A Major Project Report On

IMPLEMENTING PRIVATE CLOUD USING OPEN SOURCE SOFTWARE

Submitted in partial fulfilment of the requirements

for the award of the degree of

MASTER OF TECHNOLOGY

IN

Computer Science & Engineering

By

Rajat Sehrawat

(Roll No. 2K12/CSE/13)

Under the guidance of

Divyashikha Sethia

Assistant Professor

Department of Software Engineering

Delhi Technological University, Delhi

Department of Computer Engineering

Delhi Technological University, Delhi

2012-2014

Department of Computer Engineering
Delhi Technological University,
Shahbad Daulatpur, Main Bawana Road, Delhi – 110042

Tel : 011-27871043 Fax : 011-27871023

CERTIFICATE

This is to certify that the thesis entitled "**Implementing Private Cloud Using Open Source Software**" submitted by Rajat Sehrawat (2K12/CSE/13), to the Department of Computer Engineering of Delhi Technolgical University, Delhi in partial fulfilment of the requirements for the award of the degree of Master of Technology (M. Tech.) in Computer Science & Engineering is an authentic record of the work carried out by him under my guidance and supervision.

In my opinion, this work fulfils the requirement for which it has been submitted. This dissertation has not been submitted to any other university or institution for any degree.

Divyashikha Sethia,
Assistant Professor,
Department of Software Engineering,
Delhi Technological University, Delhi.

Department of Computer Engineering
Delhi Technological University,
Shahbad Daulatpur,Main Bawana Road, Delhi – 110042

Tel : 011-27871043 Fax : 011-27871023

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled "**Implementing Private Cloud Using Open Source Software**" in partial fulfilment of the requirement for the award of the degree of Master of Technology (M. Tech.) in Computer Science & Engineering and submitted in the Department of Computer Engineering. Delhi Technological University, Delhi is an authentic record of my own work carried out under the supervision of Divyashikha Sethia, Assistant Professor, Department of Software Engineering. The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other Institute / University.

Name	**:**	**Rajat Sehrawat**
Roll No.	**:**	**2K12/CSE/13**

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

Date :

Divyashikha Sethia,
Assistant Professor,
Department of Software Engineering,
Delhi Technological University, Delhi.

Department of Computer Engineering
Delhi Technological University,
Shahbad Daulatpur,Main Bawana Road, Delhi – 110042

Tel : 011-27871043 Fax : 011-27871023

ACKNOWLEDGEMENT

With due regards, I hereby take this opportunity to acknowledge a lot of people who have supported me with their words and deeds in completion of my research work as part of this course of Master of Technology in Software Engineering.

To start with I would like to thank the almighty for being with me in each and every step of my life. Next, I thank my parents and family for their encouragement and persistent support.

I would like to express my deepest sense of gratitude and indebtedness to my guide and motivator, **Divyashikha Sethia**, Assistant Professor, Department of Software Engineering, Delhi Technological University for her valuable guidance and support in all the phases from conceptualization to final completion of the project.

I wish to convey my sincere gratitude to **Prof. Rajeev Kapoor,** Head of Department, and all the faculties and PhD. Scholars of Department of Computer Engineering, Delhi Technological University who have enlightened me during my project.

I humbly extend my grateful appreciation to my friends whose moral support made this project possible.

Last but not the least, I would like to thank all the people directly and indirectly involved in successfully completion of this project.

Rajat Sehrawat

Roll No: 2K12/CSE/13

ABSTRACT

This work is based on setting up private cloud to be used as a platform for deploying various applications over it. In this work a private cloud has been deployed in LANS lab, DTU using Apache CloudStack Infrastructure as Service (IaaS) with ActiveState Stackato Platform as a Service (PaaS) over it. The main cloud setup is used to provide a Health Secure Service for a NFC based healthcare service as suggested in [1].

