CLOUD IAAS VISUALISATION CONTROLLER

Project Report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science & Engineering

By

Shubhangi Agrawal(121316)

Divyansh Garg(123216)

Under the supervision of

Mr. Punit Gupta

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Certificate

Candidate's Declaration

I hereby declare that the work presented in this report entitled **"CLOUD IAAS VISUALIZATION CONTROLLER"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from August 2015 to June 2016 under the supervision of **Mr. Punit Gupta** (Assistant Professor (Grade-I), Information Technology). The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Student Name:

Shubhangi Agrawal(121316)

Divyansh Garg(123216)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)
Supervisor Name: Mr. Punit Gupta
Designation: Assistant Professor (Grade-I)
Department name : Information Technology
Dated :

Acknowledgement

I would like to use this opportunity to express my gratitude to everyone who supported me throughout the course of this B.Tech project. I am thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. I am sincerely grateful to them for sharing their truthful and inspiring Objective views on a number of issues related to the project.

I am especially grateful to **Mr. Punit Gupta**, Project Supervisor, for his valuable suggestions, support and constant encouragement during the course of the project. His perpetual energy, motivation, enthusiasm and immense knowledge inspired me to discipline myself in efficiently executing my multiple responsibilities simultaneously.

Date:

Shubhangi Agrawal (121316)

Divyansh Garg (123216)

Table of Contents

List of Figures

List of Tables

ABSTRACT

IaaS (Infrastructure as a Platform) public cloud is one mainstream service mode for public cloud computing. The design aim of one IaaS public cloud is to enlarge the hardware-usage of whole platform, optimize the virtual machine deployment and enhance the accept rate of service demand. In this paper we create one service model for IaaS public cloud, and based on the waiting-line theory to optimize the service model, the queue length and the configuration of scheduling server. And create one demand-vector based scheduling model, to filter the available host machine according to the match of demand and metadata of available re-source. The scheduling model can be bonded with the virtual machine motion to reallocate the resources to guarantee the available rate of the whole platform.

CHAPTER 1

INTRODUCTION

Cloud computing has emerged as one of the leading technology in the field of IT. It is one of the technologies that provide cloud storage to manage the data. Cloud storage act as a repository in which the data is maintained, managed and is made available to the end users. When a user store its photos online instead of on the home computer, or use webmail or a social networking site, the user is using a "cloud computing" service.

Cloud computing refers to the delivery of computing resources over the Internet by virtualizing the computing resources and making them available for the users on pay-as-you-go basis.

Cloud is essentially provided by large distributed data centers. Cloud users are provided with virtual images of the physical machines in the data centers. This virtualization is one of the key concept of cloud computing as it is essential to virtualize the resources over the physical layer. Cloud computing applications like Dropbox, EC2, OpenStack, etc. are gaining popularity day by day for their availability, reliability, scalability and utility model. These applications made distributed computing very easy.

In this work, we have proposed a web portal and a cloud IAAS controller which can be used by the user to interact with the server so that the user can demand for the computing resources over the internet and the server may provide the resources to the user.

1.1 Cloud Computing

A definition for cloud computing can be given as an emerging computer paradigm where data and services reside in massively scalable data centers in the cloud and can be accessed from any connected devices over the internet.
Cloud computing is a model for enabling 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. It is the practice of using a network of remote servers hosted on the internet to store, manage, and process data, rather than a local server or a personal computer. Cloud resources are usually not only shared by multiple users but are also dynamically reallocated per demand.
From a business point of view, cloud computing is a method to address the scalability and availability concerns for large scale applications which involves lesser overhead.

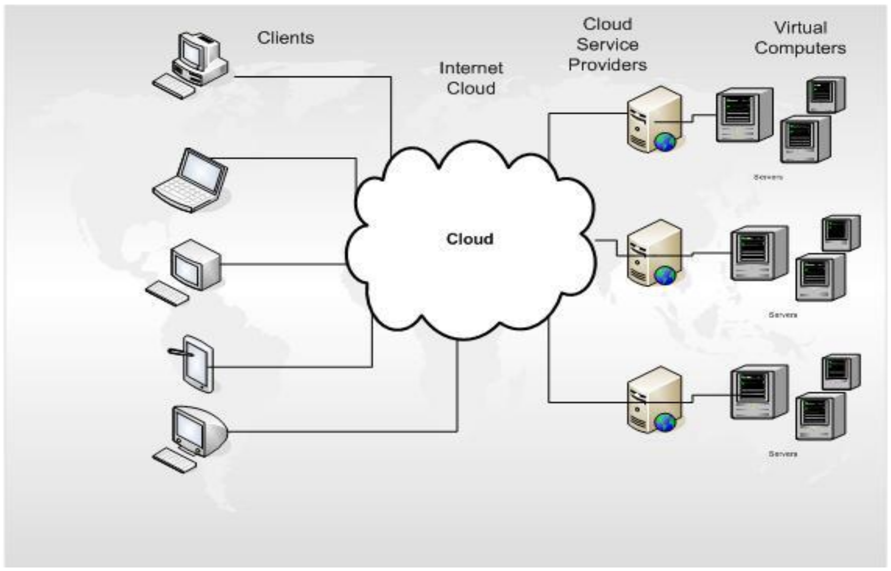


Figure 1: The components of Cloud Computing

1.1.1 Service Models

There are different kind of services provided by different servers in the cloud computing environment. These services may be a software, or a certain environment to run an application or other computing resources. These cloud services are provided over the internet in three different models. These service models are:

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

1.1.1.1 Software as a Service (SaaS)

In SaaS, the user can use different software services that are provided by the cloud service provider over the internet. The service is provided on pay-as-you-go basis and all the upgrades and patching are done by the service provider.

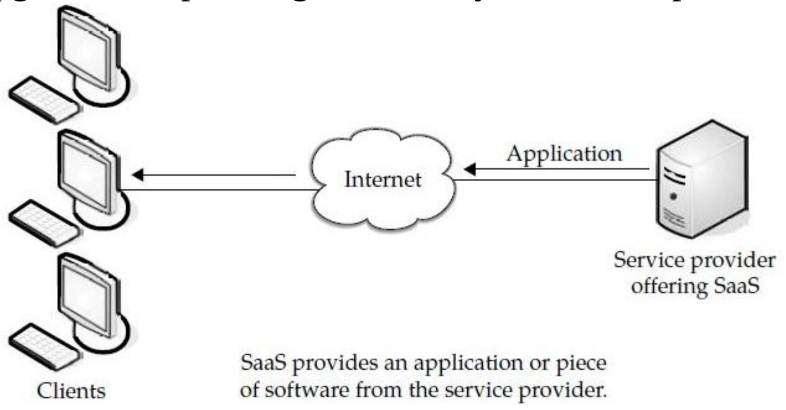

Application
Internet
Service provider offering SaaS
Clients
SaaS provides an application or piece of software from the service provider.

Figure 2: Software as a Service

1.1.1.2 Platform as a Service (PaaS)

In PaaS model, a cloud service provider delivers hardware and software tools that are needed for application development, to its users as a service. A PaaS provider hosts the hardware and software on its own infrastructure thus offering a developing environment to the application developers. Thus, the provider frees the user from having to install hardware and software to develop or run a new application.

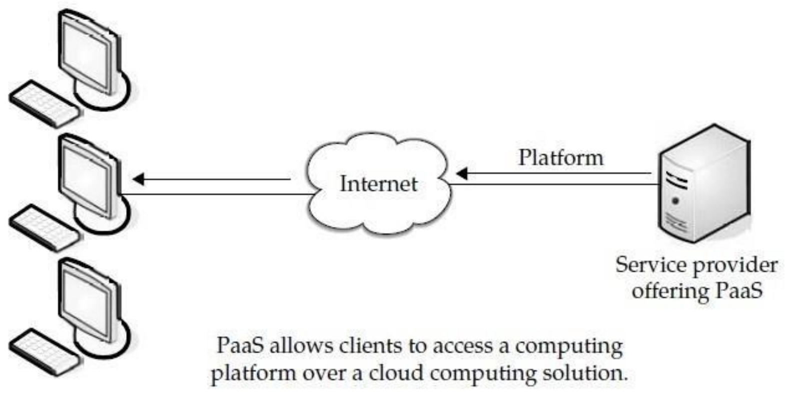
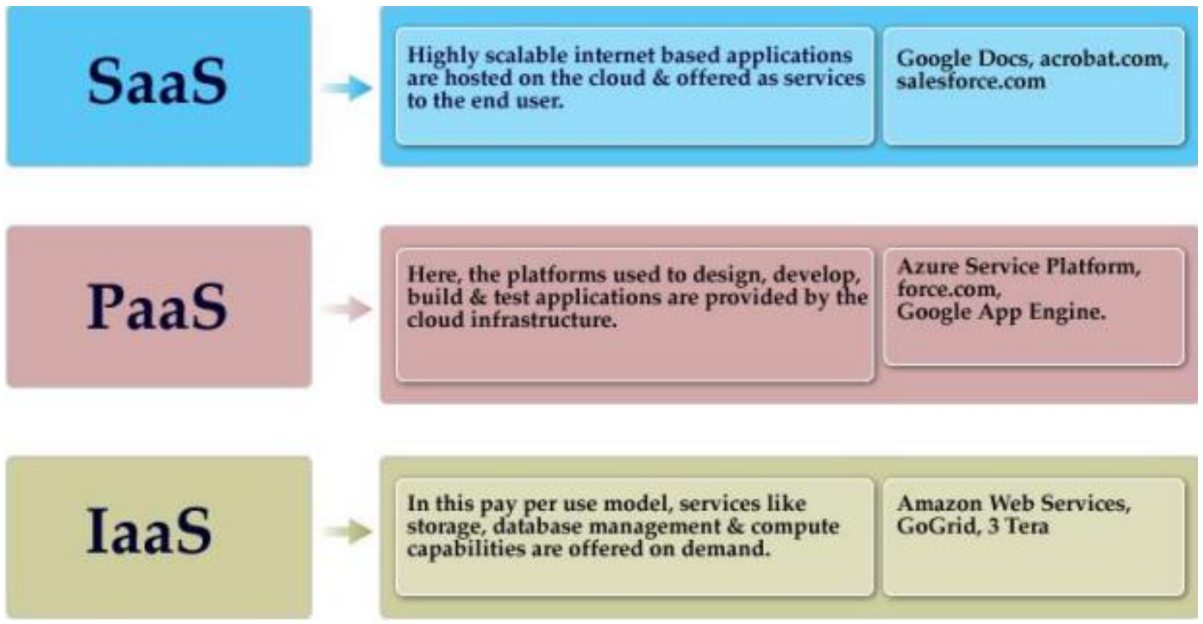

Internet
Platform
Service provider offering PaaS
PaaS allows clients to access a computing platform over a cloud computing solution.

