Implementation of a Private Cloud: A Case Study

Prachi Deshpande, S. C. Sharma and S. K. Peddoju

Abstract Availability of Cloud environment setup is sometimes difficult at personal level for a researcher, especially a beginner. In this case, open source tools, softwares can help a great deal to build and deploy a private cloud. Deploying cloud for research purpose is quiet time consuming due to scattered information and variety of options. In this paper, an attempt has been done for providing collective information of various measures to deploy a private cloud. In-depth analysis of different real-time errors during installation of a cloud setup along with solutions is also reported in this paper. This attempt will certainly helpful in minimizing the time and efforts to setup the cloud and makes deployment easier. Open source operating system (OS) Ubuntu-12.04 long time support (LTS) and OpenNebula cloud computing framework are used for deployment of this private cloud.

Keywords Cloud · Deployment · Hypervisor · KVM · OpenNebula · Open source · VM

1 Introduction

Rapid information processing and access is becoming need of the day. Evolution of cloud computing has its roots to fulfill this ever increasing demand of processing information in real-time manner. Cloud consists of a cluster of resources,

P. Deshpande (✉) · S. C. Sharma · S. K. Peddoju
Indian Institute of Technology Roorkee, Roorkee 247667, Uttarakhand, India
e-mail: deprachi3@gmail.com

S. C. Sharma
e-mail: scs60fpt@gmail.com

S. K. Peddoju
e-mail: prof.sateesh@gmail.com

services, and technologies distributed over network. Principle aim of cloud computing is to reduce the cost of service usage, infrastructure development and maintenance, availability and very importantly security in data processing [1]. According to [2], cloud is a parallel and distributed computing system consisting of a collection of inter-connected 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. According to National institute of standards and technology (NIST) [3], cloud computing is a model for enabling ubiquitous, 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.

Cloud computing is a assimilation of features of different technologies and standards like virtualization of resources, web services and architectures, distributed computing, and data center automation and autonomic computing [4]. It inherits some of its characteristics from mainframe computer, client–server model, grid computing, utility computing, peer-to-peer [5] computing and autonomic computing [2, 6]. In the infrastructure as a service (IaaS) layer of cloud, basic services like computers (physical or virtual), servers, virtual local area networks, firewalls, file based and raw storage are provided to user.

The users need to pay as per its utilization. 'Amazon elastic compute (EC) 2', 'HP cloud', 'Rackpace' are some of the IaaS providers. Computing platform like OS and programming language platform is provided to users via platform as a service (PaaS). 'Cloud Fondry', 'Google App Engine' and 'Jelastic' are some of the PaaS providers. Different application development is made possible by virtue of web servers. In software as a service (SaaS) layer, cloud providers install and operate application software in the cloud. Users can access the software from cloud clients [2]. Based on coverage area and distribution, the cloud deployment models are classified as a private cloud, public cloud, hybrid cloud, community cloud. Many organizations when share the capabilities of cloud, it is referred as community cloud [4]. For research purpose, a private cloud is sufficient to carry out the experimentations. The basic information and various tools for a cloud setup are discussed in Sect. 2. Installation of cloud setup is conversed in Sect. 3 along with its trouble shooting aspect. The performance of the cloud setup is presented in Sect. 4.

2 Backbone for a Private Cloud Installation

This section provides off-the-shelf information for the installation of a private cloud. For research purpose, private cloud has to be built by using available open source platforms. These platforms are acting as a tool for managing the cloud resources and services. The key virtualization technology in a cloud setup is a hypervisor. Hypervisor is a software which is used in between the hardware and OS for virtualization purpose. It allows hardware virtualization so that multiple virtual machines (VM) can run on a single host. Memory management, CPU scheduling are also performed by hypervisor. This virtualization technology falls into different categories as full virtualization, paravirtualization, and hardware assisted virtualization [4]. Few of the hypervisors available in the market are discussed below. From the feature comparison, one can decide about the suitability of a specific hypervisor according to working environment and desired results.

