BUILDING A OPENSTACK SWIFT SDK PLATFORM FOR OBJECT STORAGE

Pooja Patil[1], Vidya R. Kulkarni[2], Rashmi M. Jogdand[3]
[1]Student, CSE Dept., VTU GIT Belgaum, India
[2]professor, CSE Dept., VTU GIT Belgaum, India
[3]professor, ISE Dept., VTU GIT Belgaum, India

Abstract
Cloud computing has become a trend now because of its scaling properties and near unlimited capabilities. Cloud can be used as storage, software services and infrastructure mode. In storage mod, cloud is used for storing and retrieving files. Each cloud vendor provides a unique SDK to store and retrieve which the application developers have to use to access cloud. Open Stack is open source cloud. It provides SDK to store and access cloud storage in terms of objects. In this paper, we adapt the open stack SDK to serve as front end for commercial public clouds.
Keywords: Object cloud storage; OpenStack Swift

I INTRODUCTION

The requirement for computing resources is growing in larger steps every day. Especially after smartphone revolution, the demand for computing and storage has grown large. With smart phone and internet availability contents are getting generated exponentially and demand for storing content has become important. Cloud storage is the solution to meet this unlimited demand for storage. In cloud, ground of servers are distributed in different parts and these servers are connected by internet. Cloud provides virtual machine for storage and computing. Due to pay as go model, the users of cloud don't need to pay for installation of large volume of storage. They only pay for volume of data consumed. This attracts many people to use cloud storage. Cloud provides storage SDK to storage and retrieves from cloud. Each vendor provides their own version of cloud SDK[19]. Due to these applications depending on cloud storage cannot be ported easily from one cloud to another. This becomes a hurdle for cloud users because they get tied to one operator and not able to move from one cloud to another cloud.

Open stack is open source private cloud. It provides a object based cloud storage. But when migrated to public cloud, the open stack applications are not able to easily move to public cloud. In this project, we consider this problem and give a porting for open stack sdk for public cloud storage.

II NEW SYSTEM DEFINITION AND ITS NEED

To develop Object Cloud Storage System, which enables users to store and retrieve their contents in the cloud through REST API's. Current cloud storage systems do not have facility for search and retrieval of related contents. But for users, it is very important functionality to search related contents.

Proposed System

Necessity of the proposed system:

Searching related contents through object based search though API will save lot of bandwidth for users, since users don't need to visit all the files to get his related contents.

Challenges of the proposed system:

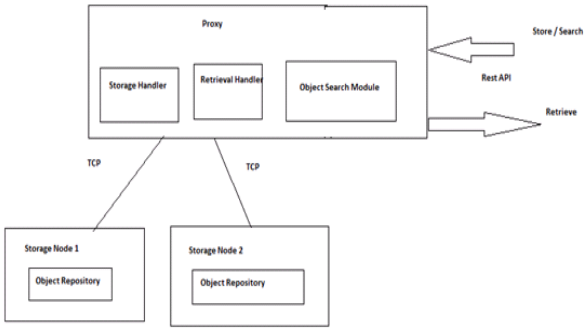
The contents can be any types like text document, picture, audio, video, etc. and the format may also be different. So designing a search system for any content type and format is the challenging task.

III LITERATURE SURVEY

Below are some of the existing tools[12] for storing data in the cloud but there exist some drawback with each one of them:

Tools	Drawbacks
OpenNebula[10]	Very Centralized
Storpool[18]	Risk due to distributed storage
Nimbus[1, 8]	Not so open
Eucalyptus[11]	No uniform way of storage
VMware Hypervisor[5]	Not portable

IV SYSTEM DESIGN



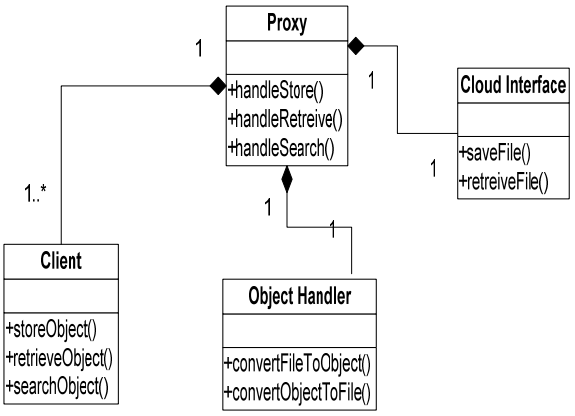
The system is split to two sub system components:

1. Proxy
2. Storage Node

Proxy: Proxy module is the front end interface for the object cloud storage system. It provides REST API interface to store, retrieve and search objects. Proxy internally maintains the storage index mapping to keep track of location of objects in storage nodes. It also distributes objects for load sharing. It implements the search system to do content based search on contents and provides related contents to users.

