Front. Cur. Trends. Engg. Tech. Vol. 1(1), *pp.* 16-22 (2016)
OPEN ACCESS ISSN: 2456 - 1185
Published by ACET. Available online: www.fctet.in

Optimized task scheduling with secure storage and deuduplicating data for IAAS cloud

K.Vivekrabinson[1*] and R. Parvadha devi[1]

Abstract: Infrastructure as a Service is a form of cloud computing that provides virtualized computing resources over the Internet. This paper provides a short overview of how to create instance base on user request and how to set priority and schedule the instance. In this project use generalized priority algorithm and PSO (particle swarm optimization based scheduling) to manage the instance. The use of generalized priority algorithm is to prioritize the user requests based on size, Memory and bandwidth and also use to rank the instance based on cloudlet. The working procedure of PSO used to outsource its task to external clouds when its own resources are not sufficient to meet the demand and at the same time guarantee the QoS (Quality of Service). It is similar to outsourcing during hike period. Users also can store their data to cloud. By the help of advanced encryption standard and diff algorithm increase reliability and efficiency by encrypting user files and remove duplicate data from cloud.

Keywords: Cloud storage, cloudlet, priority, outsourcing, encryption and duplicate check

1. INTRODUCTION

1.1. Cloud computing

Cloud computing is the conveyance of computing services over the Web. Cloud services allow individuals and organizations to utilize software and hardware that are managed by outsiders at remote locations. Examples of cloud services incorporate online document storage, social systems administration locales, webmail, and online business applications. The cloud computing model allows access to information and PC assets from anywhere that a system association is available. Cloud computing gives a shared pool of assets, including data storage space, systems, PC preparing control, and specialized corporate and client applications.

Cloud computing has three major service models and they are Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). The IaaS model gives only the hardware and system; the client installs or builds up its own particular operating frameworks, software and applications. In PaaS, an operating framework, hardware, and system are given, and the client installs or adds to its own particular software and applications. In Software as a Service model, a pre-made application, along with any required software, operating framework, hardware, and system are given.

1.2. Scheduling

In computing, scheduling [10], [14] is the technique by which work indicated by a few means is allocated to resources that finish the work. The work might be virtual calculation components, for example, strings, procedures or information streams, which are thus booked onto equipment resources.

A scheduler is the thing that does the scheduling action. Schedulers are frequently executed so they keep every PC asset occupied (as in burden adjusting), permit numerous clients to share framework resources viably, or to accomplish an objective nature of administration. Scheduling is essential to calculation itself, and a characteristic part of the execution model of a PC framework; the idea of scheduling makes it conceivable to have PC multitasking with a single central processing unit (CPU). In order to overcome this problem we propose cloud system with scheduling technique. It was achieved by the help of Optimized Scheduling Algorithm.

1.3. Workload Balancing

In computing, load balancing [9], [11] distributes workloads across multiple computing resources, such as computers, a

Received: 4 January 2016; Revised: 16 March 2016; Accepted: 28 March 2016; Published online: 06 April 2016

*Correspondence to: vivekrabsin1993@gmail.com

[1]IDepartment of Computer science and engineering, Kalasalingam Intuition of Technology, Anand Nagar, Krishnankoil, Tamil Nadu, India

computer cluster, network links, central processing units or disk drives. Load balancing intends to advance asset use, augment throughput, minimize reaction time, and maintain a strategic distance from overload of any single asset. Utilizing different parts with load balancing rather than a solitary segment might expand dependability and accessibility through excess. Load balancing more often than not includes devoted programming or equipment, for example, a multi layer switch or a Space Name Framework server process. Load balancing isolates traffic between network interfaces on a network attachment premise, while channel holding infers a division of traffic between physical interfaces at a lower level, either per bundle or on information join premise with a protocol such as shortest path bridging.

2. RELATED WORK

Amit Agarwal et al [1] have proposed to an arrangement of strategies to control the request of work to be performed by a PC framework. A decent scheduler adjusts its planning procedure as per the changing environment and the kind of assignment. In this examination paper we introduced a Generalized Priority algorithm for proficient execution of undertaking and correlation with FCFS and Round Robin Scheduling. Algorithm ought to be tried in cloud Sim toolbox and result demonstrates that it gives better execution contrasted with other conventional scheduling algorithm.