- *XEN'*—It allows separation of the domains which lead to strong isolation and security. The OS acting as domain0 (Dom0) has direct communication with hardware of host. Dom0 manage and launch domainUser (DomU) which is a guest OS. Network management is based on 'First-in-first-Out (FIFO)' scheduling. XEN can provide paravirtualization and full virtualization. 'Dom0' and 'DomU' structure makes XEN different from other hypervisors due to its isolation property. XEN can be compared with kernel virtual machine (KVM) when hardware VM with XEN is provided, as it is similar to KVM, which provides full virtualization [7].
- *KVM*—It is an open source product by 'RedHat'. It is a module that can be used for converting Linux kernel into a hypervisor. It has features like live migration of VM, support for different types of guest OS, isolation policy decision for VM by administrator to provide security, and hardware devices having Linux support.
- *VMware VSphere*—VSphere is the product by VMware. The main features of VSphere are distributed locking, supported with sharing cell, load balancing based on CPU topology, disk management using latency aware priority based scheduler for network management and priority based network input output control and distributed switches [8].

Different parameters can be used to compare hypervisors to choose among available options. KVM is the solution for virtualizing Linux kernel and it provides full virtualization by direct communication with hardware. KVM is opted in this work as there is no performance issues of host are observed.

Major issues in managing the cloud resources are ensuring the availability of resources on demand and load balancing while providing services to the users [9]. There are different types of virtual infrastructure (VI) managers available for this purpose such as:

- **Eucalyptus**: This is the first open-source framework provided for building 'IaaS' clouds. It has compatibility with 'Amazon EC2' and has XEN, KVM, VMware back ends, virtual networking, and EC2 compatible command line interface (CLI) and Web portal interfaces [10].
- **OpenNebula:** It is the most famous cloud setup open source platform. Four application programming interface (API) available with it, which makes it different from other VI service providers. These APIs are 'XML-RPC' and 'Libvirtd' for local interaction, 'EC2 (Query)' APIs and the OpenNebula cloud

API (OCA) for remote operation purpose. It has compatibility for XEN, KVM, and VMware hypervisors. Also it has powerful security management, virtual networking compatibility with other open source products like open Vswitch, dynamic resource allocation and VM migration [11].

- **oVirt:** It is a virtualization management application and similar in concept with VSphere of VMware. 'oVirt' management interface can manage hardware nodes, storage and network resources, and deployment and monitoring of virtual machines running in data center. It can be built with KVM hypervisor and on the 'RHEV-M' management server released by RedHat, live migration of VM, user and group based security solutions are supported [12].
- **Apache cloud stack:** It is a project of Apache software foundation (ASF) open source software for deploying public and private infrastructure-as-a-service (IaaS) clouds. Its features include compatibility with hosts running XEN Server/ XCP, KVM, and/or VMware 'ESXi' with VSphere. It has 'Amazon S3/EC2' compatible API. Network services like firewall, NAT, VPN are provided. Web based user interface for managing cloud is also provided by this platform [13].
- **Nimbus:** It is a open-source toolkit focused on providing IaaS capabilities. Nimbus provides Linux-based controller, EC2-compatible, modeling and accessing stateful resources with web services resource framework (WSRF) interfaces; XEN and KVM backend; interface to the Amazon EC2 public cloud and virtual networks. Table 1 provides a brief summary of capacities of different open source platforms [14].

2.1 Private Cloud Setup

In the present work, to build a private cloud infrastructure, the open source software 'OpenNebula' is used. The particular choice is due to its features like:

(a) Deployment can be done with almost all Linux distributions.
(b) Physical infrastructure management with monitoring and control facilities like creation and deletion of physical hosts and VM, setting and monitoring intervals, VM instance management etc.
(c) Availability can be achieved with the support of hooks. They can notify error state of host so that the VMs on failed host can be redeployed on another host. Persistent backend keeps OpenNebula daemon safe even after crash for recovery.
(d) User security management through *'auth'* subsystem for managing users, additional support of *SSH*, and X.509.
(e) It has a great deal of adaptability to manage any hardware and software combination along with integrating capacity with any other cloud product.

