Cloud Computing: Concepts, Virtualization, IaaS and PaaS

Sanjay Chaudhary, SEAS - AU
Gaurav Somani,
Central University, Rajasthan

Virtualization - Introduction

Virtualization is the technique to create an efficient and isolated duplicate of a real machine [1].

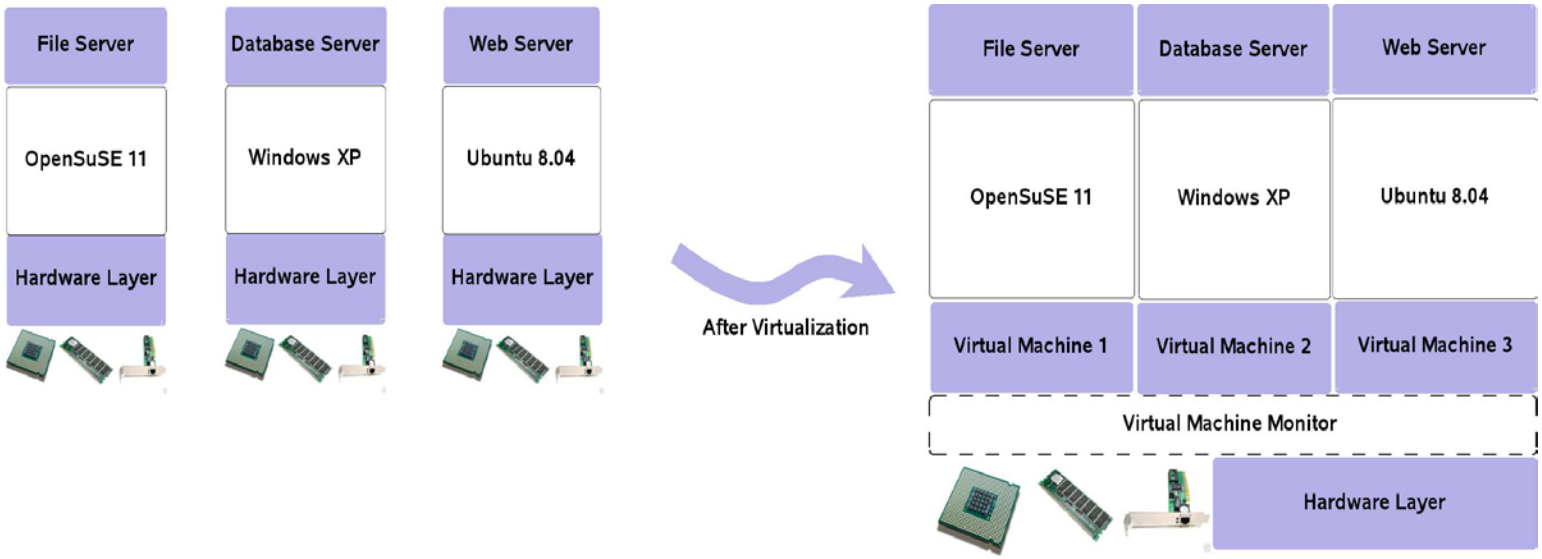


Figure 1. Implementing Virtualization

Why to virtualize?

1. Flexibility
2. Availability
3. Fault Isolation
4. Scalability
5. Hardware Utilization
6. Security
7. Cloning

Why not to virtualize?

1. Virtualization Overhead
2. Single point of failure
3. Real time applications

Virtual Machines

- Virtualize whole system environment :
  - OS, Applications
- Virtual Machine Monitor (VMM) OR Hypervisor:
  - Software layer that provides virtualization support
  - Manages multiple virtual machines
- Virtual Machine:
  - Replica of a machine environment
- Guest OS:
  - OS running inside a VM

Formal Model of Virtual Machines [1]

❑ **Popek, Goldberg 1974**

❑ **VMM characteristics**

VMM should have 3 characteristics:

1. Equivalence: A program running in a VM should behave identically to running in native mode except for performance)
2. Efficiency: Programs running in a VM should show only minor decreases in speed
3. Resource control: VMM should have complete control of system resources

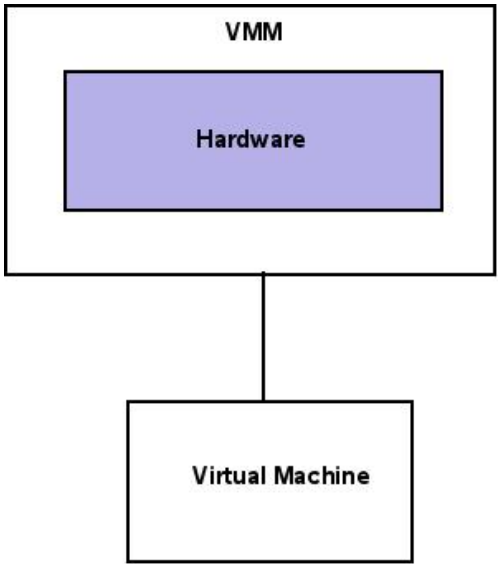


Figure 3 : Formal Model of Virtual Machine Monitor [1]

Formal Model of Virtual Machines [1] cont.

- ***"An architecture can be virtualized if sensitive instructions are a subset of privileged instructions"***

  Intuition: VMM can capture all sensitive instructions.

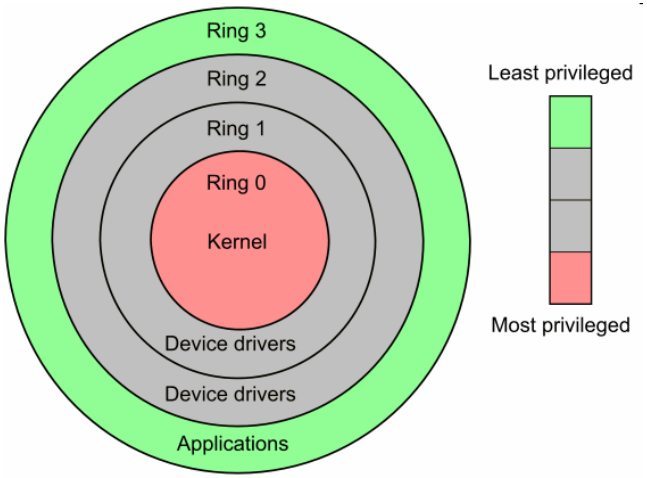
- Types of instructions:
  - **Sensitive:** Change system state (e.g.: resource allocation, protected data, etc.)
  - **Innocuous:** Regular instructions.
  - **Privileged** : Can be executed only in kernel mode (Trap in user mode).

- Many architectures (e.g.: Intel x86) do not satisfy this theorem, Total17 instructions do not satisfy this, E.g.: POPF instruction for setting interrupt flag.
  - SGDT, SIDT, PUSHF, POPF.....

- **Efficiency:** All innocuous instructions should be executed natively.

x86 Privilege Rings

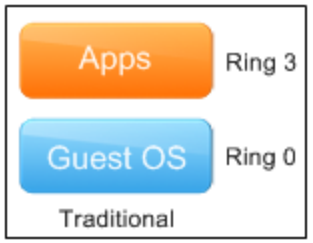
- x86 CPUs provide a range of protection levels also known as rings in which code can execute. Ring 0 has the highest level privilege and is where the operating system kernel normally runs. Code executing in Ring 0 is said to be running in system space, kernel mode or supervisor mode. All other code such as applications running on the operating system operate in less privileged rings, typically Ring 3.



Rings in virtualization

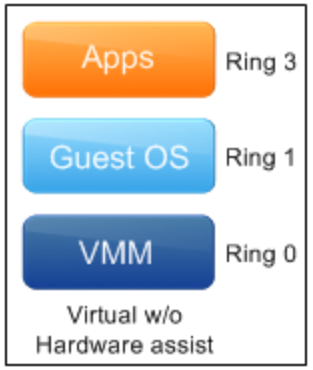
Traditional systems

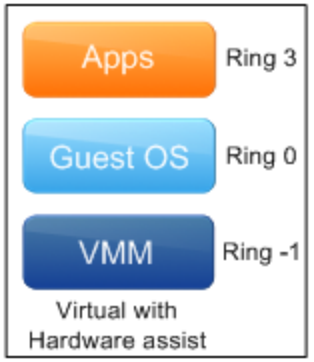
- Operating system runs in privileged mode in Ring 0 and owns the hardware
- Applications run in Ring 3 with less privileges runs in privileged mode in Ring 0



Virtualized systems

- VMM Guest OS inside VMs are fooled into thinking they are running in Ring 0, privileged instructions are trapped and emulated by the VMM
- Newer CPUs (AMD-V/Intel-VT) use a new privilege level called Ring -1 for the VMM to reside allowing for better performance as the VMM no longer needs to fool the Guest OS that it is running in Ring 0.

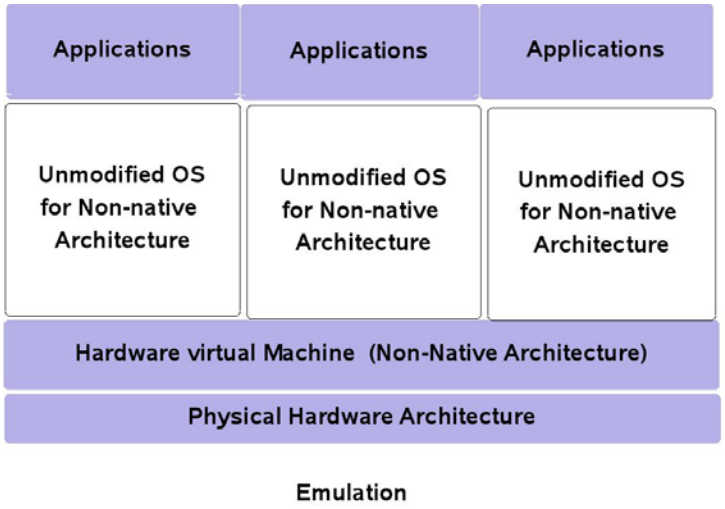


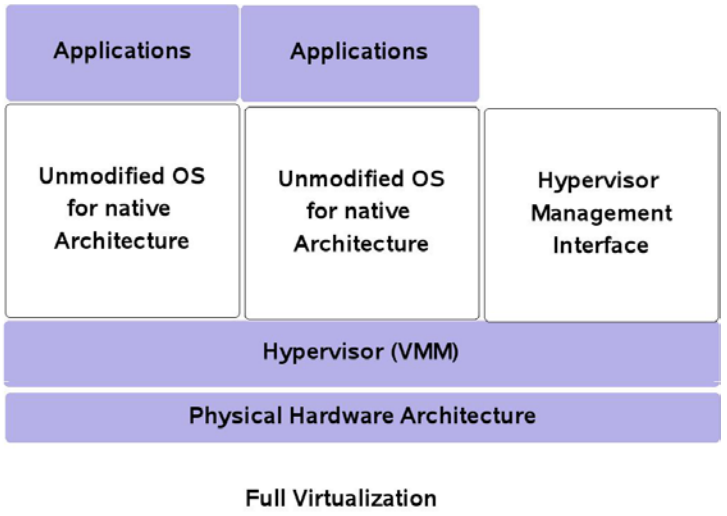


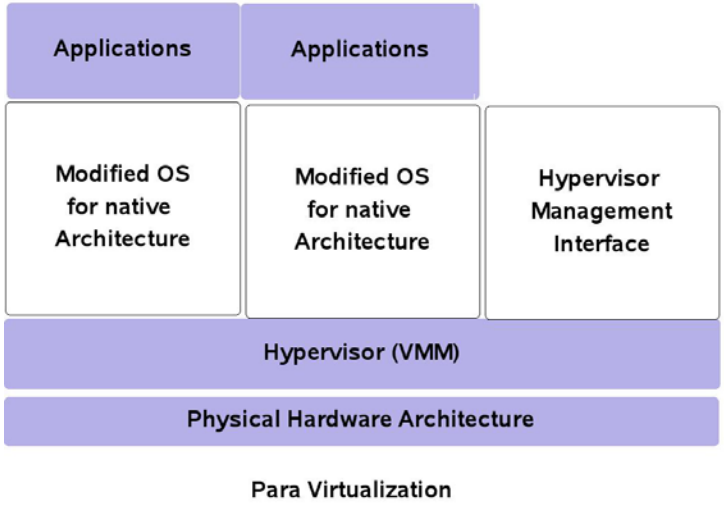
X86 Processor Virtualization

- x86 architecture is not fully virtualizable
  - Certain privileged instructions behave differently when run in unprivileged mode
  - Certain unprivileged instructions can access privileged state
- Instructions do not satisfy this, E.g.: POPF instruction for setting interrupt flag.
  - SGDT, SIDT, PUSHF, POPF.....
- Techniques to address inability to virtualize x86
  - Replace non-virtualizable instructions with easily virtualized ones statically (Paravirtualization)
  - Perform Binary Translation (Full Virtualization)

Kinds of Virtualization







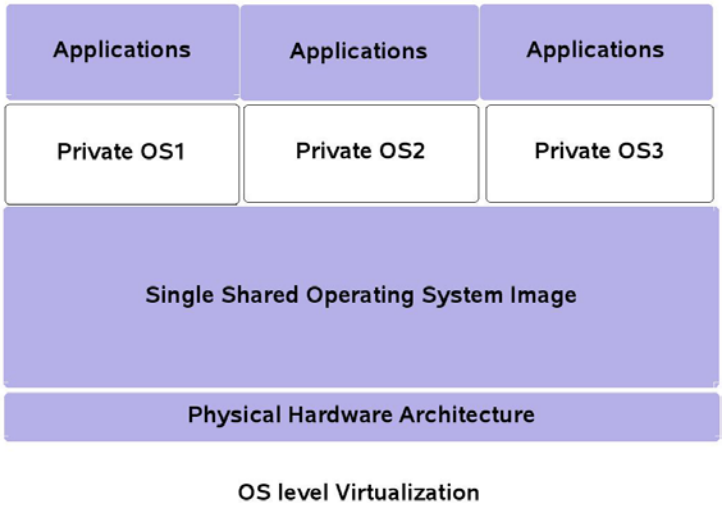
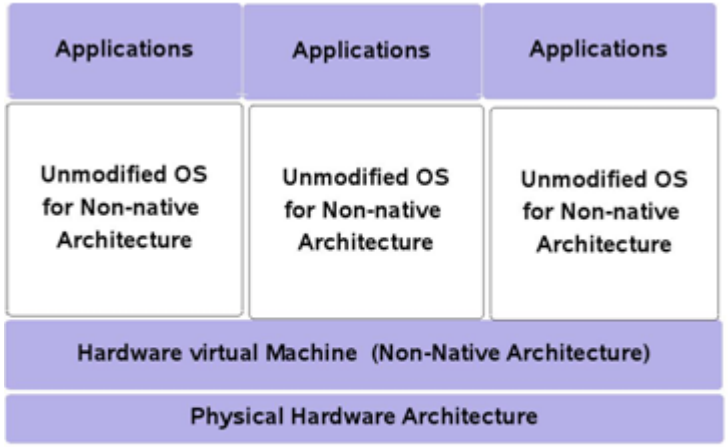


Figure 4: Kinds of Virtualization [2]

Emulation

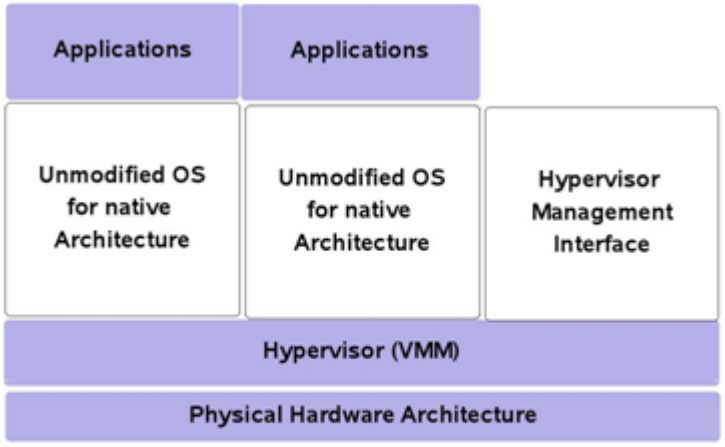
- An **emulator** duplicates the functions of one system using a different system, so that the second system behaves like the first system
- Processor emulator example – QEMU
- Works on dynamic binary translation.
- http://www.qemu.org



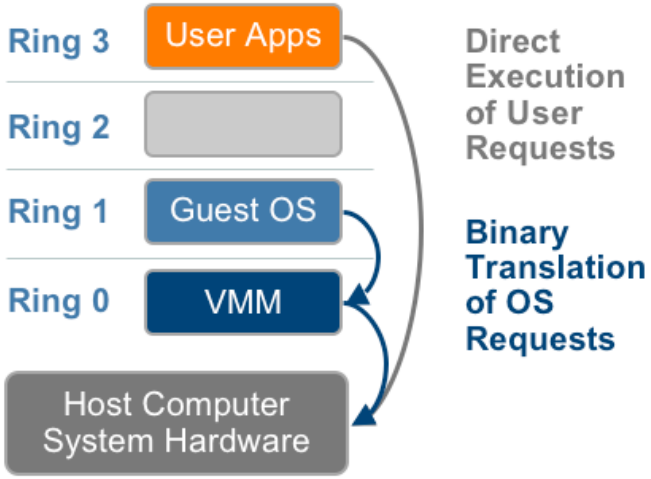
Emulation

Full Virtualization

- Xen and KVM can use full virtualization.
- Full virtualization uses hardware features of the processor to provide total abstraction of the underlying physical system (Bare-metal) and create a new virtual system in which the guest operating systems can run.
- No modifications are needed in the guest operating system.
- The guest operating system and any applications on the guest are not aware of the virtualized environment and run normally.

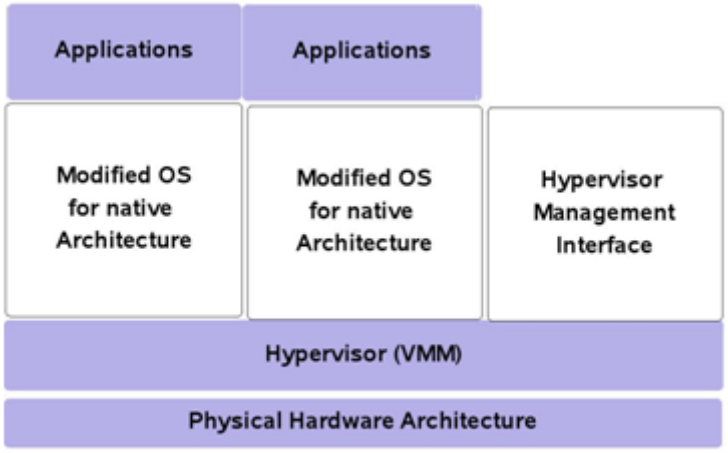


Full Virtualization

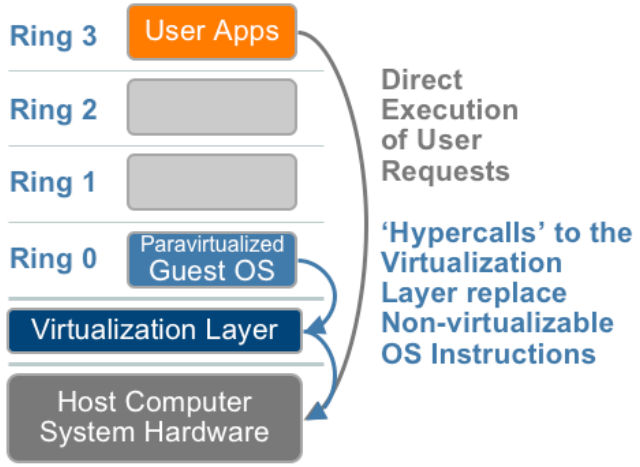


Para Virtualization

- It presents a software interface to virtual machines that is similar but not identical to that of the underlying hardware.
- The intent of the modified interface is to reduce the portion of the guest's execution time spent performing operations which are substantially more difficult to run in a virtual environment compared to a non-virtualized environment.
- The paravirtualization provides specially defined 'hooks' (Hypercalls) to allow the guest(s) and host to request and acknowledge these tasks
- Paravirtualization requires the guest operating system to be explicitly ported. (Modified)
- Example - Xen



Para Virtualization



Xen Virtual machine monitor

- Developed at the University of Cambridge Computer Laboratory.
- GNU General Public License (GPL)
- The x86 architecture support with guest OS modification
- Live migration of running virtual machines between physical hosts.
- Intel and AMD Virtualization Technology for unmodified guest operating systems
- Xen host kernel code runs in ring 0, while the hosted domains run in ring1 or Ring 3.