Raja Manish Singh et al [16] have proposed task scheduling assumes a key part in distributed computing frameworks. Scheduling of tasks is impossible on the premise of single criteria however under a great deal of standards and regulations that we can term as an agreement in the middle of clients and suppliers of cloud. This agreement is only the nature of administration that the client needs from the suppliers. Giving great nature of administrations to the clients as indicated by the agreement is a conclusive task for the suppliers as in the meantime there are an extensive number of tasks running at the supplier's side. The task scheduling issue can be seen as the finding or looking an ideal mapping/task of set of subtasks of various tasks over the accessible arrangement of assets (processors/PC machines) with the goal that we can accomplish the craved objectives for tasks. In this paper we are performing similar investigation of the distinctive calculations for their suitability, practicality, flexibility in the setting of cloud situation, after that we attempt to propose the half and half approach that can be embraced to upgrade the current stage further. So it can encourage cloud-suppliers to give better nature of administrations.

Chun-Yan LIU et al [2] have proposed a proficient way to deal with task scheduling algorithm remains a long-standing test in distributed computing. Regardless of the different scheduling algorithms proposed for cloud environment, those are for the most part upgrades taking into account one algorithm. And it's not entirely obvious impediments of the algorithm itself. Going for attributes of assignment scheduling in cloud environment, this paper proposes an undertaking scheduling algorithm taking into account genetic-ant colony algorithm. We exploit solid positive criticism of ant colony optimization (ACO) on meeting rate of the algorithm into account. But the decision of the underlying pheromone crucially affects the merging rate. The algorithm makes utilization of the worldwide inquiry capacity of genetic algorithm to understand the ideal arrangement rapidly, and after those changes over it into the underlying pheromone of ACO. The recreation tests demonstrate that under the same conditions, this algorithm over weighs genetic algorithm and ACO even has effectiveness advantage in extensive scale situations. It is a productive task scheduling algorithm in the distributed computing environment.

RamMohan N.R et al [17] have proposed the computational humanity is flattering extremely bulky and multifaceted. Cloud computing is becoming one of the most expanding methodologies in the computing industry. It is a novel approach for the deliverance of IT services on the World Wide Web. This model provides computing resources in the puddle for consumers, all the way through Internet. In cloud computing, resource allocation and scheduling of numerous aggregate web services is an imperative and demanding quandary. This paper estimates the various network resource allocation strategies and their applications in Cloud Computing Environment. A brief description for network resource allocation in Cloud Computing, based on differentially adapted dynamic proportions, has also been done.

Our approaches various from the above mentioned work and overcome the problems like optimize the user requests and balance the workload at cloud server by outsourcing. At first the requests are optimized by the help of optimized scheduling algorithm and then the workload are balanced by the help of particle swarm optimization algorithm by monitoring the resource usage. Once the threshold limit reached it transfer the workload to external cloud.

3. SYSTEM DESIGN

3.1. Block diagram

The Figure 1 represents overall block diagram for how to request for instance and how to create, allocate them. It also shows how they are store and retrieve the files in cloud.