Figure 1 shows a private cloud setup using OpenNebula. One has to use the OpenNebula facilities to create and manage their own VM and virtual networks. It has following components:

Table 1 Open source platform for cloud setup

Cloud software (IaaS)	Virtual network	Interface to public cloud	Hypervisors compatibility	Security options	Dynamic resource allocation
Eucalyptus	Yes	EC2	XEN, KVM	Firewall filters (security groups), SSH and WS-security	Yes
OpenNebula	Yes	Amazon EC2	XEN, KVM, VMware	Authentication mechanisms and access control lists, firewall	Yes
oVirt	Yes		KVM	Supports Red Hat directory server, Free IPA or microsoft active directory for user and administrator authentication	Yes
Apache cloud stack	Yes	AWS EC2, S3	VMware, KVM, Citrix XEN server, and Citrix XEN cloud platform	Secure single sign-on, isolation of user for managing resources, firewall	Yes
Nimbus	Yes	EC2	XEN, KVM	Public key based authentication and authorization. Use of globus certificate credentials	Require integration with OpenNebula

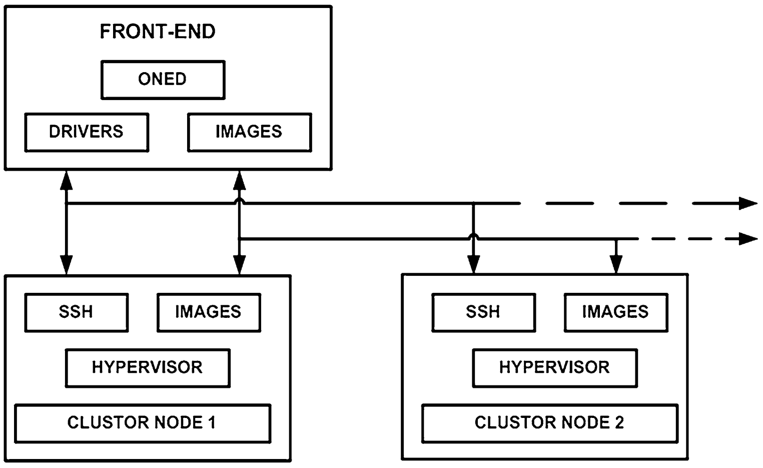


Fig. 1 OpenNebula cloud architecture [11]

- **Front-end:** OpenNebula installation is performed on this machine and it must have access to image repository and network connection with cluster nodes.
- **Cluster nodes**: It is the host on which hypervisor is installed. It provides the resources to the VM.
- **Image repository:** It is a storage medium that holds base images of the VM.
- **OpenNebula daemon:** It is the core service of the system. It managing service called 'oned'.
- **Drivers:** These are programs used by the core to interface with a specific cluster subsystem, i.e. a given hypervisor or storage file system.
- **Oneadmin:** It is the administrator of the private cloud that performs any operation on the VM, virtual networks, nodes or users.

For effective use and deployment of a private cloud, one must be familiar with the internal architecture of OpenNebula. The details of the OpenNebula internal architecture can be found in [11]. The OpenNebula internal architecture is made-up of three layers: the uppermost layer is of tools provided with Open Nebula. The middle layer is known as the core. It consist of VM, host, and virtual network management components. The last layer is of drivers. Drivers are required to plug-in different virtualization technologies into core. The following section describes the important parameters from internal architecture of OpenNebula which is essential to be known for installation of a cloud setup.

2.1.1 Tools

This layer represents various tools available with OpenNebula. CLI is provided with OpenNebula to manually control the VI under 'ONE daemon'. For this

Table 2 Commands available in OpenNebula

Command	Interpretation
onevm	To submit, control and monitor VM
onehost	To add delete and monitor hosts
Onevnet	To add delete and monitor virtual networks

purpose various commands are available with OpenNebula. Few of them are listed in Table 2.

The second tool is scheduler. The scheduler is an independent entity in the OpenNebula architecture. It uses the XML-RPC interface provided by 'ONE daemon' to invoke actions on VM. The scheduler distributed in the technology preview provides the following functionality:

- *User-driven consolidation:* The scheduler allocates VM to those cluster nodes which congregate the capacity constraints of the VM.
- *Matchmaking policy:* The scheduler sort out resources which do not meet the constraints of the VM. Then a grading process is made to select the best Host. This is accomplished by pertaining the ranking expression defined in the VM template. Third party tools can also be created using XML-RPC API or OpenNebula-client API.

2.1.2 OpenNebula Core

The core consists of a set of components to control and monitor VMs and hosts. The core performs its actions by invoking suitable drivers. The functional components of OpenNebula core are summarized in Table 3.

2.1.3 Drivers

OpenNebula has a set of pluggable modules to interact with specific middleware (e.g. virtualization hypervisor, file transfer mechanisms or information services). These modules are known Drivers. Few important drivers of OpenNebula are listed out in Table 4.