Figure 3: Platform as a Service (PaaS)

1.1.1.3 Infrastructure as a Service (IaaS)

IaaS is also known as Hardware as a Service (HaaS). In this, the infrastructure or actual hardware (RAM, VCPU, and HDD) is provided to customers, via virtualization, who are responsible to install operating systems and necessary software as per their usage. IaaS cloud is usually provided to users in the form of Virtual Machines (VMs), such as Amazon EC2. In an IaaS cloud, users can apply VMs on-demand to deploy and run their applications also provide services to their clients which will be helpful to clients.

- Allows application to be run on cloud supplier’s hardware by allowing user to install virtual server on the IT infrastructure.
- No need to purchase server or network equipment.
- Server, storage and network managed by vendor.
- Usually bill based on usage.


SaaS
Highly scalable internet based applications are hosted on the cloud & offered as services to the end user.
Google Docs, acrobat.com, salesforce.com
PaaS
Here, the platforms used to design, develop, build & test applications are provided by the cloud infrastructure.
Azure Service Platform, force.com, Google App Engine.
IaaS
In this pay per use model, services like storage, database management & compute capabilities are offered on demand.
Amazon Web Services, GoGrid, 3 Tera
Figure 4: Cloud computing services

1.1.2 Deployment Models

Cloud computing models can be categorized into 3 deployment models:

- Private Cloud
- Public Cloud
- Hybrid cloud

1.1.2.1 Private Cloud

Private cloud is cloud infrastructure operated solely for a single organization. It can be managed internally or by a third-party and can be hosted either internally or externally by a third-party.

1.1.2.2 Public Cloud

A cloud is called a "public cloud" when the services are provided over a network that is open for public use. The services offered by a public cloud may be free of cost or offered on a pay-as-you-go model. Cloud service providers like Amazon

AWS, Microsoft and Google own and operate the infrastructure at their data center and access is generally via the Internet.

1.1.2.3 Hybrid Cloud

Hybrid cloud gets its name as it is a combination of two or more clouds (private, community or public) that remain distinct entities but work together, offering the benefits of multiple deployment models.

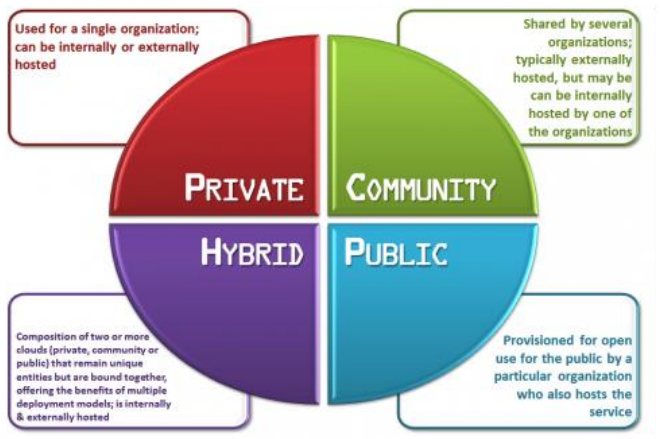

Used for a single organization; can be internally or externally hosted
Shared by several organizations; typically externally hosted, but may be can be internally hosted by one of the organizations
PRIVATE
COMMUNITY
HYBRID
PUBLIC
Composition of two or more clouds (private, community or public) that remain unique entities but are bound together, offering the benefits of multiple deployment models; is internally & externally hosted
Provisioned for open use for the public by a particular organization who also hosts the service

Figure 5: Cloud computing models

1.1.3 Advantages of Cloud

The advantages for using cloud services can be of technical, architectural, business etc.

1. <u>Cloud Providers' point of view</u>
    - Most of the data centers today are underutilized. They are mostly 15% utilized, or sometimes less than that. These data centers need additional capacity to cope with the huge load that sometimes get in the server usage.
    - Large companies having those data centers can easily rent those computing power to other organizations and get profit out of it and also make the resources needed for running data center (like power) utilized properly.

2. Cloud Users' point of view
    - Cloud users need not to take care about the hardware and software they use and also they don't have to be worried about maintenance.
    - Virtualization technology gives the illusion to the users that all the resources are available physically to the user.
    - Cloud users can use the resources on demand basis and pay as much as they use.
    - Scalability is one of the major advantages to cloud users. Users get as much resources as they need and have to pay only for those resources that they use.

1.1.4 Motivation towards cloud in recent time

Cloud computing is not a new idea but it is an evolution of some old paradigm of distributed computing. The advent of the enthusiasm about cloud computing in recent past is due to some recent technology trend and business models.

- High demand of interactive applications
  Applications with real time response and with capability of providing information either by other users or by nonhuman sensors gaining more and more popularity today. These are generally attracted to cloud not only because of high availability but also because these services are generally data intensive and require analyzing data across different sources.

- Extensive desktop application
  Some desktop applications, like MATLAB, are becoming so compute intensive that a single desktop machine is no longer enough to run them. So they are developed to be capable of using cloud computing to perform extensive evaluations.

- Parallel batch processing
  Cloud inherently supports batch-processing and analyzing Tera-bytes of data very efficiently. Programming models like Google's map-reduce and Yahoo!'s open source counterpart Hadoop can be used to do these hiding

operational complexity of parallel processing of hundreds of cloud computing servers.

1.1.5 Characteristics of Cloud Computing

Cloud computing exhibits a number of characteristics which makes it so popular in today's computing world. These characteristics are:

- ***On demand self-service*** is one of the major characteristics of cloud computing that makes it unique. A customer can unilaterally provision computing resources such as network, storage, software, etc. without needing any human interaction with the service provider.

- ***Resource Pooling*** is a term where provider servers multiple clients. Service provider shares its computing resources and cost across a large pool of users allowing for centralization and peak load capacity.

- ***Measured service*** are provided to the user so that the user pays only for those resources used, going back to the affordable nature of cloud.

- ***Cost*** is greatly reduced and capital expenditure is converted into operational expenditure. This lowers barriers to entry an infrastructure is typically provided by the third party and does not need to be purchased for one-time or infrequent extensive computing tasks.

- ***Device and location independence*** enables a user to access system using a web browser regardless of their location or what device they are using, e.g., PC, mobile, tablet, etc. The user can connect from anywhere as infrastructure is off-site and is provided by a third-party.

- ***Reliability*** improves through the use of multiple redundant sites, which makes it suitable for business continuity and disaster recovery.

- ***Scalability:*** The resources are provided by the cloud service provider dynamically, i.e., "on-demand" basis without user having to engineer for peak loads. Performance is monitored and architectures are constructed using web services as the system interface.

- ***Productivity*** may be increased when multiple users can work on the same data simultaneously, rather than waiting for it to be saved and emailed. Time may be saved as information does not need to be re-entered when fields are matched, nor do users need to install application software upgrades to their computer.

- ***Self Healing:*** Any application or any service running in a cloud computing environment has the property of self-healing. In case of failure of the application, there is always a hot backup of the application ready to take over without disruption.

- ***Virtualized***: The applications in cloud computing are fully decoupled from the underlying hardware. The cloud computing environment is a fully virtualized environment.

- ***Rapid elasticity***: If anything, the cloud is flexible and scalable to suit for immediate business needs. You can quickly and easily add or remove users, software features, and other resources.

1.1.6 Cloud Architecture

The cloud providers actually have the physical data centers to provide virtualized services to their users over the Internet. The cloud providers often provide separation between application and data. The underlying physical machines are generally organized and usually geographically distributed. Virtualization plays an important role in the cloud computing scenario. The data center hosts provide the physical hardware on which virtual machines resides. User can use any OS supported by the virtual machines.

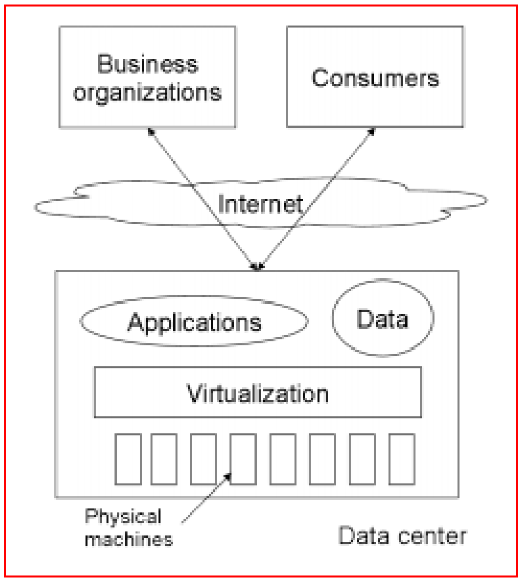


Figure 6: Basic Cloud Computing Architecture

Operating systems are designed for specific hardware and software. It results in the lack of portability of operating system and software from one machine to another machine as different machines uses different instruction set and architecture. The concept of virtual machine solves this problem by acting as an interface between the hardware and the operating system called as system VMs.

Virtualization can be very roughly said to be as software translating the hardware instructions generated by conventional software to the understandable format for the physical hardware. Virtualization also includes the mapping of virtual resources like registers and memory to real hardware resources. The underlying platform in virtualization is generally referred to as host and the software that runs in the VM environment is called as the guest. The *Figure 7* shows very basics of virtualization. Here the virtualization layer covers the physical hardware. Operating System accesses physical hardware through virtualization layer. Applications can issue instruction by using OS interface as well as directly using virtualizing layer interface. This design enables the users to use applications not compatible with the operating system.

