DATA REPLICATION UPDATION FOR DISTRIBUTED DATA SERVICE IN CLOUD COMPUTING

Mr.K.Saravanan[1], Dr.S.Kirubakaran[2], Dr N.Rajeshkumar[3]

[1,2,3] *KPR Institute of Engineering and Technology*
[1]*saravanan.k@kpriet.ac.in,* [2]*kirubakaran.s@kpriet.ac.in*
[3] *rajeshkumar.n@kpriet.ac.in*

ABSTRACT

Cloud Computing promises to deliver on an objective that building on compute and storage virtualization technologies, consumers are able to rent infrastructure "in the Cloud" as needed, deploy applications and store data, and access them via Web protocols on a pay-per-use basis. For application and data deployment in cloud, high availability and service level agreement must be satisfied. To leverage the performance and the number of concurrent access of application and data, dynamic application deployment and data replication are usually employed. With data replication in cloud computing, the consistence of multiple data replicas must be maintained. However, to maintain the consistence of data replicas and to keep the accessibility and availability of data service is usually a paradox. The consistence of multiple data replicas is discussed in this paper and asynchronous replication update approaches are used to separate the process of data replication and data access in cloud computing, which improves throughput of the data access and reduce response time and propagation time. Programming is performed using PHP platform and back up management using MySQL and advanced update data replication standard process is implemented for updating in cloud.

Keywords: replication, cloud computing

INTRODUCTION

Cloud computing refers to applications and services offered over the Internet. These services are offered from data centers all over the world, which collectively are referred to as the "cloud." This metaphor represents the intangible, yet universal nature of the Internet. It 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. This cloud model is composed of five essential characteristics**,** three service models, and four deployment models.

The idea of the "cloud" simplifies the many network connections and computer systems involved in online services. In fact, many network diagrams use the image of a cloud to represent the Internet. This symbolizes the Internet's broad reach, while simplifying its complexity. Any user with an Internet connection can access the cloud and the services it provides. Since these services are often connected, users can share information between multiple systems and with other users. The idea of cloud computing is based on a very fundamental principle of reusability of IT capabilities. The difference that cloud computing brings compared to traditional concepts of "grid computing", "distributed computing", "utility computing", or "autonomic computing" is to broaden horizons across organizational boundaries.

1.1 ESSENTIAL CHARACTERISTICS

Cloud computing typically entails

- **On-demand self-service:**
  A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.
- **Broad network access:**
  Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).
- **Resource pooling:**
  The providers computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.
- **Rapid elasticity:**
  Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.

LITERATURE SURVEY

Replication is a well recognized sticking point in the cloud, Laura Cristiana Voicu in (2010) discuss in detail about the refresh of individual replicas in a completely distributed way. Finally, we present our approach to cloud data management, based on a recent protocol for data grids.

Dormann, Will & Rafail in (2006) summarize that the cloud provides many options for the everyday computer user as well as large and small businesses. It opens up the world of computing to a broader range of uses and increases the ease of use by giving access through any internet connection. However, with this increased ease also come drawbacks. You have less control over who has access to your information and little to no knowledge of where it is stored. You also must be aware of the security risks of having data stored on the cloud. The cloud is a big target for malicious individuals and may have disadvantages because it can be accessed through an unsecured internet connection.

F. Akal specified that the sites are divided between update and read-only sites in (2005), Read only sites only allow read access to data. Updates occur on the update sites and are synchronized among the update sites and finally propagated to the read-only sites. K. B¨ohm assumes an eager replication among update sites and in addition we apply lazy replication mechanisms between update and read-only sites that take into account have different levels of freshness.

J.Cho and H.Garcia-Molina in (2000), to improve the freshness on lazy replication where transaction execute entirely at the initiation site, which may not be the case in practical settings. A recent protocol for finer grained data replication that supports freshness and lazy update propagation for many read-only nodes has been presented.

Yochai Ben-Chaim and Avigdor Gal in (2006) discuss the function by periodically sending data in bulk format in the replication techniques named snapshot replication. Usually it is used when the subscribing servers can function in read-only environment, and also when the subscribing server can function for some time without updated data. Functioning without updated data for a period of time is referred to as latency.

Hans P. Reiser in (2011) proposed Byzantine Fault Tolerance for the Cloud that assumes a service is replicated on multiple virtual machines deployed in the cloud, potentially across multiple cloud providers. Each replica may fail in arbitrary ways. The number of replicas that are faulty at a given time is bounded. Furthermore, in a first variant of the architecture we assume that the cloud infrastructure itself is trusted, i.e., that it fails only by crashing.

Giuliana Santos Veronese in (2010) suggested that BFT atomic multicast is used for replication, on the basis of BFT consensus. Our current prototype is based on the EBAWA algorithm, which uses a trusted component (USIG – unique sequential identifier generator) and requires only 2f +1 replica. We extend this algorithm for recovery and reconfiguration.