Storage Nodes: These nodes connect via TCP connection to the Proxy and store, retrieve the objects on demand by the Proxy. Proxy is the main intelligence module and storage node obeys the command from the Proxy.

V CLASS AND DFD DIAGRAM OF THE SYSTEM



The functionality in Client class is below

StoreObject	User interface function to be called by users for storing the object
RetrieveObject	User interface function to be called by the users to retrieve the object
SearchObject	User interface functions to be called by the users to search object

The functionality in Proxy class is below

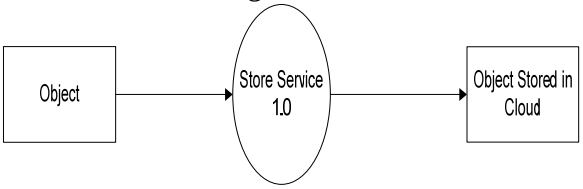
HandleStorage	This function will handle Storage of object to azure cloud
HandleRetrieve	This function will handle retrieve object from cloud
HandleSearch	This function will handle Search for the object based on keywords.

Data Flow Diagram of the system

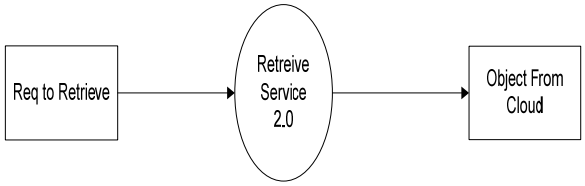
The data flow diagram documents the input, output and process in the system.

Level 0 Data flow diagram

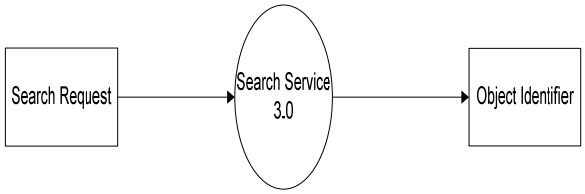
The top level process is given in the level 0 data flow diagram.



Input	Object to store
Output	The object is stored to cloud
Process	Store Service



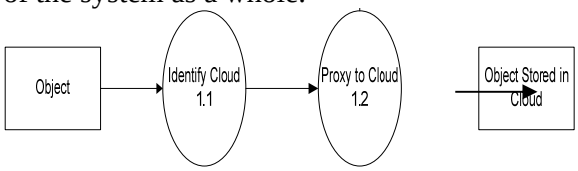
Input	Request to retrieve object with object id
Output	The object fetched from cloud
Process	Retrieve Service



Input	Search Request with meta data
Output	The matching object identifier
Process	Search Service.

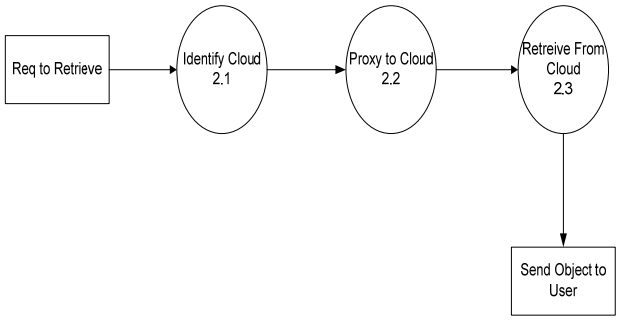
Level 1 Data flow diagram

The Level 1 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole.



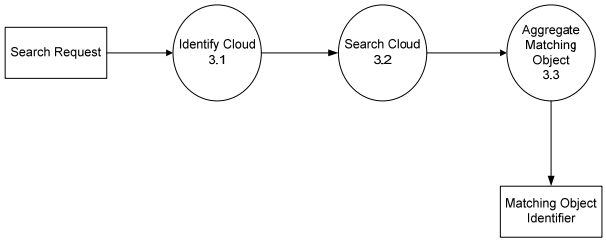
The store service is split to following sub process

Identify cloud	Cloud identified is Microsoft Azure
Proxy to cloud	Sent the request to cloud to store object
Get response from cloud	Get response form cloud after storage and return to user.



The retrieve service is split to following sub process

Identify cloud	Cloud identified is Microsoft Azure
Proxy to cloud	Sent the request to cloud to download object
Retrieve from cloud	Fetch the object and send to user.

