, T. Yu, "RunTest: assuring integrity of dataflow processing in cloud computing infrastructures", In Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS '10), ACM, New York, NY, USA, 293-304