ijecs open access www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 6 June, 2014 Page No. 6422-6431

Analysis of Performance for Data Center under for Private Cloud through Cloud Computing

Ankush Dhiman, Mauli Joshi

Research Scholar M.Tech (CSE)
MVEC, Jagadhri, INDIA
dhiman.ankush@outlook.com

Assistant Professor in Computer Science & Engineering
MVEC, Jagadhri, INDIA
mauli.joshi@gmail.com

Abstract: Cloud computing is a sort of technology that now aiming to power next generation data centers and to enable application service providers to lease data center capacities to deploy applications depending on user QoS (Quality of Service) needs. Adopting the cloud computing is like signing the new form of a website. In cloud computing, the GUI that controls it make is directly control the hardware resource and your application. But one of the difficult phase in cloud computing is to deploy in real environment. It is difficult to know the exact cost as well as requirement until and unless we own the service not only that whether it will support the existing applications that is available on traditional data center or had to design a new application for the cloud computing environment. Some of the parameters such as security issue, fault tolerance and latency are need to keen care before deploying that we can enable to know solely before implementation. But by using simulation, we can do the experiment before deploying it to real environment. Simulation provide a way to understand the real environment of cloud computing and after achieving the successful result we can commence deploying application in cloud computing environment. By exploiting the simulator, we can save the time as well as cost also.

Keywords: Cloud Computing, Virtual Machine, Simulator, CloudSim, Data Center, Virtualization, Cloudlet

1. Introduction

Cloud Computing is a marketing term that delivers the service as Infrastructure, Platform and Software as a service in pay-as-you-go model to users. It is also rendered as utility computing. Moving an existing application or new application to the cloud will almost surely have some trade-offs in term of performance. Organizations considering moving to cloud computing definitely ought to suppose in prices for up the network infrastructure needed to run applications within the cloud.

On the other hand, bandwidth continues to enhance and approaches like compression, dynamic caching, pre-fetching and other related web acceleration technologies will result in major performance enhancements for end users, often exceeding 50%. The application has to design or re-design in such a manner that it will support cloud computing in order to attain maximum performance.

The growth of cloud computing has taken all the attention of several communities like student, business, researchers, consumer and government organization. Main reason for emergence of cloud computing concept is *Big data*. In everyday life, a lot of data in the size of PETA bytes are uploaded in the digital world that required lots of storage and computing resources.

Cloud based and traditional web application embraces web hosting, social networking, content delivery and real time instrumented data processing. These applications have distinct configuration, composition and deployment requirements. In a real cloud environment, measuring the performance of allocation and scheduling policies for distinct applications and service models under varied conditions is exceedingly challenging issue because (1) clouds exhibit varied demand, system size, supply pattern and (2) customers have heterogeneous and competing QoS needs. Deploying the cloud computing in real infrastructure like Amazon EC2, Google limits the experiments to the size of the infrastructure and build the replica of result an especially tough endeavor. The most reason for this being the conditions prevailing within the Internet-based environments area unit on the far side the management of developers of resource allocation and application scheduling algorithms.

2. Background

2.1 Definition

The Cloud Computing is not evolved yet completely and also

there exists no wide accepted definition. Based on our expertise, we tend to propose an early definition of Cloud computing as follows:

"*Cloud computing is online service to provide resources as service on demand when needed by users through internet that can be leased and released with least management effort and human intervention with service provider*"

2.2 Delivery Models

Three delivery models were defined by NIST which are: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service.

Infrastructure as a Service – IaaS is a service model that enable user to share hardware resources for executing services using virtualization technology. Its major objective is to make resources like servers, storage and networks more expeditiously accessible by applications and operating systems. Examples of IaaS incorporate GoGrid, Amazon S3.

Platform as a Service – PaaS refers to the delivery of computing platform and solution stack as a service without software installation or downloads for IT managers, developers or end users. In this model, user does not manage the infrastructure but he controls deployed applications possibly their configurations. Examples of PaaS incorporate Force.com, Google App Engine.

Software as a Service – SaaS is a process in which Application Service Provider (ASP) offer distinct software applications over the internet without installing and operating on user's computer. Example of SaaS incorporates Saleforce.com Google Apps.

2.3 Deployment Models

In the deployment model of cloud, storage, platform, networking and software infrastructure are offered as services which can be scale up and down as per user's requirement. As per NIST definition, Cloud Computing has four prime deployment models which are:

Private Cloud – Also called Internal Cloud, a private cloud has dedicated infrastructure to particular organization in which scalable resources provided by cloud vendor pool together and available to cloud users to share and exploit. Private Cloud is of two types: *On-Premise* – Hosted within organization own facility and *External Hosted* – Used by one organization but hosted by third party. An example of Private Cloud embraces Eucalyptus Systems.