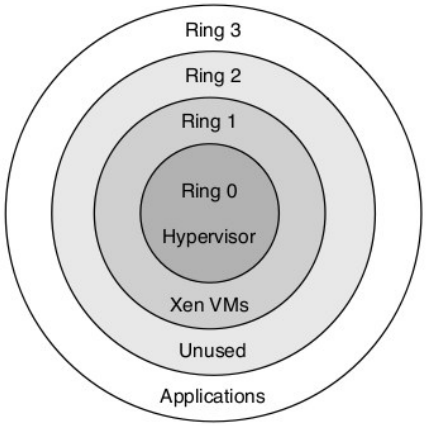


Figure 5 : Privilege Rings in x86 [3][7]

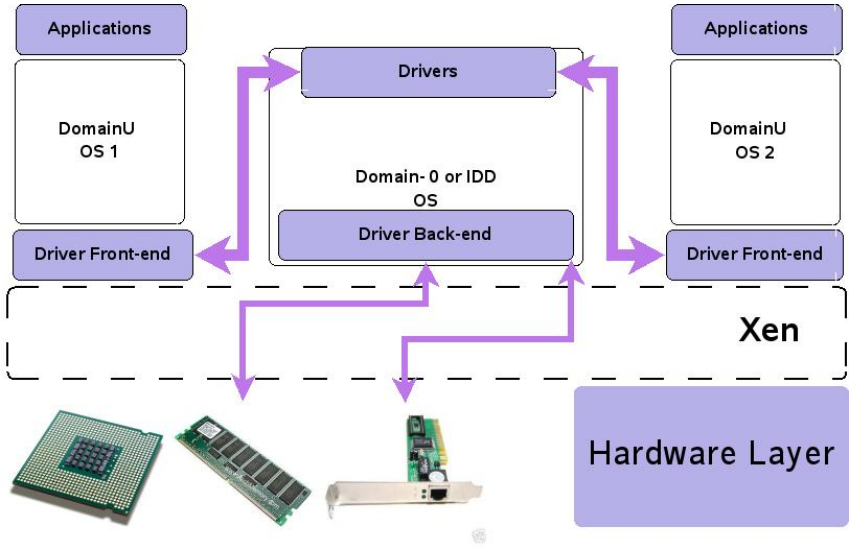
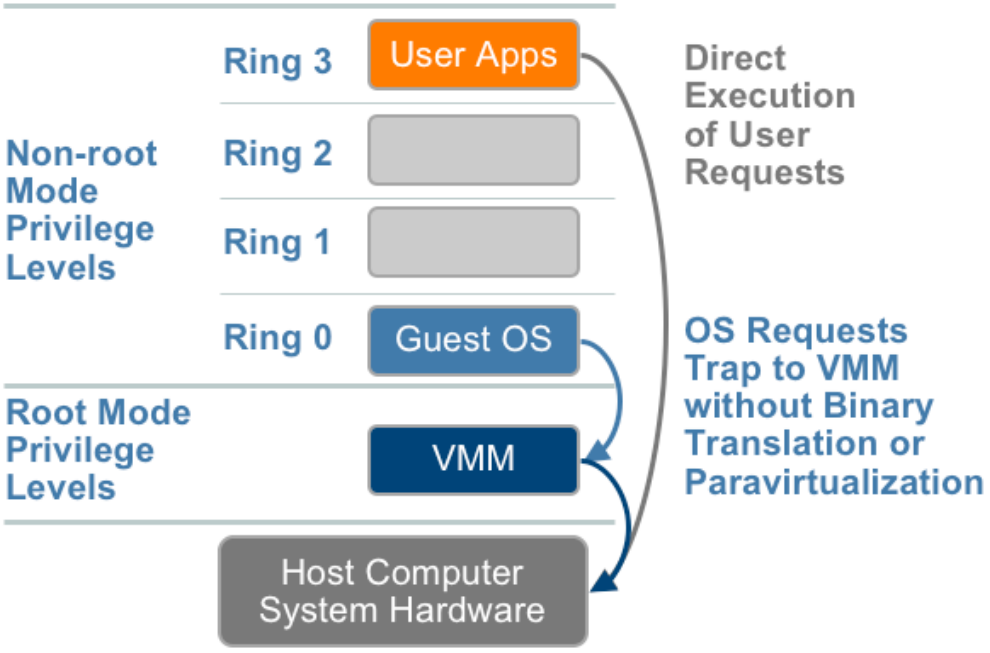


Figure 6 : Xen architecture based on [3][7]

Hardware Assisted Virtualization

- Support from Intel Virtualization Technology (VT-x) and AMD's AMD-V
- Both target privileged instructions with a new CPU execution mode feature that allows the VMM to run in a new root mode below ring 0
- Xen, VMWare



Architectural Support: Intel VT-x

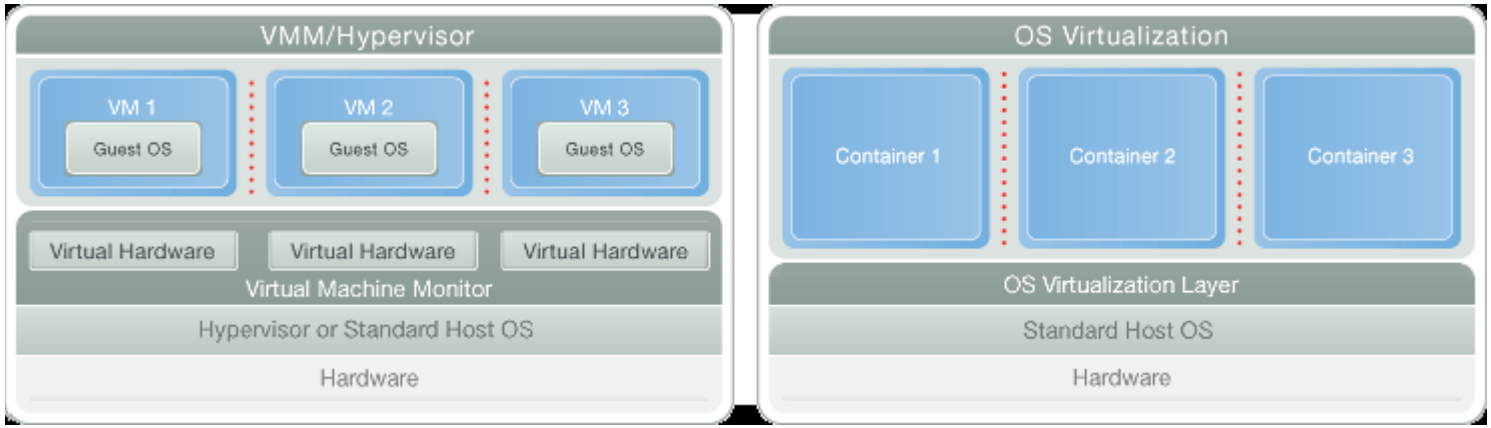
Two modes:

- Root (VMM)
- Non-root (Guest OS)

- Transition operations:
  - VM entry, VM exit
  - Transitions occur for sensitive instructions
- VM control structure
  - Stores VM state (e.g., registers)
  - Similar to process control block

OS level/Shared OS Virtualization

- Shared OS image
- Isolation, abstraction and security
- Examples - Parallels, Virtuozzo



Resource Virtualization

We need to virtualize:

- CPU
- Memory
- I/O devices
- Network

Processor Virtualization

- VMM runs in privileged mode
- Guest OS and applications run in non-privileged mode
- VMM time slices between different VMs similar to process time slicing
- Whole VM state preserved on switching
- Need to trap sensitive instructions
- Dynamic binary translation: Patch sensitive instructions to trap into the VMM
- Hypervisor: Allows explicit *hyper-calls*

Interrupt Handling

- Interrupts managed by VMM
- Guest OS disables interrupts => VMM queues up subsequent interrupts
- Guest enables interrupts => deliver queued interrupts
- VMM traps special instructions, e.g., POPF, so that VM sees disabled/enabled interrupts
- Timer interrupts are handled by VMM

Resource Virtualization

Various resource Types:

- Processor

- Memory

- Network

- I/O devices

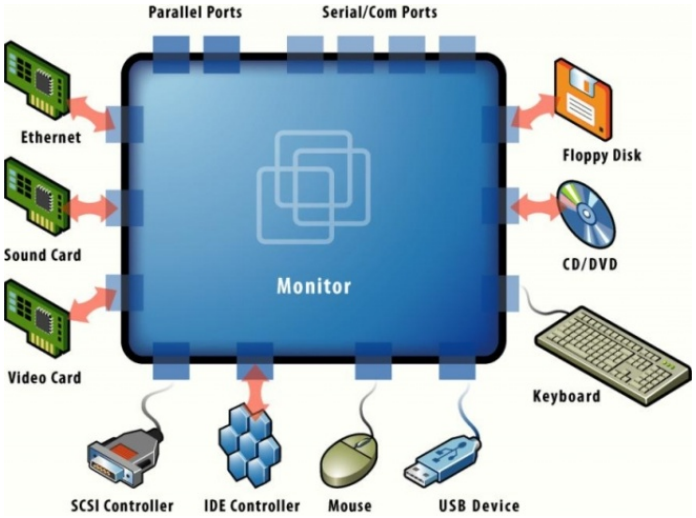


Image Source : [19,20,21,22]

Processor Virtualization

- VMM time slicing
- Binary translation (VMware)
- hyper-calls (Xen)
- Intel VT-x and AMD VT architectural support

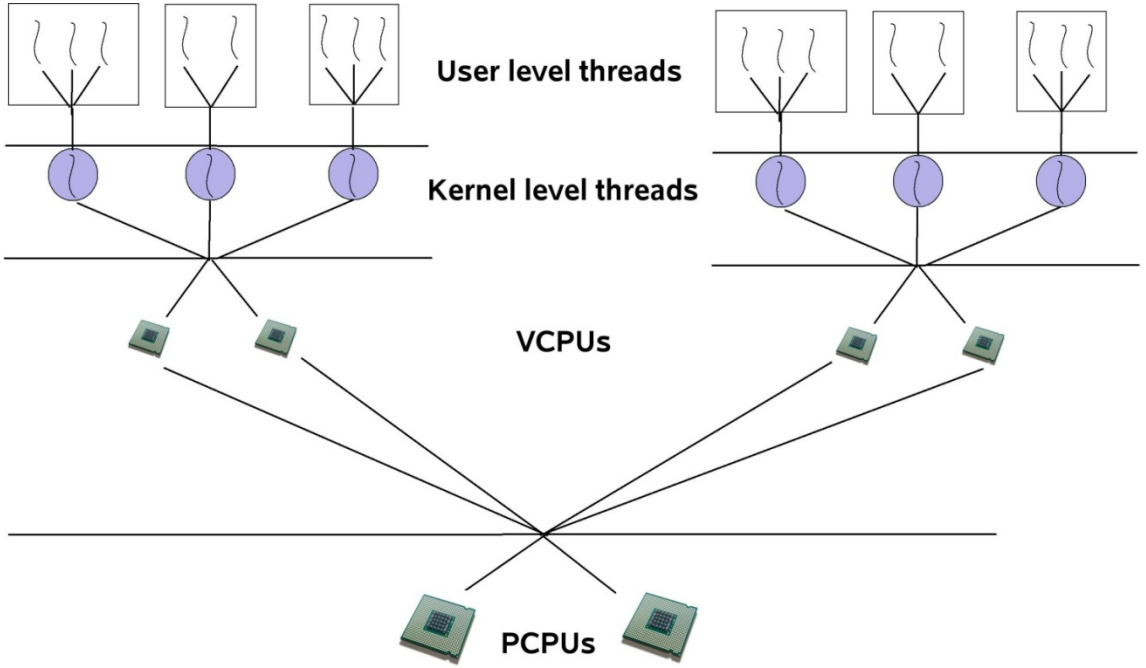


Figure 4 : Scheduling in Xen (based on [4])

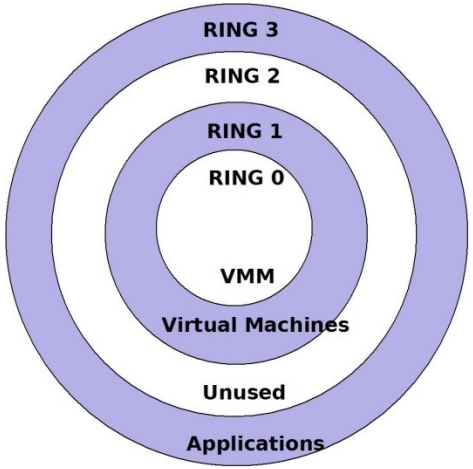


Figure 3 : Privilege Rings in x86 [3][7]

Memory Virtualization

- Guest OS sees flat “physical” address space.
- Page tables within guest OS:
  - Translate from virtual to physical addresses.
- Second-level mapping:
  - Physical addresses to machine addresses.
- VMM can swap a VM’s pages to disk.

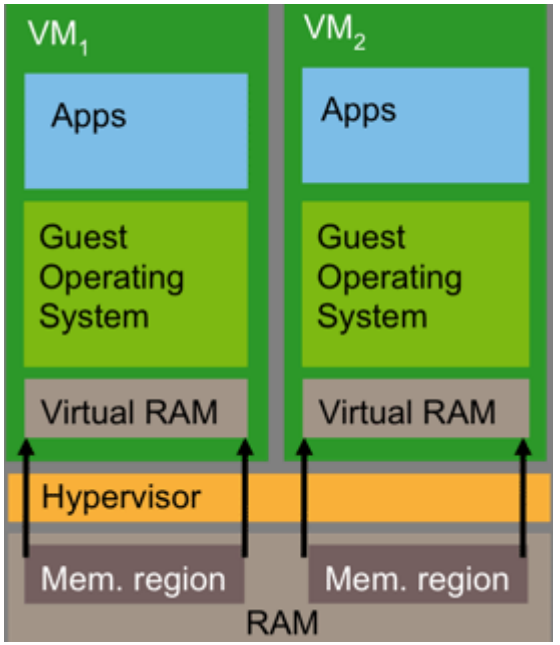


Figure 5: Memory Virtualization [15]

I/O Virtualization

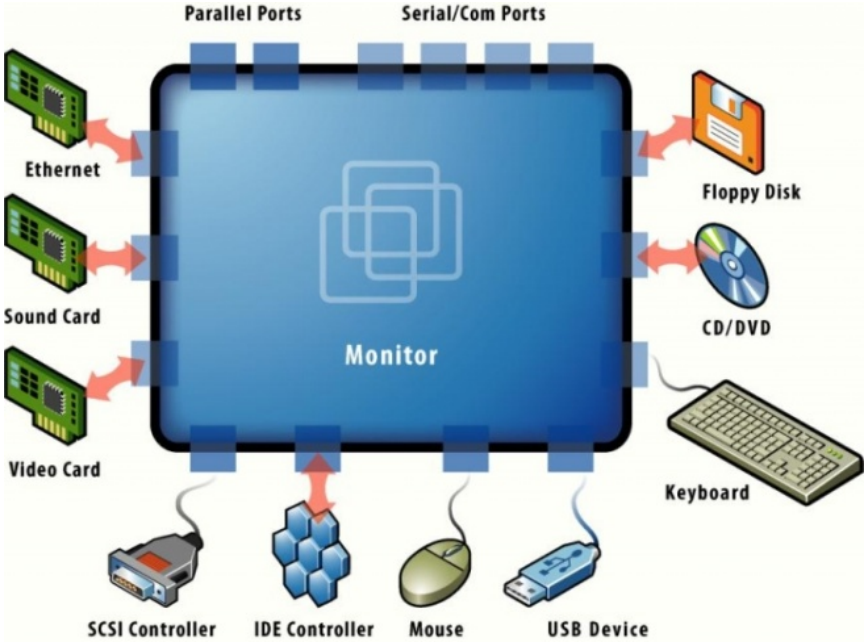


Figure 6: I/O devices [21]

Problems:

- Different devices have different characteristics.
- Large number of devices.

Techniques:

- VMM virtualizes devices, translates into native device I/O.
- VMM gives access to I/O devices, but controls access.

Host-based I/O [3]

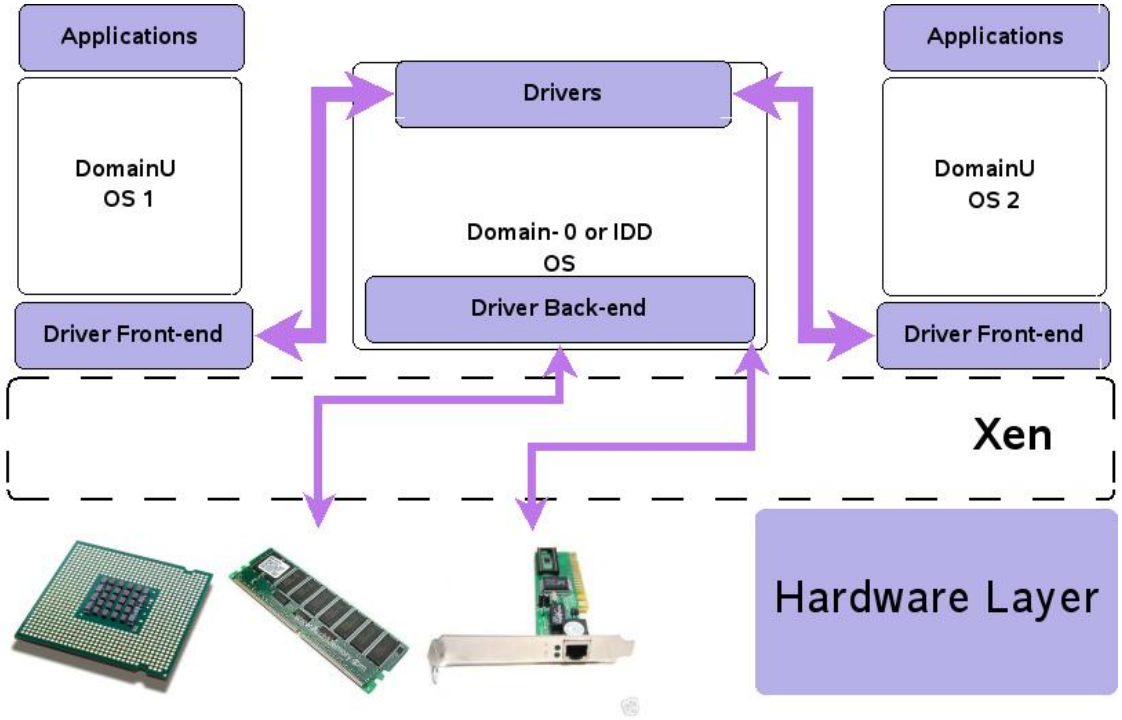


Figure 7: Device Driver handling by host OS

- VMM uses device drivers in the host OS.
- Guest I/O commands converted to host I/O system calls.
- E.g.: virtual disk is mapped to a file on host.
- Disk reads/writes become file reads/writes.

Virtual Network [16]

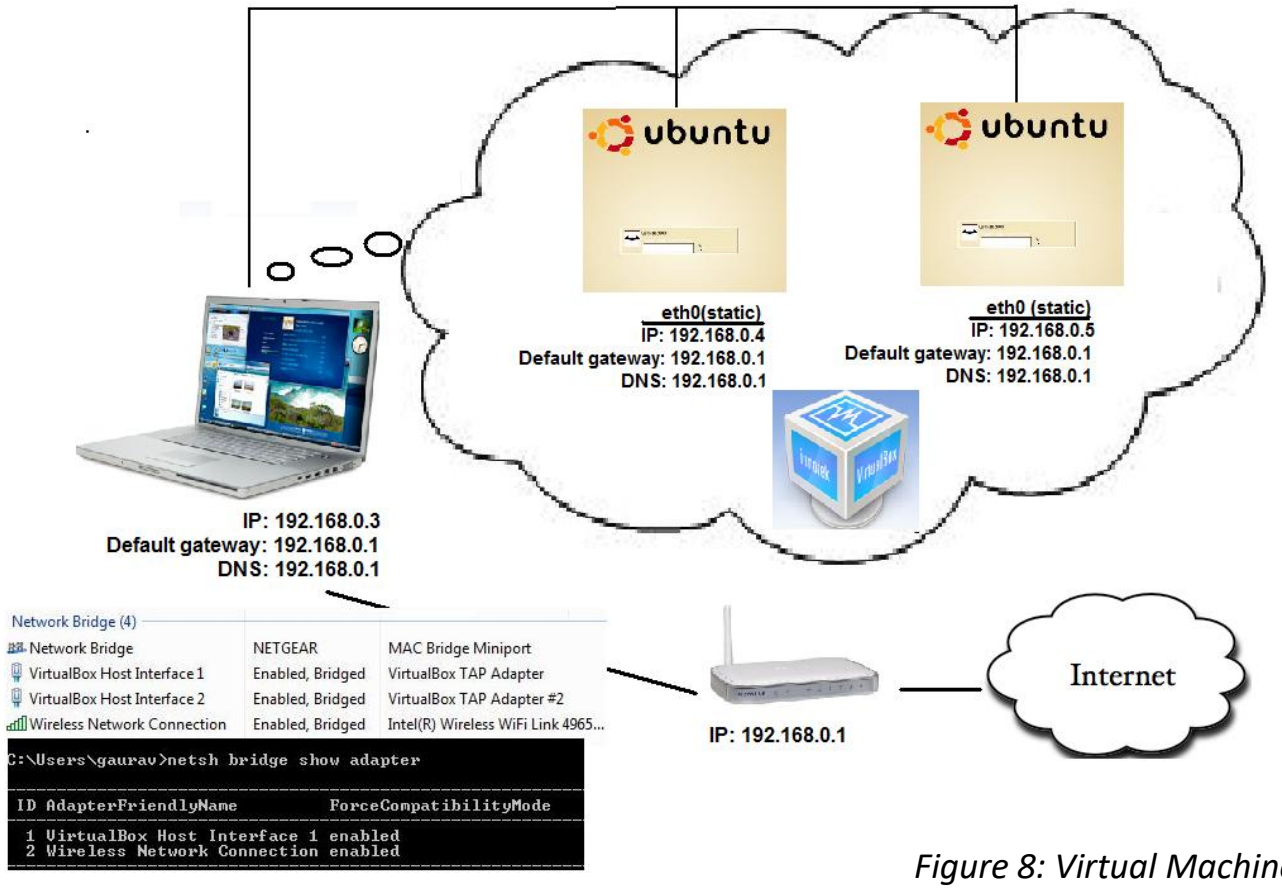


Figure 8: Virtual Machine's Network

- Each VM is assigned a separate IP address.
- Communication among VMs: No physical networking required.
- How do machines across the network talk to a specific VM?
- Physical network device in promiscuous mode => Listens to all packets, picks up VM-specific packets.
- NAT, Bridged, Host-only or internal network connections.