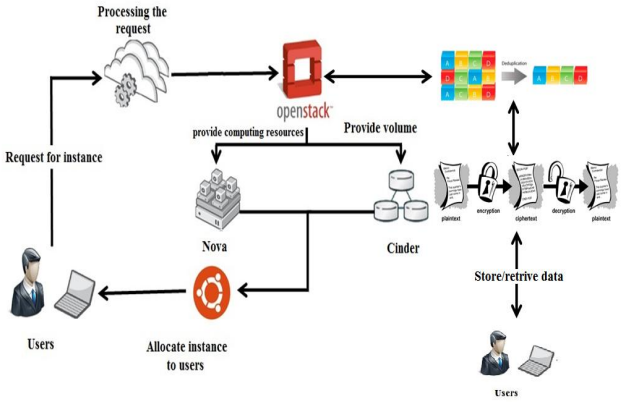

Figure 1. Block diagram of the proposed research work

Initially the customer should register their identity using registration process. Once the registration gets completed the information are validated by admin and create the account for user. User can access their cloud at any time. For first time login the user get some demo about how to use openstack and also it has some set of demo instance. Users can work with them. And also user can request for some other instance like ubuntu, fedora. The entire request provides by cloud users are gathered by cloud admin. The main work of admin is to process those request and prioritize them based on the size of memory and bandwidth by the help of optimized priority algorithm. The instance creation process consists following steps. At first, the dashboard (horizon) passes the request to the compute component (nova). And then nova passes the request to identity component (keystone) for authentication. Once verification gets complete nova request the networking component (neutron) for an ip address. Then nova request the image component (glance) for an image of user requested. Finally after getting the image, Nova mounts it on a VM host. During the boot process of the VM, it requests Neutron (DHCP component) for an IP address. Once the user get instance from cloud he can work with them without any disturbance. At last he pays the bill for what he use mostly called “pay as you use” concept. By the help of this user can access highly configured system using his low configuration system. Because all hardware and software requirements are provided from cloud and not depend on our local system. Once all the resources in the cloud are utilized and no resource for a new request then we can outsource the resources from another cloud by the help of neutron and balance the workload by the help of particle swarm optimization algorithm. This algorithm will helpful for managing the resources those who are outsourced.

User can upload their files using file upload options. To achieve reliability once the user uploaded the file diff algorithm check the uploaded file with existing file stored in cloud for whether the uploaded data already present at cloud or not. If the is not a duplicated one then the encryption process automatically triggered otherwise the new file override the existing file. If user request for particular file while encrypting, we use temporary cache memory to support the user. It helps to achieve reliability. We achieve reliability, using advanced encryption technique. After the completion of these two processes the file gets stored into cloud. If the user wants to download the file once again he have to login to cloud and click download option. Once the user click download button file get automatically decrypted and converted to original file. Now the user can view the file which he has uploaded without any inconvenience. The role of admin is to manage, create, delete all the users as well as manage the cloud.

3.2. Particle Swarm Optimization

Particle swarm optimization [12], [13] is a heuristic global optimization method and also an optimization algorithm, which is based on swarm intelligence. It comes from the research on the bird and fish flock movement behavior. The algorithm is widely used and rapidly developed for its easy implementation and few particles required to be tuned.

While hunting down sustenance, the winged creatures are either scattered or go together before they find the spot where they can discover the nourishment. While the flying creatures are hunting down sustenance starting with one place then onto the next, there is dependably a winged creature that can notice the nourishment extremely well, that is, the feathered creature is recognizable of the spot where the sustenance can be found, having the better sustenance asset data. Since they are transmitting the data, particularly the great data whenever while seeking the sustenance starting with one place then onto the next, directed by the great data, the flying creatures will in the end run to the spot where nourishment can be found. To the extent particle swam optimization calculation is

concerned, arrangement swam is contrasted with the winged creature swarm, the feathered creatures' moving starting with one place then onto the next is equivalent to the improvement of the arrangement swarm, great data is equivalent to the most positive thinker arrangement, and the nourishment asset is equivalent to the most confident person arrangement amid the entire course. The most hopeful person arrangement can be worked out in particle swarm optimization calculation by the collaboration of every person. The particle without quality and volume serves as every person, and the straightforward behavioral example is managed for every particle to demonstrate the unpredictability of the entire particle swarm. This calculation can be utilized to work out the unpredictable self assured person issues.

3.3. *Diff algorithm*

The program diff [4], [8] reports differences between two files, expressed as a minimal list of line changes to bring either file into agreement with the other. Diff has been engineered to make efficient use of time and space on typical inputs that arise in vetting version-to-version changes in computer-maintained or computer-generated documents. Time and space usage are observed to vary about as the sum of the file lengths on real data, although they are known to vary as the product of the file lengths in the worst case. The central algorithm of diff solves the 'longest common subsequence problem' to find the lines that do not change between files. Practical efficiency is gained by attending only to certain critical 'candidate' matches between the files, the breaking of which would shorten the longest subsequence common to some pair of initial segments of the two files.

3.4. *Advanced encryption standard*

The Advanced Encryption Standard or AES [7, 15] is a symmetric block cipher used by the U.S. government to protect classified information and is implemented in software and hardware throughout the world to encrypt sensitive data.
It has the following steps to encrypt the user data

a. KeyExpansion : It require 128 bit key for each encryption
b. AddRoundKey: Each byte of file compared with key using bitwise xor.