Public Cloud – Also called External Cloud, a public cloud is a cloud in which computing infrastructure is hosted by cloud vendor at the vendor's premises. It offers services to general public on pay-as-you-go basis. An example of Public Cloud embraces Google App Engine.

Hybrid Cloud – Hybrid Cloud is a combination of private and public cloud that interoperates. In this case, users outsource non-business critical information and processing to the public cloud while keeping business-critical services and data in their control. An example of Hybrid Cloud embraces Amazon Web Services (AWS).

Community Cloud – A Community Cloud is basically exploited and controlled by a set of organizations that have shared interests such as specific security requirements or a common mission. An example of Community cloud embraces Facebook.

2.4 Essential Characteristics

As per NIST definition, the Cloud Computing has five essential characteristics which are described below:

On-Demand Self-Service – User can easily access and utilize the resources like network space or server when required through the internet without human intervention with service provider.

Broad Network Access – Cloud capabilities are available to users through the internet and can be accessed from distinct devices such as Mobile Phones, Desktop Computers, Tablets and Laptops.

Measured Service – Cloud Computing provides service on pay-per-usage model which may vary from service to service. Measured service reduces the operating cost as it charges users on pay-as-you-go basis.

Resource Pooling – Resource Pooling permits a Cloud Provider to provide its services to multiple users using Multitenant model. Resources available like physically or virtually can be leased or released as per user demand.

Rapid Elasticity – One of the significant attribute of Cloud Computing is the computing resources can be provisioned expeditiously and released without human intervention when no longer demanded thus lower the operating cost.

3. Related Work

Chang Jie Guo, Wei Sun, Ying Huang, Zhi Hu Wang, Bo Gao et al. explains that the development of a native multi-tenant application poses many new research challenges brought by heterogeneous requirements. The design and implementation principles to instruct the native multi-tenant offering development and operation management are proposed and a programming model and framework to simplify and speed up the application development.

Lijun Mei, W.K. Chan, T.H. Tse et al. found that cloud computing is an emerging computing paradigm. It aims to share data, calculations, and services transparently among users of a

massive grid. Although the industry has started selling cloud-computing products, research challenges in various areas, such as UI design, task decomposition, task distribution, and task coordination, are still unclear. Therefore, to study the methods to reason and model cloud computing as a step toward identifying fundamental research questions in this paradigm.

Sudip Chahal, Jay Hahn-Steichen, Das Kamhout, Rick Kraemer et al. in an Enterprise Private Cloud Architecture and Implementation Roadmap states that the private cloud is a shared multi-tenant environment built on a highly efficient, automated, and virtualized infrastructure. The private cloud is a shared multi-tenant environment built on a highly efficient, automated, and virtualized infrastructure.

Dustin Amrhein, Patrick Anderson, Andrew de Andrade, Joe Armstrong, Ezhil Arasan B, Richard Bruklis, Ken Cameron, Reuven Cohen et al. discussed that the Cloud Computing Use Case group brought together cloud consumers and cloud vendors to define common use case scenarios for cloud computing. The use case scenarios demonstrate the performance and economic benefits of cloud computing and are based on the needs of the widest possible range of consumers. The goal is to highlight the capabilities and requirements that need to be standardized in a cloud environment to ensure interoperability, ease of integration and portability.

Hong Cai, Ning Wang, Ming Jun Zhou et al. describes the core SaaS multi-tenancy models consisting of tenant interceptor, tenant context, tenant map, tenant propagation, remote resources (such as database server, LDAP server, message queue server) in a distributed multi-tier Web system and also introduces the end to end process of making an existing Web application to be multi-tenancy enabled, and separating concerns of different roles involved.

4. Virtualization

To define the concept of virtualization is very difficult as it has distinct meanings according to context in which it is exploited. It isolates the computer hardware resources into various execution environments. In this section, firstly we will define the term of virtualization then we explain the aspects of virtualization. This will assist you to understand the concept of virtualization very clearly.

4.1 Definition

"Virtualization can be defined as technology which allows sharing single physical instance of a resource or server among multiple tenants and organizations."

4.2 Aspects of Virtualization

Several aspects of virtualization are described below as (See figure1):

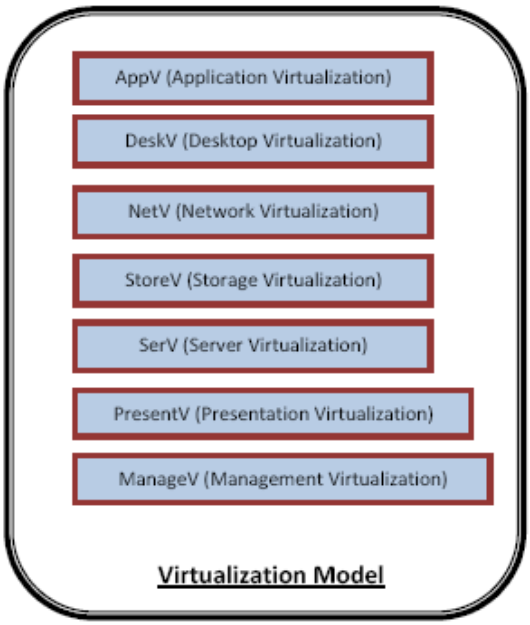

Figure 1: Aspects of Virtualization

Application Virtualization (AppV) – Application Virtualization is based on the same principle as server virtualization except that it does not provide an engine to run an entire operating system. In this virtualization aspect, Application is not installed on the operating system.