Virtual machine Scheduling

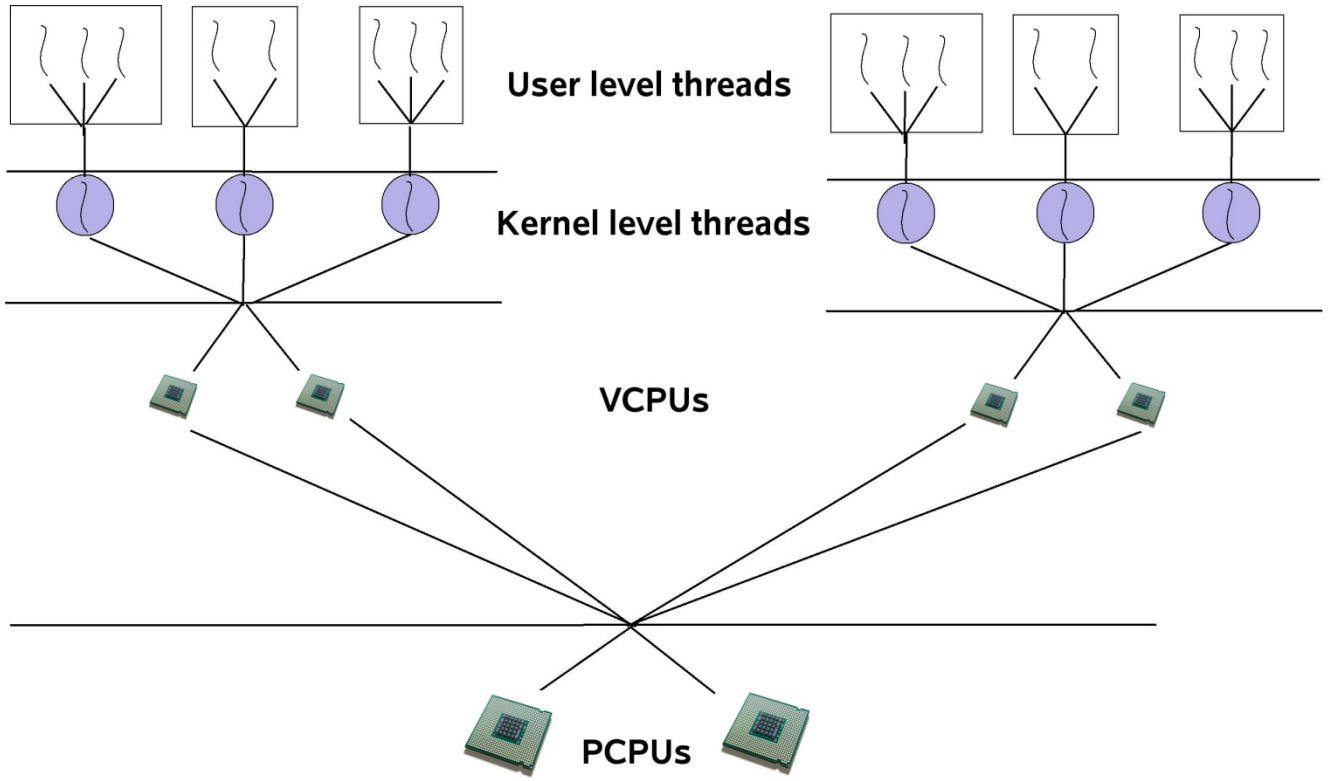
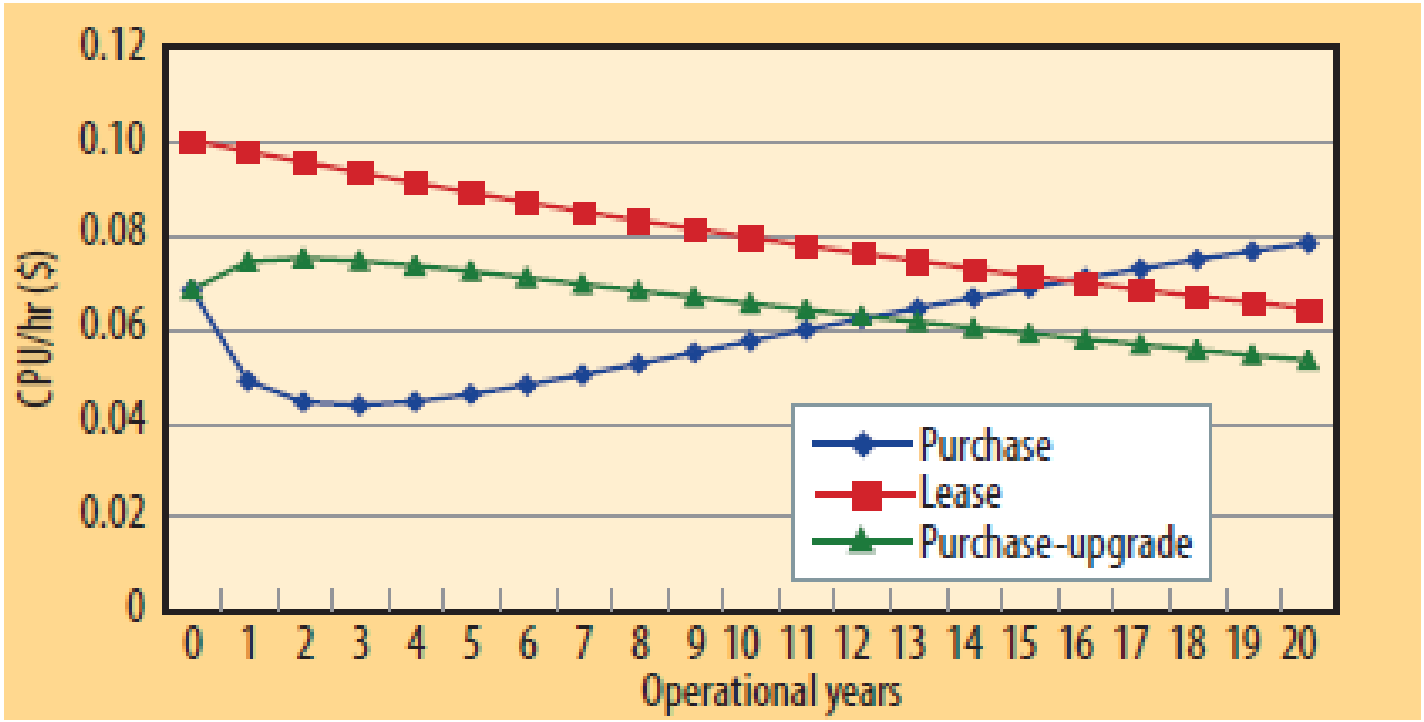


Figure 7 : Scheduling in Xen (based on [4])

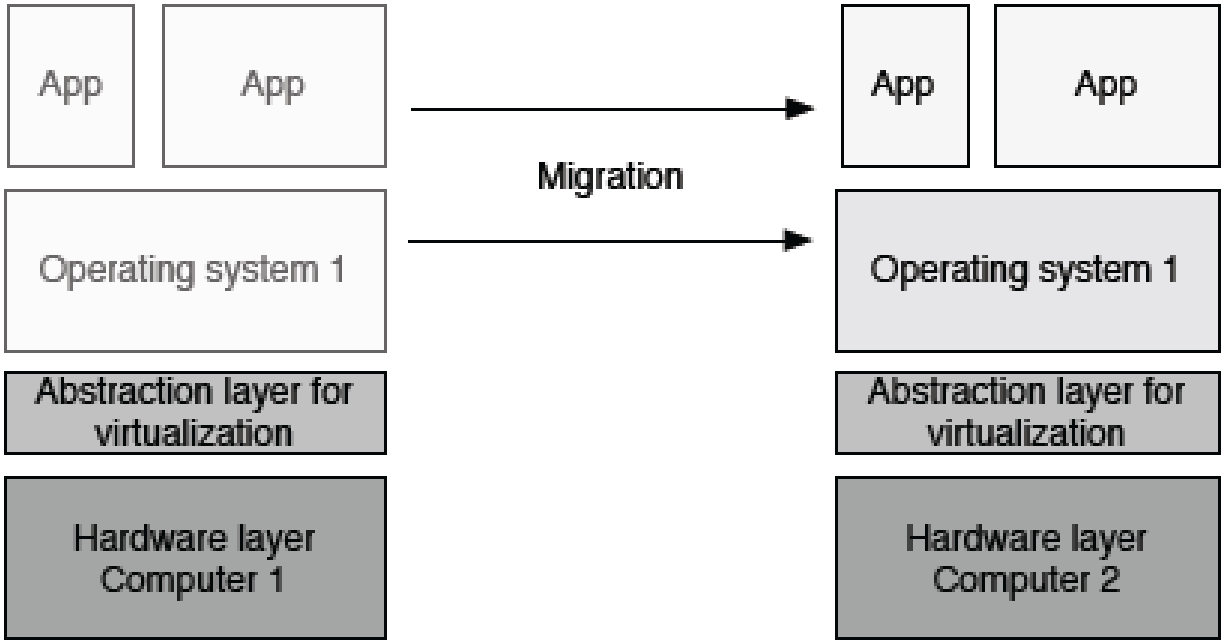
Three Tiers of schedulers

- User space threads to kernel level threads in guests
- Guest kernel mapping threads to VCPUs
- VCPUs to Physical processors

Real Cost of a CPU hour



VM Migration



Motivation : VM Migration

- Load management
- Maintenance of original host
- Why VM Migration?
  - Avoid difficulties of process-level migration approaches, such as residual dependencies
  - In-memory state can be transferred in a consistent and efficient fashion. We can migrate an on-lime game server or streaming server without requiring clients to reconnect.
  - Separation of concerns between users and operators. Very powerful tool for cluster administrators. Separate hardware and software considerations, consolidate clustered hardware into a single coherent managed domain

Approaches

- **Static**
  - To provide mobility to users who work on different physical hosts at different times, e.g. transfer OS from work to home while on the subway.
  - Optimize for slow links and longer time spans, stops OS execution for the duration of the transfer, with a set of enhancements to reduce the transmitted image size.
- **Live Migration**
  - Move a running virtual machine or application between different physical machines without disconnecting the client or application.
  - Requires shared storage provided by SAN or NAS.

Design Issues

- Migrating Memory
  - How to move while minimizing downtime and total migration time?
- Delta Copying
- Don't disrupt active services through resource contention by migrating the OS.
- Local Resources
  - What to do with resources associated with the physical machine when they are migrating away from?
    - Memory
    - Connections to local devices such as disks and network interfaces

Design Overview

- Stage 0: Pre-Migration
- Stage 1: Reservation
- Stage 2: Iterative Pre-Copy
- Stage 3: Stop-and-Copy
- Stage 4: Commitment
- Stage 5: Activation

Where does innovation lie?

- Resource Allocation
- VM Placement
- VM Configuration
- SLA fulfillment
- Migration over internet/public clouds

Open Virtualization Format

- OVF enables efficient, flexible, and secure distribution of enterprise software, facilitating the mobility of virtual machines and giving customers vendor and platform independence.
- Customers can deploy an OVF formatted virtual machine on the virtualization platform of their choice.
- Distributed Management Task Force(DMTF) uses existing packaging tools to combine one or more VM together with a standards-based XML wrapper that provides the virtualization platform -- from VMware, Microsoft, Citrix, or others -- a portable package, which includes installation and configuration parameters for the VMs.

Virtualization in cloud

"A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements (SLA) established through negotiation between the service provider and consumers. [7]."

Open Source IaaS Cloud Platforms[10]
OpenNebula, Haizea, XenCloud, and Eucalyptus etc.

Open Source VMMs[3]
Xen, KVM, Linux VServer, and UML etc.

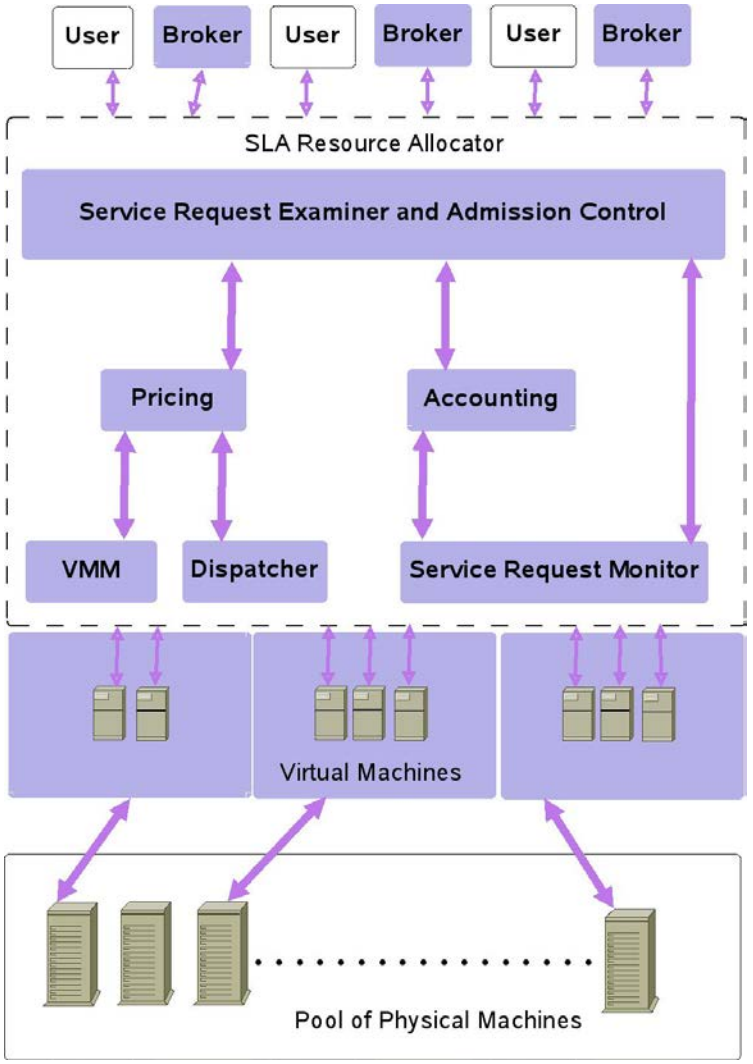
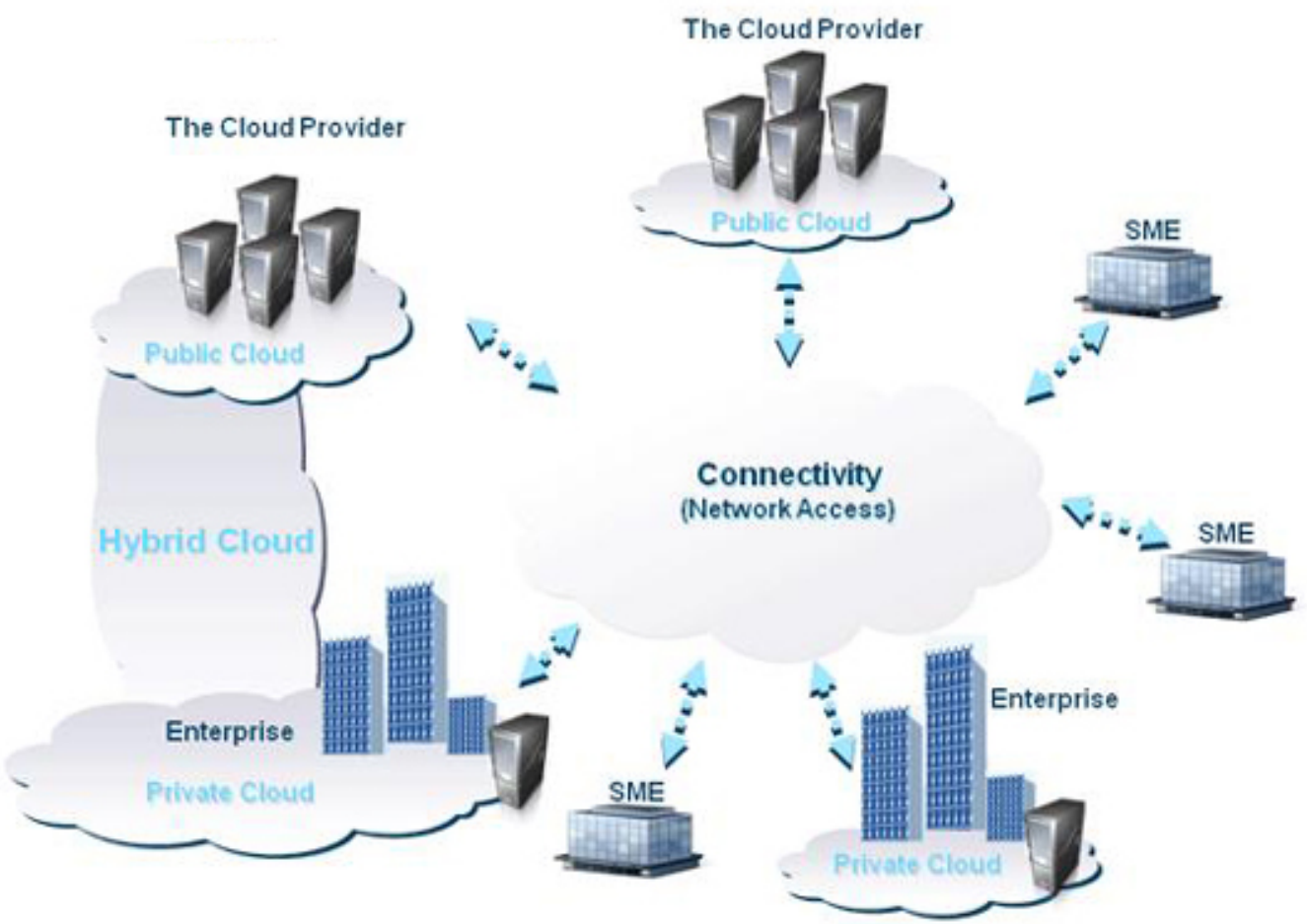


Figure 10 :Architecture of cloud computing[7]

Public/Private/Hybrid Clouds [7]



Important Issues?

- Resource Allocation and Scheduling [23]
  - Initial VM Placement
  - VM Configuration
- SLA Matching [7]
- Live Migration over internet/public clouds [13]

VM Migration

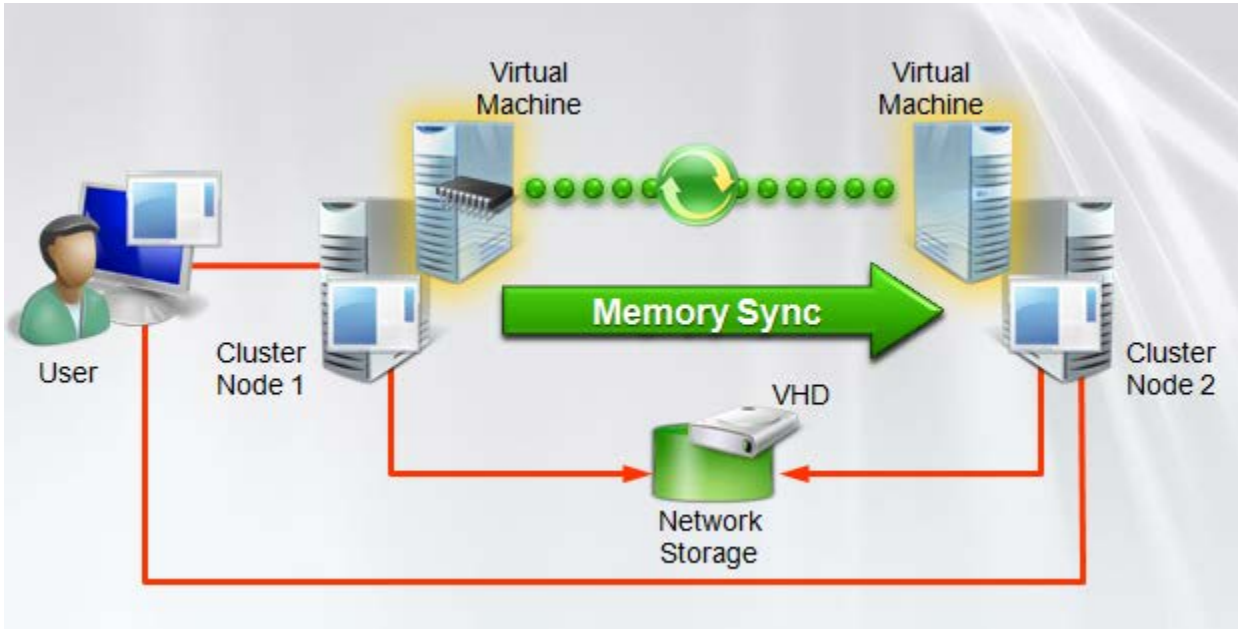


Figure 11 : Live Migration of a VM [13]

VM Placement [24]

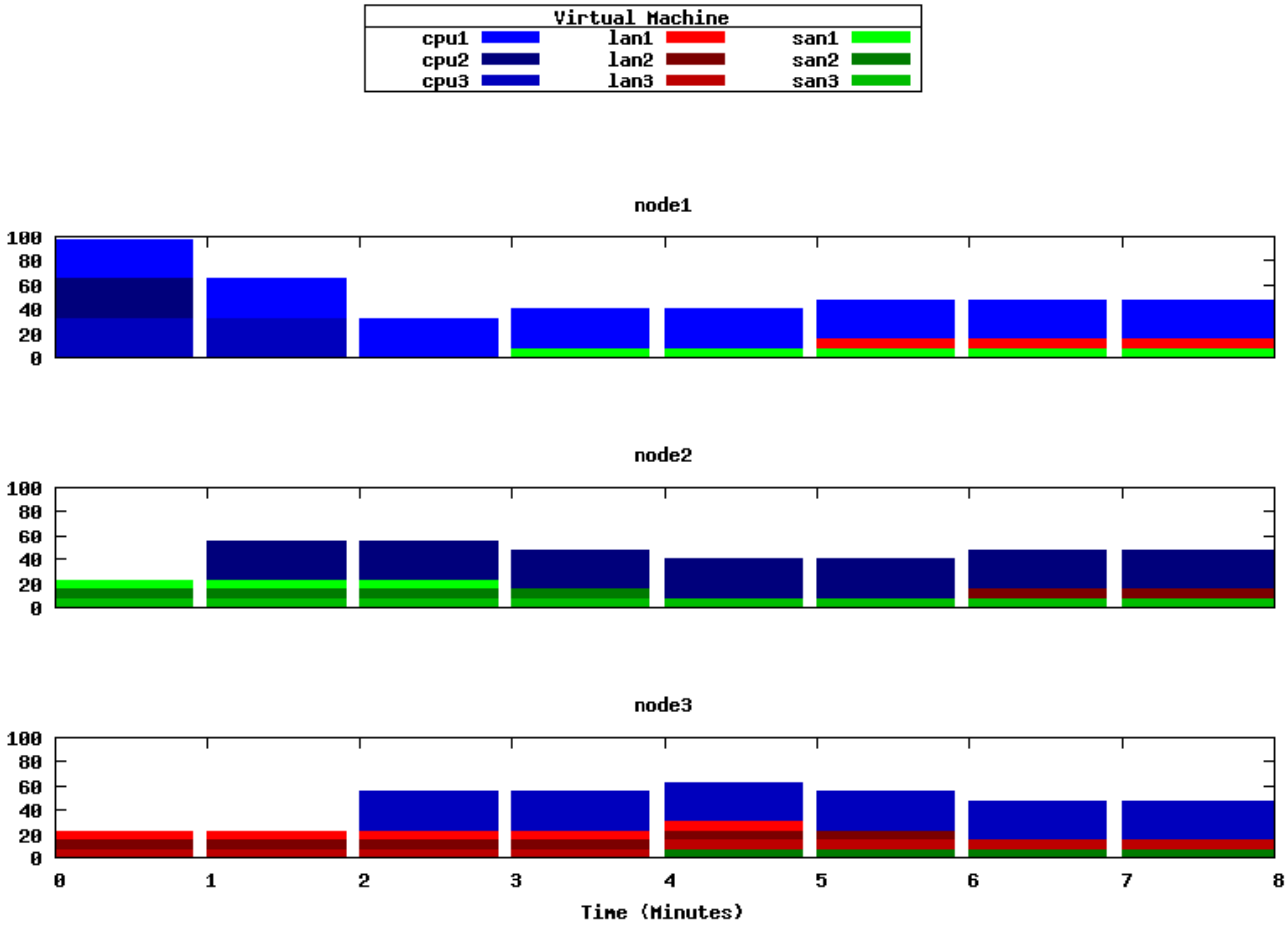


Figure 12 : Balance Policy per host CPU loads as stacked bars of virtual machine load[24].