Markus Klems in (2009) proposed the replication method named Merge replication which is the process of distributing data from Publisher to Subscribers, allowing the Publisher and Subscribers to make updates while connected or disconnected, and then merging the updates between sites when they are connected. Merge replication allows various sites to work autonomously and at a later time merge updates into a single, uniform result.

D. Wentzlaff in (2010) suggested transactional replication which works by sending changes to the subscriber as they happen. As an example, SQL Server processes all actions within the data using Transact-SQL statements. Each completed statement is called a transaction. In transactional replication, each committed transaction is replicated to the subscriber as it occurs. You can control the replication process so that it will accumulate transactions and send them at timed intervals, or transmit all changes as they occur. You use this type of replication in environments having a lower degree of latency and higher bandwidth connections.

Tobias Distler, R¨udiger Kapitza in (2011), suggested that Synchronous replication is an important consideration in the behaviour of the storage subsystem when the connection between the primary and secondary subsystem is temporarily disrupted. Synchronous replication does not provide protection against data corruption and loss of data due to human errors. Snapshot technology must be used with synchronous replication to provide full protection against both losses of access to data (protected by replication) and loss of data due to data corruption.

Rafail, Jason in (2008), each provider serves a specific function, giving users more or less control over their cloud depending on the type. When you choose a provider, compare your needs to the cloud services available. Your cloud needs will vary depending on how you intend to use the space and resources associated with the cloud. If it will be for personal home use, you will need a different cloud type and provider than if you will be using the cloud for business. Keep in mind that your cloud provider will be pay-as-you-go, meaning that if your technological needs change at any point you can purchase more storage space (or less for that matter) from your cloud provider.

E. Pacitti and E. Simon in (2000) proposed a research challenge in freshness of data which describes absolute freshness and delay freshness. Absolute freshness of a data object is defined by means of the timestamp of the last committed update transaction that has updated. These timestamps can be used by the clients in order to define their freshness requirements.

C. Clark, K. Fraser in (2005), discussed Virtual machine cloning method in which the concept of UNIX process forking is applied to an entire VM stack. Multiple replicas of a VM are created. These are logically complete and identical replicas, including the VM‟s disk, with the exception of an ID that allows each clone to identify itself and the programmer to reason about the result of cloning. The cloned VMs are transient in that once the task they were created to accomplish is finished, they are discarded and their state is completely eliminated.

Lijun Mei in (2008) suggested that data replication in cloud computing has the consistence of multiple data replicas which must be maintained. However, to maintain the consistence of data replicas and to keep the accessibility and availability of data service is usually a paradox.

Paulo Sousa, Nuno Ferreira NevesOne in (2007) brings to knowledge that, advantage of cloud computing is the dynamicity of resource provisioning. It makes use of this advantage by enabling dynamic runtime modifications of replication groups. For example, in a replication group, increasing the number of replicas typically reduces the cost of read-only operations, while increasing the cost of modifying operations. Dynamically adapting the number of replicas may be used to increase service quality, but requires careful modifications to the BFT protocols in use.

H. Zhang, G. Jiang in (2009) makes to know about the On-premise private clouds which also known as internal clouds are hosted within one‟s own data centre. This model provides a more standardized process and protection, but is limited in aspects of size and scalability. IT departments would also need to incur the capital and operational costs for the physical resources. This is best suited for applications which require complete control and configurability of the infrastructure and security.

Song Yang et al. (2018) reduce the latency and improve data allocation with minimum total storage cost.

Atakan Aral et al. (2018) proposed a distributed dissemination approach that relies on dynamic creation/ replacement or removal of replicas from the nodes by monitoring the data request coming from various locations.

Mouhamad Diye et al. (2017) proposed an uniform crossover to produce offspring from heterogenous data and mutation process is used to avoid local optima of the offspring. This offspring produced must not compromised on SLA.

DATA REPLICATION PROCESS

Data replication is the process of creating and maintaining multiple instances of the same data and the process of sharing data or data design changes between data in different locations without having to copy the entire data. In most implementations of data replication, one data server maintains the master copy of the data and the additional data servers maintain slave copies of the data. The two or more copies of a single data remain synchronized. The original data is called a Design Master and each copy of the data is called a replica. Together, the Design Master and the replicas make up a replica set. There is only one Design Master in a replica set. The term "Replication" this represents the process of sharing information to ensure consistency between redundant resources, such as software or hardware components, to improve reliability, fault-tolerance or accessibility. It could be data replication if the same data is stored on multiple storage devices or computation replication if the same computing task is executed many times.