Desktop Virtualization (DeskV) – In desktop virtualization, virtual machines is exploited to provision desktop machines.

Network Virtualization (NetV) – Network virtualization is a technique that allows taking control of available bandwidth by dividing it into various channels that can be further given to some specific resources for particular usage.

Storage Virtualization (StoreV) – Storage virtualization is virtualization technique in which distinct physical storage devices are integrated that appears to single storage device or pool. It can be of any type such as Direct Attached Storage (DAS), Network Attached Storage (NAS) and Storage Area Networks (SANs) which can be connected through various protocols like Network File System (NFS), Internet Small Computer System Interface (iSCSI).

Server Virtualization (SerV) – Server virtualization is a technique in which single server runs multiple virtual machines. Every virtual machine is independent of others and can run distinct operating system and application.

Presentation Virtualization (PresentV) – Presentation virtualization is also rendered as Server Based Computing. It is an application delivery method which delivers users applications and desktops from shared server or server based computing.

Management Virtualization (ManageV) – Management virtualization is a virtualization technique in which the physical and virtual datacenter are managed to give one single unified infrastructure for the provision of services. Best known example of Management Virtualization is Resource Pooling.

5. Multitenancy

In this section, we will first describe the concept of "Tenant" then "Multitenancy".

5.1 Tenant

The idea of "Tenant" in context of Cloud Computing is not as simple as it might first appear. In Cloud Computing environment, each user is called tenant. A tenant has provided the ability to customize some parts of the application such as color of the User Interface (UI) or business terms but he is not enabled to customize the application's code. So here we can define the Tenancy as:

"*Tenancy is a term used in cloud computing that represents each customer who has slightly given the ability to customize parts of application rather than fully customization."*

We can easily understand the concept of Tenant through the simple example of Amazon Web Services (AWS) which is a cloud service provider offerings services like storage and backup, hosting, e-commerce and media hosting etc are few one. Companies such as Urban Spoon, Autodesk and Second Life are tenants of AWS, in which they utilize AWS storage and compute resources to power their customer offerings.

5.2 Classification of Tenant

On the basis application usage, tenant can be classified into two types as (See figure 2):

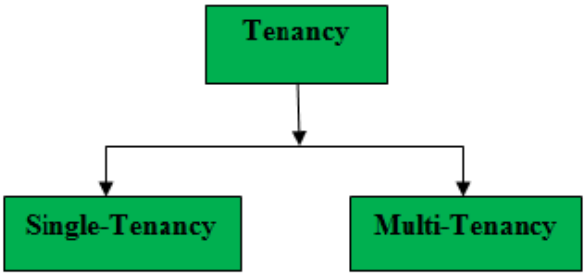

Figure 2: Classification of Tenancy

5.2.1 Single Tenant

Single-tenancy refers to a situation in which a single user or set of related users make use of dedicated resource. Traditional on-premise software applications are *Single-tenant Applications* (See figure 3).

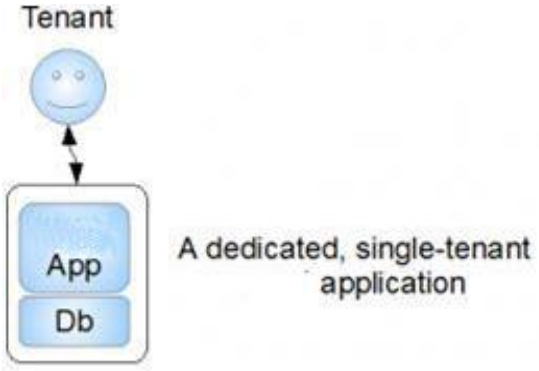

Figure 3: A Single-tenant Application

For example, you might purchase and install an office application suite such as Microsoft Office for your PC or Mac. You are the only one that uses this instance of the app.

Some of the benefits of Single-tenant Applications are as follows:

- Enables the customer to customize the app as per requirement
- Because application executes on dedicated machine so no one can easily access the data that the application maintains
- Provide control related to application incorporating scheduling, maintaining and backups etc.