Resource Allocation in Cloud

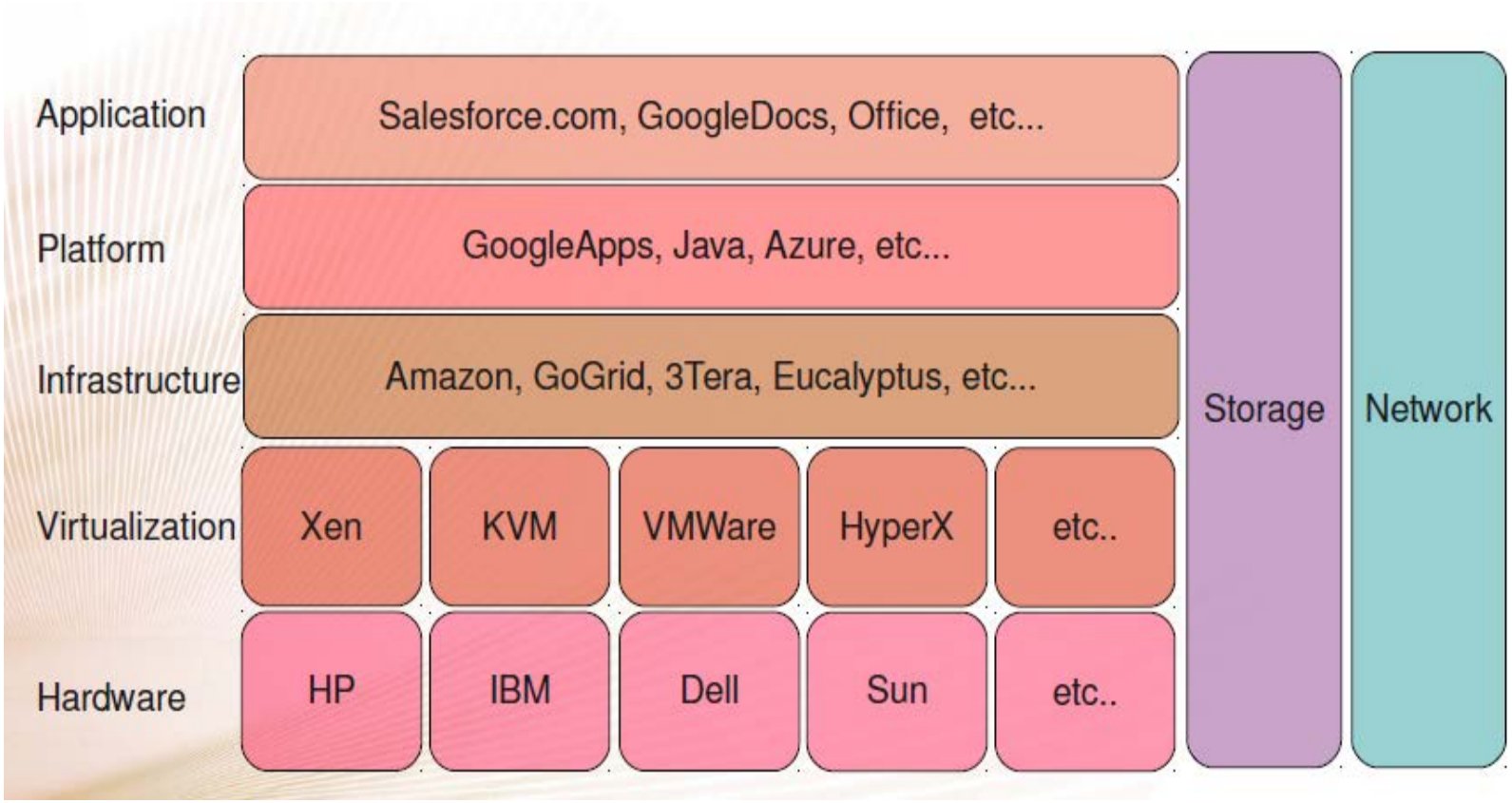
- Initial Placement Considerations[9][18]
  - Co-hosted VMs and their interference
  - Load Balancing among various servers
- Elasticity of resources
  - Feedback control mechanisms and prediction
- Resource outage and live migration of operating systems[13]
- VM scheduling algorithms and real time considerations [5]

Cloud Installation

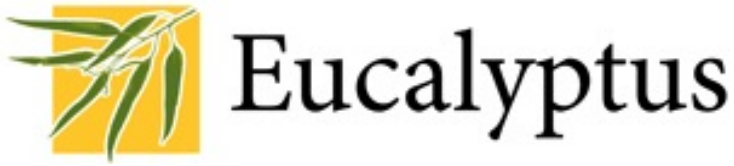
OpenNebula And Haizea

- OpenNebula is Virtual Infrastructure (VI) software toolkit, which is used to control a VMs lifecycle.
- It manages the VM image and storage, the network fabric (such as DHCP) services to tie in VMs with the environment, and hypervisors which create and control the VM. It can deploy groups of virtual machines to be treated as a single unit.
- Haizea can be used to extend OpenNebula's scheduling capabilities, allowing it to support advance reservation of resources and queueing of best effort requests.
- OpenNebula and Haizea complement each other, since OpenNebula provides all the enactment muscle (OpenNebula can manage Xen, KVM, and VMWare VMs on a cluster) and Haizea provides the scheduling brains.

Available options



Option Source Cloud

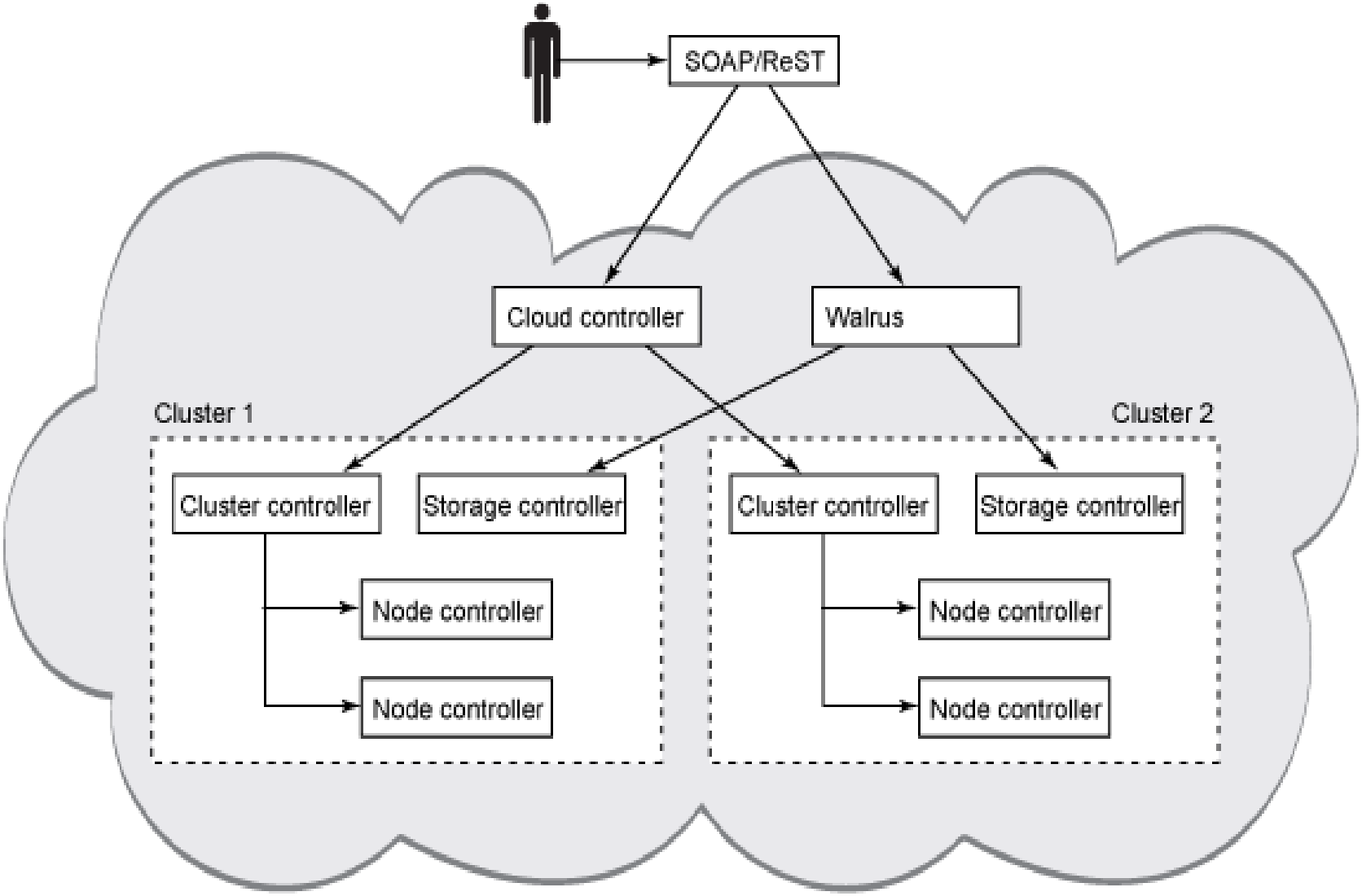



ubuntu enterprise cloud

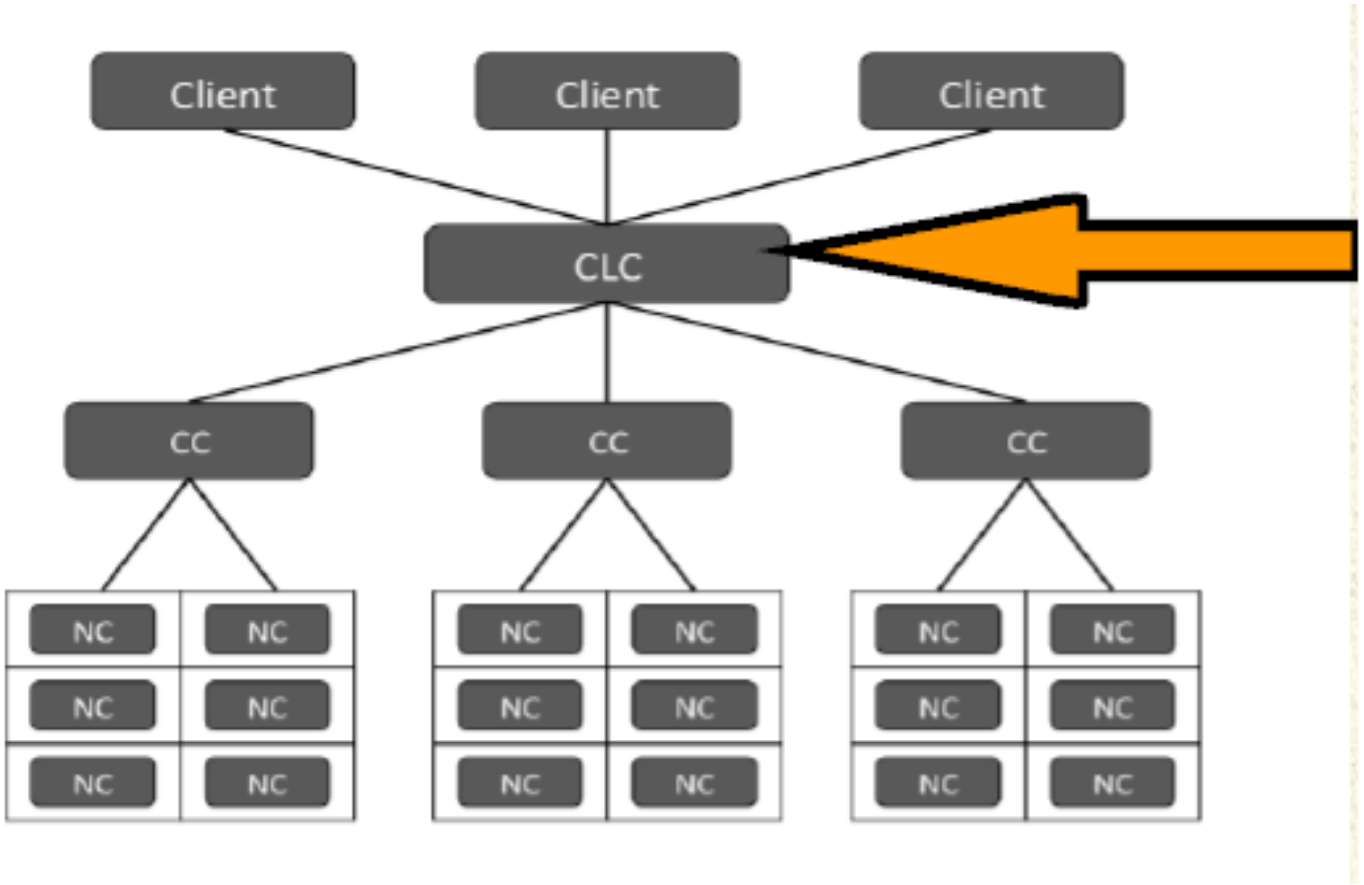
Private Cloud Components

- Cloud controller (CLC)
- Cluster controller (CC)
- Node controller (NC)
- Storage controller(SC)

Private Cloud-Multi Cluster

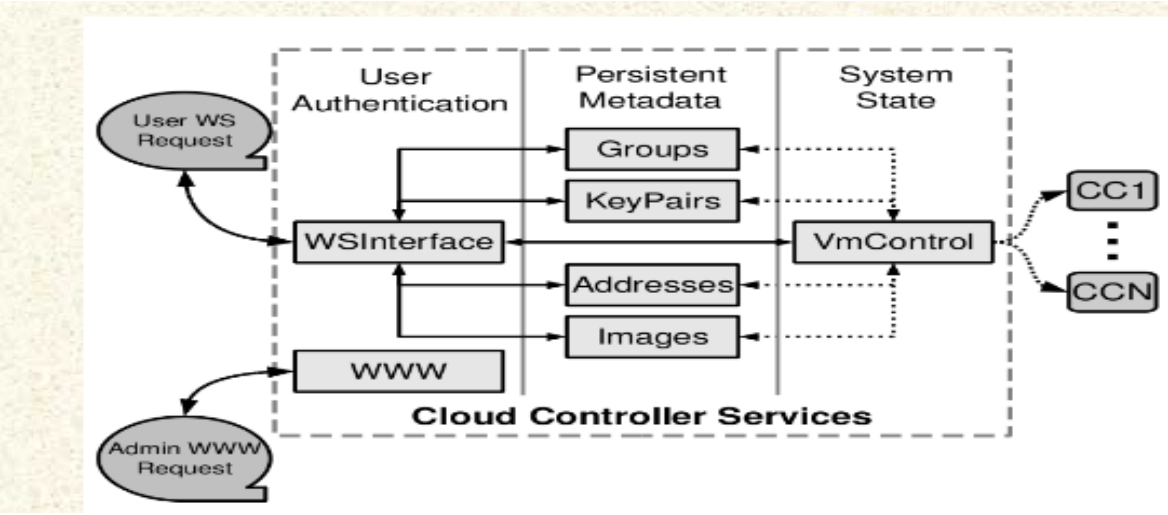


Architecture

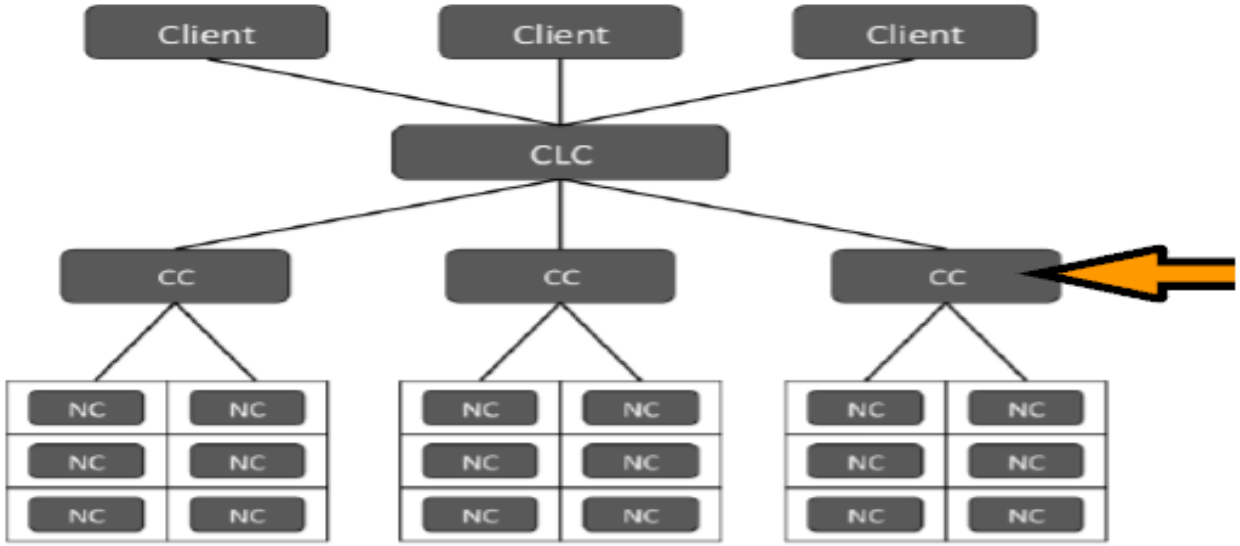


Cloud Controller

- User/Admin entry point
- High level VM instance scheduling decision
- Processing of SLA
- Maintaining system and user metadata



Cluster Controller



- Node controller
- Schedule incoming VM instance on node

Node Controller

- Execute on physical resource
- Image instance startup
- Inspection
- Shutdown
- Only on NC is required per physical node
- Call hypervisor (XEN or KVM) to control VM images

STEP 1: Prerequisites

Front End(All In One)

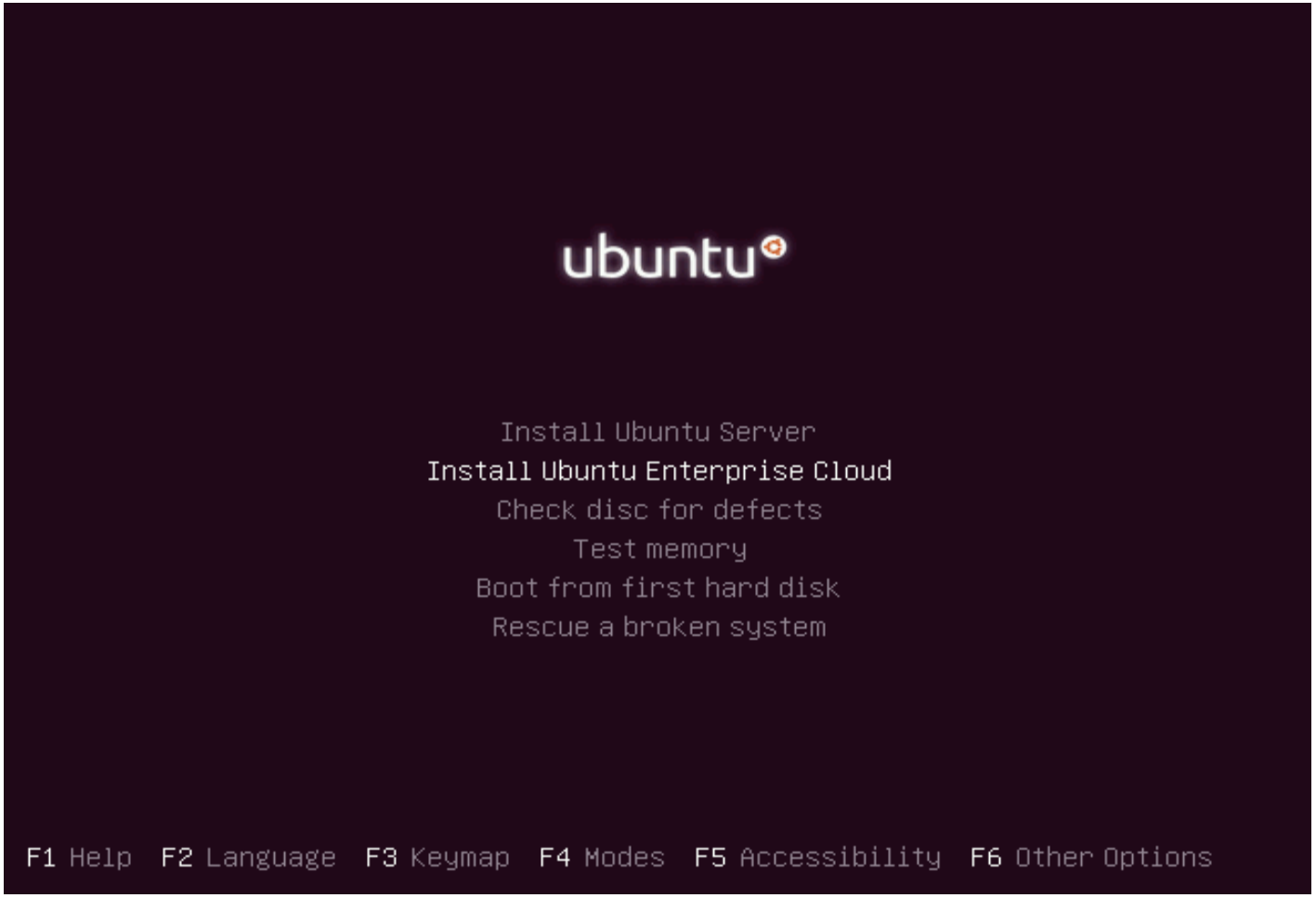
- Cloud controller(clc)
- Cluster Controller(cc)
- Storage Controller

