Private clouds provide a cost-effective alternative for very large organizations. A private cloud has

essentially the same structural components as a commercial one: the servers, the network, virtual

machines monitors (VMMs) running on individual systems, an archive containing disk images of virtual

machines (VMs), a front end for communication with the user, and a cloud control infrastructure. Opensource

cloud computing platforms such as Eucalyptus [269], OpenNebula, and Nimbus can be used as

a control infrastructure for a private cloud.

Schematically, a cloud infrastructure carries out the following steps to run an application:

• Retrieves the user input from the front end.

• Retrieves the disk image of a VM from a repository.

• Locates a system and requests the VMM running on that system to set up a VM.

• Invokes the DHCP7 and the IP bridging software to set up a MAC and IP address for the VM.

We discuss briefly the three open-source software systems, Eucalyptus, OpenNebula, and Nimbus.

Eucalyptus (www.eucalyptus.com) can be regarded as an open-source counterpart of Amazon’s EC2,

(see Figure 3.4). The systems supports several operating systems including CentOS 5 and 6, RHEL 5

and 6, Ubuntu 10.04 LTS, and 12.04 LTS.

The components of the system are:

Virtual machine. Runs under several VMMs, including Xen, KVM, and Vmware.

• Node controller. Runs on every server or node designated to host a VM and controls the activities

of the node. Reports to a cluster controller.

• Cluster controller. Controls a number of servers. Interacts with the node controller on each server

to schedule requests on that node. Cluster controllers are managed by the cloud controller.

• Cloud controller. Provides the cloud access to end users, developers, and administrators. It is accessible

through command-line tools compatible with EC2and through aWeb-based Dashboard. Manages

cloud resources, makes high-level scheduling decisions, and interacts with cluster controllers.

• Storage controller. Provides persistent virtual hard drives to applications. It is the correspondent

of EBS. Users can create snapshots from EBS volumes. Snapshots are stored in Walrus and made

available across availability zones.

• Storage service (Walrus). Provides persistent storage and, similarly to S3, allows users to store

objects in buckets.

The system supports a strong separation between the user space and the administrator space; users

access the system via a Web interface, whereas administrators need root access. The system supports

a decentralized resource management of multiple clusters with multiple cluster controllers, but a single

head node for handling user interfaces. It implements a distributed storage system, the analog of

Amazon’s S3 system, called Walrus. The procedure to construct a virtual machine is based on the generic

one described in [323]:

• Theeuca2ools front end is used to request a VM.

• The VM disk image is transferred to a compute node.

• This disk image is modified for use by the VMM on the compute node.

• The compute node sets up network bridging to provide a virtual network interface controller (NIC)8

with a virtual Media Access Control (MAC) address.9

• In the head node the DHCP is set up with the MAC/IP pair.

• VMM activates the VM.

• The user can now ssh10 directly into the VM.

The system can support a large number of users in a corporate enterprise environment. Users are

shielded from the complexity of disk configurations and can choose their VM from a set of five configurations

for available processors, memory, and hard drive space set up by the system administrators.

Open-Nebula (www.opennebula.org) is a private cloud with users actually logging into the head node

to access cloud functions. The system is centralized and its default configuration uses NFS (Network

File System). The procedure to construct a virtual machine consists of several steps: (i) the user signs

into the head node using ssh; (ii) the system uses the onevm command to request a VM; (iii) the VM template disk image is transformed to fit the correct size and configuration within the NFS directory

on the head node; (iv) the oned daemon on the head node uses ssh to log into a compute node; (v) the

compute node sets up network bridging to provide a virtual NIC with a virtual MAC; (vi) the files

needed by the VMM are transferred to the compute node via the NFS; (vii) the VMM on the compute

node starts the VM; and (viii) the user is able to ssh directly to the VM on the compute node.

According to the analysis in [323], the system is best suited for an operation involving a small-to

medium-sized group of trusted and knowledgeable users who are able to configure this versatile system

based on their needs.

Nimbus (www.nimbusproject.org) is a cloud solution for scientific applications based on the Globus

software. The system inherits from Globus the image storage, the credentials for user authentication,

and the requirement that a running Nimbus process can ssh into all compute nodes. Customization in

this system can only be done by the system administrators.

Table 3.5 summarizes the features of the three systems [323]. The conclusions of the comparative

analysis are as follows: Eucalyptus is best suited for a large corporation with its own private cloud

because it ensures a degree of protection from user malice and mistakes. OpenNebula is best suited

for a testing environment with a few servers. Nimbus is more adequate for a scientific community less

interested in the technical internals of the system than with broad customization requirements.

OpenStack is an open-source project started in 2009 at the National Aeronautics and Space Administration

(NASA) in collaboration with Rackspace (www.rackspace.com) to develop a scalable cloud

operating system for farms of servers using standard hardware. Though recently NASA has moved its

cloud infrastructure to AWS in addition to Rackspace, several other companies, including HP, Cisco,

IBM, and Red Hat, have an interest in OpenStack. The current version of the system supports a wide

range of features such as application programming interfaces (APIs) with rate limiting and authentication;

live VM management to run, reboot, suspend, and terminate instances; role-based access control;

and the ability to allocate, track, and limit resource utilization. The administrators and the users control

their resources using an extensible Web application called the Dashboard.