Single tenant has some advantages but it has drawbacks also which are described below:

- Requires dedicated set of resources to fulfill the requirements of a particular organization
- You must learn how to install and configure the application for yourself. With PC software, this is usually not much of a concern.
- Once installed, you have to maintain the application yourself. Software maintenance incorporates tasks like the installation of software updates and patches.
- You can easily recover and back up your data if some problem arises.

5.2.2 Multitenant

Multitenancy refers to situation in which many users make use of shared resources. In other words, it is an architecture in which a single instance of a software application serves multiple users. Many modern, commercial, on-demand applications are *Multi-tenant Applications* (See Figure 4).

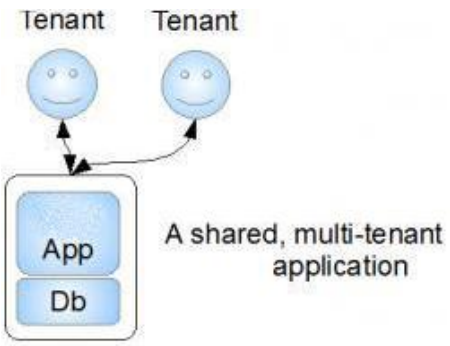

Figure 4: A Multitenant Application

For example, you might access Google Apps using a web-browser for your word processing, spreadsheet, and presentation software needs. At the same time, many other people around the world are also using Google Apps to do their work, all without ever having to download, install, and maintain applications and data.

5.3 Multitenancy in Application Tier

The Application Tier can contain a lot of servers offering various components of the overall service. Some of the common ones that will be impacted by multi-tenancy are:

Identity Management – Identity management (IdM) is a component if included in SaaS then it need to be carefully designed so as not to compromise with security. The IdM server manages the interaction with various customer repositories and ensures that access to these directories is managed individually for each tenant.

Integration – Integration servers are embraced with getting data from or sending it to other systems often within corporate environment. Thus makes critical to handle multiple tenants in a secure way, keeping access credentials robustly separated and protected.

Communication – If we talk about communication servers, multitenancy has more impact on inbound communication than outbound communication. Inbound communications are generally needed to be routed to a specific tenant instance whereas outbound communications are inherently addressed to specific users.

5.4 Multitenancy and Cloud Computing

In cloud computing environment, two different extremes are considered in multi-tenant that is: *Virtual Multi-Tenancy* and *Organic Multi-Tenancy*.

5.4.1 Virtual Multitenancy

Virtual Multitenancy refers to situation in which virtualization is at the foundation of the system architecture. In this scenario, consider applications multi-tenant that concurrently execute within virtual machine (VMs) on top of the same virtual infrastructure. To illustrate virtual multi-tenancy, consider the following example applications that use Amazon Web Services (AWS) (See figure 5).

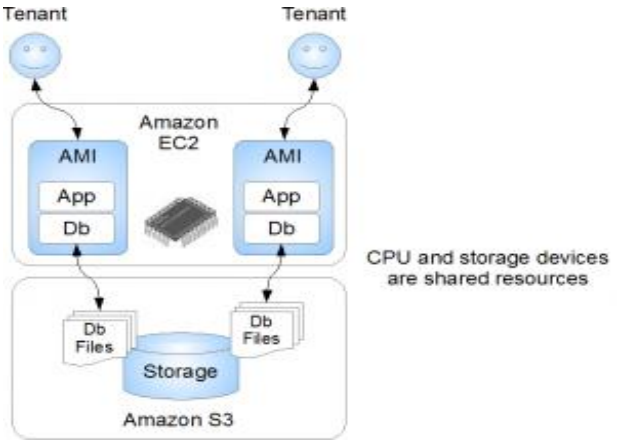

Figure 5: Virtual Multitenancy

5.4.2 Organic Multitenancy

Organic Multitenancy refers to a design in which every component such as database schema, operating systems, the application server, firewall etc within the system architecture of a single software instance is shared among all tenants (See figure 6). The term organic is used because of its design unlike with virtual multi-tenancy requires software developers to explicitly consider multi-tenant design patterns and code them into the application. To illustrate organic multi-tenancy, consider the following application that uses Force.com.