Cluster Node

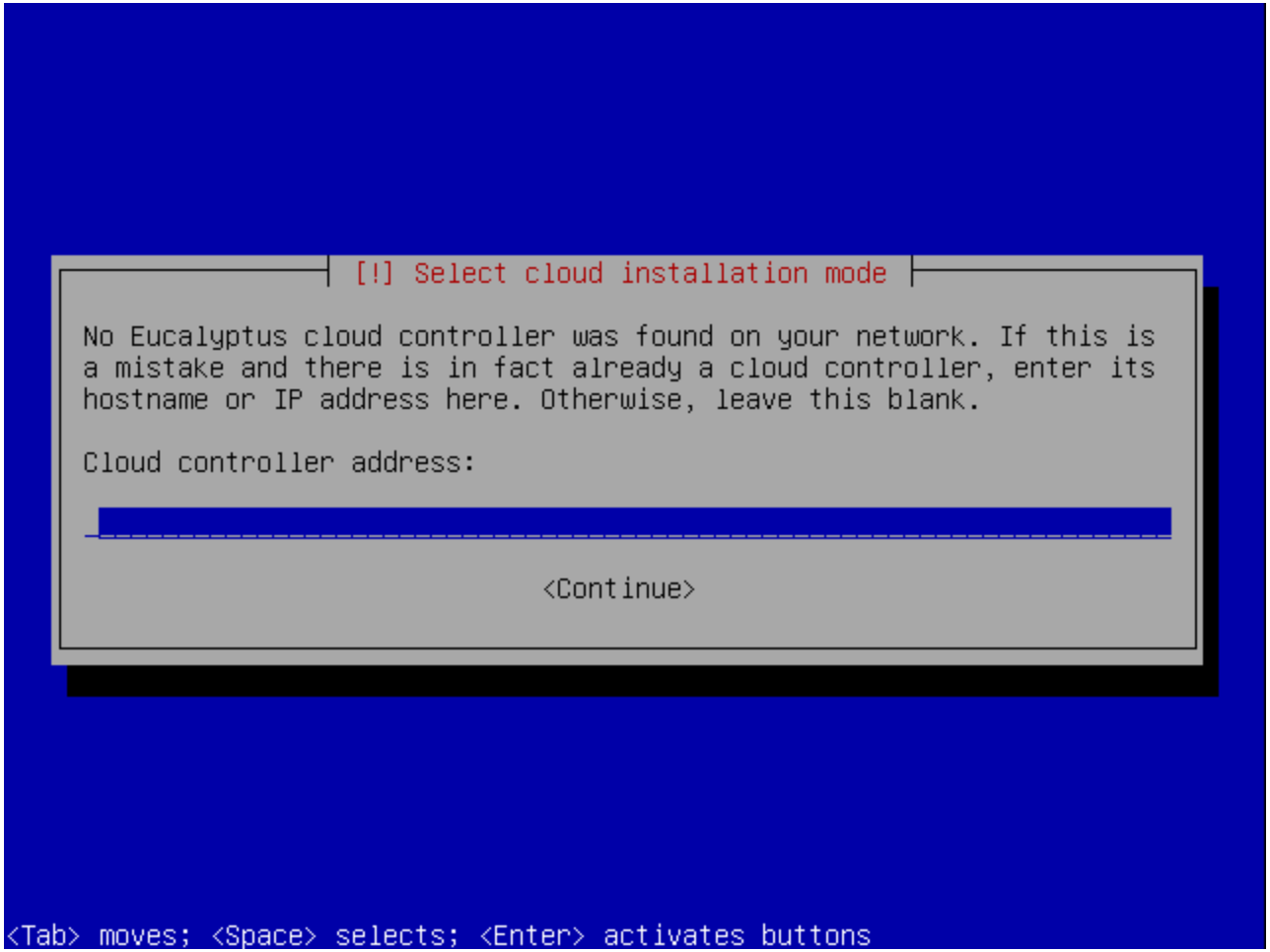
-Node Controller

- Virtualization controller

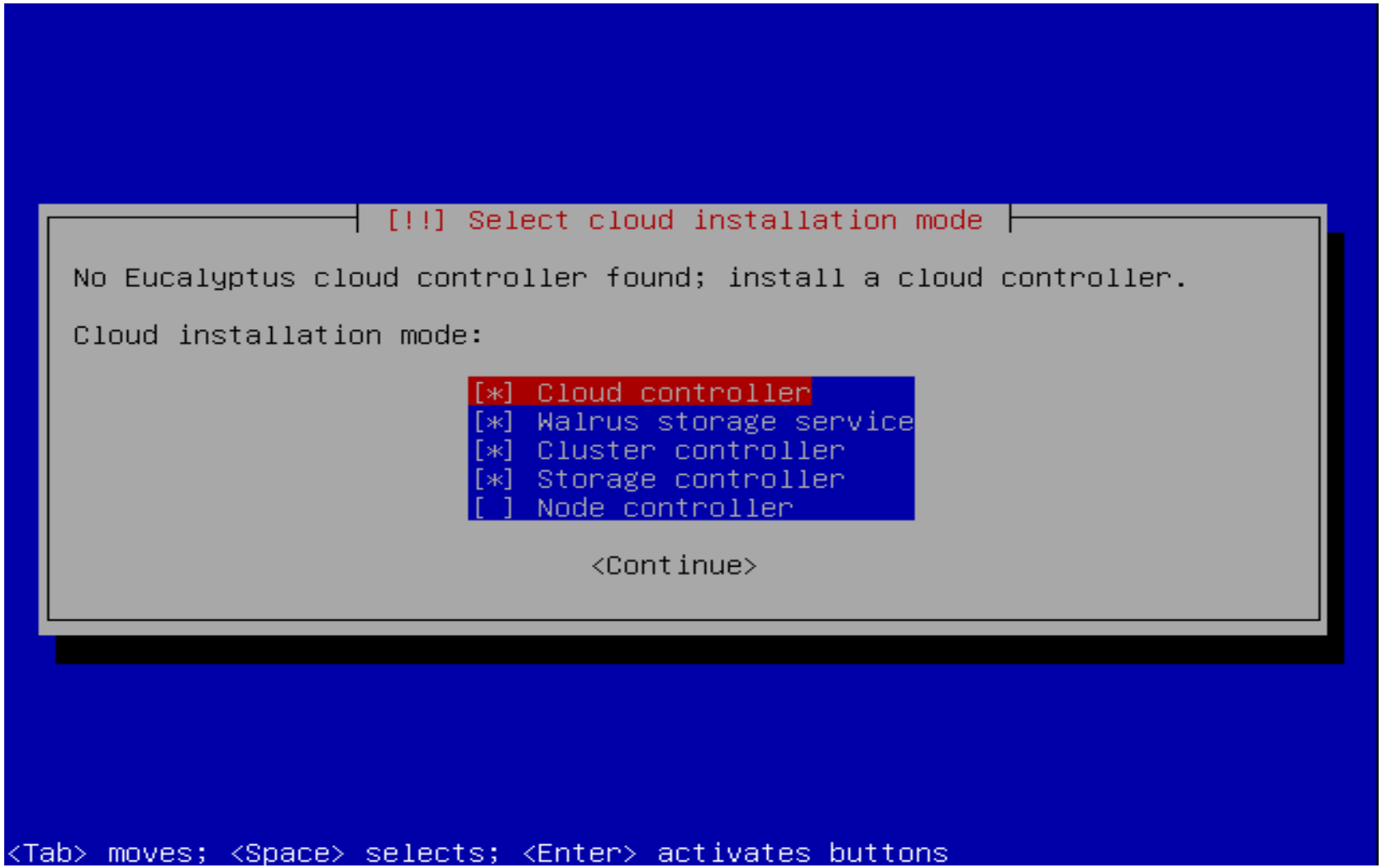
STEP 2: Install the Cloud/Cluster/Storage Front End Server



Does Cloud Controller Exist in Network?

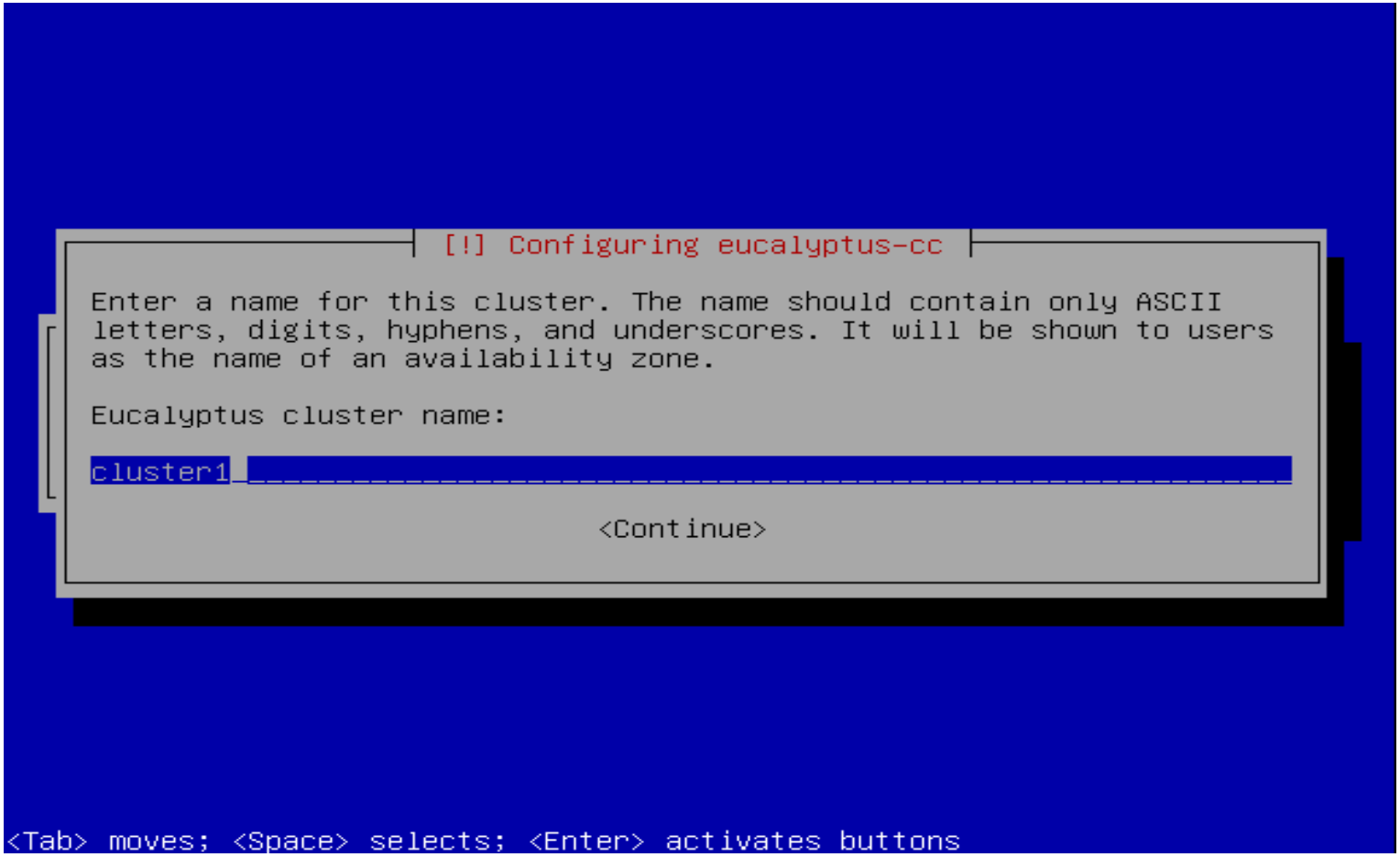


IF NOT- You can then choose which components to install

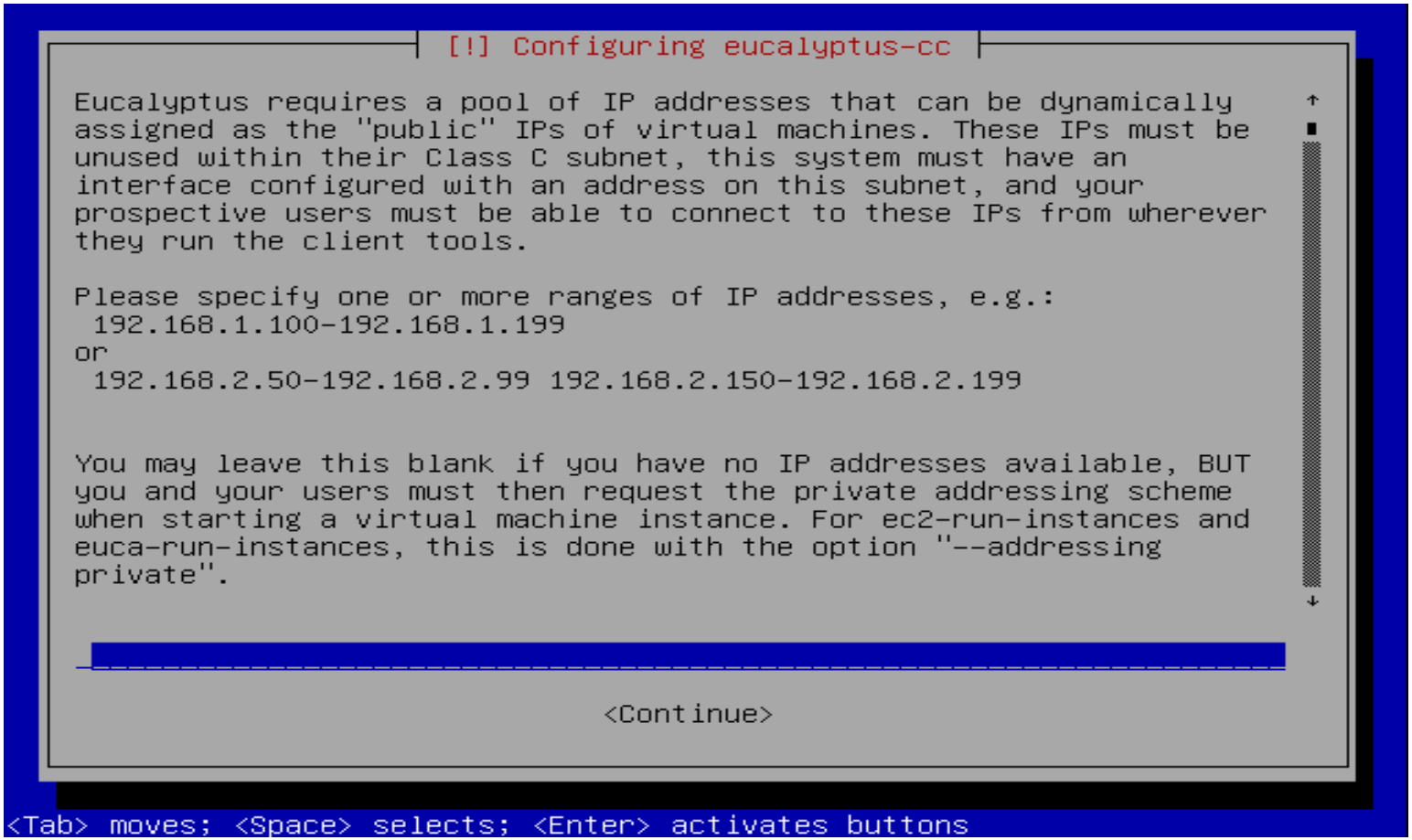


It will ask two other cloud-specific questions during install:

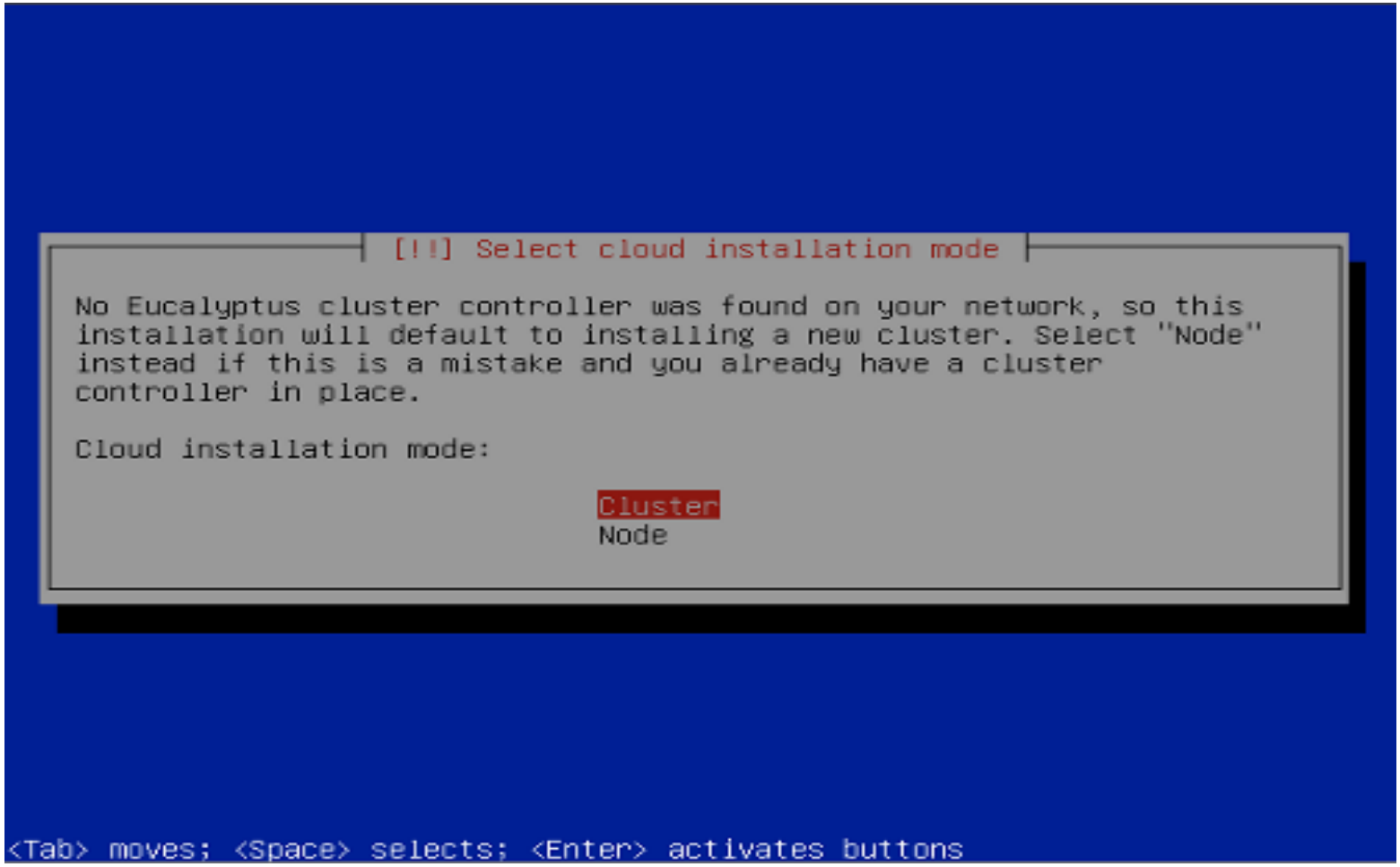
1-What is your cluster name ? Eg- cluster1



A range of private IP addresses on the LAN that the cloud can allocate to instances, e.g. *192.168.1.200-192.168.1.249*.

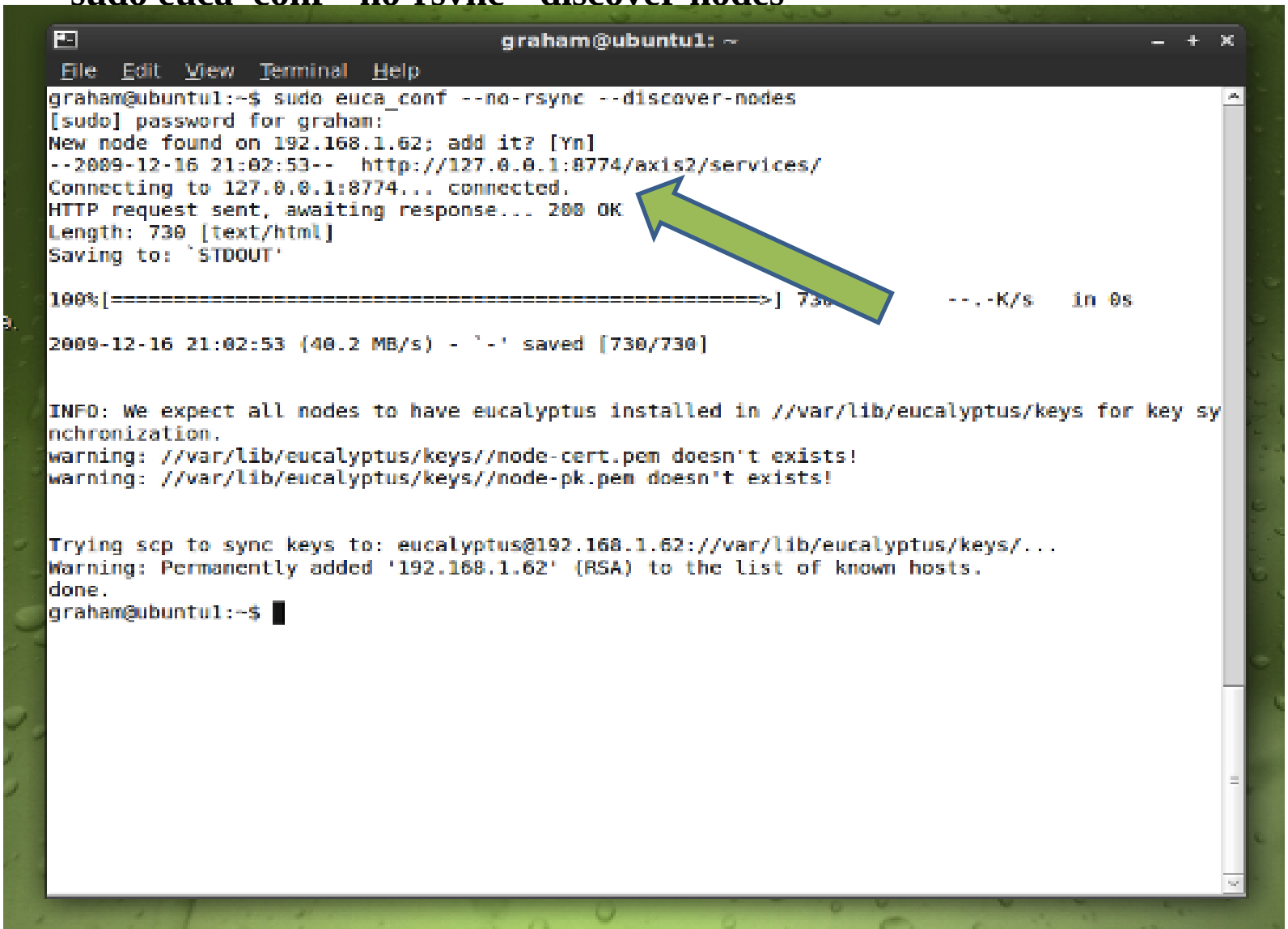


STEP 3: Install the NC at Each Node

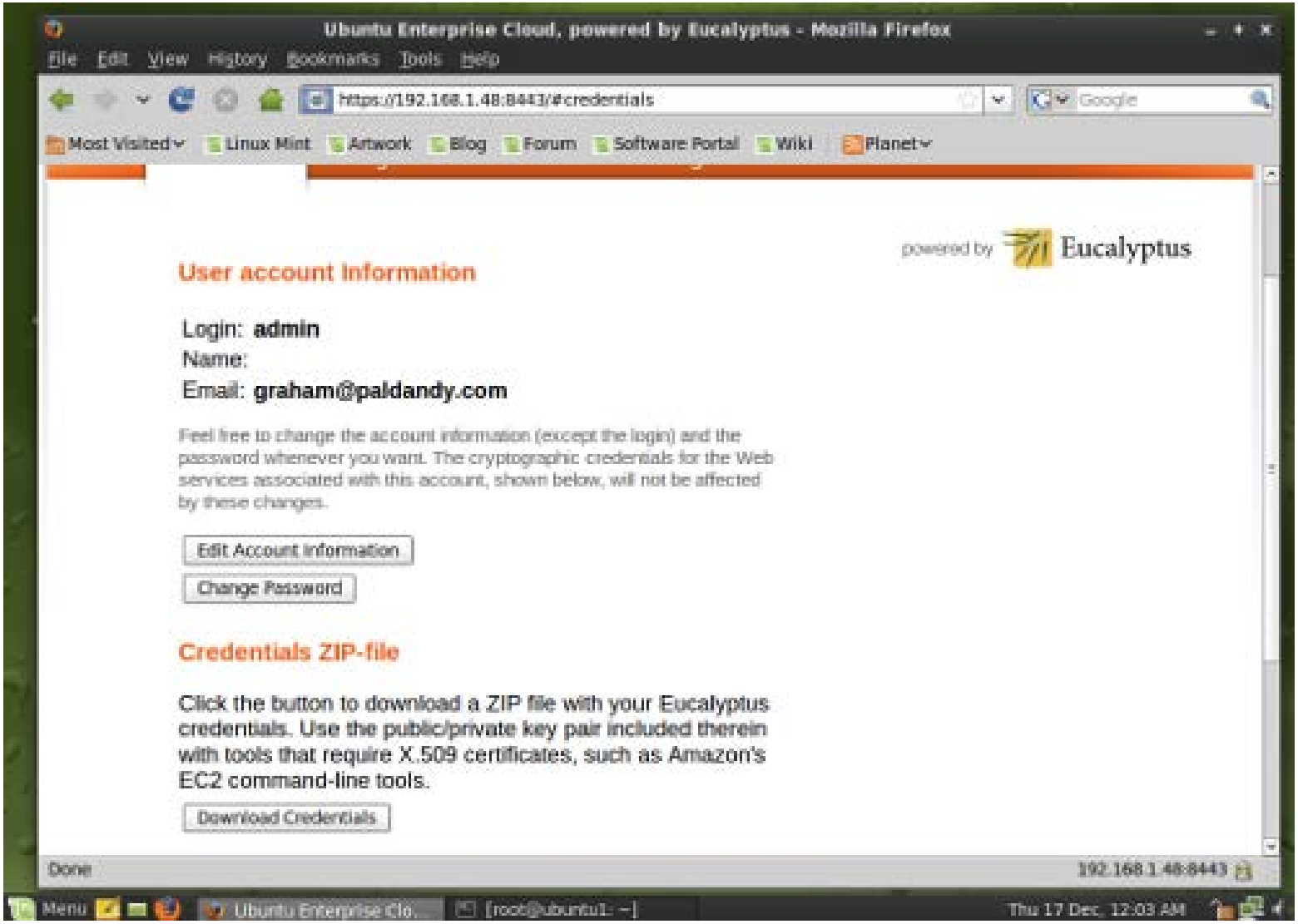


STEP 3: Configuration(Discover all nodes)