3 Cloud Setup Using OpenNebula

OpenNebula can be used for cloud setup in two modes as:

(a) *System-wide mode:* In this mode OpenNebula-services like binaries and configuration files etc. are kept at standard locations under root file system.

Table 3 Functional units of OpenNebula core

Unit	Application
Request manager	To handle client request
VM manager	To manage and monitor VMs
Host manager	To manage and monitor physical resources
Database manager	Persistent storage for ONE data structure
	'Sqlite3' is the default database used
Transfer manager	To manage transfer operations i.e. file, VM
Virtual network manager	To manage virtual networks

Table 4 Drivers and functionality

Drivers	Functionality
Information driver	To support information management
Transfer driver	To support transfer operation
VM driver	To manage VM functionality

(b) *Self-contained mode:* In this mode one can place OpenNebula distribution in self contained form at a particular directory.

Once acquainted with the capacities of OpenNebula platform, one can move for the setup of an own private cloud. In this work, the cloud setup is carried out in self-contained mode. For this purpose, the required frontend and cluster machine preferences are as:

- Frontend- Ubuntu-12.04 Desktop LTS with opennebula-3.8.3
- Cluster node- Ubuntu-12.04 Desktop LTS with KVM hypervisor installed.

As various Linux distributions are available, the selection of OS for setup depends on ease of working with the OS and the desired features. Ubuntu has a user friendly GUI and supports most of the virtualization technologies like *'Libvirt'* and KVM. The OpenNebula private cloud installation can be carried out with following steps:

1. Connection of frontend and cluster in LAN.
2. Creation of user account *'oneadmin'* and folder *'/srv/cloud/one'*, at front end and cluster nodes.
3. Network file sharing in between frontend and cluster node: Ubuntu provides network file sharing facility by mounting the directory *'/srv/cloud/one'* that is created at frontend and at cluster node. At frontend side, add IP address of cluster and the options required for mounting a directory in *'/etc/exports'* file.
4. At cluster node side, add IP address of frontend and directory to be mounted in *'/etc/fsab'* file.

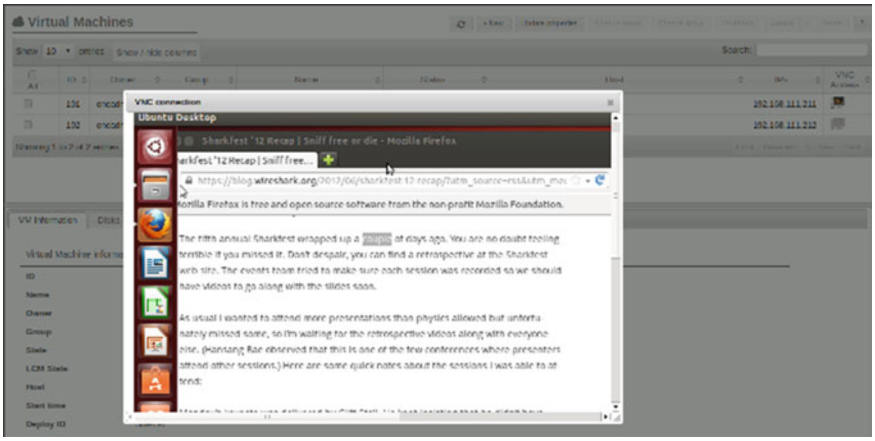
Table 5 Cloud installation and troubleshooting

Installation Setup	Errors	Solution
Network file sharing		
Front end side	Errors are not prominent for NFS	Set the parameters correctly to avoid errors
Cluster side	Bad syntax' error in *fstab* or *mtab* file	It occurs due to improper syntax specially spaces
	mount.nfs is unable to mount the directory	This error occurs due to arguments in '*fstab*' or '*mtab*' file
OpenSSH setup		
Frontend side	SSH cannot connect to host port 22, connection refused	Make sure '*sshd*' configuration file must have port 22 line uncommented
		Still error persists use '*netstat*' command to check if TCP port 22 is blocked
	After copying public key Command prompt asks to enter a password of cluster machine	The hostname and the IP address of the cluster machine can be added in '*/etc/hosts file*' of frontend
		Check permission of '*ssh*' directory
Open nebula Installation on frontend machine		
	Cannot create database or user	To check particular MYSQL version installed on machine and issue above commands according to MYSQL version compatible syntax on frontend
Host creation hurdles	Onehost list shows error status for host	Password less connection between frontend and host must be lost try to *SSH* host from frontend
		Try to ping host from frontend
VM deployment error	Unable to open disk path: permission denied	Add user '*oneadmin*' to disk group at cluster side
	Unable to create domain	Add user '*oneadmin*' to KVM group at cluster side