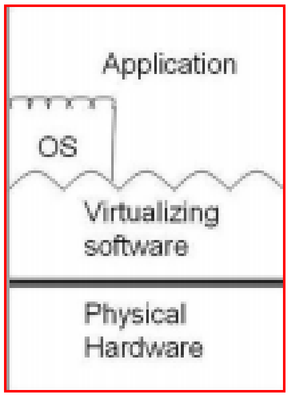


Figure 7: Virtualization Basic

1.2 Objectives

- Designing an efficient cloud IaaS scheduler to efficiently schedule the resources for the clients.
- Designing an efficient cloud IaaS controller to manage the guest virtual machines.
- Designing an online dashboard portal for the client with which the client can interact with the server and request the resources.

1.3 Organization of report

This report is organized into four chapters.

Chapter 1 describes what is cloud computing, various core services provided by cloud computing, cloud service and deployment models, advantages of using cloud computing, characteristics of a cloud, cloud architecture, problem statement and objective of report.

Chapter 2 describes about the previous research work related to the proposed problem statement. In this chapter, we have put research papers that highlight some key concepts of cloud computing.

Chapter 3 describes the proposed work and the algorithm for resource allocation.

Chapter 4 decribes the conclusion of the report.

CHAPTER 2
LITERATURE REVIEWED

2.1 Virtualization

Virtualization refers to the act of creating a virtual (rather than actual) version of something. It is the software implementation of a computer which will execute different programs like a real machine. In cloud computing, the computing resources such as computer hardware platforms, operating systems, storage devices, and computer network are virtualized by the service provider and given to the end user according to the user’s needs.

Virtualization provides flexible environment like:

- virtualization of operating environment
- virtualization of hardware resources
- Virtualization of an OS.

Some of the key benefits of virtualization are:

- Expanding hardware capabilities, allowing each single machine to do more simultaneous work.
- Efforts to control costs and to simplify management through consolidation of servers.
- The need to control large microprocessor and cluster installations, for example in server farms and render farms.
- The ability to run complex, OS-dependent applications in different hardware or OS environments.

2.1.1 Full Virtualization

This technique provides a complete simulation of the underlying hardware including the full instruction set, input/output operations, interrupts, memory access, and whatever other elements are used by the software that runs on the base machine.

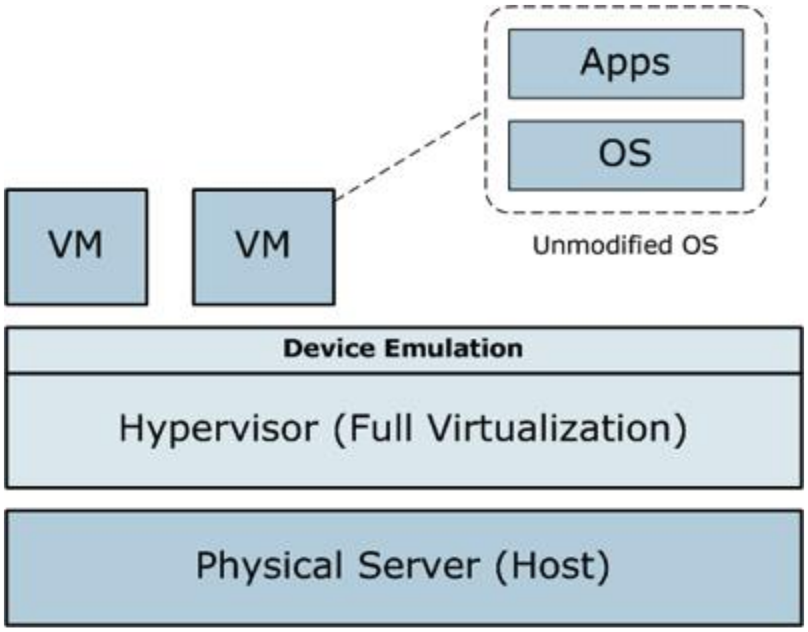


Figure 8: Full Virtualization

Full Virtualization techniques provides:

- Emulation of the hardware on another machine.
- Sharing of computing resources among multiple systems.

2.1.2 Para Virtualization

Para-virtualization technique presents a software interface to virtual machines that is similar, but not identical to that of the underlying hardware. It allows multiple operating systems to run on single machine by using the system resources efficiently. E.g., VMware and VirtualBox software.

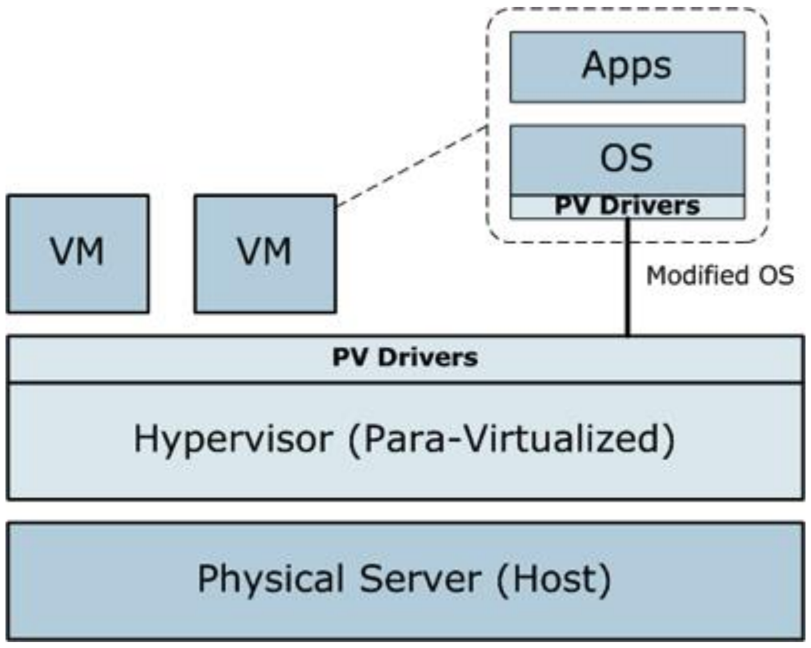


Figure 9: Para Virtualization

Para virtualization has following advantages:

- **Migration:** As the hardware can be replaced easily, migrating or moving the different parts of a new machine is faster and easier.

- **Capacity management:** In a virtualized environment, it is easier and faster to add more hard drive capacity and processing power. As the system parts or hardware can bemoved or replaced or repaired easily, capacity management is simple and easy.

Linux para-virtualization support

At the USENIX conference in 2006 in Boston, Massachusetts, a number of Linux development vendors (including IBM, VMware, Xen, and Red Hat) collaborated on an alternative form of para virtualization that provides a hypervisor-agnostic interface between the hypervisor and guest kernels.

2.2 Hypervisor

A hypervisor is a piece of computer software or firmware that creates and runs virtual machines. It provides the runtime environment to the guest virtual machine. A computer on which a hypervisor is running one or more virtual machines is defined as a *host machine*. Each virtual machine is called a *guest machine*. The hypervisor presents the guest operating systems with a virtual operating platform and manages the execution of the guest operating systems.

The virtual machines (VMs) are unaware of which type of hypervisor is implemented, as they interact only with the hypervisor Itself but not with the physical machine. The VMs get an illusion that the hypervisor on which they are running, is the only physical machine.

A hypervisor is of two types:

- ***Type-I hypervisor (or Hardware hypervisor)***: The type I hypervisors run directly on the host's hardware and control the host’s hardware to manage guest operating systems. For this reason, they are sometimes called bare metal hypervisors.

- ***Type-2 hypervisor*:** These hypervisors are also called software hypervisors as the run on the operating system just as other computer programs do. Type-2 hypervisors abstract guest operating systems from the host operating system. Examples are VMWare Workstation, VirtualBox, etc.

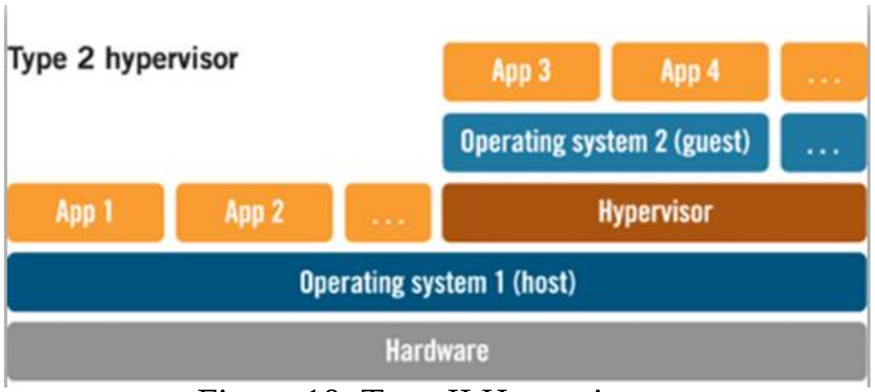


Figure 10: Type II Hypervisor

2.3 KVM

KVM (Kernel Virtual Machine) is a Linux kernel module that allows a user to utilize the hardware virtualization features of various processors. It provides the core virtualization infrastructure to the system. It requires a processor with hardware virtualization extension. KVM enhances the virtualization property of the hardware so that it can be virtualized using QEMU.

By itself, KVM does not perform any emulation. It provides following functionalities:

- Set up the guest VM's address space.
- Feed the guest simulated I/O.
- Map the guest's video display back onto the host.

2.4 QEMU

QEMU(Quick Emulator) is a hypervisor that performs hardware virtualization. It is a machine emulator and virtualizer that enables the system to run a variety of guest operating systems. It is used along with KVM to run virtual machines. It can also be used purely for CPU emulation for user-level processes, allowing applications compiled for one architecture to be run on another.

When used as a machine emulator, it can run Operating Systems and programs made for one machine on a different machine. (e.g., an ARM board) on a different machine (e.g. your own PC).