The Health Secure service provides security as a service for a secure interaction between patient and doctor. The application assumes that patient records are retained with him on a personal electronic device like mobile. In case the records are lost they can be retrieved from the backup of records maintained in the cloud. The cloud provides a PaaS application where patients and doctors can be registered and their security information as well as access control information can be retained on a SQL database. The patient or any hospital staff will connect to private cloud for key retrieval or information retrieval of patient. If hospital staff wants to add new medical record into system then it should be done in very secure way. For this purpose private cloud is necessary which will deliver on-demand service to authenticated devices in a secure manner and users can utilize a shared and elastic infrastructure. Another positive aspect of cloud computing is scalability. If there is peak load or high traffic for a site, cloud can handle easily without any additional hardware infrastructure & without disturbing user's normal work. Another advantage of cloud is fault tolerance, i.e. hardware failure can be easily traced out and rectified.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

1.1 Cloud computing

Cloud Computing is emerging computing paradigm which pools various resources such as computing power, storage, network and software for provision of services on the internet in a remotely accessible fashion. Cloud Computing is not a new technology, instead it is modern way of providing services using technology. Cloud Computing is a powerful and flexible software environment, in which users pay as they go. The migration of applications on the Cloud is increasing day by day. Some users try to build their own private Cloud which is a custom-fit to their requirements. These users make use of existing technologies such as CloudStack, OpenStack, Eucalyptus, Nimbus, OpenNebula, Azure and many more [29].

The official definition given by the cloud Security Alliance and National Institute of Standards and Technology (NSIT) [10] for cloud computing -
"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".
Gartner [27] defines cloud computing as-
"A style of computing where massively scalable, IT-enabled capabilities are provided as a service across the Internet to multiple external customers".
Cloud computing refers to applications delivered as services to the end users over the Internet where the hardware and systems software are placed at remote locations that provide those services. It facilitates sharing of digital information, software platforms and technological resources between a large number of clients and users on Internet.

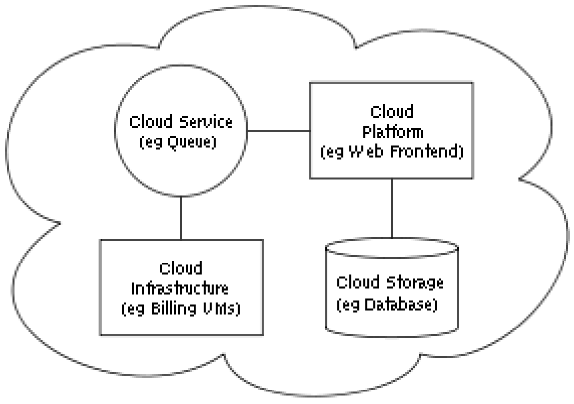

Cloud Service (eg Queue)
Cloud Platform (eg Web Frontend)
Cloud Infrastructure (eg Billing VMs)
Cloud Storage (eg Database)

Figure 1: Cloud Computing Sample Architecture [8]

1.1.1 Features of cloud computing [24]

- **On Demand Self Service**

It is a prime feature of cloud which refers to the provision of cloud resource available to user whenever they requires. On-demand self service methodology passes users to requested resources on run time.

- **Broad Network Access**

Broad network access is the property by which the private cloud resources hosted within company firewall are made available to wide range of devices like Desktops, Laptops, Tablets, Mobile Phones, etc. These resources can also be accessible from remote locations.

- **Resource Pooling**

Pool of computing resource shared between the users as per their requirements. This can be achieved by Multi-Tenant application architecture, where multiple user share same instance. Each user has his/her own isolated space dedicated for his/her use.

- **Scalable & Elastic**

Cloud service has to scale up or scale down its capacity based on user's or application's need. Scalability is the power of cloud service by which it can grow or shrink while Elasticity is property of immediate addition or removal of resources. Scalability achieved by virtuousness of its architecture while Elasticity is implicitly gained by deploying on a cloud.

- **Measured Service**

Cloud services are measured in similar to electricity usage. Aside from measuring and monitoring, the resource usage can also be controlled. The price lists are entirely based on the amount of the service used by the users, which may be measured in terms of hours, data transfers or amount of computing resources it required with time. Here a user only pays for what he/she use (similar to cab services).