(continued)

Table 5 (continued)

Installation Setup	Errors	Solution
VM deployment error	Unknown OS type ‘*hvm*’	Check if KVM module is loaded using command ‘*lsmod |grep kvm*’
		Ensure correct loading of the module. If it is loaded then try ‘*modprobe kvm*’
		If same error persists then check if ‘*lsmod |grep kvm*’ returns ‘*kvm_intel*’. Still errors persist, then try *‘modprobe kvm kvm-intel’* at cluster side
GUI errors	*Novnc* icon active but does not show VM screen error [Failed to connect to server(1006)]	kill existing proxy
		You can get proxy details using command ‘*ps aux | grep websockify*’
		Relaunch proxy manually to get reason of error
		Tip: till you resolve the problem you can use virtual machine manager on host to see VM



lozilla Firefox is free and open source software from the non-profit Mozilla Foundation.

The fifth annual Sharkfest wrapped up a couple of days ago. You are no doubt feeling terrible if you missed it. Don't despair, you can find a retrospective at the Sharkfest web site. The events team tried to make sure each session was recorded so we should have videos to go along with the slides soon.

As usual I wanted to attend more presentations than physics allowed but unfortunately missed some, so I'm waiting for the retrospective videos along with everyone else. (Hansang Bae observed that this is one of the few conferences where presenters attend other sessions.) Here are some quick notes about the sessions I was able to attend:

Fig. 2 VM creation

5. After steps 1–4, mount *'/srv/cloud/one'* using command line at cluster node side.
6. Establishment of communication between front end and clusters using *'openSSH'* facility secure shell access through key generation is performed. Generation of the key at frontend and copying it to cluster node side allows passwordless entry into the host machine.
7. Installation of dependency softwares and packages in frontend as per requirement of OpenNebula version.
8. Installation of OpenNebula in the frontend.
9. Configuration of hypervisor in cluster nodes—Installing bridge utilities and changing *'libvirtd.conf'* and *'qemu.conf'* files.
10. Installation of 'Sunstone' graphical user interface [15].

Starting with the network file sharing, Table 5 provides possible errors with solutions to mitigate them for speedy cloud installation.

4 Results and Discussion

This paper provides the details of the available open source solutions for building private cloud. OpenNebula private cloud can be setup with the help of this paper by minimizing the possible errors with less time. In the present work, a P-IV system with 2-GB RAM at front end and 8-GB RAM at backend is used for cloud setup. Figure 2 depicts the functioning of cloud in terms of VM creation. It indicates that the OpenNebula cloud setup is successfully functioning. Internet can be accessed via the running VM for OpenNebula just like normal OS. The access to the outside world is provided to VM by 'Libvirt' bridge.