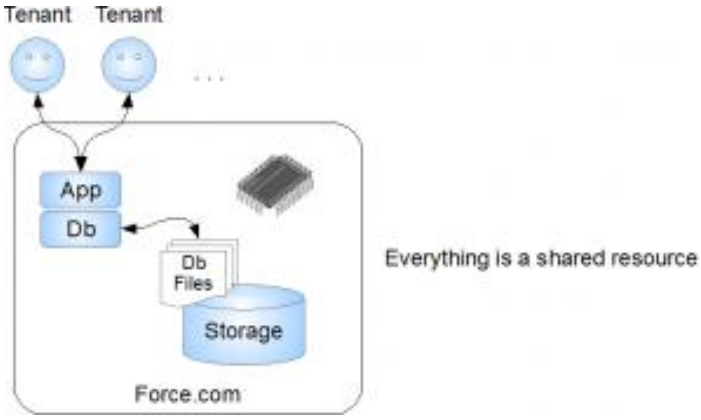

Figure 6: Organic Multitenancy

6. Research Proposal

Attempting to weave multitenancy throughout the fabric of an application's core logic and its underlying infrastructure is a complex undertaking. To build metadata-driven, multitenant applications from scratch without any prior experience is destined to be a time consuming and error-prone effort. Ultimately, many would be SaaS providers struggle to succeed in building multitenant applications and end up wasting valuable time that could have been spent targeted on the innovation of core application functionality and features. One problem is that traditional application development frameworks and platforms are not equipped to handle the special needs of modern Internet applications. As a result, new types of

platforms are emerging to help simplify the development and deployment of multitenant applications. Force.com is the first and most mature general purpose, multitenant, Internet application development platform available today.

Multi-tenancy is basically a new software architecture principle in the realm of the Software as a Service (SaaS) business model. It permit to make full use of the economy of scale, as multiple customers – "tenants" – share the same application and database instance. All the while, the tenants enjoy a highly configurable application, making it appear that the application is deployed on a dedicated server. The key benefits of multi-tenancy are enhanced utilization of hardware resources and improved ease of maintenance, in particular on the deployment side.

We have exploited distinct algorithm to create a data centre with varying number of host and run cloudlet on it. The algorithm shows how to create a data centre with 'n' number of hosts and run on different cloudlet on it. The cloudlet list show the Virtual Machine list creates to implements a cloud for Multi-Tenant's Applications. Algorithm also shows how to create a datacenter with one host and run two or more than two cloudlets on it. The cloudlets run under Virtual Machines i.e. VMs with the same MIPS (Millions of Instructions per Seconds) needs and take the same time to complete the execution. Algorithm also shows how to create a datacenter with two or more hosts and run two cloudlets on it. The cloudlets run in VMs with distinct MIPS requirements. The cloudlets will take distinct time to complete the execution depending on the requested VM performance.

To Implement these algorithm we use Java Language for CloudSim Tool and whatever the Result produce by CloudSim Simulator on the basis of those parameters we produce the Graphical Result in Cloud Analyst Tool.

7. Overview of Simulator

7.1 CloudSim

CloudSim is basically a library for simulation of cloud computing scenarios. CloudSim provides basic classes for describing virtual machines, data centers, applications, users, computational resources and policies. CloudSim supports VM scheduling at two levels: at Host level and VM level. At *Host Level* – It is possible to specify how much of the overall processing power of each core in a host will be assigned at each VM. *At VM Level* – In this the VMs assign specific amount of the available processing power to the individual task units that are hosted within its execution engine.

Some CloudSim functionalities are provided as follows:

- Supports for modeling and simulation of federated clouds.
- Supports for modeling and simulation of large scale cloud computing data centers
- Supports for dynamic insertion of simulation elements, stop and resume of simulation.
- Supports for modeling and simulation of data center network topologies and message passing applications.
- Supports for modeling and simulation of energy aware computational resources.

The fundamental classes of CloudSim which constitutes the basic building blocks of simulator are described as follows:

- Data Center – This class of CloudSim models the core infrastructure level services offered by resource providers in a cloud computing environment.
- DatacenterBroker – This class models a broker that is responsible for mediating between users and service providers depending on users QoS needs and deploys service tasks across clouds.
- SANStorage – This class models a storage area which is commonly available to cloud based data centers for storing large chunks of data.
- VirtualMachine – This class models an instance of a VM whose management is the responsibility of host component during its life cycle.
- Cloudlet – This class models the cloud based application services that are commonly deployed in the data centers.
- CloudCoordinator – This class offers federation capacity to a data center.
- BWProvisioner – This class models the provisioning policy of bandwidth to VMs which are deployed on a host component.
- MemoryProvisioner – This class models the provisioning policy for allocating memory to VMs.
- VMProvisioner – This class represents the provisioning policy that a VM monitor utilizes for allocating VMs to hosts.
- VMMAllocationPolicy – This class implemented by a host component which models the policies required for allocating processing power to VMs.

EXPERIMENTAL SETUP AND RESULT

To evaluate the performance of cloud, results were simulated in Window 8 basic (64-bit), i3 processor, 2348 M Processor, 2.3 GHz of speed with memory of 4 GB and the language used is Java. First code of simulation is tested on one data center with one host and run on one Cloudlet.

Initialize the CloudSim Package:

Int num_user = 1; //number of cloud users

Calendar calendar = Calendar.getInstance ();

Boolean trace_flag = false; //mean trace events

Initialize the CloudSim Library:

CloudSim.init (num_user, calendar, trace_flag);

Create Datacenters: These are the resource providers in CloudSim. We need at least one of them to run a CloudSim simulation.