If at any time one master replica is designated to process all the requests, then we are talking about the primary-backup scheme (master-slave scheme) predominant in high-availability clusters. On the other side, if any replica processes a request and then distributes a new state, then this is a multi-primary scheme (called multi-master in the data field). Even thought the process of Data Replication it is used to create instances of the same or parts of the same data, we must not confuse it with the process of backup since replicas are frequently updated and quickly lose any historical state. Backup on the other hand saves a copy of data unchanged for a long period of time.

3.1 PROBLEMS OF CURRENT DATA REPLICATION PROCESSES

The problem of maintaining the consistence of data replication is discussed here. Some approaches are used to separate the process of data replica updating and data access in cloud.

3.1.1 Snapshot replication

Snapshot replication is helpful when:

- Data is mostly static and does not change often.
- It is acceptable to have copies of data that are out of date for a period of time.
- Replicating small volumes of data in which an entire refresh of the data is reasonable.

3.1.2 Merging replication

Merge Replication is helpful when:

- Multiple Subscribers need to update data at various times and propagate those changes to the Publisher and to other Subscribers.
- Subscribers need to receive data, make changes offline, and later synchronize changes with the Publisher and other Subscribers.
- You do not expect many conflicts when data is updated at multiple sites (because the data is filtered into partitions and then published to different Subscribers or because of the uses of your application). However, if conflicts do occur, violations of ACID properties are acceptable.

3.1.3 Transactional replication

Transactional replication is helpful when:

- You want incremental changes to be propagated to Subscribers as they occur.
- You need transactions to adhere to ACID properties.
- Subscribers are reliably and/or frequently connected to the Publisher.

3.1.4 Synchronous Replication

When synchronous replication is used, an update made to a data volume at the primary site is synchronously replicated to a data volume at a secondary site. This guarantees that the secondary site has an identical copy of the data at all times. The availability and reliability characteristics of Multi-AZ deployments (AZ is short for "Availability Zone") make them well suited for critical production environments. I'd like to tell you about this new feature and how it works; here's a diagram to get you started:

3.2 DEFECTS OF CURRENT DATA REPLICATION PROCESS

There are number of problems with this existing process:

- As data start growing the time taken to complete the replication will go on increasing.
- It takes lot of time to replicate and synchronize both ends.
- There is low consistency as lot of parties has to be synchronized.
- There can be conflicts while merge replication if the same rows are affected in more than once subscriber and publisher.
- Disaster recovery is provided only for data and not for applications or other data stores at the primary site.
    - In Transaction replication, Network congestion can increase latency, which can increase the recovery time.
    - Latency is the time elapsed between committing a transaction at the primary database and committing that transaction at the standby database.
- Some data loss can occur in certain situations.
- The primary system must wait for the secondary system before proceeding, leading to an increased response time. Because of the increased response time and communication delays, synchronous replication is often impractical unless the secondary system is physically located close to the primary system.

PROPOSED METHOD: ASYNCHRONOUS REPLICATION

Enterprise Replication uses *asynchronous* data replication to update the databases that reside at a replicated site after the primary database has committed a change. One of the most important things for a brand, company or institute is its data. The data is very important for maintaining the records, transactions or information. Any company or institute cannot afford to lose them under any conditions.

In an asynchronous mode of operation, I/O operations are written to the primary storage system and then sent to one or more remote storage systems at some later point in time. Due to the time lag, data on the remote systems is not always an exact mirror of the data at the primary site. This mode is ideal for disk-to-disk backup or taking a snapshot of data for offline processes, such as testing or business planning. The time lag enables data to be replicated over lower-bandwidth networks, but it does not provide the same level of protection as synchronous replication. Asynchronous replication is less sensitive to distance and link transmission speeds. However, because replication might be delayed, data can be lost if a communication failure occurs followed by a primary site outage.

Considerations when sizing link requirements and configuring the storage subsystem include the following:

- Incoming ROC (for more information about ROC, see the “Rate of Change” section below).
- Speed of the replication process in the storage subsystem (how quickly data can be pushed out to the replication link).
- Line speed and line latency.
- Size of the replication buffer (queue space) in the storage subsystem; the buffer has to be large enough to cope with peak ROCs.

4.1 CHOOSING ASYNCHRONOUS REPLICATION

Both synchronous and asynchronous replication processes have its own merits and demerits and it is completely the specific needs of the company which can determine which technique to use. A complete assessment of the business has to be done before making any choices. The decision is quite complex and generally depends on different aspects of the company like assessments of potential risks and business impact analysis. However, choosing asynchronous replication is highly advisable when all the aforementioned factors take place in an enterprise.