When used as a virtualizer, it achieves very good performance by executing the guest code directly on the host CPU. QEMU supports virtualization using kernel module in Linux.

2.5 Virt-Manager

The Virtual Machine Manager, also known as virt-manager, is a desktop-driven virtual machine manager with which users can manage virtual machines (VMs).

Virtual Machine Manager allows users to:

- Create, edit, start and stop VMs
- View and control of each VM's console
- See performance and utilization statistics for each VM
- View all running VMs and hosts, and their live performance or resource utilization statistics.

2.6 VNC

Virtual Network Computing (VNC) is a graphical desktop sharing system that is used to remotely control another computer. It transmits the keyboard and mouse events from one computer to another, relaying the graphical screen updates back in the other direction, over a network.

- VNC is platform-independent.
- Multiple clients may connect to a VNC server at the same time.
- To connect to the VNC server, the client need to provide the address of the host and the port number.
- The guest virtual machine is shared with the client using web-browser.

The VNC client (or viewer) is the program that watches, controls, and interacts with the server. The client controls the server.

2.7 LIBVIRT

Libvirt is an open source API, daemon and management tool for managing platform virtualization. It can be used to manage KVM, QEMU and other virtualization technologies. These APIs are widely used in the orchestration layer of hypervisors in the development of a cloud-based solution.

Libvirt is collection of software that provides a convenient way to manage virtual machines and other virtualization functionality, such as storage and network interface management. These software pieces include a long term stable C API, a daemon (libvirtd), and a command line utility (virsh). A primary goal of libvirt is to provide a single way to manage multiple different virtualization providers.

2.7.1 Goals of Libvirt:

Goal is to provide a common and stable layer sufficient to manage domains on a node.

- A **node** is a single physical machine.
- A **hypervisor** is a layer of software allowing to virtualize a node in a set of virtual machines.

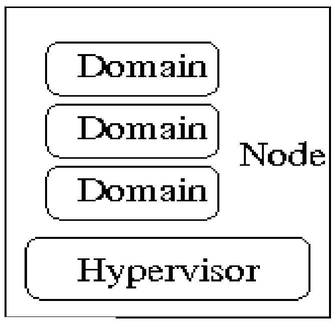


Fig. 11: Goals of Libvirt

2.7.2 Features of libvirt:

Some of the major libvirt features are:

- ***VM management***: Various domain lifecycle operations such as start, stop, pause, save, restore, and migrate.
- ***Remote machine support***: All libvirt functionality is accessible on any machine running the libvirt daemon, including remote machines. A variety of network transports are supported for connecting remotely, e.g., SSH.
- ***Storage management:*** Any host running the libvirt daemon can be used to manage various types of storage: create file images of various file formats, mount NFS shares, enumerate existing LVM volume groups, create new LVM volume groups and logical volumes, partition raw disk devices, mount iSCSI shares, and much more.
- ***Virtual NAT and Route based networking***: Any host running the libvirt daemon can manage and create virtual networks. Libvirt virtual networks use firewall rules to act as a router, providing VMs transparent access to the host machines network.

1.8 Web Portal

A web portal is a web site that provides the functionality to the user so that the user can communicate with the server and get the desire service of the server. The user can directly interact with the server using a web portal and demand for the resources that it need.

A web portal is designed using a combination of technologies that include a scripting language, HTML, CSS, JavaScript, JSP, etc.

HTML

Hyper Text Markup Language (HTML) is the standard markup language used to create web pages. Along with CSS, and JavaScript, HTML is a cornerstone technology, used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.

HTML can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages. Web browsers can also refer to Cascading Style Sheets (CSS) to define the look and layout of text and other material.

Cascading Style Sheets (CSS)

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language.

It is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate CSS file, and reduce complexity and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers.

JSP

JavaServer Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML,XML, or other document types. Released in 1999 by Sun Microsystems, JSP is similar to PHP and ASP, but it uses the Java programming language. JSP pages use several delimiters for scripting functions. The most basic is **<% ... %>**, which encloses a JSP *scriptlet.* A scriptlet is a fragment of Java code that is run when the user requests the page. Other common delimiters include **<%= ... %>** for *expressions,* where the scriptlet and delimiters are replaced with the result of evaluating the expression, and *directives*, denoted with **<%@ ... %>**.

Java code is not required to be complete or self-contained within a single scriptlet block. It can straddle markup content, provided that the page as a whole is syntactically correct. For example, any Java *if/for/while* blocks opened in one scriptlet must be correctly closed in a later scriptlet for the page to successfully compile.

Content which falls inside a split block of Java code (spanning multiple scriptlets) is subject to that code. Content inside an *if* block will only appear in the output when the *if* condition evaluates to true. Likewise, content inside a loop construct may appear multiple times in the output, depending upon how many times the loop body runs.

JAVA

Java is a general-purpose computer programming language. It is class-based and object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (bytecode compiler), GNU Class path (standard libraries), and Iced Tea-Web (browser plugin for applets).

The latest version is Java 8, which is the only version currently supported for free by Oracle, although earlier versions are supported both by Oracle and other companies on a commercial basis.

2.9 Research Related Work

The following sections describe the literature background for the proposed problem statement given in chapter 1. Here, various authors have proposed the research work describing various cloud IaaS tools and techniques and various resource allocation algorithms when more than one client requests for resources.

1.)Title: Role of virtualization in cloud computing
Author: KamyabKhajehei

Abstract: KamyabKhajehei, a scholar from Islamic Azad University, Iran, has discussed about the concept of virtualization technology that let us solve our problem statement.

The author says that virtualization is one of the technologies that given the ability to create the abstraction of computer with ability to perform all the behavior of the actual computer. By using this concept, the cloud computing technology came into picture. By using the virtualization technology, we can virtualize storage, network and hardware resources, and provide a virtualized environment for an application or program to run.

Network virtualization is a major aspect in cloud computing. Systems use the TCP/IP based protocols to communication. Each computer has one specific, unique IP. But in cloud computing environment, there are virtual machines (VM’s) and IP’s are available from pool of network which is obtained by network virtualization.This concept plays a big role in IaaS service of cloud computing where the virtualized infrastructure (hardware, memory, network, storage) is provided on the network.

There are two scenarios for the communication between two VMs:
The first scenario is communication between two VM on a same physical host. In this case, all devices will be virtualized devices like virtual switches or virtual firewalls.

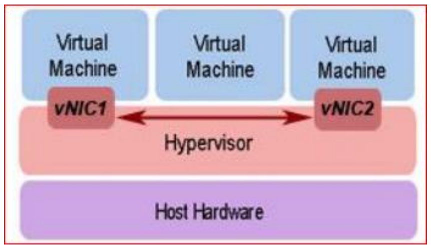


Figure 12: VM communication on same physical host

In the second scenario, two VMs are on different physical hostsin which the communication is a combination of physical communication and virtual communication. The virtual NIC will be linked to the physical NIC of host VM. After that the communication will be between physical NICs and at the last step, the packets will send to another virtual NIC.

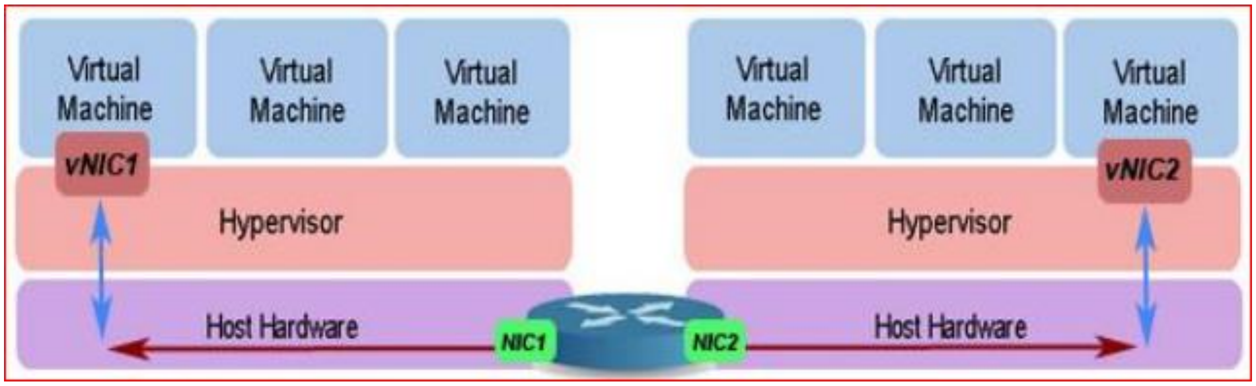


Figure 13: VM communication on two different hosts

2.) <u>Title</u>: A Comparative Study of Current Open-Source infrastructure as a Service Frameworks

<u>Author</u>: Theo Lynn, Graham Hunt, David Corcoran, John Morrison and Philip Healy

Abstract: The authors of the paper have comparatively analyzed different cloud computing products, launched by different vendors worldwide, and the tools these products provide for IaaS.

- <u>OpenStack</u>:
  The IaaS tool provided by OpenStack is NOVA. NOVA is the cloud IaaS controller and manages all the compute resources of the OpenStack cloud. Processes within NOVA include:
  a) ***Nova schedule:*** It takes a VM request and determines where it should be placed.
  b) ***Nova compute:*** It creates and terminates VM instances. It includes support KVM, QEMU, etc.

  Other tools such as Swift (Object storage), Glance (Image service), Cinder, Neutron, Horizon, etc. work with Nova to provide full IaaS platform.

- <u>CloudStack:</u>
  The management server is the CloudStack software that work as an IaaS tool for the cloud. It provides a web interface for administrators and end users, API interfaces, etc. It manages assignment of guest VMs, assignment of IP addresses, allocates storage during VM installation, and manages snapshots, disk images, and ISO images. It is a single point of configuration for the IaaS cloud and informs the user about the resources used.