Rounds :

a. SubBytes: Each byte is replaced with another byte based on lookup table.
b. ShiftRows: Based on the offset value if shift the bytes in each row.
c. MixColumns: Combine the bytes of each column with one another using invertible linear transformation.
d. AddRoundKey

Final Round (no MixColumns) :

a. SubBytes
b. ShiftRows
c. AddRoundKey

Similarly the decryption process consist the reverse process of encryption and the steps are

a. Inverse ShiftRows
b. Inverse SubBytes
c. Inverse AddRoundKey

Inverse MixColumns

4. SYSTEM IMPLEMENTATION

The system [5] is implemented by the help of openstack tool and base operating system as ubuntu. Some of the screen shot of the system are given below.

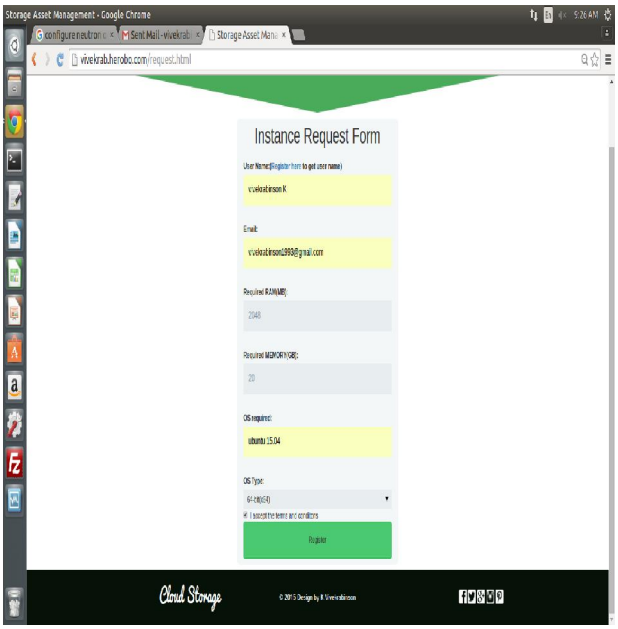

Figure 2. Shows instance request form for users. It consist username, mail-id, required ram and secondary storage, type of operating system needed and os architecture. These details are stored into database.

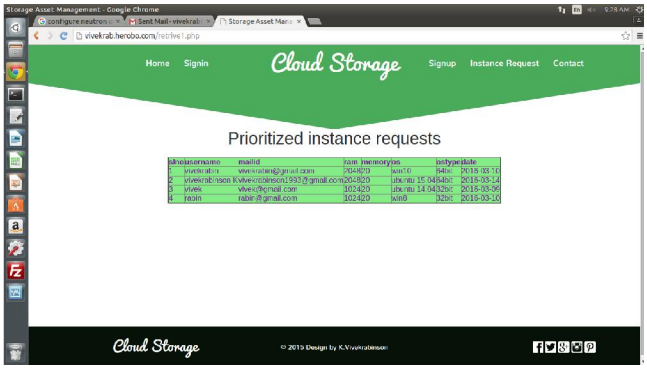

Figure 3. Shows the prioritized request of all the users. It prioritize the request based on primary memory and data of requested. It is an automatic process and no need human interaction to prioritize them.

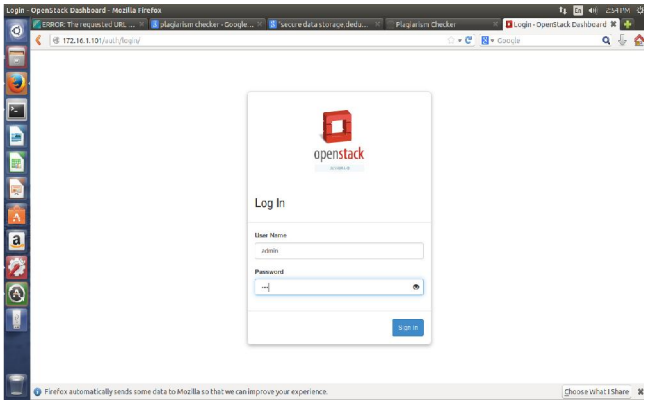

Figure 4. Shows login page for all users and admin. And it was accessed by the help of clicking sign in link in the website.