4.1.1 The efficiency of asynchronous replication

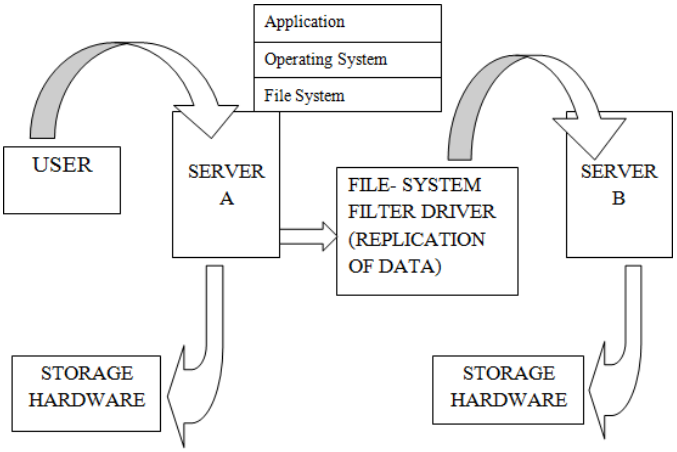

Fig 1: Asynchronous replication

In the above asynchronous replication diagram (fig 1) first the user executes a transaction or enters new information into the database. Then the data flows from the servers application layer to operating system and file system in close to real time. Data then flows to the storage hardware. From there file system filters driver captures a copy of data and sends the copy to server B. it has the back-up copy of the data, which is then stored.

Asynchronous replication allows the following replication models:

- Primary-target (Primary-Target Replication System)

  All database changes originate at the primary database and are replicated to the target databases. Changes at the target databases are not replicated to the primary.

- Update-anywhere (Update-Anywhere Replication System)

  All databases have read and write capabilities. Updates are applied at all databases.

The update-anywhere model provides the greater challenge in asynchronous replication. For example, if a replication system contains three replication sites that all have read and write capabilities, conflicts occur when the sites try to update the same data at the same time. Conflicts must be detected and resolved so that the data elements eventually have the same value at every site.

4.2 PRIMARY-TARGET REPLICATION SYSTEM

In the primary-target replication system, the flow of information is in one direction. In primary-target replication; all database changes originate at the primary database and are replicated to the target databases. Changes at the target databases are not replicated to the primary. A primary-target replication system can provide one-to-many or many-to-one replication:

- One-to-many replication

  In one-to-many (*distribution*) replication, all changes to a primary database server are replicated to many target database servers. Use this replication model when information gathered at a central site must be disseminated to many scattered sites.

- Many-to-one replication

  In many-to-one (*consolidation*) replication, many primary servers send information to a single target server. Use this replication model when many sites are gathering information (for example, local field studies for an environmental study) that needs to be centralized for final processing.

4.3 UPDATE-ANYWHERE REPLICATION SYSTEM

In update-anywhere replication, changes made on any participating database server are replicated to all other participating database servers. This capability allows users to function autonomously even when other systems or networks in the replication system are not available.

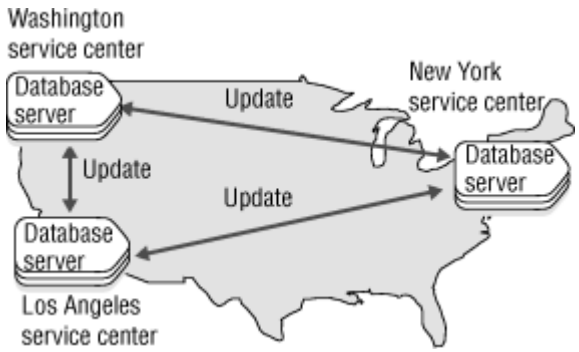

Fig 4.2 Update-Anywhere Replication System

Because each service centre can update a copy of the data, conflicts can occur when the data is replicated to the other sites. To resolve update conflicts, Enterprise Replication uses *conflict resolution*. Review the following information before you select your update-anywhere replication system:

- Administration

  Update-anywhere replication systems allow peer-to-peer updates, and therefore require *conflict-resolution*. Update-anywhere replication systems require more administration than primary-target replication systems.

- Information consistency

  Some risk is associated with delivering consistent information in an update- anywhere replication system. You determine the amount of risk based on the type of conflict-resolution rules and routines you choose for resolving conflicts. You can configure an update-anywhere replication system where no data is ever lost; however, you might find that other factors (for example, performance) outweigh your need for a fail-safe mechanism to deliver consistent information.

- Capacity Planning

  All replication systems require you to plan for capacity changes. For more information, see Preparing Data for Replication. In particular, if you choose the time stamp or time stamp plus SPL routine conflict-resolution rule, see Delete Table Disk Space and Shadow Column Disk Space.

- High Availability

  If any of your database servers are critical, consider using HDR to provide backup servers.