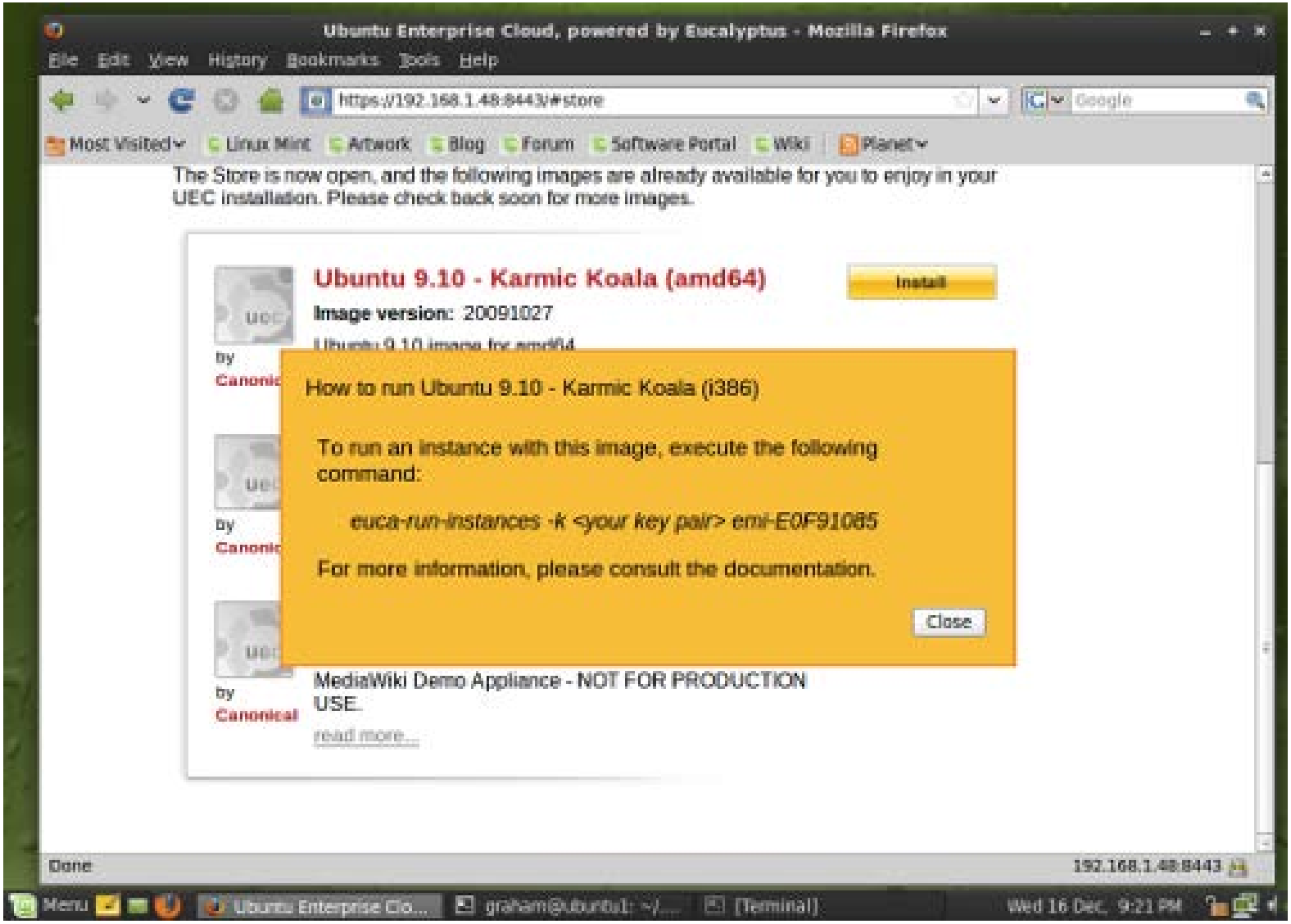
sudo euca_conf --no-rsync --discover-nodes'



STEP 4: Credentials

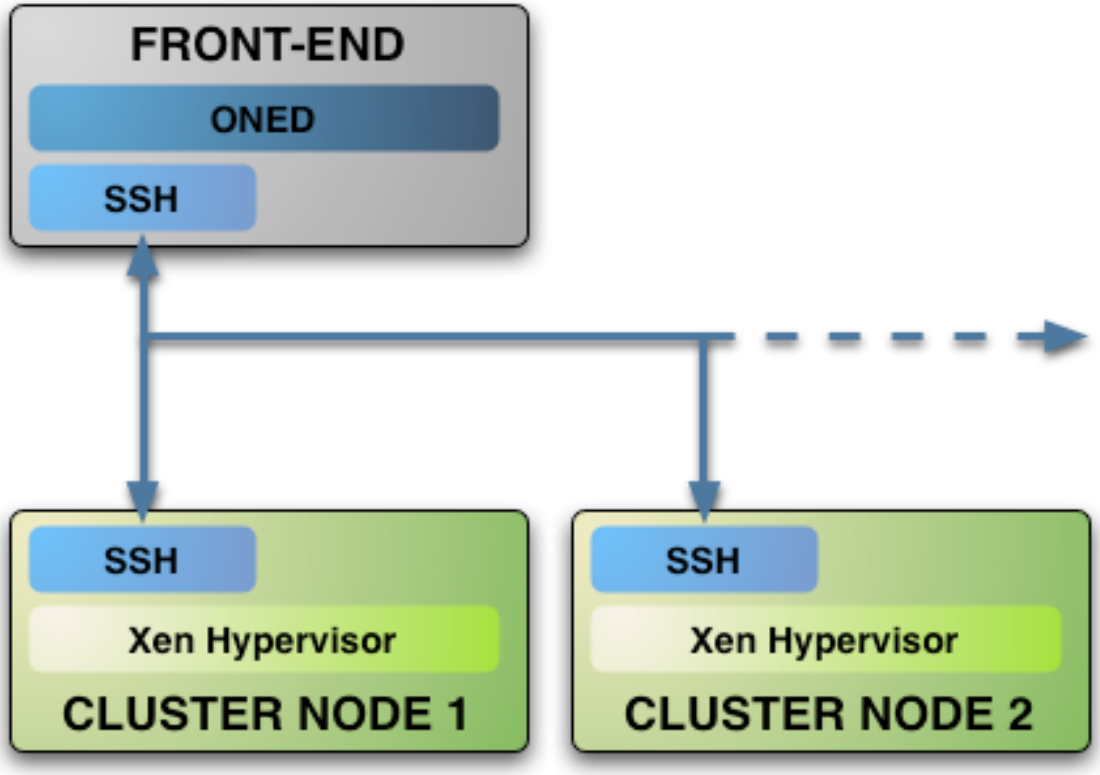


STEP 5: Run an OS image




OpenNebula.org
The Open Source Toolkit for Cloud Computing

Same Setup as UEC



Front End (OpenNebula Server)

Prerequisites configuration

- **ruby** >= 1.8.6 and < 1.9.0
- **sqlite3** >= 3.5.2
- **xmlrpc-c** == 1.06
- **openssl** >= 0.9
- **ssh**

- Install all above packages on front end open nebula server

All steps to be executed at console/command prompt

STEP-1

- Install Ubuntu Desktop 11.04
- Install required packages at all node
- Connect all the nodes in one LAN

2- Download and install the latest Xen:

root@lab213: cd /usr/src

hg clone -r 4.0.0-rc8 http://xenbits.xensource.com/xen-unstable.hg

root@lab213: cd xen-unstable.hg

root@lab213: make xen

root@lab213: make tools

root@lab213: make stubdom

root@lab213: make install-xen

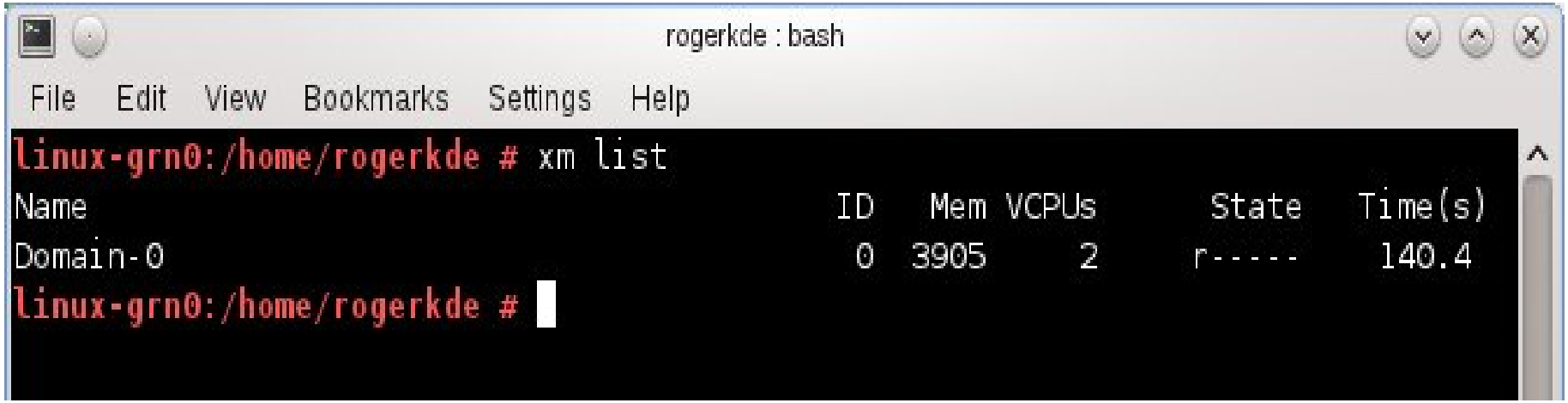
root@lab213: make install-tool

root@lab213: make install-stubdom

root@lab213: update-rc.d xend defaults 20 21

3- Reboot into your new dom0!

- Reboot in to the new xenified kernel

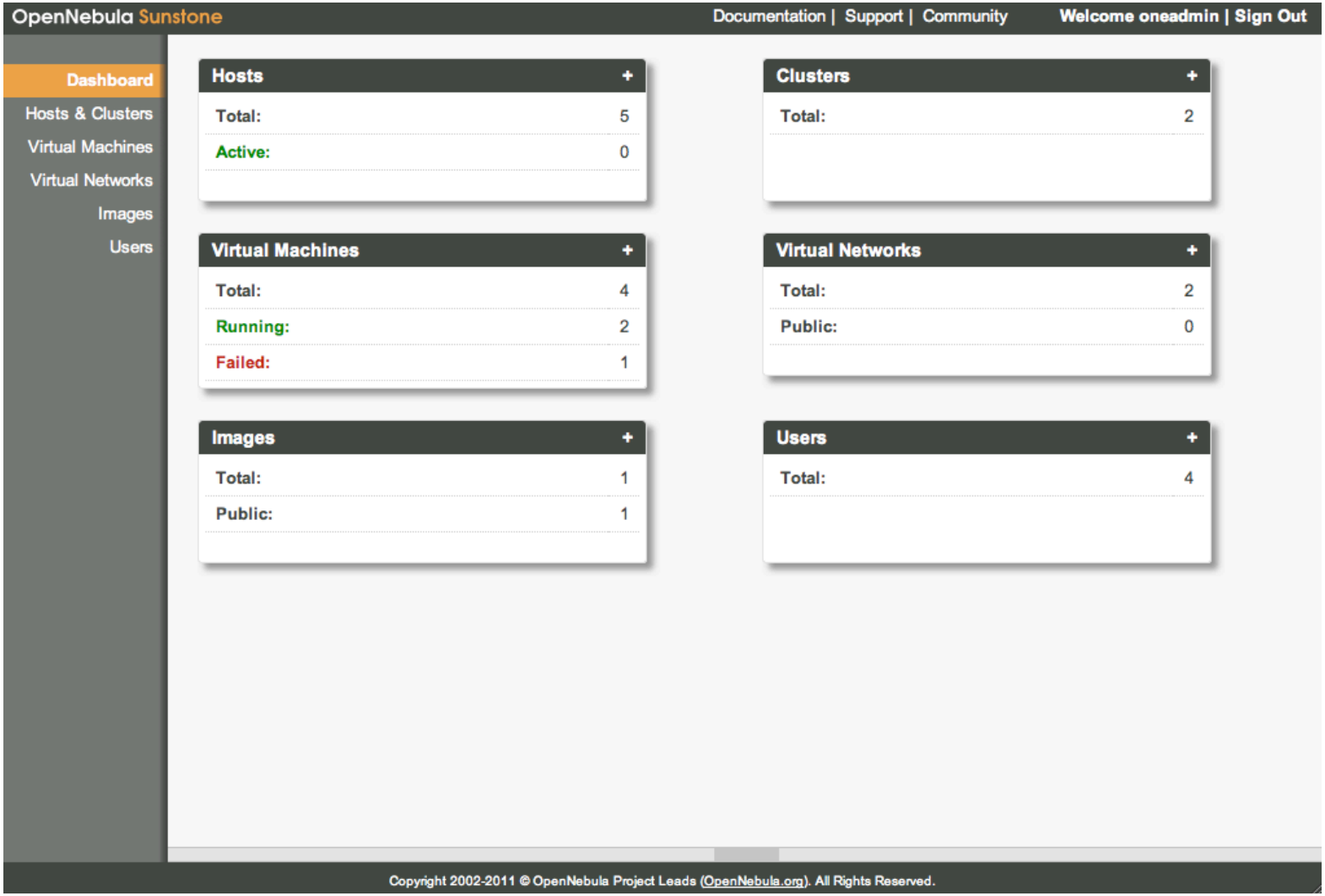


Install OpenNebula On Server Node

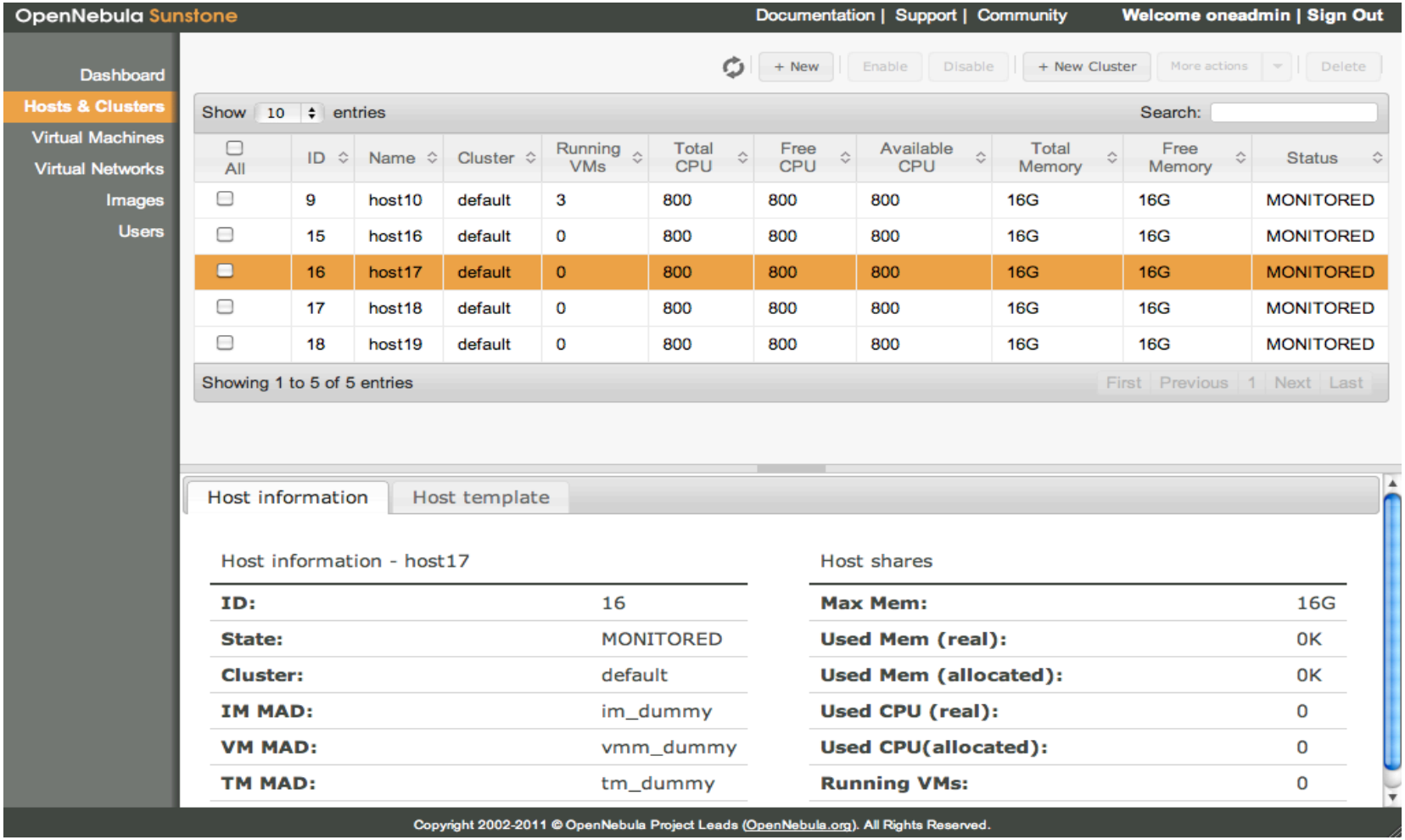
- Download and install
- Configure it with all xen nodes
- Install some management tools to control virtual machines

Management Consol Screen.

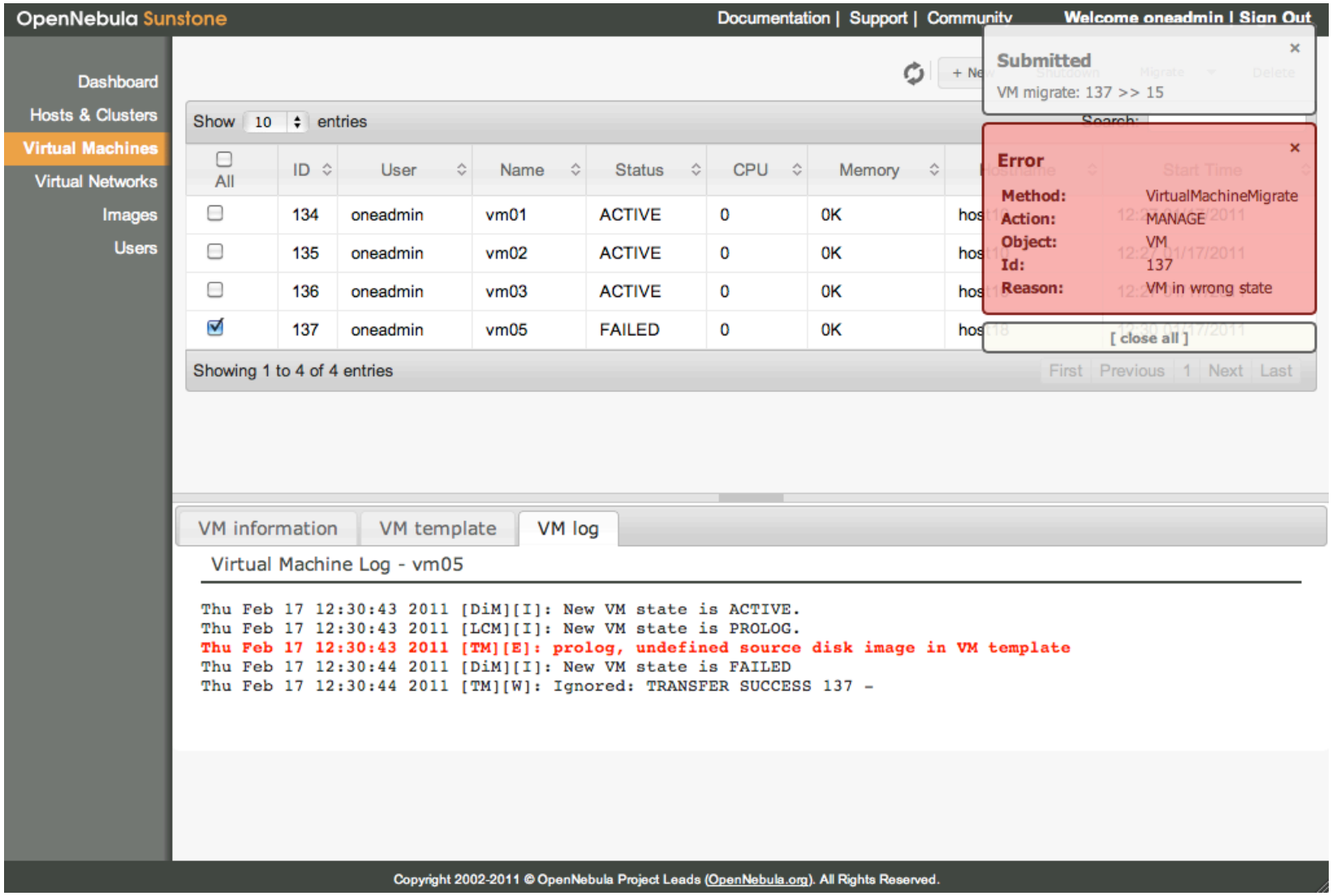
Dashboard


OpenNebula Sunstone
Documentation | Support | Community
Welcome oneadmin | Sign Out
Dashboard
Hosts & Clusters
Virtual Machines
Virtual Networks
Images
Users
Hosts
Total: 5
Active: 0
Clusters
Total: 2
Virtual Machines
Total: 4
Running: 2
Failed: 1
Virtual Networks
Total: 2
Public: 0
Images
Total: 1
Public: 1
Users
Total: 4
Copyright 2002-2011 © OpenNebula Project Leads (OpenNebula.org). All Rights Reserved.