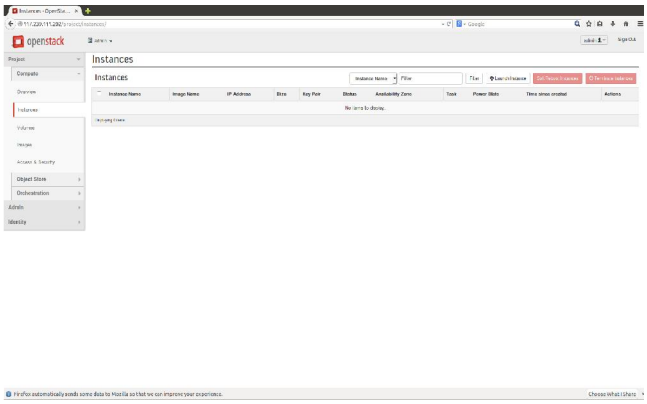

Figure 5. shows the openstack instance management screen which is used to create, launch, modify and terminate the instance.

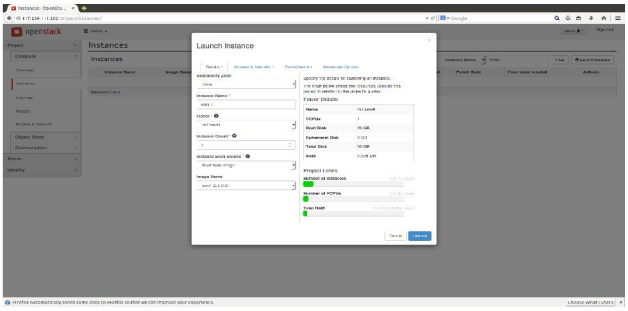

Figure 6. Shows instance creation process. This consist instance name used to identify the instance, instance flavor like tiny, small. This flavor represents the ram size allocation to particular instance. If we assign small as its flavor then it take 2 GB ram for that instance. Next we should provide instance boot source like boot from image or screenshot or some other. Finally we have to load image to complete creation process

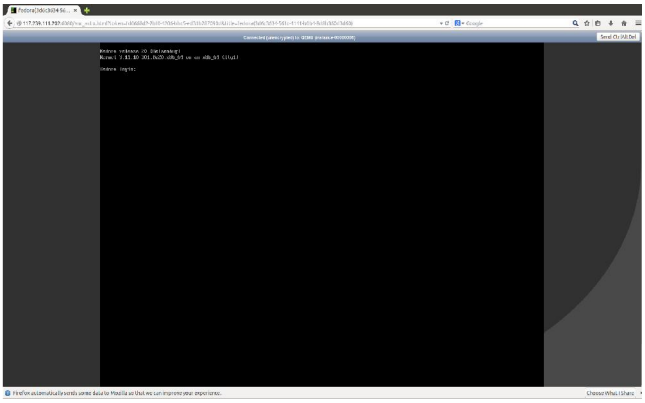

Figure 7. shows the running console of fedora instance. We can access this instance from anywhere by any base operating systems (windows, apple etc).

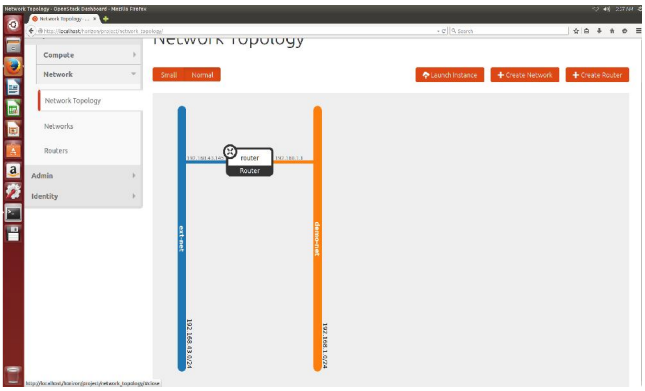

Figure 8. Shows the workload balancing of cloud environment. Here we deployed particle swarm optimization algorithm to monitor the cloud resources. If all the resources are utilized then the extra workload transferred to external cloud which was connected via neutron network.