PERFORMANCE EVALUATION

For evaluating the performance of the proposed asynchronous system, PHP language is used with MYSQL as background. PHP is a general-purpose server-side scripting language originally designed for Web development to produce dynamic Web pages. It is one of the first developed server- side scripting languages to be embedded into an HTML source document rather than calling an external file to process data. The code is interpreted by a Web server with a PHP processor module which generates the resulting Web page. It also has evolved to include a command-line interface capability and can be used in standalone graphical applications. PHP can be deployed on most Web servers and also as a standalone shell on almost every operating system and platform free of charge. A competitor to Microsoft's Active Server Pages (ASP) server-side script engine and similar languages, PHP is installed on more than 20 million Web sites and 1 million Web servers.

5.1 CLOUD DEPLOYMENT

MySQL can also be run on cloud computing platforms such as Amazon EC2.
Listed below are some common deployment models for MySQL on the cloud:

- **Virtual Machine Image** - cloud users can upload a machine image of their own with MySQL installed, or use a ready-made machine image with an optimized installation of MySQL on it, such as the one provided by Amazon EC2.
- **MySQL as a Service** - some cloud platforms offer MySQL "as a service". In this configuration, application owners do not have to install and maintain the MySQL database on their own. Instead, the database service provider takes responsibility for installing and maintaining the database, and application owners pay according to their usage. Two notable cloud-based MySQL services are the Amazon Relational Database Service, and the Xeround Cloud Database, which runs on EC2, Rack space, HP Cloud, and Heroku.
- **Managed MySQL cloud hosting** - the database is not offered as a service, but the cloud provider hosts the database and manages it on the application owner's behalf. As of 2011, of the major cloud providers, only Terre mark and Rack space offer managed hosting for MySQL databases.

5.2 RESULTS AND DISCUSSION

The data replication is hosted on a "site", which may be a physical standalone server or a virtualization server. The data replica is deployed at this site and the site is registered in a whole data replication network. For a site hosting a data replica, the updating related components include: replica management, update receiver, update propagator and update executor. Replica management component is for the synchronization between sites, completing network communication and transfer synchronization. The functions of update receiver are receiving update messages and data updates, transmitting the data to update executor which updates the background data store. The processing performed in update propagator are packing the data updates and transfer to next step.

In the registration form the user register their detail in the database. From this the user name and password to enter into cloud be received.

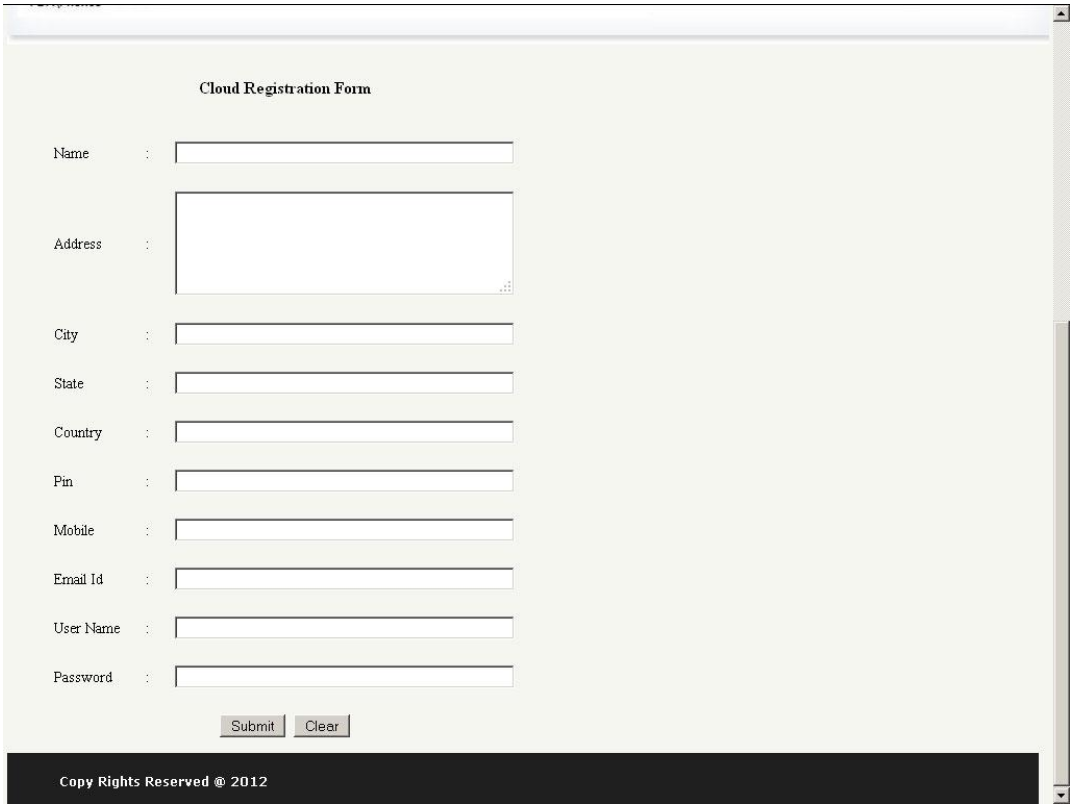

Fig 2: Screen shot of Cloud Registration Form

Database is created by using the PHP software and data information be stored in the cloud with certain information.