Datacenter datacenter0 = createDatacenter ("Datacenter_0");

Create Broker:

DatacenterBroker broker = createBroker ();

Int brokerId = broker.getId ();

Create one Virtual machine:

Vmlist = new ArrayList<Vm>();

VM description:

Int vmid = 0;

Int mips = 1000;

Long size = 10000; //image size (MB)

Int ram = 512; //vm memory (MB)

Long bw = 1000;
Int pesNumber = 1; //number of cpu

String vmm = "Xen"; // VMM name

Create VM:

Vm vm = new Vm (vmid, brokerId, mips, pesNumber, ram, bw, size, vmm, new CloudletSchedulerTimeShared ());

Add the VM to the vmList:

vmList.add (vm1);

Submit vm list to the broker:

Broker.submitVmList (vmList);

Creation of Cloudlets:

Cloud
letList = new ArrayList<Cloudlet>();

Cloudlet Properties:

Int id = 0;

PesNumber = 1;

long length = 400000;

long fileSize = 300;

long outputSize = 300;

UtilizationModel utilizationModel = new UtilizationModelFull ();

Cloudlet cloudlet = new Cloudlet (id, length, pesNumber, fileSize, outputSize, utilizationModel, utilizationModel1, utilizationModel1);

Cloudlet.setUserId (brokerId);

Cloudlet.setVmId (vmid);

Add the Cloudlets to the list:

cloudLetList.add (cloudlet1);

Submit cloudlet list to the broker:

Broker.submitCloudletList (cloudletList);

Start Simulation:

CloudSim.startSimulation ();

CloudSim.stopSimulation ();

Final step: Print result when simulation is over

List<Cloudlet> newList =
broker.getCloudletReceivedList ();

printCloudletList (newList);

// Print the debt of each user to each datacenter

Datacenter0.printDebts ();

The output of this simulation is:

==========OUTPUT==========

Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time
0	SUCCESS	2	0	400	0	400

*****PowerDatacenter: Datacenter_0*****

User ID	Debt
3	35.6

****

After enhancing the datacenter with two hosts and run two cloudlets on it. The cloudlets run in VMs with distinct MIPS needs. The cloudlets will take distinct time to complete the execution depending on the requested VM performance.
It prints the following output:

```
==========OUTPUT==========
Cloudlet ID  STATUS   Data center ID  VM ID  Time  Start Time  Finish Time
0            SUCCESS  2               0      1000  0           1000
1            SUCCESS  2               1      1000  0           1000
*****PowerDatacenter: Datacenter_0*****
User ID      Debt
3            35.6
**************************************
```

7.2 Cloud Analyst

With the above simulator and a lot of other it is simple to do simulation on massive scale environment. But it would be simpler if the tool is with visualization capacity. It is a tool which can separate the simulation experiment and programming exercise. By this tool, we concentrate on simulation parameter rather than technicalities of programming. Through cloud analyst it is easy to do simulation and get the result in graphical view that is simple to understand and print the report. It's easy use, ability to outline the simulation with a high degree of configurability and flexibility, graphical output, repeatability easy extension or technologies exploited in cloud analyst are Java, Java Swing, CloudSim and SimJava. SimJava is the underlying simulation framework of CloudSim and some of its features are exploited directly in CloudAnalyst.

Experiment on CloudAnalyst simulation with below parameter

Two data center with OS as Linux and VMM as Xen is created. And their physical hardware details are below:

- Memory – 204800 MB
- Storage – 100000000 MB
- No of Processors – 4
- Processor Speed – 10000
- VM Policy – Time Shared

Simulation duration is set to 60 min with 5 User bases for distinct location. Table 1 below is data provided in every user base.

Table 1 User bases for distinct location

Name	Region	Requests per User per Hr	Data Size per Request (bytes)	Peak Hours Start (GMT)	Peak Hours End (GMT)	Avg Peak Users	Avg off-Peak Users
UB1	0	60	100	3	9	1000	100
UB2	1	60	100	3	9	1000	100
UB3	2	60	100	3	9	1000	100
UB4	3	60	100	3	9	1000	100
UB5	4	60	100	3	9	1000	100

OUTPUT OF THE SIMULATION

Following are the outputs produced in simulation environment under distinct conditions. Figure 7 represents the snapshot of the simulator in geographical view.