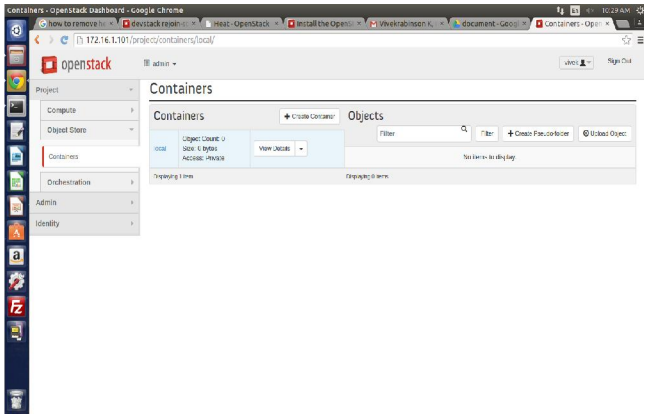

Figure 9. Shows storage information for particular users. Once user receives userid and password they can login into cloud and store their data into cloud

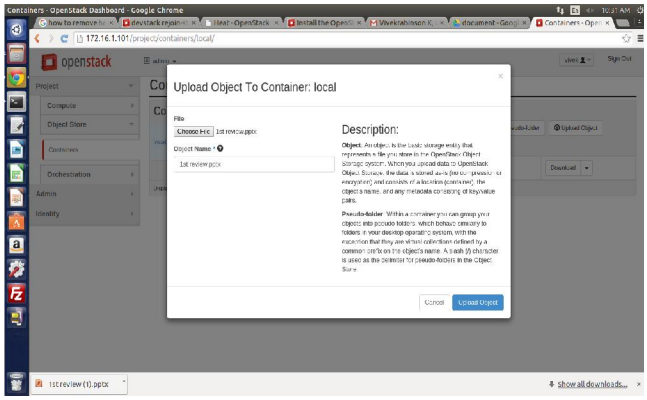

Figure 10. Shows file uploading process. To upload files into cloud user need to click upload button and choose file from local machine. And provide name for file we are going to upload.

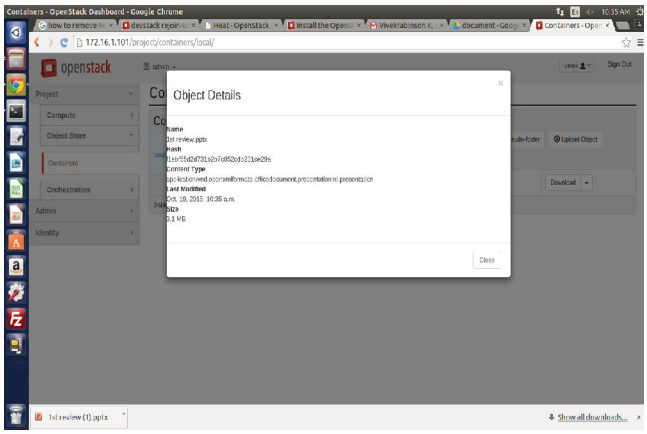

Figure 11. shows uploaded file information. And these include time and date of uploaded, name and hash key. Once we click upload button it will automatically check duplicates and encrypts the file.

5. PERFORMANCE ANALYSIS

The performance of the proposed scheme is evaluated by file size versus file storage duration. The present work, cloud is configured by the help of open stack website [5]. Further to configure nova, neutron, swift which was provided network storage services by the help of same guide.

After configuring all the components we have to create website for new user registration, instance request form and feedback and contact information. At next connect website with open stack to complete the process. The registered user information and instance request details will get stored in website alone. Once the connection established successfully we can access the cloud from anywhere.

At next we have to set break point to cloud in order to add workload balancing, duplication check and encryption to cloud. Upload particle swarm optimization algorithm to balance the workload between multiple clouds. Upload Diff algorithm into cloud for eliminate duplicated data. Using security guide, [3] enable security to data which are passed in duplication check. And the security is done by encryption process. The benefit of this tool is, it will automatically replicate data into multiple partition and will avoid single point failure.