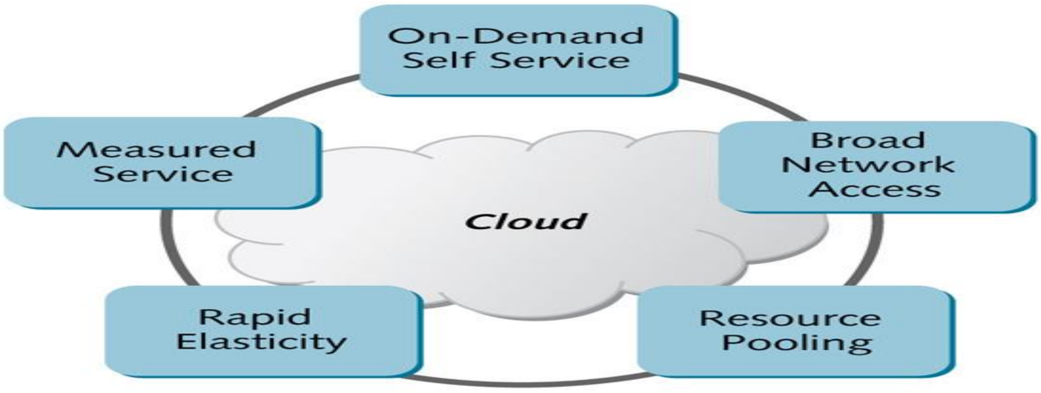


Figure 2: Features of Cloud

1.2 Cloud participants

The user of the cloud need not to know any background about the cloud and can perform his/her tasks seamlessly using cloud services. Enterprise management is responsible for managing the services and data present on the cloud.

Service provider is responsible for providing the services and service management.

1.3 Cloud, cloud services and access

The cloud is defined as the set of networks, hardware, interfaces, services and storage that are collectively responsible for providing the services.

Cloud services are responsible for the delivery of infrastructure, storage and software over the Internet as per the user's requirements.

Access to the cloud by the user can be via multiple devices like PC, mobile tec. or multiple technologies like Internet etc.