- <u>OpenNebula</u>:
  The OpenNebula architecture consists of three layers that work together to provide cloud IaaS services to the user. Those layers are:

  a) Tools layer provides interfaces to communicate with users and allows users to manage VMs through the interfaces, including command line interface (CLI) and libvirt API. This layer also contains a scheduler that manages the functionality of the core layer.

b) ***Drivers*** layer contains components that communicate directly with the operating system.

c) ***Core*** layer perform user requests and control resources. In this layer, disk images for VMs are stored using datastores. The monitoring system gathers information on the hosts and the VMs. It contains the authentication system, which comes with a built-in user/password authentication, SSH Authentication and LDAP Authentication.

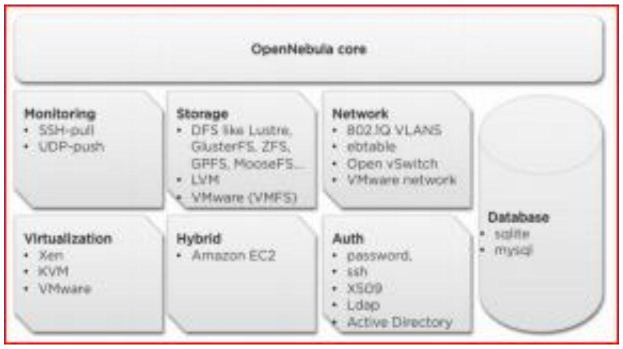


Figure 14: OpenNebula Architecture

- <u>Eucalyptus</u>

  It is a closed-source cloud developed for Infrastructure as a Service. Its ***Cloud Controller (CLC)*** component provides IaaS services. It is responsible for exposing and managing the underlying virtualized resources and offers a web-based administrative interface.

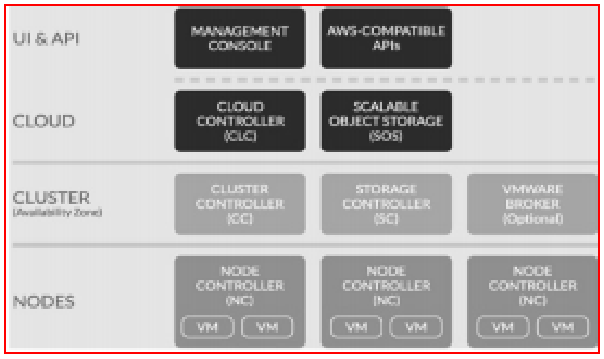


Figure 15: Eucalyptus Architecture

- <u>Nimbus</u>
  It is an open source IaaS project. Its main components are:

  a) The ***Workspace Service*** is a VM manager.
  b) The ***metadata Server*** responds to HTTP queries from VMs.
  c) The ***Cloud Client*** enables users to launch instances quickly.
  d) The ***Workspace-Control*** program can start, stop and pause VMs, implement VM image reconstruction and management, connect the VMs to the network, and deliver contextualization information.

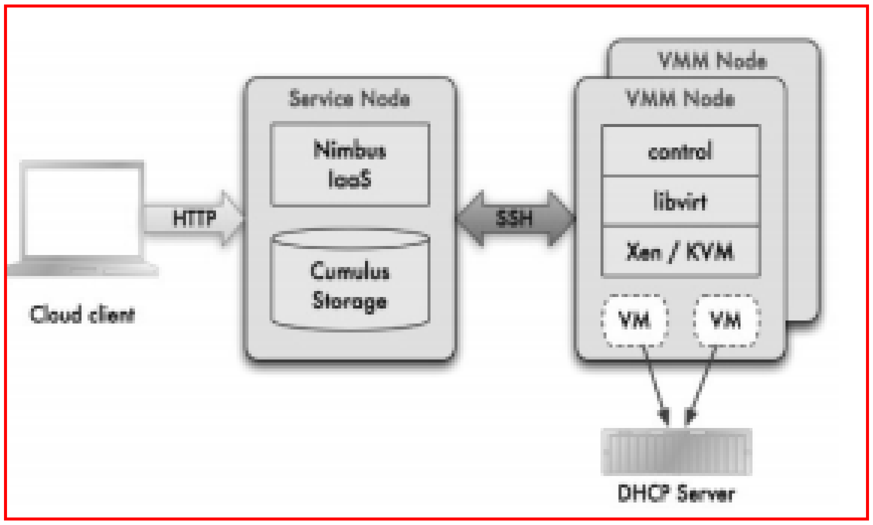
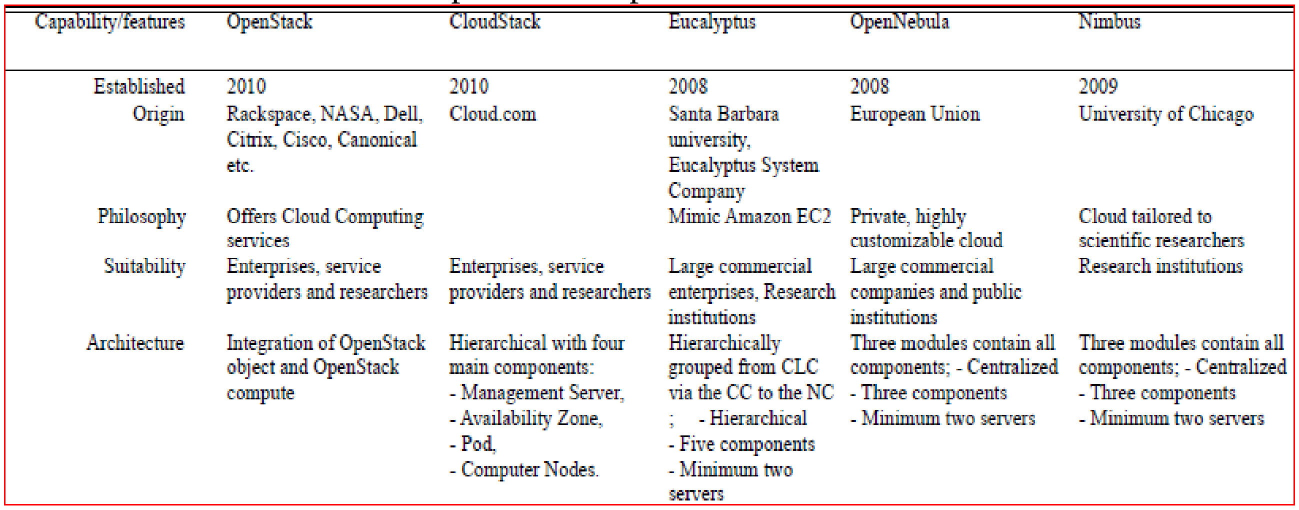


Figure 16: Nimbus Architecture

Table 1: Comparison of open source IaaS solutions



Capability/features	OpenStack	CloudStack	Eucalyptus	OpenNebula	Nimbus
Established	2010	2010	2008	2008	2009
Origin	Rackspace, NASA, Dell, Citrix, Cisco, Canonical etc.	Cloud.com	Santa Barbara university, Eucalyptus System Company	European Union	University of Chicago
Philosophy	Offers Cloud Computing services		Mimic Amazon EC2	Private, highly customizable cloud	Cloud tailored to scientific researchers
Suitability	Enterprises, service providers and researchers	Enterprises, service providers and researchers	Large commercial enterprises, Research institutions	Large commercial companies and public institutions	Research institutions
Architecture	Integration of OpenStack object and OpenStack compute	Hierarchical with four main components: - Management Server, - Availability Zone, - Pod, - Computer Nodes.	Hierarchically grouped from CLC via the CC to the NC ; - Hierarchical - Five components - Minimum two servers	Three modules contain all components; - Centralized - Three components - Minimum two servers	Three modules contain all components; - Centralized - Three components - Minimum two servers

API Support	Native API, Amazon EC2 API, CloudFiles REST API.	Amazon EC2 API, S3	Amazon EC2 API,	Native API in Ruby and JAVA. XML-RPC API for interfaces creation. OGF OCCI & Amazon EC2 APIs.	EC2 APIs, S3 APIs, JAVA client APIs.
Amazon Support	EC2, S3	EC2, S3	EC2, S3, EBS, IAM, AMI	EC2, EBS, AMI	EC2, S3
Cloud Implementation (deployment)	Public Hybrid Private	Public Hybrid Private	Private Hybrid	Private Hybrid	Private Community
Hypervisor	KVM, Xen, VMware ESX, ESXi, Hyper-v, LXC, QEMU, UML, PowerVM, Bare metal	VMware, Oracle VM, KVM, XEN	KVM, Xen, VMware	KVM, Xen, VMware ESX, ESXi	Python, Bash, Ebtables, Libvirt, KVM, Xen
Programming Language	Python	Java	Java, C, Python	Java, Ruby and C++	Java, Python
Community	+++++	++++	+++	+++	++
Release Frequency	<4 months	4 months	>4 months	>6 months	<4 months
Ease of use	+++++	+++++	++	+++	+++
Supported OS	Linux, Windows, Requires x86 Server	Depending on the Hypervisor and hardware - Mac OS X, Asianux, CentOS, Debian, DOS, Fedora, FreeBSD, Novell Netware, Oracle Enterprise Linux, Ubuntu, Red Hat Enterprise Linux, Sun Solaris, SUSE Linex Enterprise, Windows.	Linux (Ubuntu, Fedora, CentOS, OpenSUSE et Debian)	CentOS, Debian, Fedora, RHEL open-SUSE, SLES, and Ubuntu.	Most Linux distributions
Storage	Object and block storage supported. Volumes are persistent (data retained until the volume is deleted, independently from the VM). File storage is supported through Swift (organizing the files in containers).	Supports for iSCSI, NFS, SMB/CIFS; support for OpenStack Swift and Amazon S3	Support for iSCSI, EBS, Amazon S3. Hardware support for industry-standard Storage Hardware.	Hardware support for Fibre Channel, iSCSI, NAS shared storage, SCSI / SAS / SATA. Non-shared and shared file systems (NFS, LVM with CoW, VMFS, etc.).	
Networking	VLAN NO VLAN Public IP's Private IP's SDN IDS Load- balance Firewalls VPN; OpenStack Compute	VLAN, Public IP,	VLAN NO VLAN Public IP's Private IP's; DHCP server on the cluster controller	VLAN NO VLAN Public IP's Private IP's Ebtables OVSwitch; Manual configuration	DHCP server installed on nodes
User Interface	Web interface (i.e. Dashboard) and Command line interface to deploy VMs and a console to manage the VMs.	Web interface and Command Line Interface (CLI)	euca2ools (CLI)	Web interface and Command Line interface (CLI)	Web-Services, specifically: Nimbus Web
Security	API includes protection against DoS attacks or faulty clients. The project concept is introduced by Nova, allowing administrators to manage other user accounts and the project resources. Keystone used for identity management.	CloudStack Secuirty Groups	The Cloud Controller generates a public/private key code pairing for user authentication	Authentication by passwords, secure shell and RSA key code pairings Lightweight Directory Access Protocol ; Authentication framework based on passwords, SSH RSA key-pairs or LDAP. Various administration roles. Multi-tenancy for public clouds.	Public Key Infrastructure
Error Robustness	Replication	Replication	Separate clusters reduce likelihood of correlated errors; Cluster controller's separation	Permanent database to store information about hosts, networks and virtual machines; Database backend (registers virtual machine information)	Regular check and backup of worker nodes; Periodic verification of cloud nodes
Load Balancing	The Cloud Controller	TCP Load Balancer	The Cloud Controller	Nginx	Le Context Broker