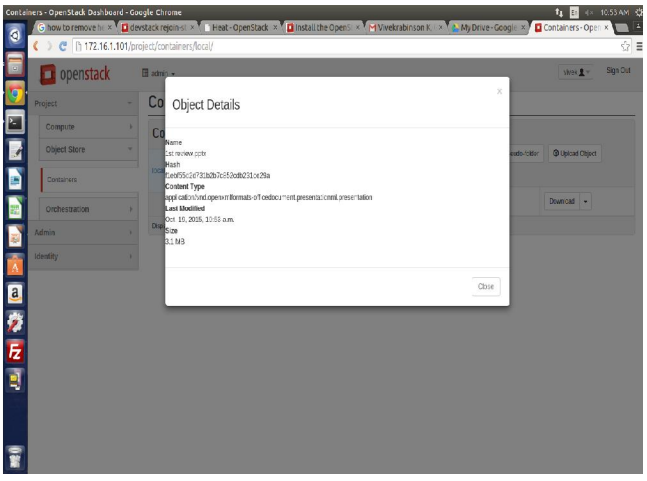

Figure 12. Shows the process of uploading the same file which was already present in the cloud. If we upload the same file it caught on duplication check and the older file will get eliminated. Because the new file may have some new information. And the details of newly uploaded shown in above figure.

The graphical chart (Figure 13) shows the efficiency of the cloud. The graph is plotted for size of file (X-axis) and time to store (Y-axis). Compared to some other cloud like open nebula and eucalyptus, open stack provide faster data storage and load balancing. The main difference between open stack with some other cloud are, it segregate the file into number of pieces and then stored them. Thus it improve the efficiency by reduce the amount of time required to store the file.

The main advantage of this project is mobile friendly. User can access their cloud using their handheld device with the help of internet. Due to this we no need to buy mobile with larger internal storage. Simply buy a basic mobile with internet connection and upload the files whenever we need.

6. CONCLUSION

The present system provided cloud environment to users and prioritize the user request those who are requested for instances and workload was balanced via outsourcing. User can store and retrieve their files into cloud from anywhere and any device. The reliability of the system and eliminate duplicated files from the cloud achieved by diff algorithm. Cloud system security is well managed and tightened by using the AES encryption standard.

REFERENCES

[1] Amit Agarwal and Saloni Jain, "Efficient Optimal Algorithm of Task Scheduling in Cloud Computing Environment", *International Journal of Computer Trends and Technology (IJCTT),* vol 9, num 7, pp. 344-349, 2014.

[2] L. I. U. Chun-Yan, Z. O. U. Cheng-Ming, W. U. Pei, "A task scheduling algorithm based on genetic algorithm and ant colony optimization in cloud computing", 13th International Symposium on Distributed Computing and Applications to Business, Engineering and Science, 2014, pp. 68-72.

[3] http://ais.seecs.nust.edu.pk/colonial/siteData/Documentation%20on%20adding%20encryption%20to%20OpenStack%20Swift_v1_4.pdf

[4] http://c2.com/cgi/wiki?DiffAlgorithm

[5] http://docs.openstack.org/juno/install-guide/install/apt/content/

[6] http://docs.openstack.org/security-guide/tenant-data/data-encryption.html

[7] https://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[8] https://en.wikipedia.org/wiki/Diff_utility

[9] https://en.wikipedia.org/wiki/Load_balancing_%28computing%29

[10] https://en.wikipedia.org/wiki/Scheduling_%28computing%29

[11] http://searchnetworking.techtarget.com/definition/load-balancing

[12] http://www.mnemstudio.org/particle-swarm-introduction.htm

[13] http://www.swarmintelligence.org/tutorials.php

[14] http://www.tutorialspoint.com/operating_system/os_process_scheduling.htm

[15] http://www.tutorialspoint.com/cryptography/advanced_encryption standard.htm

[16] Raja Manish Singh, Sanchita Paul, Abhishek Kumar, "Task Scheduling in Cloud Computing: Review", International Journal of Computer Science and Information Technologies, Vol. 5 (6), pp. 7940-7944, 2014.

[17] N. R. RamMohan, E. Baburaj, "Resource Allocation Techniques in Cloud computing-Research Challenges for Applications", Fourth International Conference on Computational Intelligence and Communication Networks, 2012, pp. 556-560.

[18] Xingquan Zuo, Guoxiang Zhang, Wei Tan, "Self-Adaptive Learning PSO-Based Deadline Constrained Task Scheduling for Hybrid IaaS Cloud", IEEE Transaction on automation science and engineering, Vol. 11, no. 2, pp. 564-573, 2014.