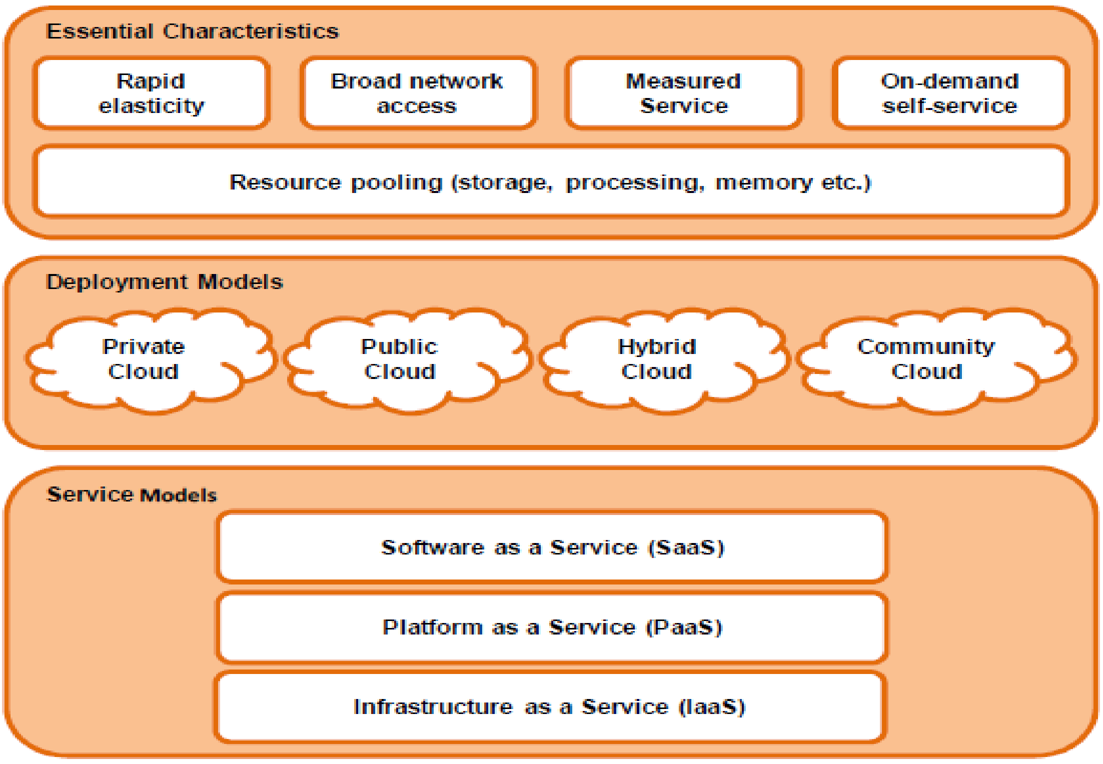

Essential Characteristics
Rapid elasticity
Broad network access
Measured Service
On-demand self-service
Resource pooling (storage, processing, memory etc.)
Deployment Models
Private Cloud
Public Cloud
Hybrid Cloud
Community Cloud
Service Models
Software as a Service (SaaS)
Platform as a Service (PaaS)
Infrastructure as a Service (IaaS)

Figure 3: Overview of cloud computing [11]

Chapter 2

DEPLOYMENT MODELS

2.1 **Public Cloud**

This type of cloud is made for public use i.e. the end-user of this cloud can be anyone who wishes to use the service. It is provided big companies like Amazon, Google etc. In this type of cloud data is stored at remote locations which are difficult to be accessed.

Advantages of Public cloud are-

- Easy setup.
- Pay for only what you use.
- Increased scalability.
- Reliable and Robust backup.

2.2 **Private cloud**

This type of cloud is made for an enterprise and is used by that enterprise only for their personal work. Here only the enterprise members can access the cloud and no one else is able to use the cloud.

Advantages of Private cloud

- Reduces costs.
- Provides elasticity.
- Dynamic storage capacity.
- Easy management of the infrastructure.
- More control

2.3 **Hybrid cloud**

This type of cloud is composition of more than two clouds public, community and private cloud. It is managed by external as well as internal providers.

2.4 **Community cloud**

This type of cloud is shared between a number of enterprises and it supports a particular group or community that has a same goals and objectives.

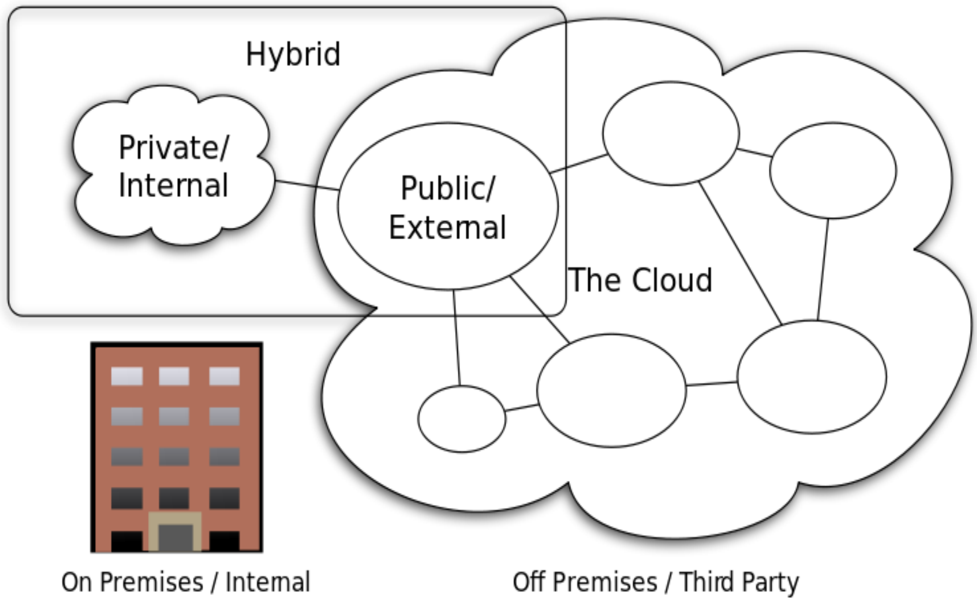


Figure 4: Deployment Models [6]

Chapter 3

SERVICE MODELS

Based on the type of services and the flexibility that the cloud computing provider provides the user, cloud services are broadly classified into three basic service models.