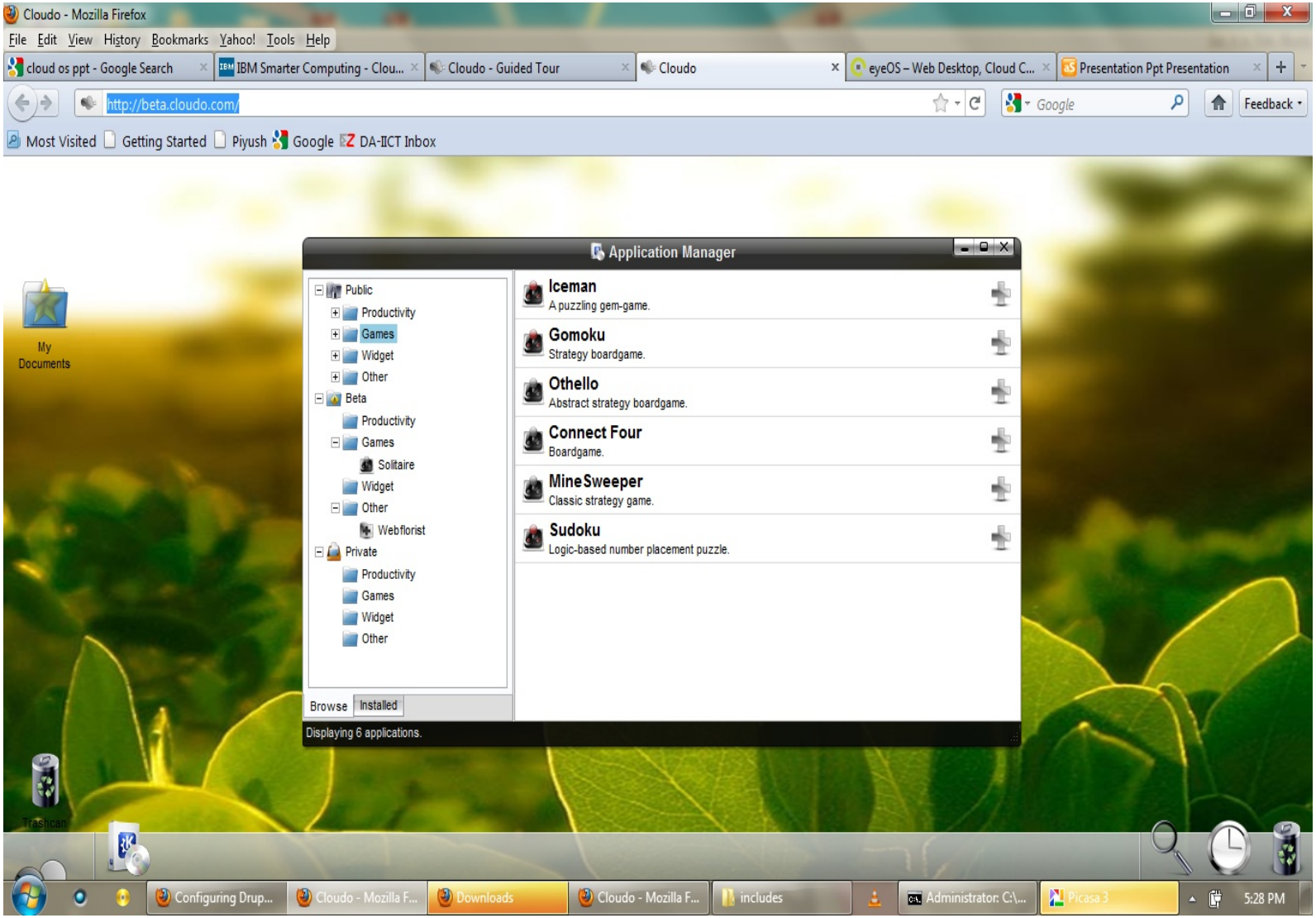
Hosts and Clusters



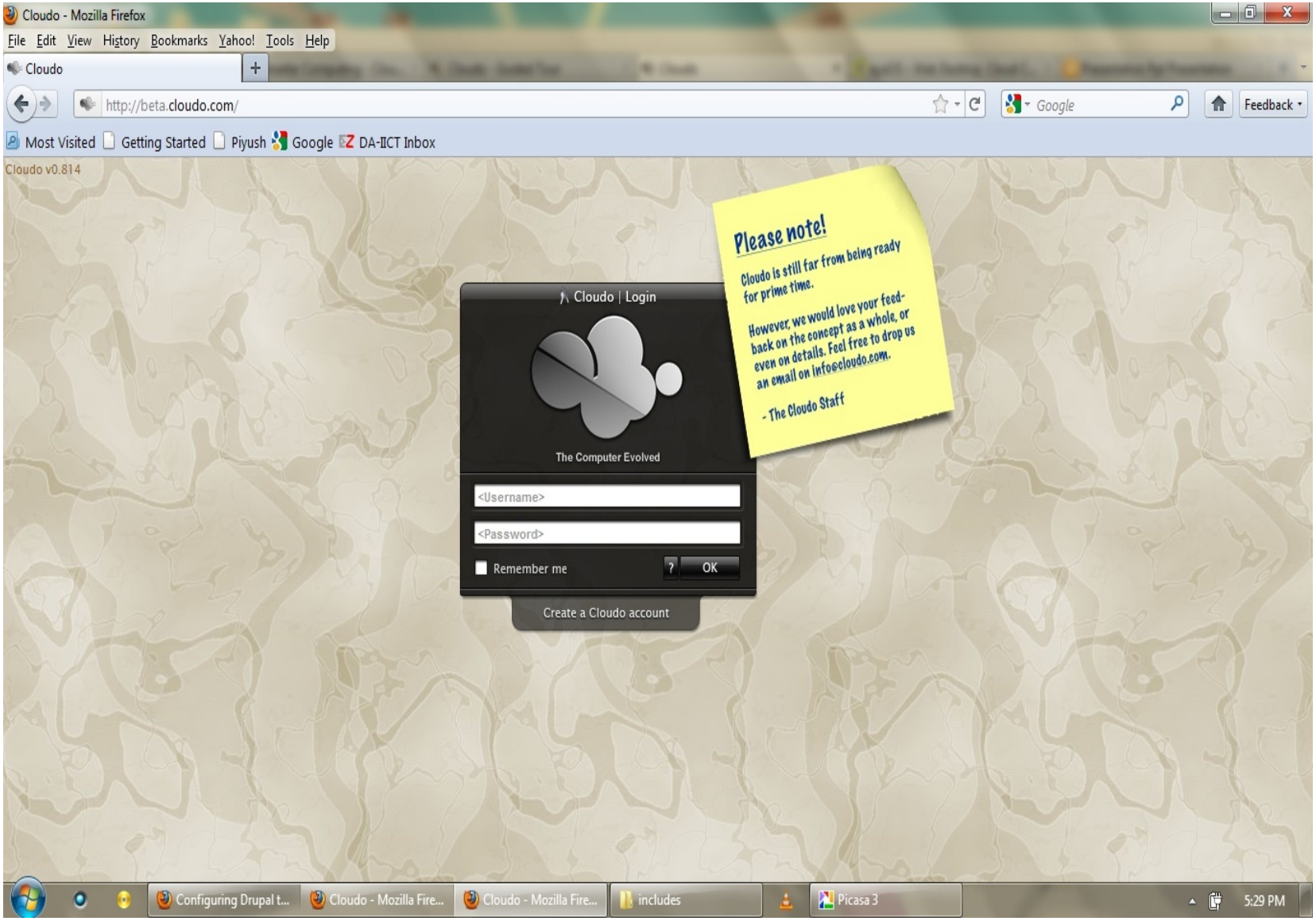
Created Virtual Machine

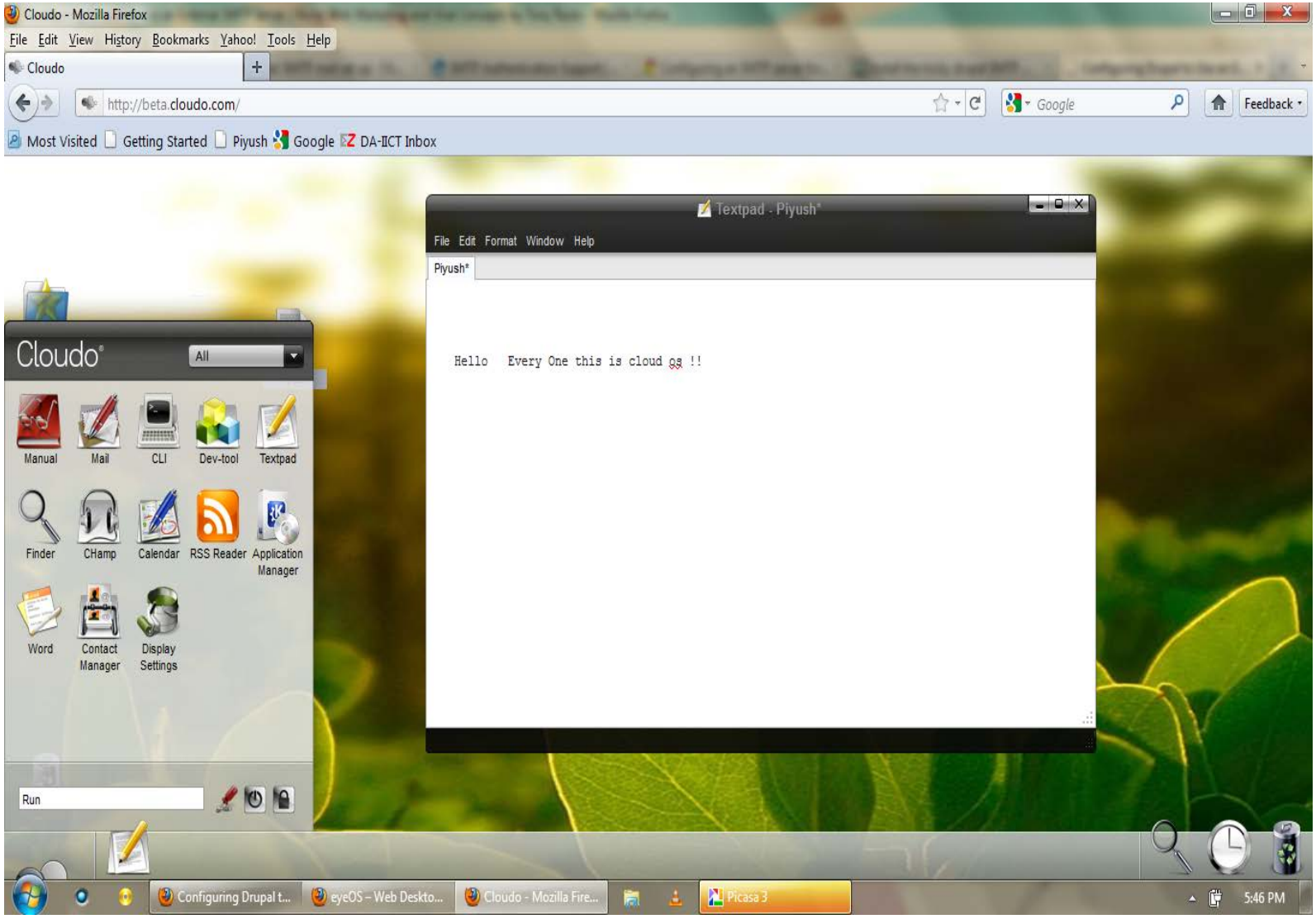


Web Based Cloud OS Screen



http://www.cloudo.com/




Cloudo - Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
Cloudo
http://beta.cloudo.com/
Google
Feedback
Most Visited
Getting Started
Piyush
Google
DA-IICT Inbox
Textpad - Piyush*
File Edit Format Window Help
Piyush*
Hello Every One this is cloud os !!
Cloudo
All
Manual
Mail
CLI
Dev-tool
Textpad
Finder
CHamp
Calendar
RSS Reader
Application Manager
Word
Contact Manager
Display Settings
Run
Configuring Drupal t...
eyeOS – Web Deskto...
Cloudo - Mozilla Fire...
Picasa 3
5:46 PM

References

[1] Gerald J. Popek and Robert P. Goldberg, "Formal Requirements for Virtualizable Third generation Architectures, *Communications of ACM*, Volume 17, Number 7, 1974.

[2] Jeanna N. Matthews, Eli M. Dow, Todd Deshane, Wenjin Hu, Jeremy Bongio, Patrick F. Wilbur, and Brendan Johnson. *Running Xen: A Hands-On Guide to the Art of Virtualization. Prentice Hall PTR*, Upper Saddle River, NJ, USA, 2008.

[3] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. *In SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 164–177, New York, NY, USA, 2003. ACM.

[4] David Chisnall. *The Definitive Guide to the Xen Hypervisor (Prentice Hall Open Source Software Development Series).* Prentice Hall PTR, Upper Saddle River, NJ, USA, 2007.

[5] Ludmila Cherkasova and Diwaker Gupta and Amin Vahdat, "Comparison of the three CPU schedulers in Xen, *ACM SIGMETRICS Perform. Eval. Rev.*, Volume 35, Number 2, 2007.

[6] Diego Ongaro, Alan L. Cox, and Scott Rixner. Scheduling i/o in virtual machine monitors. *In VEE '08: Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, pages 1–10, New York, NY, USA, 2008. ACM.

[7] Paul R. Barham, Boris Dragovic, Keir A. Fraser, Steven M. Hand, Timothy L. Harris, Alex C. Ho, Evangelos Kotsovinos, Anil V.S. Madhavapeddy, Rolf Neugebauer, Ian A. Pratt, Andrew K. Wareld, "Xen 2002", *Technical Report, University of Cambridge Computer Laboratory*, 2003.

[8] Jeanna Neefe Matthews, Wenjin Hu, Madhujith Hapuarachchi, Todd Deshane, Demetrios Dimatos, Gary Hamilton, Michael McCabe, and James Owens. Quantifying the performance isolation properties of virtualization systems. *In ExpCS '07: Proceedings of the 2007 workshop on Experimental computer science*, page 6, New York, NY, USA, 2007. ACM.

[9]Ludmila Cherkasova and Rob Gardner. Measuring cpu overhead for i/o processing in the Xen virtual machine monitor. *In ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference,* pages 24– 24, Berkeley, CA, USA, 2005. USENIX Association.

[10] www.opennebula.org

[11] Dongsung Kim, Hwanju Kim, Myeongjae Jeon, Euiseong Seo, and Joonwon Lee. Guest- aware priority-based virtual machine scheduling for highly consolidated server. *In Euro-Par'08: Proceedings of the 14th international Euro-Par conference on Parallel Processing*, pages 285–294, Berlin, Heidelberg, 2008. Springer-Verlag.

[12] Bin Lin and Peter A. Dinda. Vsched: Mixing batch and interactive virtual machines using periodic real-time scheduling. *In SC '05: Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, page 8, Washington, DC, USA, 2005. IEEE Computer Society.

References

[13] Clark, C., Fraser, K., Hand, S., Hansen, J. G., Jul, E., Limpach, C., Pratt, I., and Warfield, A. 2005. Live migration of virtual machines. In *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2* (May 02 - 04, 2005). USENIX Association, Berkeley, CA, 273-286.

[14] VMWare White papers "Understanding *Full* Virtualization, *Paravirtualization,* and Hardware Assist" , http://www.vmware.com/ /files/pdf/VMware_paravirtualization.pdf

[15] http://www.eetimes.com/design/automotive-design/4016811/Embedded-systems-virtualization-Consider-a-Hypervisor

[16] http://opensourceexperiments.files.wordpress.com/2008/04/my-network-diagram.jpg

[17] http://blogs.gartner.com/david_cappuccio/2009/02/15/data-center-efficiency-beyond-pue-and-dcie/

[18] Gaurav Somani and Sanjay Chaudhary. 2009. Application Performance Isolation in Virtualization. In *Proceedings of the 2009 IEEE International Conference on Cloud Computing* (CLOUD '09). IEEE Computer Society, Washington, DC, USA, 41-48.

[19] http://www.intel.com/pressroom/kits/pentiumee/index.htm

[20] http://www.omega-cottage.co.uk/articles/networks/ethernet/build_your_own_network_pt1.html

[21] http://gallery.techarena.in/data/516/medium/Device_and_IO_virtualization.JPG

[22] http://www.upgradecomputermemory.com/images/products

[23] N. Bobroff, A. Kochut, and K Beaty, Dynamic Placement of Virtual Machines for Managing SLA Violations, In Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Management (IM2007), pp. 119-128, 2007.

[24] Autonomic Virtual Machine Placement in the Data Center Chris Hyser, Bret McKee, Rob Gardner, Brian J. Watson HP Laboratories HPL-2007-189

Platform as a Service (PaaS)

Outline

- **PaaS**
  - What is PaaS..?
  - Traditional model v/s PaaS model
  - Advantages & Disadvantages of PaaS
  - PaaS Providers
- **DaaS**
  - What is Relational database..?
  - Problem with relational database
  - Features of non- relational database
  - Cloud database Providers

What is PaaS..?

- PaaS provides a full or partial development platform for which development tool itself will be over the cloud.
- With PaaS, developers can:
  - **build web applications** without installing any tools on their computer.
  - **deploy those applications** without any specialized systems administration skills.

Traditional Model
Vs
PaaS Model

Traditional Model

- Building and running on-premise applications have always been **complex and expensive**.
- Each application requires:
  - Hardware
  - Database
  - Web Servers etc.

- Once the stack was assembled,
  - A team of developers had to navigate complex programming models like J2EE and .NET.
  - A team of network, database, and system management experts were needed to keep everything up and running.

Problem with Traditional Model

- Any **change in business requirement** will require a change to the application, which again starts lengthy development, test, and redeployment cycle.
- Software and hardware **upgrades** must be managed.
- Data must be **replicated** at different data centers so that it can be restored in case of disaster.

PaaS Model

- All the infrastructure to run applications will be over the Internet.
- Developers do **not need to worry** about the storage or hosting.
- Developers write the code and the PaaS provider uploads that code & present it on the internet.
- The PaaS provider manages upgrades, patches, and other routine system maintenance.

Advantages of PaaS

- **Simplified Deployment**

  Developers can focus on development and innovation without worrying about the infrastructure.

- **Prebuilt Business Functionality**

  Some PaaS vendors also provide prebuilt business functionality so that users can avoid building everything from scratch hence helping jump-start projects.

- **Lower Risk**

  No requirements of up-front investment in hardware and software. Developers only need a PC and an Internet connection to start building applications.

- **Instant Community**

  PaaS vendors frequently offer online communities where developers can get ideas, share best practices, and seek advice from others.

- **Pay-per-use model**

  We have to pay only for what we use so there will be reduction in cost.

- **Scalability**

  Applications deployed can scale from one to tens of thousands of users without any changes to the application.

Disadvantages of PaaS

- **Vendor lock-in**

  One have to write applications according to the platform provided by PaaS provider so migrating an application to other PaaS provider would be a problem.

- **Data Privacy**

  Corporate data, whether it can be critical or not, will be private so if it is not located within the walls of the company there can be risk in terms of privacy of data.

- **Integration with the rest of the systems applications**

  It may happen that some applications are local and some are in cloud. So there will be increased complexity when we want to use data which is in cloud with the local data.

Windows Azure

AppHarbor

cloudControl
web - application - platform

LONGJUMP

phpfog

Google
app engine

CloudBees

Google App Engine

- To build application in GAE, you set up an account and get access to the App Engine Software Development Kit (SDK).
- Since this is Google , the platform does not support Microsoft Technologies…!!
- Google App Engine mainly two programming languages:
  1. App Engine's **Java** runtime environment includes standard Java technologies like JVM, Java Servlets, and the Java language.
  2. App Engine's **Python** runtime environment includes a fast Python interpreter and the Python standard library.

Cont…

- **Google Datastore:** App Engine provides a distributed data storage service which features a query engine and transactions.
- **GQL:** App engine also provides query language for querying data.
- **Intended for small applications:** so any request that takes longer than 30 seconds to complete or that sends more than 10MB of data is immediately terminated.

Microsoft Azure

- Azure is Microsoft's equivalent of the GAE so it is slightly more focused on the use of **.NET components.**
- Microsoft Azure mainly provides three components:
  1. **Windows Azure** provides a Windows based environment to develop applications on Microsoft's data centers.
  2. **SQL Azure** provides access to a relational database hosted at Microsoft's' data centers.
  3. **.NET Services** provide access to applications running on the cloud.
- Azure also supports non-Microsoft development languages including **PHP and Python**.

Sales**force.com**

- The force.com allows users to access a application development and execution platform from a browser.
- The force.com provides support for language **Apex**. Apex has Java like syntax. But it is not possible to run any Java or .NET programs on the force.com platform.
- According to Salesforce, more than **100,000 applications** have been built on Salesforce platform.

Google App Engine Demo

What do you need…?