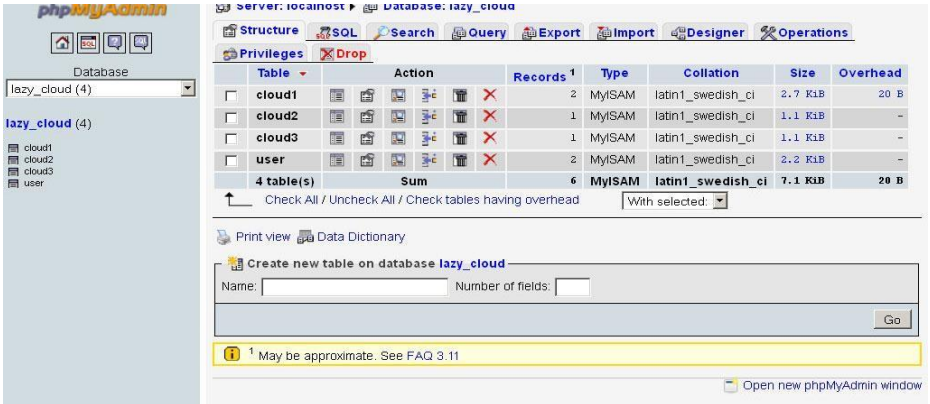

Fig 3: Screen shot of Creation of Data base

Uploaded files are stored with the user detail and we can remove the file by the drop command. Then the replication process be done by the asynchronous replication.

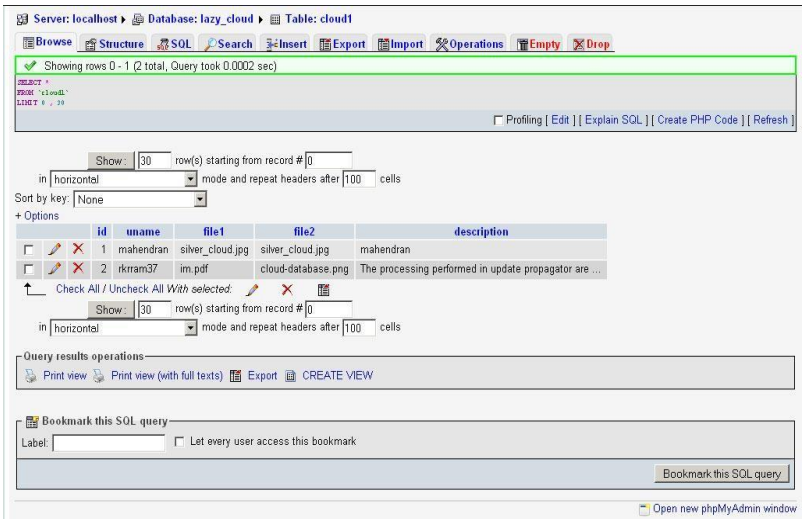

Fig 4: Screen shot of File be uploaded

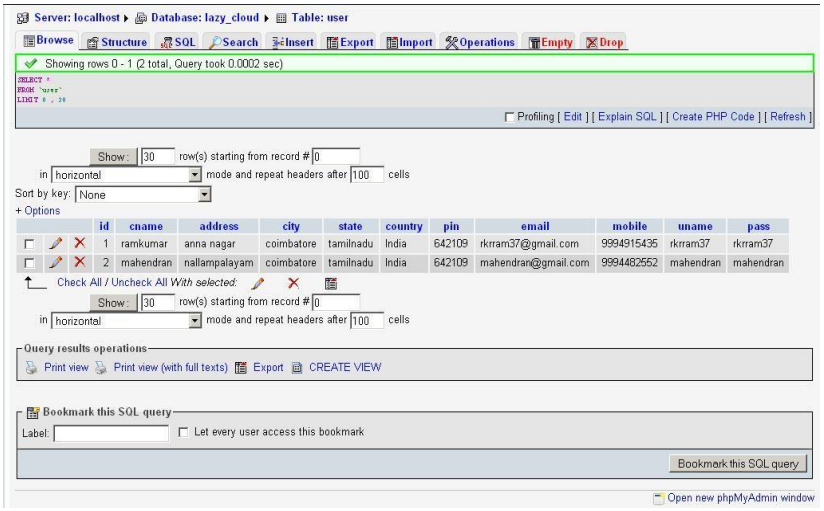

Fig 5: Screen shot of User Table with Updated Files

The replicated files are updated and stored in the cloud database. From this we can view the response time and throughput values for the normal and asynchronous replication.