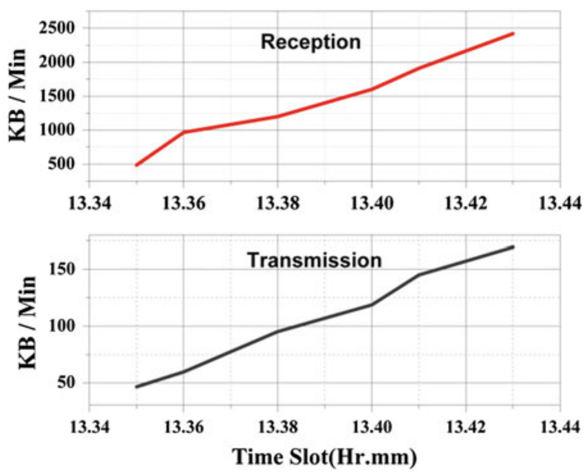


Fig. 3 Network transmission and reception rate for VM

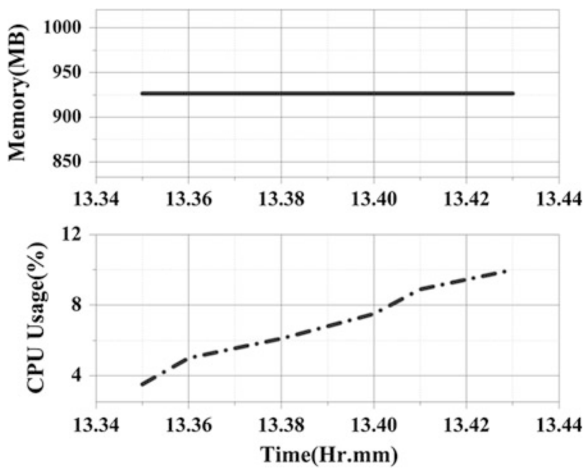


Fig. 4 CPU and memory usage of VM

The performance of the cloud setup has been verified by internet access through the VM. Initially the cloud setup is operated for a time slot of 10 min. Figure 3 shows the network transmission and reception for VM. The transmission and reception rate is relative with the corresponding increase or decrease of the network traffic. It indicates the successful deployment of cloud setup as the communication with the external world (internet) is carried out through VM.

Figure 4 shows the CPU usage by VM. It shows that CPU usage is varying with time slot. Also the maximum CPU usage is 12 % at its peak whereas memory of VM is constant as of allocated at the time of its creation. This indicates that host is dynamically adjusting the workload by using only 12 % CPU capacity and thus the remaining CPU and memory capacity can be made available to the other users (VM) on demand. This is the main difference between a cloud based system and a traditional system.

5 Conclusions

Parting the traditional information processing methodologies, cloud technology is growing rapidly. Hence it has got attention of researchers and academicians for its improvement. In this context, a private cloud set up is carried out using various open source tools. The present work highlights the requirements, procedure and tools for deployment of a private cloud. The practical solutions, based on the personal experience, to the various errors that arise during cloud set up are also listed out here. A KVM hypervisor is used here as it doesn't show any compatibility issues with the host. The performance of the cloud is verified by the cost functions such as transmission and reception rate, CPU usage and memory usage.

It shows that only 12 % of the CPU capacity is used during the peak traffic load and keeping rest of the capacity available for other users. The present attempt will certainly helpful to setup a cloud environment for research purpose with minimal efforts.

References

1. What is Cloud Technology (IEEE 802.16). www.freecloudinfo.com
2. Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I.: Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener. Comp. Syst. **25**, 599–616 (2009)
3. Brown, E.: NIST issues cloud computing guidelines for managing security and privacy. NIST Spec. Publ. 800–144 (2012)
4. Voorsluys, W., Broberg, J., Buyya, R.: Introduction to cloud computing: principles and paradigms. Wiley Press, New York (2011)
5. Zhang, Q., Cheng, L., Boutaba, R.: Cloud computing: state-of-the-art and research challenges. J. Internet Serv. Appl. **1**(1), 7–18 (2010)
6. Buyya, R., Calheiros, R.N., Li, X.: Autonomic cloud computing: open challenges and architectural elements. In: Proceedings of 3rd International Conference of Emerging Applications of Information Technology (EAIT 2012), Kolkata, India. IEEE Press, USA (2012)
7. XEN project Governance followed by individuals for developing and using Xen products. http://www.xenproject.org/
8. Hwang, J., Zeng, S., Wu, K., Wood, T.: A component-based performance comparison of four hypervisors. In: 13th IFIP/IEEE International Symposium on Integrated Network Management (IM) Technical Session (2013)
9. Sotomayor, B., Montero, R.S., Lorente, I.M., Foster, I.: Virtual infrastructure management in private and hybrid clouds. IEEE Internet Comput. **13**(5), 14–22 (2009)
10. Eucalyptus-products and services. http://www.eucalyptus.com
11. Official documentation on open nebula cloud framework. www.opennebula.org/documentation
12. oVirt 3.2 release notes. http://www.ovirt.org/OVirt_3.2_release_notes
13. Apache cloud stack. http://cloudstack.apache.org/
14. Nimbus infrastructure implementation. http://www.nimbusproject.org/doc/nimbus/
15. Cloud-b-lab: Tutorials on cloud computing, OPENNEBULA 3.4.1 in Ubuntu 12.04 Precises—single machine installation for learning and testing purpose