- Google Account
- Google App Engine SDK and Web Tool Kit

  http://code.google.com/appengine/downloads.html#Google_App_Engine_SDK_for_Java
- Eclipse IDE (3.5, 3.6 or 3.7)

  http://www.eclipse.org/downloads/
- Google Plugin for Eclipse

  http://code.google.com/eclipse/
- Little bit knowledge of JAVA

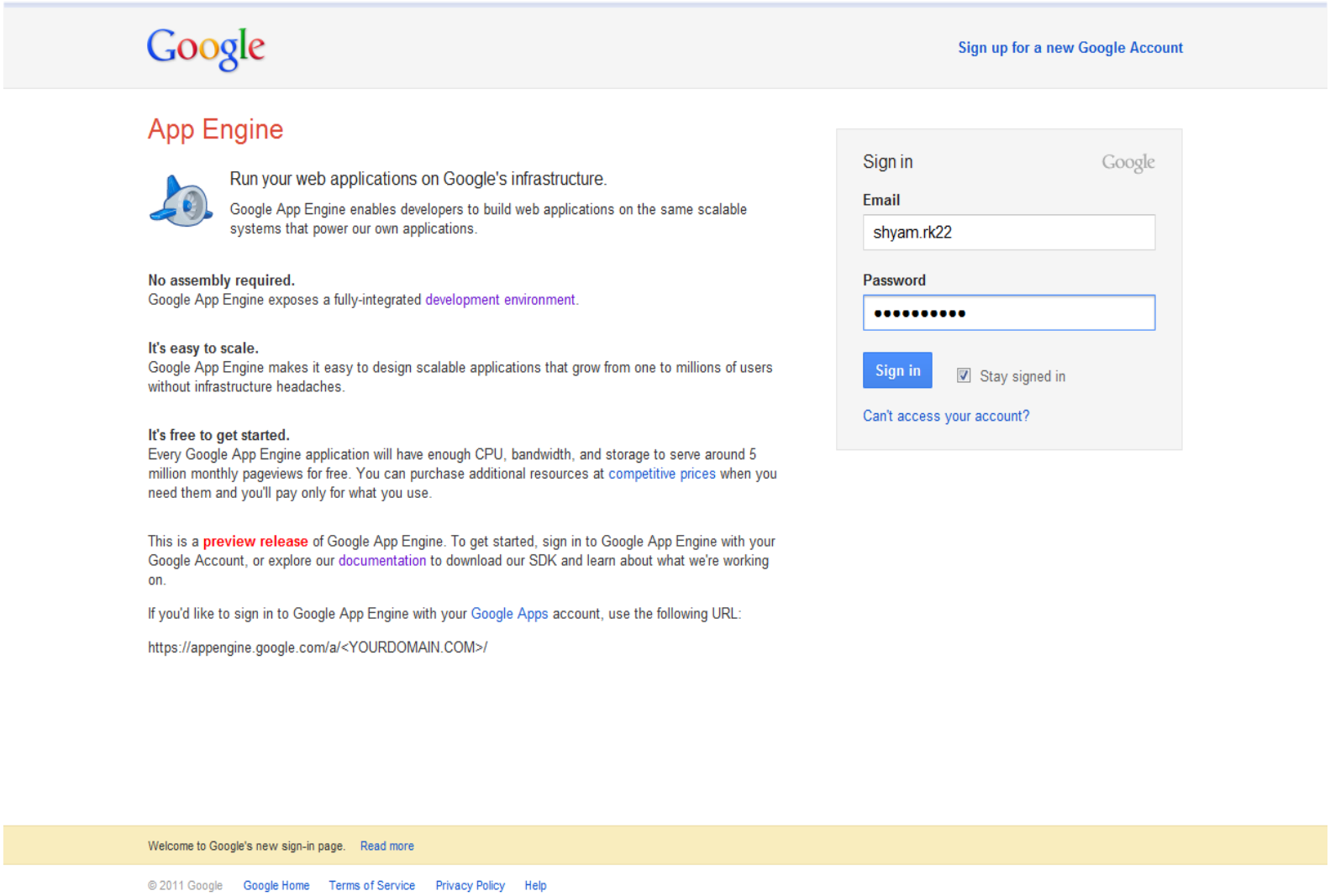
  Go to Library…☺

How to Start…?

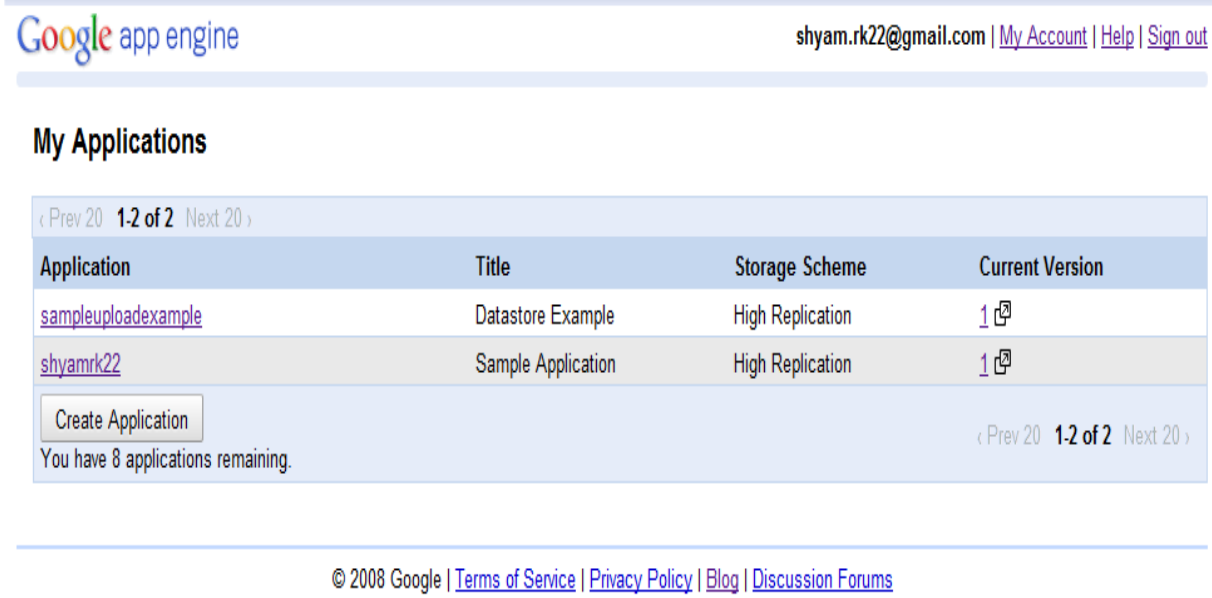
- Start making a sample application

  http://code.google.com/appengine/docs/java/gettingstarted/creating.html
- Run it on local host
- If application working properly, deploy it on Google.
- Give URL name to your application and It's done…
- Now your application can be accessed anywhere, anytime through internet…☺

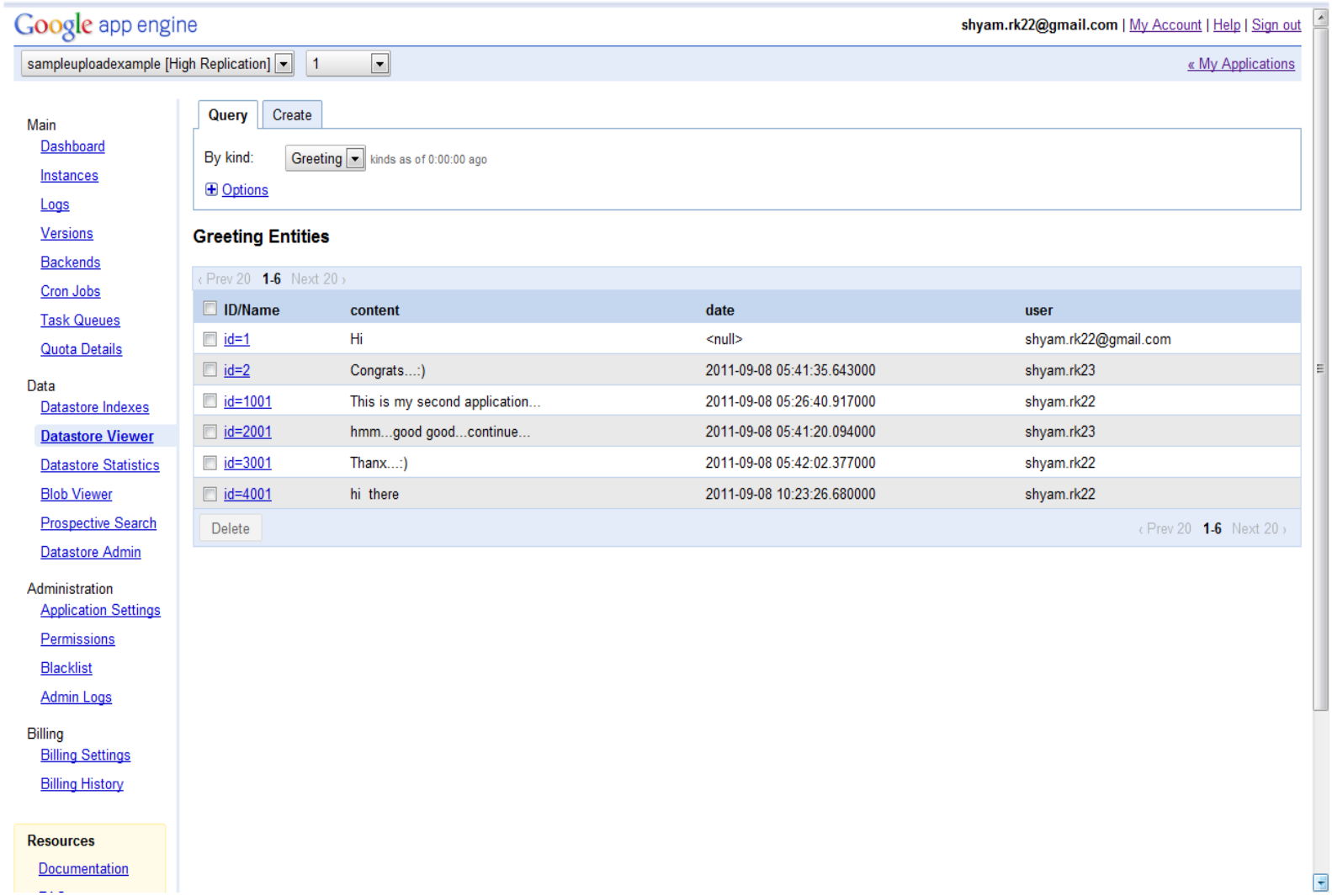
appengine.google.com



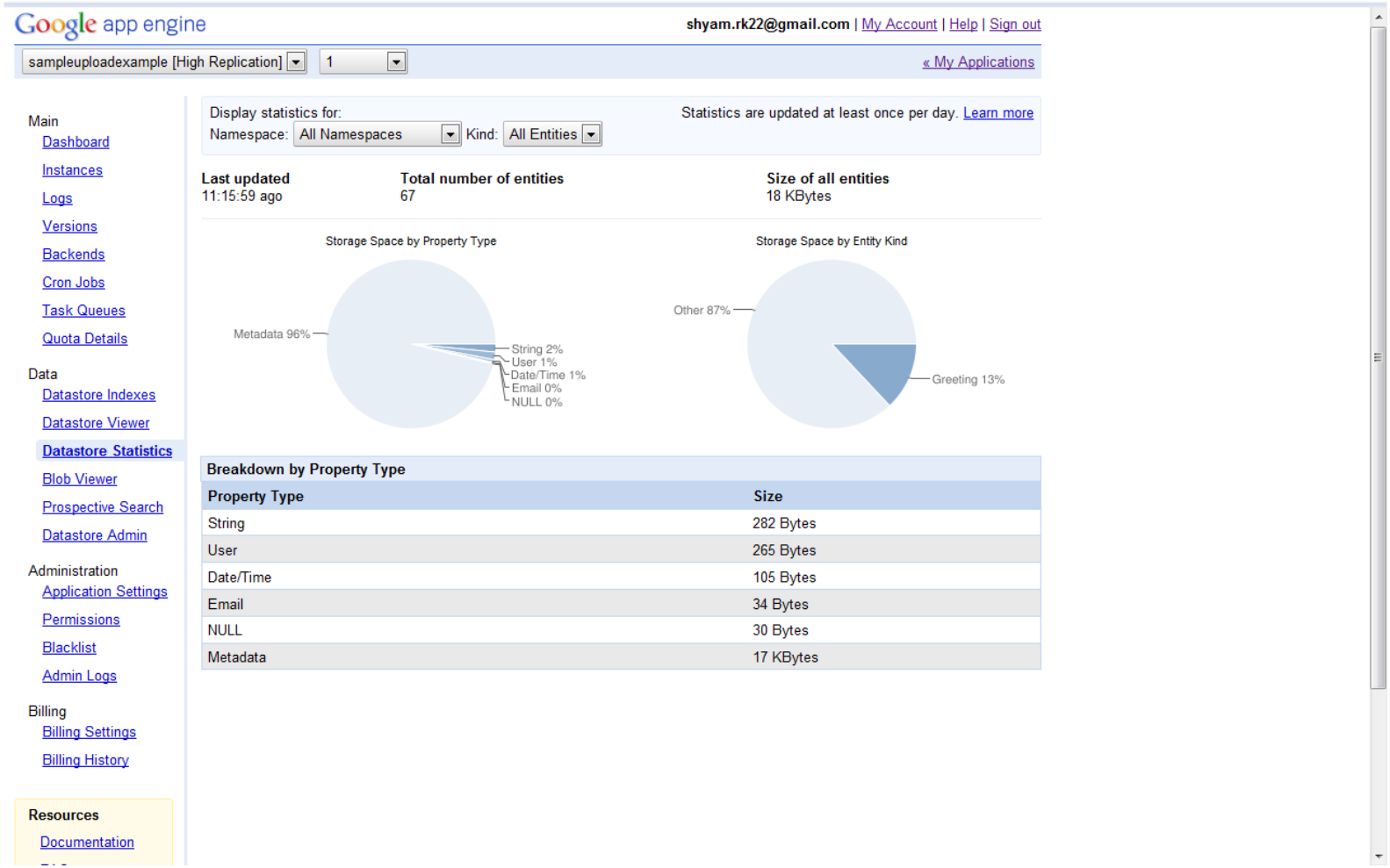
Applications



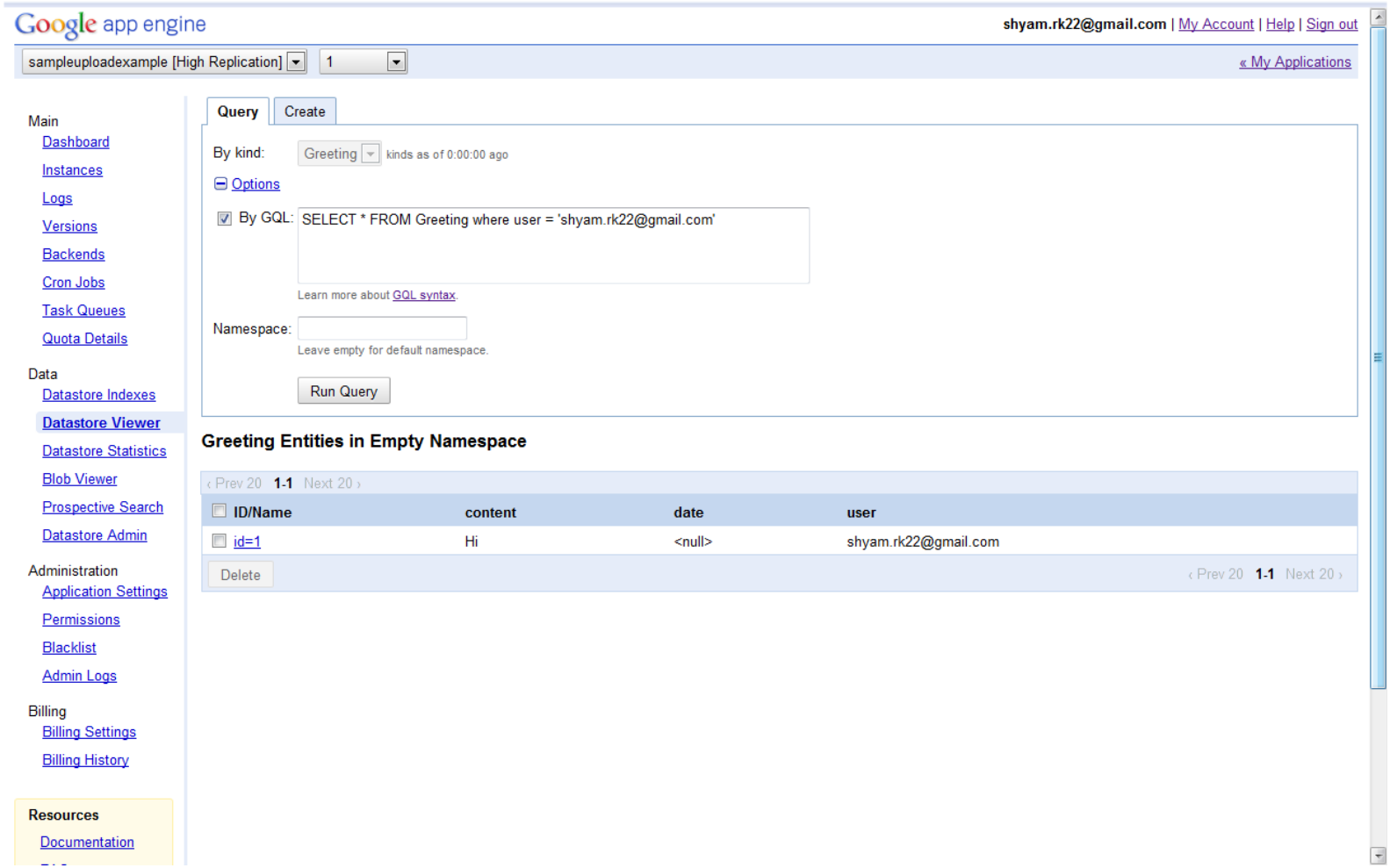
Google Datastore



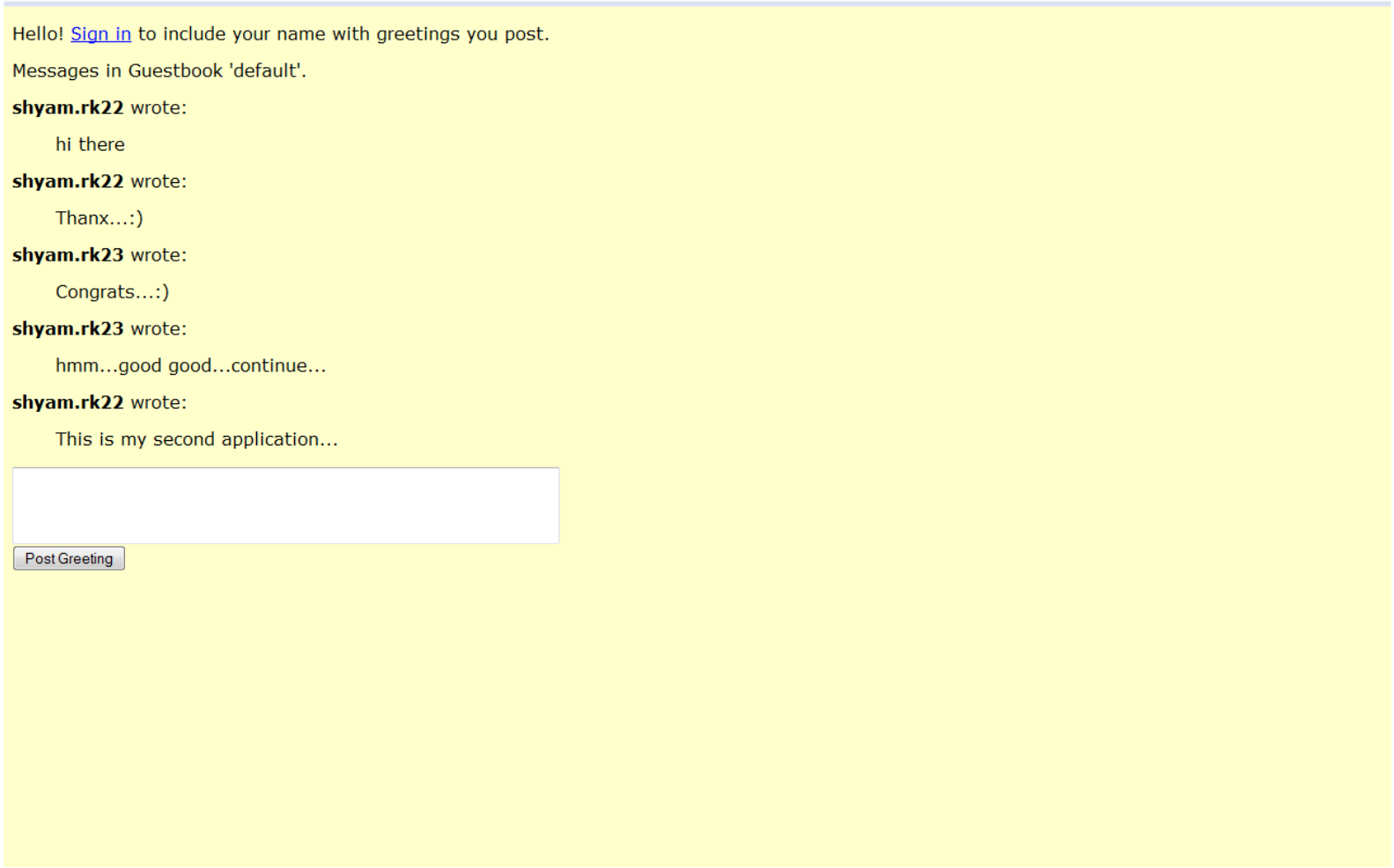
Datastore Statistics



GQL query


Google app engine
shyam.rk22@gmail.com | My Account | Help | Sign out
sampleuploadexample [High Replication]
1
« My Applications
Main
Dashboard
Instances
Logs
Versions
Backends
Cron Jobs
Task Queues
Quota Details
Data
Datastore Indexes
Datastore Viewer
Datastore Statistics
Blob Viewer
Prospective Search
Datastore Admin
Administration
Application Settings
Permissions
Blacklist
Admin Logs
Billing
Billing Settings
Billing History
Resources
Documentation
Query
Create
By kind:
Greeting
kinds as of 0:00:00 ago
Options
By GQL:
SELECT * FROM Greeting where user = 'shyam.rk22@gmail.com'
Learn more about GQL syntax.
Namespace:
Leave empty for default namespace.
Run Query
Greeting Entities in Empty Namespace
‹ Prev 20 1-1 Next 20 ›
ID/Name
content
date
user
id=1
Hi
<null>
shyam.rk22@gmail.com
Delete
‹ Prev 20 1-1 Next 20 ›

Application



Application

Hello, shyam.rk22! (You can sign out.)

Messages in Guestbook 'default'.

shyam.rk22 wrote:

hi there

shyam.rk22 wrote:

Thanx...:)

shyam.rk23 wrote:

Congrats...:)

shyam.rk23 wrote:

hmm...good good...continue...

shyam.rk22 wrote:

This is my second application...

Post Greeting

Application

Application