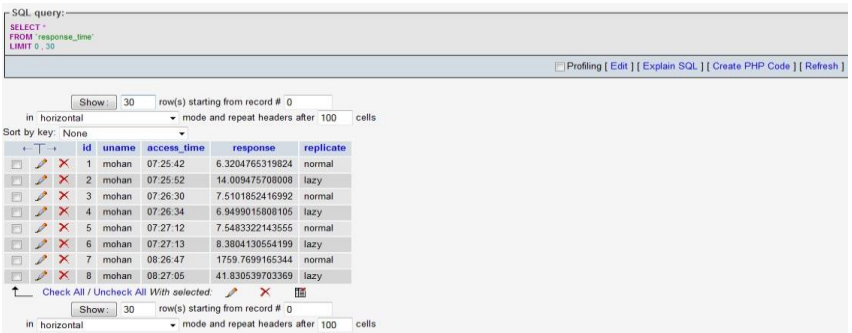

Fig 6: Screen shot of Response time with Updated Files

In the fig 7 response time and no of clients be compared. Response time be increased if the client be increased. In our proposed method the response time be less when compared to other process. Generally Response time want to be less to access the data in cloud in quick succession.

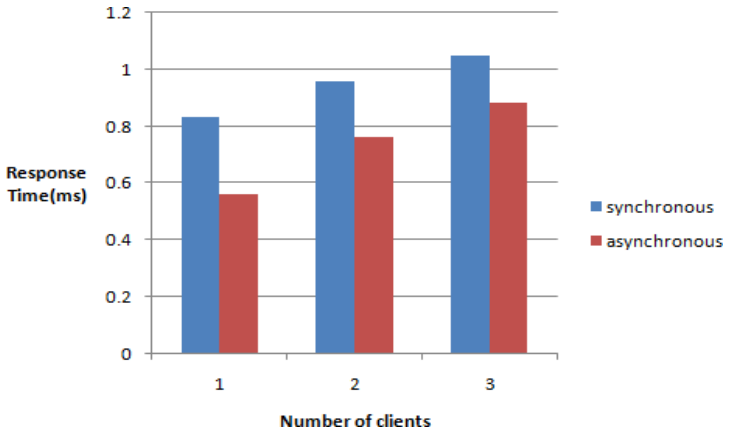

Fig 7: Response time vs. No of clients

In the fig 8 average throughput and no of request be compared. Throughput be increased if the client be increased. In our proposed method throughput be high when compared to other process. Generally throughput want to be high in cloud to efficint data access.

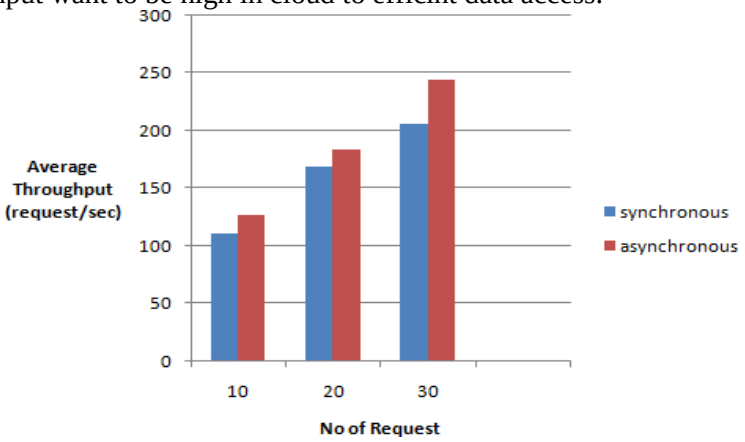

Fig 8: Average throughput vs. No of Request

CONCLUSION AND FUTURE WORK

CONCLUSION

In the current economic environment, where the expectations of efficiencies and cost savings are growing from IT organizations, enterprise clouds provide a good opportunity to get started with cloud computing and reap the associated benefits of agility, cost saving, and on-demand services while meeting the stringent enterprise security, performance and reliability requirements. Data replication and the corresponding updating problems is a research topic in distributed computing. In this project, a method for data updates of data replicas in cloud computing is discussed, which is asynchronous update propagation. The method can improve the throughput of the data server and reduce average response time and propagation time. The method is especially useful for cases where the database transactions are short and high throughput is the main object for the system. Experimental results show that the methods are effective and scalable. Also, the asynchronous update approaches can reinforce the delivery semantic and improve the freshness of data.