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

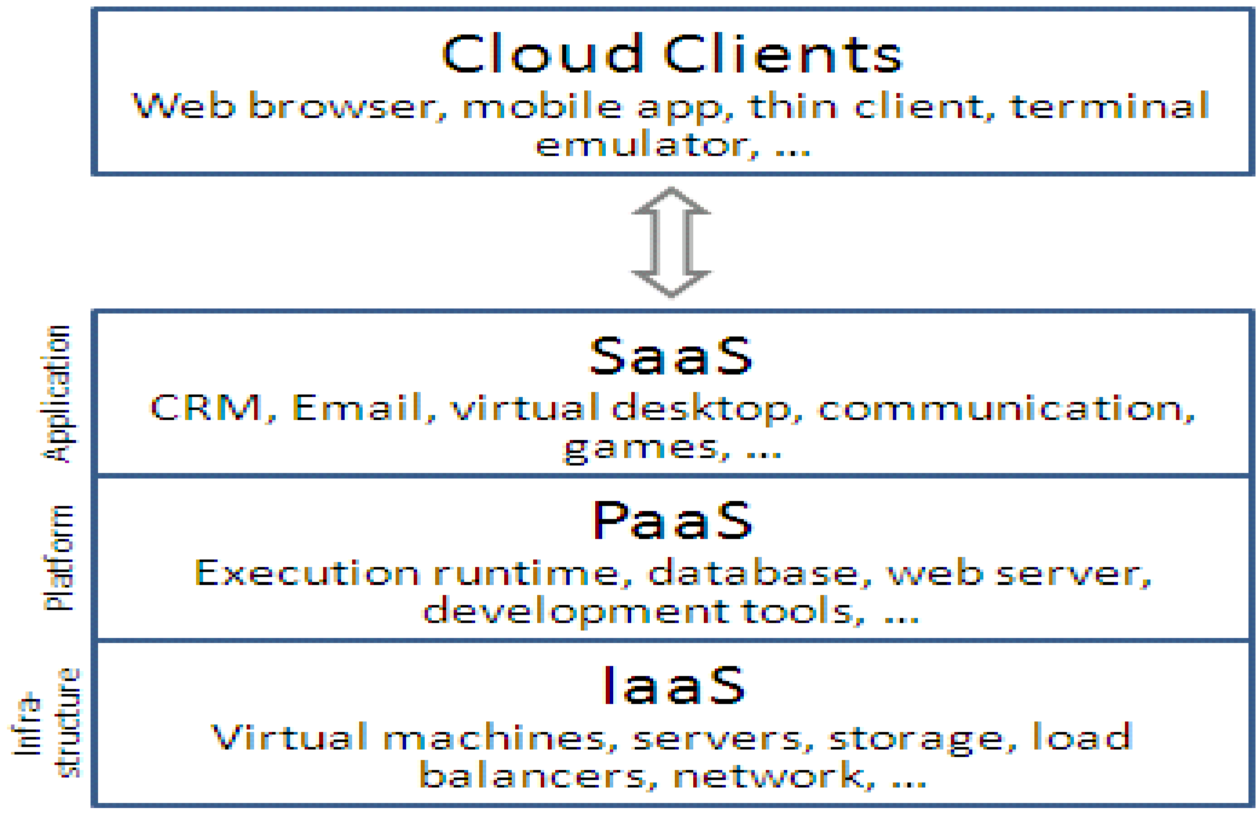

Cloud Clients
Web browser, mobile app, thin client, terminal emulator, ...
Application
SaaS
CRM, Email, virtual desktop, communication, games, ...
Platform
PaaS
Execution runtime, database, web server, development tools, ...
Infra-structure
IaaS
Virtual machines, servers, storage, load balancers, network, ...