3.)Topic: Cloud Storage Architecture and KeyTechnologies

Author: Wenying Zeng, Yuelong Zhao, KairiOu, Wei Song

Abstract:

The authors have presented here the general architecture of cloud storage system. Cloud storage is a storage service mode with which the service providers supply storage capacities and data storage services through the Internet to the clients. The proposed architecture of cloud storage is layered and cooperation storage service system with multiple devices, many application domains, and many service forms.

Architecture of Cloud Storage:

Cloud storage is composed of thousands of storage devices clustered by network, distributed file systems and other storage middleware to provide cloud storage service for users. The typical structure of cloud storage includes storage resource pool, distributed file system, service level agreements (SLA), and service interfaces, etc.

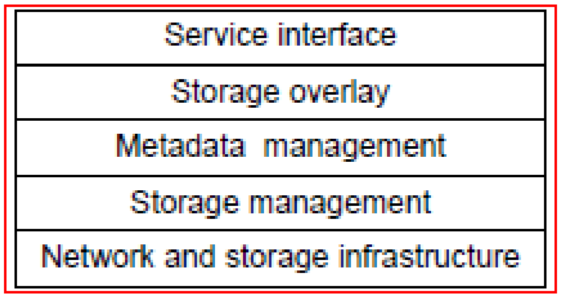


Figure 17: Cloud Storage Architecture

In ***network and storage*** infrastructure, there are distributed wired and wireless networks, storage devices networks.

In ***storage management*** geographical distributed storage resources are organized by domains and logical entities, data can be stored by files or blocks in storage media.

The ***metadata management*** clusters the global domain data storage metadata information and collaborate different domains to load balance.

In ***storage overlay layer*** the virtualization and service retrieving and redirection can be fulfilled. It may be thought as middleware which links storage devices distributed to a virtual storage networks and expose simplified and standard data structures to service interfaces.

In ***service interface layer*** the cloud storage system provides clients uniform interface to access, and filter the illegal clients out of the system. Service delivering mode is a key aspect in cloud storage. Storage resources can be thought as commerce products, and there are much commerce theories and experiences can be lead to cloud storage services.

Ant Colony Optimization

When the storage clients request for the cloud storage services, which storage providers or storage servers will be selected will be based on ant colony optimization.

The clients may select a cloud storage resources initially at random, but when next request is sent, the past experiences referred and it updates the related values by the current storage service, just as the pheromone in ant optimization algorithms.

Game Theory in cloud storage

Game theory in cloud storage is applied in the construction and cooperation phases. Let us assume that there are multiple cloud storage service providers. They can get more benefits if they cooperate and make cloud storage federation in providing storage services to clients. In storage federation, who provide storage resources to clients can be seen as game theory problem.
The service provider would then provide storage resources to the nearest geographic locations to reduce transports delay and communication cost.

4.) Title : A SURVEY OF VARIOUS SCHEDULING ALGORITHM IN CLOUD COMPUTING ENVIRONMENT
Author: Pinal Salot

Abstract:
The author Pinal Salot has discussed some of the important scheduling algorithms that play a major role in scheduling the resources for the clients in cloud computing environment. The author says that scheduling of resources is one of the major part of cloud IaaS as all the client that connect to the server requests for different resources at different time. The server must be efficient enough to efficiently schedule the resources for the client.

In this research work, the author has discussed about some of the resource scheduling algorithms used in cloud computing environment.

- ***First Come First Serve Algorithm***: Job in the queue which come first is served. This algorithm is simple and fast.

- ***Min–Min algorithm***: This algorithm chooses small tasks to be executed firstly, which in turn large task delays for long time.

- ***Max – Min algorithm***: This algorithm chooses large tasks to be executed firstly, which in turn small task delays for long time.

- ***Most fit task scheduling algorithm***: In this algorithm task which fit best in queue are executed first. This algorithm has high failure ratio.

- ***Priority scheduling algorithm***: Each process is assigned a priority, and priority is allowed to run. Equal-Priority processes are scheduled in FCFS order.
  The shortest-Job-First (SJF) algorithm is a special case of general priority scheduling algorithm where the priority is the inverse of the next CPU burst. That is, the longer the CPU burst, the lower the priority and vice versa.

- ***Round Robin algorithm***: In the round robin scheduling, processes are dispatched in a FIFO manner but are given a limited amount of CPU time called a time-slice or a quantum. If a process does not complete before its CPU-time expires, the CPU is preempted and given to the next process waiting in a queue. The preempted process is then placed at the back of the ready list.

Here, some of the existing scheduling algorithms are also discussed.

- **Resource-Aware-Scheduling algorithm (RASA)**: Saeed Parsa and Reza Entezari-Maleki proposed a new task scheduling algorithm RASA. It is composed of two traditional scheduling algorithms; Max-min and Min-min. RASA uses the advantages of Max-min and Min-min algorithms and covers their disadvantages.

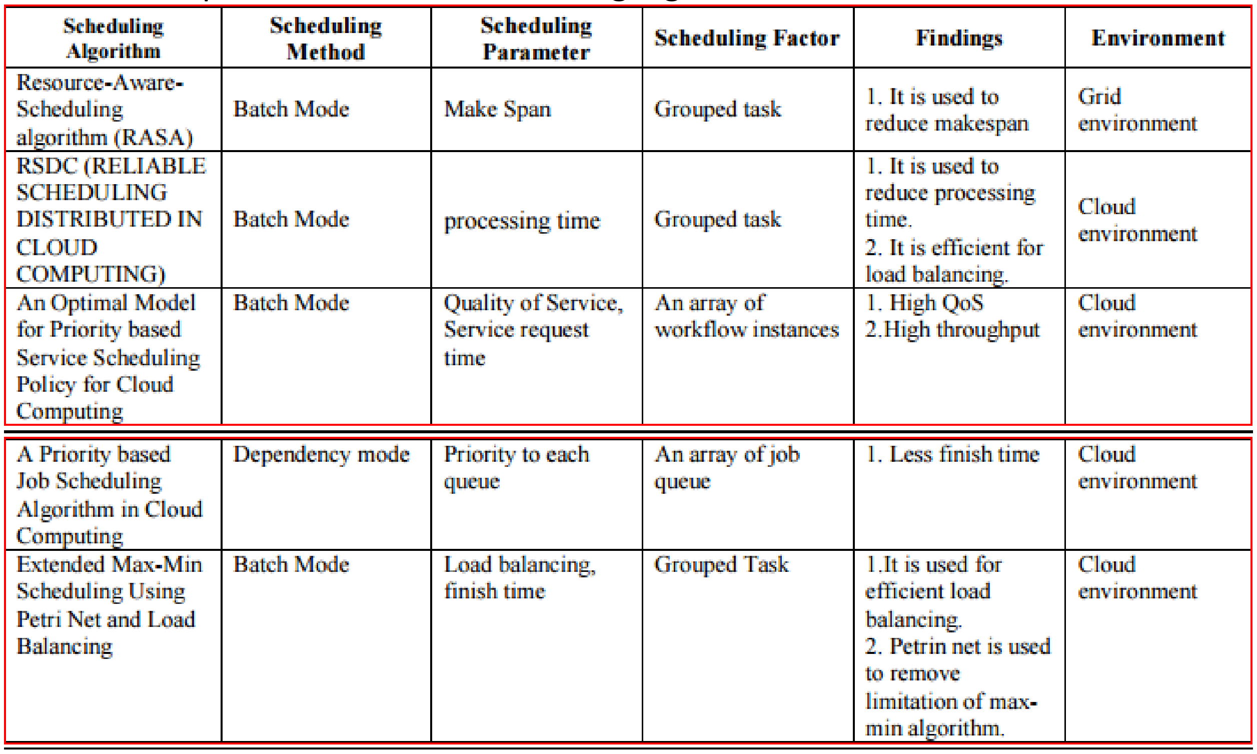
- **Reliable Scheduling Distributed in Cloud Computing (RSDC):** Arash Ghorbannia Delavar, Mahdi Javanmard, Mehrdad Barzegar Shabestari and Marjan Khosravi Talebi proposed a reliable scheduling algorithm in cloud computing environment. In this algorithm major job is divided to sub job. In order to balance the jobs the request and acknowledge time are calculated separately. The scheduling of each job is done by calculating the request and acknowledges time in the form of a shared job. So that efficiency of the system is increased.

- **An Optimal Model for Priority based Service Scheduling Policy for Cloud Computing Environment:** Dr. M. Dakshayini, Dr. H. S. Guruprasad proposed a scheduling algorithm based on priority and admission control scheme. In this algorithm priority is assigned to each admitted queue. Admission of each queue is decided by calculating tolerable delay and service cost. Advantage of this algorithm is that this policy with the proposed cloud architecture has achieved very high service completion rate with guaranteed QoS.