FUTURE WORK

In future work, the methods will be tested in more practical cloud computing environments with much larger volume. With the methods in this project, the consistency of database needs to be investigated. When and where to deploy data replication dynamically.

REFERENCES

[1] F. Akal, H. Schuldt, and H.-J. Schek, "Grid-Enabled Data Replication for Digital Libraries with Freshness and Correctness Guarantees", In 3rd VLDB Workshop on Data Management in Grids, Vienna, Austria, 2007.

[2] T.Anderson,Y. Breitbart, H. F. Korth, and A. Wool, "Replication, Consistency,and Practicality: are these mutually exclusive", In Proceedings ACM SIGMOD International Conference on Management of Data, pages 484–495. ACM, 1998.

[3]D. Battre´, S. Ewan, F. Hueske, O. Kao, V. Markl, and D. Warneke, "Nephele/PACTs: A Programming Model and Execution Framework for Web-Scale Analytical Processing", Proc. ACM Symp. Cloud Computing (SoCC ‟10), pp. 119- 130, 2010.

[4] J. Cho and H. Garcia-Molina, "Synchronizing a database to Improve Freshness", In ACM SIGMOD International Conference on Management of Data, pages 117– 128, 2000.

[5] Ian Foster, Yong Zhao, Ioan Raicu, Shiyong Lu, "Cloud Computing and Grid Computing 360-Degree Compared", Grid Computing Environments Workshop, 2008. pp.1-10

[6]Laura Cristiana Voicu, Heiko Schuldt, Yuri Breitbart, Hans-J¨org Schek,"Flexible Data Access in a Cloud based on Freshness Requirements", IEEE 3rd International Conference on Cloud Computing, 2010

[7] Leslie Lamport, Dahlia Malkhi, Lidong Zhou, and Microsoft Research. Vertical Paxos and Primary- Backup Replication, 2009.

[8] Lijun Mei , W. K. Chan , T. H. Tse, "A Tale of Clouds: Paradigm Comparisons and Some Thoughts on Research Issues", Proceedings of the 2008 IEEE Asia-Pacific /services computing conference (APSCC 2008),IEEE Computer Society, Press,pp.464-469.

[9] Mark A. Linsenbardt, Shane Stigler „McGraw-Hill/Osborne Media book SQL Server 2000 Administration‟ - Chapter 10, 'Replication'.

[10] Markus Klems, Jens Nimis and Stefan Tai, "Do Clouds Compute? A Framework for Estimating the Value of Cloud Computing", Lecture Notes in Business Information Processing, 2009, Volume 22, Part 4, pp.110-123

[11]. E. Pacitti, E. Simon, and R. de Melo, "Improving data freshness in lazy master schemes", Proceedings of Int. Conf. On Distributed Computing Systems (ICDCS98), Amsterdam, May 1998.

[12] U. R¨ohm, K. B¨ohm, H.-J. Schek, and H. Schuldt, "FAS: a Freshnesssensitive Coordination Middleware for a Cluster of OLAP Components". In Proceedings of 28th International Conference on Very Large Data Bases (VLDB 2002), pages 754– 765, 2002.

[13] D. Saha and A. Mukherjee. Pervasive computing: a paradigm for the 21st century. IEEE Computer, 36(3): 25–31, 2003.

[14] Tobias Distler, R¨udiger Kapitza, Ivan Popov, Hans P. Reiser, and Wolfgang Schr¨oder-Preikschat,"SPARE: Replicas on Hold. In Internet Society (ISOC)", editor, Proceedings of the 18th Network and Distributed System Security Symposium (NDSS ‟11), 2011.

[15] C. T¨urker, F. Akal, H.-J. Schek, Y. Breitbart, T. Grabs, and L. Veen, "Fine- Grained Replication and Scheduling with Freshness and Correctness Guarantees",In Proceedings of the 32st International Conference on Very Large Databases (VLDB 2005), pages 565–576, 2005.

[16] D. Wentzlaff, C. Gruenwald III, N. Beckmann, K. Modzelewski, A. Belay, L. Youseff, J. Miller, and A. Agarwal, "An Operating System for Multicore and Clouds: Mechanisms and Implementation," Proc. ACM Symp. Cloud Computing (SoCC ‟10), pp. 3-14, 2010.

[17] Yang Zhao-Hong, Gong Yun-Zhan, Liu Hai-Yan, Bi Xue-Jun "Minimizing the copy graph to improve scalability", IEEE 2003.

[18] Mouhamad Dieye & Mohamed Faten Zhani 2017, 'On Achieving High Data Availability in Heterogeneous Cloud Storage Systems', IEEE, pp. 2-10.

[19] Atakan Aral & Tolga Ovatman 2018 'A Decentralized Replica Placement Algorithm for Edge Computing', IEEE, pp. 1-14.