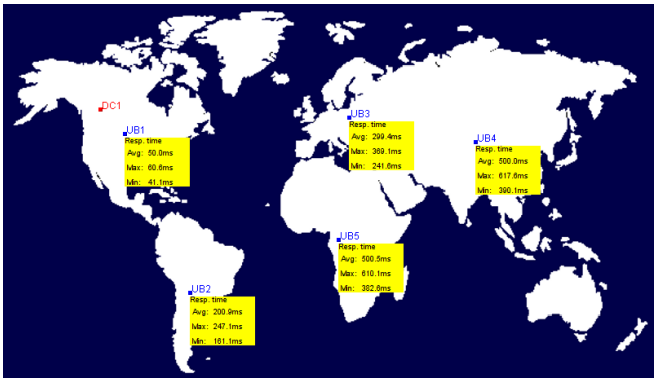

Figure 7 Screen Shot in Geographical View

- Overall Response Time

Table 2 Overall Response Time

	Avg (ms)	Min (ms)	Max (ms)
Overall Response Time:	310.54	41.11	617.61
Data Centre Processing Time:	0.32	0.02	0.86

- Response Time by the Region

Table 3 Response Time by the Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.006	41.106	60.606
UB2	200.871	161.111	247.114
UB3	299.384	241.614	369.116
UB4	499.963	390.115	617.613
UB5	500.494	382.599	610.122

- User Base Hourly Average Response Times

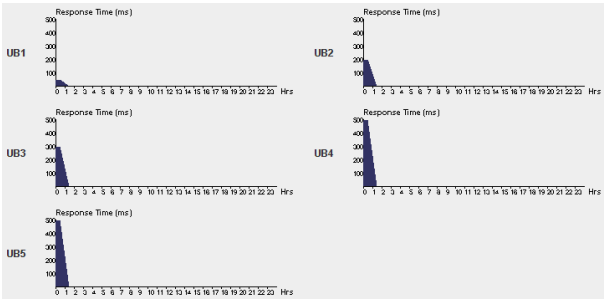

Figure 8 User Base Hourly Average Response Times

- Data Center Request Servicing Times

Table 4 Data Center Request Servicing Time

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	0.322	0.016	0.856

RESULT: With the above simulation we are able to define cost of such an environment in real world.

Cost

Total Virtual Machine Cost - $0.50
Total Data Transfer Cost - $0.32
Grand Total - $0.82
Cost in Data Center

Table 5 Data Center Request Servicing Time

Data Center	VM Cost	Data Transfer Cost	Total
DC1	0.502	0.321	0.822

8. Conclusion & Future Work

Recently lots of efforts have been made to design and develop cloud technologies focusing on defining new policies, methods and mechanisms for efficiently managing Cloud Infrastructures. Researchers require tools to test the newly developed methods and policies to evaluate the hypothesis prior to real deployment in an environment where one can reproduce tests. Simulation approach assists the one in evaluating cloud computing systems and application behaviors offer significant benefits as they permit cloud developers (1) to evaluate the performance of their provisioning and service delivery policies in a repeatable controllable environment free of cost (2) to evaluate the performance bottlenecks before real world deployment on commercial clouds.

In this paper, we have discuss two simulator based on high performance computing network that are CloudSim for cloud computing and CloudAnalyst for cloud environment cost wise. CloudSim is a customizable tool for modeling and simulation of extensible clouds. CloudSim permits extension and definition of policies in all the components of the software stack that makes it suitable as a research tool that can handle the complexities arising from simulated environments. In this paper, we have also proposed the codes for simulation. We see the several outputs in which the information about the Cloudlets, Status, Datacenter ID, Virtual Machine ID, Start Time, and Finish Time is given. And by changing the number of Host, Datacenters and Cloudlets, we saw the difference.

We have performed small simulation as well with some parameter to understand the cost of using cloud computing in real world. Through CloudAnalyst we are able to find out that if user willing to deploy on high performance network, we can exploit to study it before deploying in real world. Future work will be a relative study on load balancing algorithm on CloudAnalyst and to compare its result and predict which one is best load balancing algorithm in cloud computing environment.

References

[1] Hong Cai, Ning Wang, Ming Jun Zhou, "A Transparent Approach of Enabling SaaS Multi-tenancy in the Cloud", 2010 IEEE 6th World Congress on Services.

[2] Chang Jie Guo, Wei Sun, Ying Huang, Zhi Hu Wang, Bo Gao, "A Framework for Native Multi-Tenancy Application Development and Management''2007 9th IEEE International Conference on E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services.

[3] Lijun Mei, W.K. Chan, T.H. Tse, "A Tale of Clouds: Paradigm Comparisons and Some Thoughts on Research Issues", 2008 IEEE Asia-Pacific Services Computing Conference.

[4] Jack Brass, "Physical layer Network Isolation in Multi-tenant Clouds", International Conference on Distributed Computing Systems Workshops 2010.

[5] "Introduction to Cloud Computing Architecture", White Paper 1st Edition, June 2009.

[6] Thesis by lvailo P. Sokolov, "Cloud Computing: Overview, Concepts and Business Deployment Scenarios" Bachelor's 2009 Vienna University of Economics and Business.