- **A Priority based Job Scheduling Algorithm in Cloud Computing**: Shamsollah Ghanbari, Mohamed Othman proposed this scheduling algorithm based on multi–criteria and multi-decision priority driven scheduling algorithms. This scheduling algorithm consist of three level of scheduling: object level, attribute level and alternate level where the priority can be set by job resource ratio.

- **Extended Max-Min Scheduling Using Petri Net and Load Balancing**: El-Sayed T. El-kenawy, Ali Ibraheem El-Desoky, Mohamed F. Al-rahamawy has proposed this algorithm based on impact of RASA algorithm. Improved Max-min algorithm is based on the expected execution time instead of complete time as a selection basis.

Table 2: Comparison between scheduling algorithms



Scheduling Algorithm	Scheduling Method	Scheduling Parameter	Scheduling Factor	Findings	Environment
Resource-Aware-Scheduling algorithm (RASA)	Batch Mode	Make Span	Grouped task	1. It is used to reduce makespan	Grid environment
RSDC (RELIABLE SCHEDULING DISTRIBUTED IN CLOUD COMPUTING)	Batch Mode	processing time	Grouped task	1. It is used to reduce processing time. 2. It is efficient for load balancing.	Cloud environment
An Optimal Model for Priority based Service Scheduling Policy for Cloud Computing	Batch Mode	Quality of Service, Service request time	An array of workflow instances	1. High QoS 2.High throughput	Cloud environment
A Priority based Job Scheduling Algorithm in Cloud Computing	Dependency mode	Priority to each queue	An array of job queue	1. Less finish time	Cloud environment
Extended Max-Min Scheduling Using Petri Net and Load Balancing	Batch Mode	Load balancing, finish time	Grouped Task	1.It is used for efficient load balancing. 2. Petrin net is used to remove limitation of max-min algorithm.	Cloud environment

Chapter 3

Proposed Work and Results

3.1 Proposed Work

We have proposed to create a user interface for providing Virtual Machine to the client through internet. Cloud Computing is gaining increasing attention within enterprises of all shapes and size. Taking standard software applications, databases, and user interfaces and deploying them in a Cloud environment. Proper scoping, usage modeling, and careful design are all essential to success in the Cloud.

We have proposed to implement scheduling algorithm in order to allocate the resources to different users of cloud. The algorithm to be used is Shortest Job First (SJF).

Shortest Job First

Shortest Job First (SJF) scheduling is a priority and Non-Preemptive scheduling. It is a dynamic load balancing algorithm which handles the process with priority basis.

This algorithm distributes the load randomly by first checking the size of the process and then transferring the load to a Virtual Machine, which is lightly loaded.

Characteristics:

High efficiency, low turnaround time and minimal average waiting time.

3.2 SJF Algorithm

1. Firstly start process, and maintain the process by priority checking the size of the process and distribute the load to the virtual machine which is lightly loaded.

2. Take the input from user.

3. VM load balancer select process which load has shortest burst time among all loads will execute first.

4. If in the process any load have same burst time length then FCFS (First come First Served) scheduling algorithm used.

5. Make average waiting time length of next process.

6. Start with first process, selection as above as shortest load come first which has minimal average time and other processes are to be in queue.

7. Calculates Burst total number of time.

8. Display the Related values.

9. Now close / Stop process.

3.3 Code

3.3.1 Creating the VM

This function creates a VM which has predefined values given for creating VM.

```
public long create(Connect conn)
{  try
            {

longstartTime = System.currentTimeMillis();
{
                String dumpxml = "<domain type='qemu'>"+
                        "<name>tt2</name>"+
                        "<memory>1048576</memory>"+
                        "<currentMemory>1048576</currentMemory>"+
                        "<vcpu>1</vcpu>"+
                        "<os>"+
                       "  <type arch='i686' machine='pc-0.11'>hvm</type>"+
                       "  <boot dev='cdrom'/>"+
                        "</os>"+
                        "<features>"+
                       "  <acpi/>"+
                       "  <apic/>"+
                       " <pae/>"+
                        "</features>"+
                        "<clock offset='utc'/>"+
                        "<on_poweroff>destroy</on_poweroff>"+
                        "<on_reboot>restart</on_reboot>"+
                        "<on_crash>restart</on_crash>"+
                        "<devices>"+
                       "  <emulator>/usr/bin/qemu</emulator>"+
                       "  <disk type='file' device='disk'>"+
                        "    <driver name='qemu' cache='none'/>"+
                        "    <source file='/var/lib/libvirt/images/tt2.img'/>"+
                        "    <target dev='hda' bus='ide'/>"+
                       "  </disk>"+
"  <disk type='file' device='cdrom'> "+
"  <driver name='qemu' type='raw'/>" +
                        "<source file='/root/Desktop/CentOS-6.5-x86_64-minimal.iso'/>"+
                        "<target dev='hdc' bus='ide'/>"+
                        "<readonly/>"+
                        "<address type='drive' controller='0' bus='1' unit='0'/>"+
                        "   </disk>"+

                       "  <interface type='network'>"+
                        "    <mac address='54:52:00:02:02:2c'/>"+
                        "    <source network='default'/>"+
                       "  </interface>"+
                       "  <serial type='pty'>"+
                        "    <target port='0'/>"+
                       "  </serial>"+
                       "  <console type='pty'>"+
                          "    <target port='0'/>"+
                        "  </console>"+
                        "  <input type='mouse' bus='ps2'/>"+
                        "  <graphics type='vnc' port='-1' autoport='yes' keymap='en-
us'/>"+
                          "</devices>"+
                        "</domain>";
}
```

//The set of lines in the bracket gives all the required functionalities needed to create a VM.

```
            {
               Domain dm = null;
                        try
                  {
                                dm =conn.domainLookupByName("tt2");

                        }
                        catch (LibvirtException e) {
                                // TODO: handle exception
                        }
if(dm!=null)
                  {System.out.println("domain already exist");}
else
                  {
                     String volume="<volume>"
                         + "<name>tt2.img</name>  "
                         + "   <allocation>10000</allocation>"
+ "<capacity unit=\"M\">10001</capacity>"
                         + "       <target>"
                         + "<path>/var/lib/virt/images/tt2.img</path>"
                         + "<permissions>"
                         + "<owner>107</owner>"
                         + "<group>107</group>"
                         + "           <mode>0744</mode>   "
                         + "        <label>virt_image_t</label>    "
                         + ""
                         + "     </permissions>"
                         + "       </target>    "
```

```
                              + "  </volume>";
}
```

//dm is the object that is been created and if dm is not null then domain name tt2 already exits else new domain is created with the given requirements

```
d.create(conn)
                          }



                      }
longstopTime = System.currentTimeMillis();
longelapsedTime = stopTime - startTime;

System.out.println(elapsedTime);
delete(conn);
returnelapsedTime;
                  }
catch (LibvirtException e)
               {
                            System.out.println("exception caught:" + e);
                            System.out.println(e.getError());
return 0;
                }

    }
```

3.3.2 Stopping VM

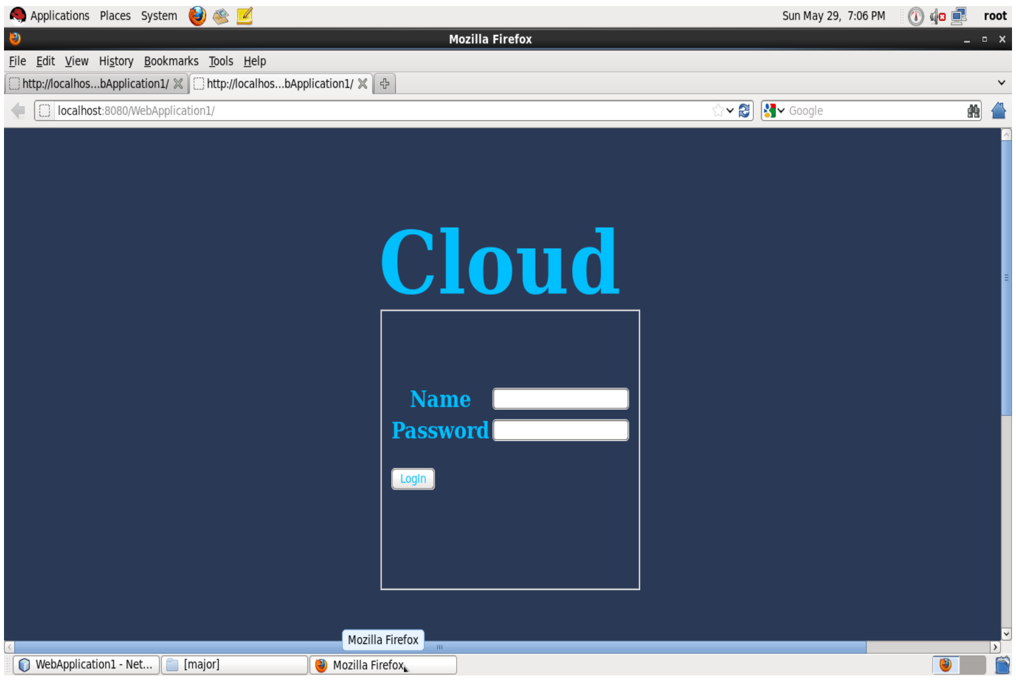
```
public void delete(Connect conn)
   {

try
   {
```

```
   Domain d=conn.domainLookupByName("tt2");
//domain tt2 is looked up and when it is found and is active, it is deleted by the
function destroy()
if(d.isActive()==1)
        {d.destroy();}
StoragePoolsp=conn.storagePoolLookupByName("default");
StorageVolsv=sp.storageVolLookupByName("tt2.img");
sv.delete(1);
d.undefine();
     }
catch (LibvirtException e)
            {
        }
   }
```

3.4 Results

3.4.1 Page1



Applications Places System
Sun May 29, 7:06 PM
root
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhos...bApplication1/
http://localhos...bApplication1/
localhost:8080/WebApplication1/
Google
Cloud
Name
Password
Login
Mozilla Firefox
WebApplication1 - Net...
[major]
Mozilla Firefox