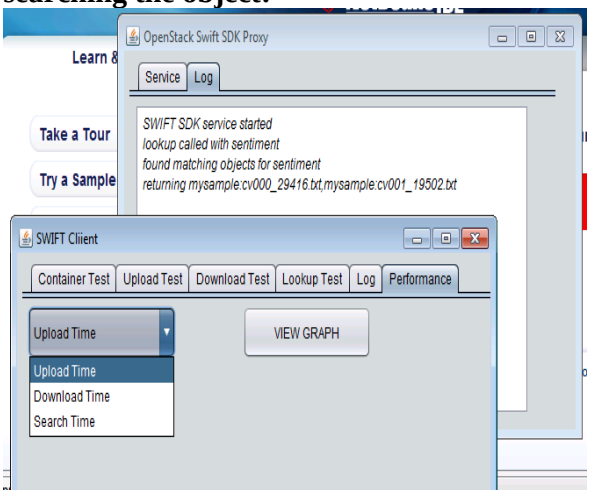


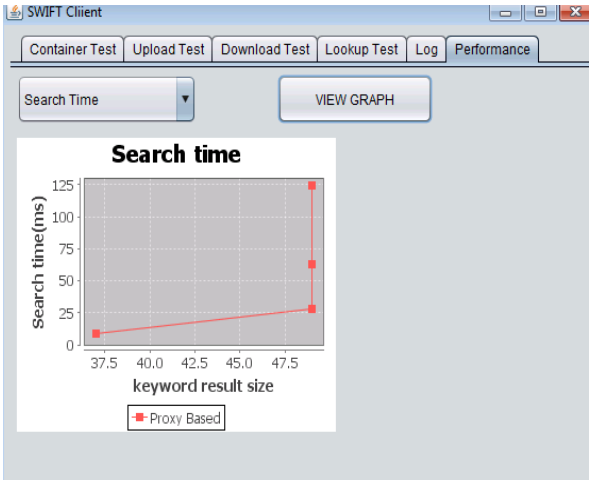
The search service is split to following sub process

Identify cloud	Cloud identified is Microsoft Azure
Search cloud	Search in cloud for matching meta data
Aggregate matching object	Fetch the matching objects and return to users.

Experimental Result

Below is the experimental result for searching the object:





CONCLUSION

The paper says that, Object Cloud Storage system provides unlimited storage to end users and flexibility to store and retrieve data in forms of objects. But most Object cloud storage system lacks search functionality. Without this functionality, it is difficult for the users to search on contents and retrieve relevant files. It motivates us to design a solution to this problem.

FUTURE WORK

In future we plan to implement advanced look up methods like search by giving an image for image object search and search by audio sample to search in Audio objects.

REFERENCES

[1] Armbrust M, Fox A, Griffith R, Joseph AD, Katz RH, Konwinski A, Lee G, Patterson DA, Rabkin A, and Zaharia M. Above the clouds: A berkeley view of cloud computing. Technical report; 2009.

[2] Amazon simple storage service (Amazon S3), http://aws.amazon.com/s3/, 2013.

[3] World's data more than doubling every two years — Driving big data opportunity, new IT roles, http://www.emc.com/about/news/press/2011/2011062801.htm, 2013.

[4] IDC says world's storage is breaking Moore's law, more than doubling every two years,http://enterprise.media.seagate.com/2011/06/insideitstorage/idc-says-worlds-storage-is breaking-mooreslawmore-than-doubling-every-two-years/, 2012.

[5] www.**vmware**.com

[6] Amazon Simple Storage Service, aws documentation, API reference (API version 2006-03-01) Amazon Web Services LLC; 2014.

[7] Krawczyk H, Bellare M, Canetti R. HMAC: Keyedhashing for message authentication; 1997.

[8] Eucalyptus URL: http://www.eucalyptus.com/ Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G.,Soman, S., Youseff, L. et al. (2009) The Eucalyptus Open-Source Cloud Computing System. In: 9th IEEE/ACM International Symposium on Cl uster Computing and the Grid .Shanghai, China 2009.

[9] B. Sotomayor, R. S. Montero, I. M. Llorente, I. Foster, Virtual infrastructure management in private and ybrid clouds IEEE Internet Comput. 13 (2009) 1422.doi: 10.1109/MIC.2009.119.

[10] OpenNebula URL: http://opennebula.org/

[11] Nimbus. URL: http://www.nimbusproject.org/

[12] Peter Sempolinski and Douglas Thain, A Comparison and Critique of Eucalyptus, OpenNebula and Nimbus, University of Notre Dame.

[13] Z.Lei, B. Zhang, W. Zhang, Q. Li, X. Zhang, and J. Peng Comparison of Several Cloud Computing Platforms. Second International Symposium on Information Science and Engineering, pages 23-27, 2009.

[14] OpenStack URL: http://www.openstack.org/

[15] Prakashan Korambath, Narcis Madern Investigating Private Cloud Storage Deployment using Cumulus, Walrus, and OpenStack/Swift

[16] OpenStack Swift Authentication System URL: http://docs.openstack.org/developer/swift/overview_auth. html

[17] Benchmarking – SwiftStack Documentation URL: https://www.swiftstack.com/docs/faqs/benchmarking.ht ml

[18] https://**storpool**.com/

[19] Sridevi Bonthu, Y S S R Murthy, M. Srilakshmi "Building an object Cloud Storage Service System using OpenStack Swift".