[7] Dustin Amrhein, Patrick Anderson, Andrew de Andrade, Joe Armstrong, Ezhil Arasan B, Richard Bruklis, et n al. "Cloud Computing Use Cases", A white paper produced by the cloud computing use case Discussion Group, Version 2.0 30 October 2009.

[8] "Application Tenancy" Progress Software Corporation, 14 Oak Park, Bedford, MA 01730 USA.

[9] Kirill Kolyshkin, Virtualization in Linux" September 1, 2006 Kir@openvz.org

[10] "Converting your Web Application into a Multi-tenant SaaS Solution" Corent technology © 2011

[11] Guoling Liu,"Research on Independent SaaS Platform" School of Information Science and Technology Shandong Institute of Light Industry Jinan, China.

[12] Enrique Jimenez Domingo, Javier Torres Niño, Angel Lagares Lemos, Miguel Lagares Lemos, Ricardo Colomo Palacios, Juan Miguel Gomez Berbís, **"**CLOUDIO: A Cloud Computing-oriented Multi-Tenant Architecture for Business Information Systems",2010 IEEE 3rd International Conference on Cloud Computing.

[13] Franclin S. Foping, Ioannis M. Dokas, John Feehan, Syed Imran, "A New Hybrid Schema-Sharing Technique for Multitenant Applications", Cork Constraint Computation Centre, University College Cork, Ireland.

[14] Sivaram Shunmugam, "Cloud computing with red hat solution", Red Hat Asia Pacific Pte Ltd.

[15] Xing Pu, Ling Liu1, YiduoMei, Sankaran Sivathanu, Younggyun Koh, Calton Pu, "Understanding Performance Interference of I/O Workload in Virtualized Cloud Environments" 2010 IEEE 3rd International Conference on Cloud Computing.

[16] Pankaj Goyal, "Policy-based Event-driven Services-oriented Architecture for Cloud Services Operation & Management", 2009 IEEE International Conference on Cloud Computing.

[17] Sudip Chahal, Jay Hahn-Steichen, Das Kamhout, Rick Kraemer, Hong Li, Chris Peters "An Enterprise Private Cloud Architecture and Implementation Roadmap" IT@Intel White Paper Intel Information Technology Business Solution.

[18] Zeeshan Pervez, Sungyoung Lee, Young-Koo Lee," Multi-Tenant, Secure, Load Disseminated SaaS Architecture'' Department of Computer Engineering, Kyung Hee University, South Korea.

[19] Qiang Li, Qinfen Hao, Limin Xiao, Zhoujun Li, "Adaptive Management of Virtualized Resources in cloud Computing Using Feedback Control" The 1st International Conference on Information Science and Engineering (ICISE2009)

[20] MingXue Wang, Kosala Yapa Bandara and Claus Pahl, "Process as a Service - Distributed Multi-tenant Policy-based Process Runtime Governance", 2010 IEEE International Conference on Services Computing.

[21] Afkham Azeez, Srinath Perera, Dimuthu Gamage, Ruwan Linton, Prabath Siriwardana, Dimuthu Leelaratne, Sanjiva Weerawarana, Paul Fremantle, "Multi-Tenant SOA Middleware for Cloud Computing" WSO2 Inc. Mountain View, CA, USA.
[22] Juniper Networks, Inc. 1194 North Mathilda Avenue Sunnyvale, CA 94089 USA "Securing Multi-Tenancy and Cloud Computing".
[23] Mohamed A. El-Refaey, Prof. Mohamed Abu Rizkaa, "Cloud Gauge: A Dynamic Cloud and Virtualization Benchmarking Suite", 2010 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises.
[24] Matthias Schmidt, Niels Fallenbeck, Matthew Smith, Bernd Freisleben, "Efficient Distribution of Virtual Machines for Cloud Computing", 2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing
[25] Lingli Fu, Tao Chen, "Building Enterprise Application Based On Cloud Computing" 2010 International Conference on E- Health Networking, Digital Ecosystems and Technologies.
[26] Bo Peng, Bin Cui and Xiaoming Li, "Implementation Issues of a Cloud Computing Platform ", Department of Computer Science and Technology, Peking University.
[27] Ian Foster, Yong Zhao, Ioan Raicu, Shiyong Lu "Cloud Computing and Grid Computing 360-Degree Compared".
[28] Wei-Tek Tsai, Qihong Shao, Jay Elston, "Prioritizing Service Requests on Cloud with Multi-tenancy", IEEE International Conference on E-Business Engineering 2010.

Author Profile

Ankush Dhiman Belong to Yamunanagar Received his Master of Computer Application degree from TIMT, Yamunanagar, Haryana-India in 2012. He is pursuing M.Tech in Computer Science and Engg from MVEC, Jagadhri, Haryana-India.

Email: dhiman.ankush@outlook.com