Figure 18: Login Form (index.html)

3.4.2 Page 2

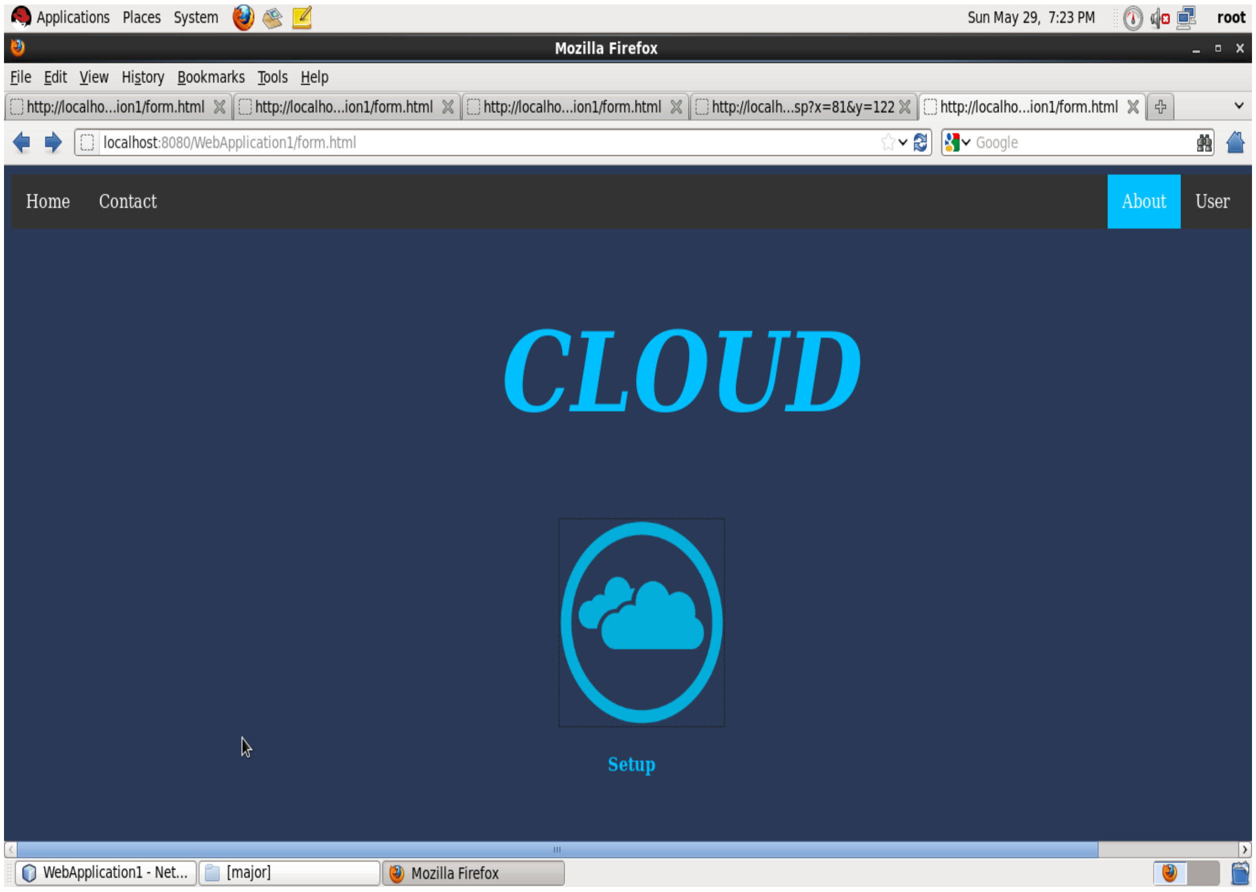

Applications Places System
Sun May 29, 7:23 PM
root
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localho...ion1/form.html
http://localho...ion1/form.html
http://localho...ion1/form.html
http://localh...sp?x=81&y=122
http://localho...ion1/form.html
localhost:8080/WebApplication1/form.html
Google
Home
Contact
About
User
CLOUD
Setup
WebApplication1 - Net...
[major]
Mozilla Firefox

Figure 19: Home Page (form.jsp)

3.4.3 Page 3

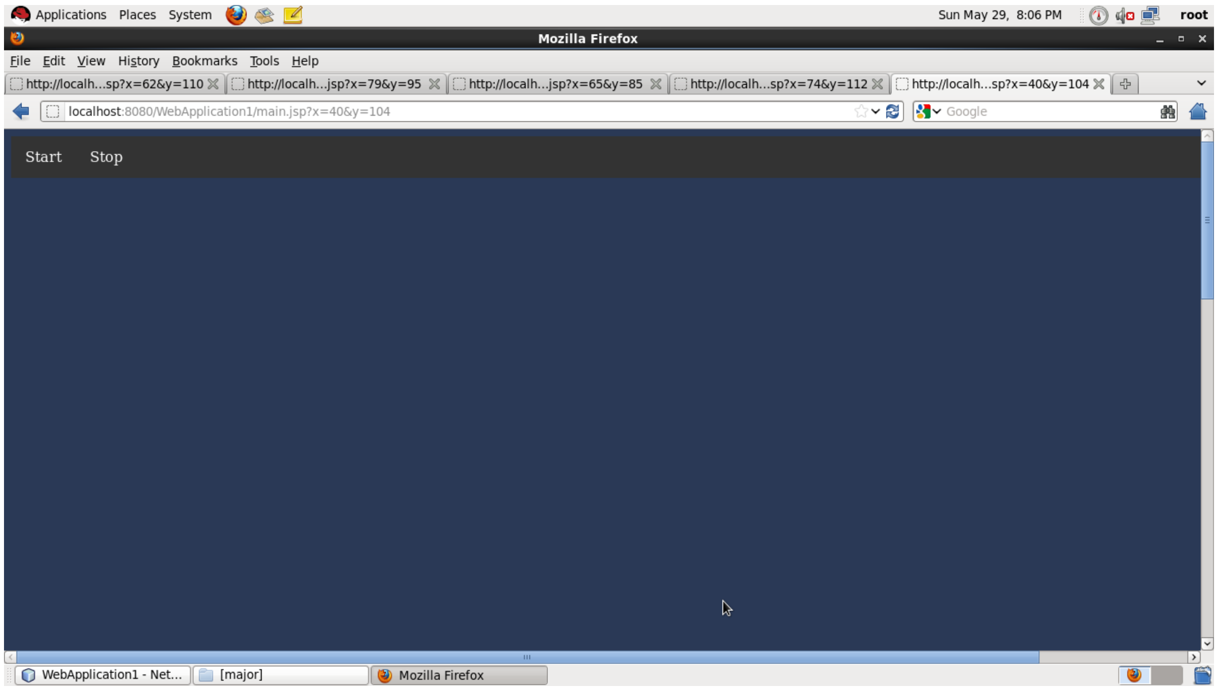

Applications Places System
Sun May 29, 8:06 PM
root
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localh...sp?x=62&y=110
http://localh...jsp?x=79&y=95
http://localh...jsp?x=65&y=85
http://localh...sp?x=74&y=112
http://localh...sp?x=40&y=104
localhost:8080/WebApplication1/main.jsp?x=40&y=104
Google
Start
Stop
WebApplication1 - Net...
[major]
Mozilla Firefox

Figure 20: User page (machne.jsp)

Chapter 4

Conclusion and Future Work

4.1 Conclusion

In this report, thorough study of different cloud computing techniques have been done. Different technologies that are used to create cloud IaaS controller have been summarized. By using these tools and techniques, we are able to successfully design a cloud IaaS visualization controller which would help the client to access the services of the server by using a web portal.

4.2 Future Work

In this report, proposed work presents existing techniques of handling load of servers in cloud environment. In future other parameters can be modified to achieve greater utilization and less power consumption.

References

1. AlexandruIosup, Member, IEEE, Simon Ostermann,NezihYigitbasi, Member, IEEE, RaduProdan, Member, IEEE, Thomas Fahringer, Member, IEEE, and Dick Epema, Member, IEEE, November 2010

2. Qi Zhang · Lu Cheng · RaoufBoutaba, "Cloud computing: state-of-the-art and research challenges", The Brazilian Computer Society 2010, 20 April 2010

3. Cloud spectator , "Comparing Linux Compute Performance of Amazon EC2, Microsoft Azure, and SherWeb", January 2015

4. Thilagavathi M* School of Information Technology and Engineering, VIT University, India,Cloud Platforms – A Comparison,Volume 3, Issue 11, November 2013

5. Peter Mell and Tim Grance, The NIST Definition of Cloud Computing, Version 15, July 2009.

6. Rajveer Kaur, SupriyaKinger,Department of Computer Science and Engineering, SGGSWU,Fatehgarh Sahib, India

7. Punjab,Analysis of Job Scheduling Algorithms in Cloud Computing,volume 9 number 7 – Mar 2014, ISSN: 2231

8. KamyabKhajehei, "*Role of virtualization in cloud computing",* International Journal of Advance Research in Computer Science and Management Studies, *ISSN: 2321-7782,* volume 2, issue 4, April 2014.

9. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "*Above the Clouds: A Berkeley View of Cloud Computing*", EECS Department, University of

California,Berkeley, Technical Report No., UCB/EECS-2009-28, pages 1-23, February 2009.

10. Wenying Zeng, Yuelong Zhao, KairiOu, Wei Song-*"Research on Cloud Storage Architecture and Key Technologies",* South China University of Technology, Guangzhou.

11. Graham Hunt, John P. Morisson, Philip Healy, Theodore Gerard Lynn, *" A Comparative study of current open-source Infrastructure as a Service", May 2015,* DOI: 10.5220/0005423300950104

12. Pinal Salot, *"A SURVEY OF VARIOUS SCHEDULING ALGORITHM IN CLOUD COMPUTING ENVIRONMENT"*, M.E, Computer Engineering, Alpha College of Engineering, Gujarat, India, ISSN: 2319 - 1163