Figure 5: Cloud Service Models [9]

3.1 <u>Infrastructure as a Service</u>

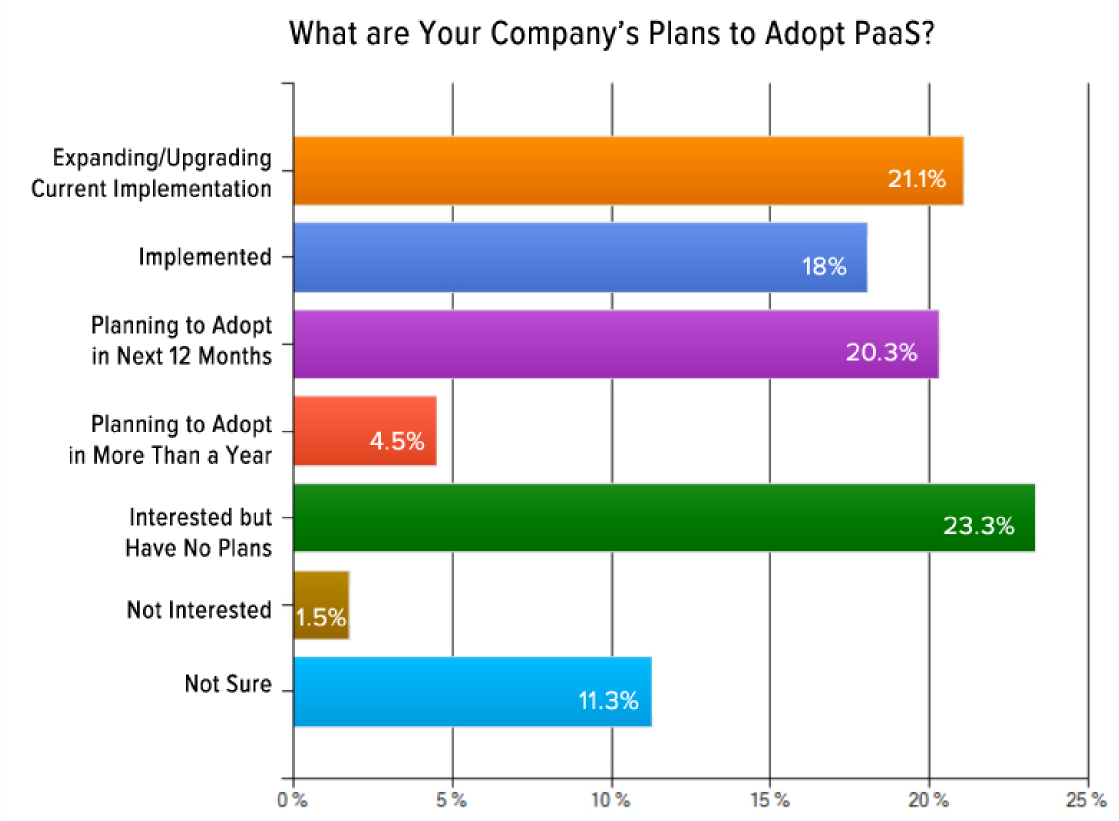
IaaS is the most basic cloud service model. Main aim of any organization is to reduce time and cost required for procuring the new improved hardware systems. IaaS targets that objective with outsourcing of equipment to support operations. In Iaas the duty of service provider is housing, controlling and equipment maintenance which forms the cloud. The VM are run by the hypervisors, like KVM or Xen as guests on the host systems. This feature allows cloud to support multiple VMs. IaaS clouds includes resources such as images of virtual machine, storage, IP addresses, load balancers, firewalls, VLANs (virtual local area networks), and software bundles.

Examples of IaaS include: OpenStack, CloudStack, Eucalyptus, AWS, Nimbus, Open Nebula [29].

3.2 Platform as a Service

The PaaS will provide computing resources through platform such as Operating System. PaaS is built on top of IaaS. It provides building blocks such as programming languages and an environment for easy and quick deployment of applications onto the cloud infrastructure. PaaS eradicates the time consuming phase of hardware and other configurations. It provides a platform for safe and secure development, deployment and management of applications. PaaS platform has evolved from IaaS and SaaS. The companies are shifting more and more towards PaaS solution. In a survey [30] of 162 people, conducted by Engine Yard in the year 2012 following results were obtained

- About 88% of the surveyed people were aware about PaaS
- Almost 21% of the surveyed companies are currently upgrading to PaaS and 18% of them are already using PaaS for their companies



.

Figure 6: PaaS Adoption Percentage Graph [30]

- Following figure shows the benefits of PaaS and their importance percentage from company perspective.

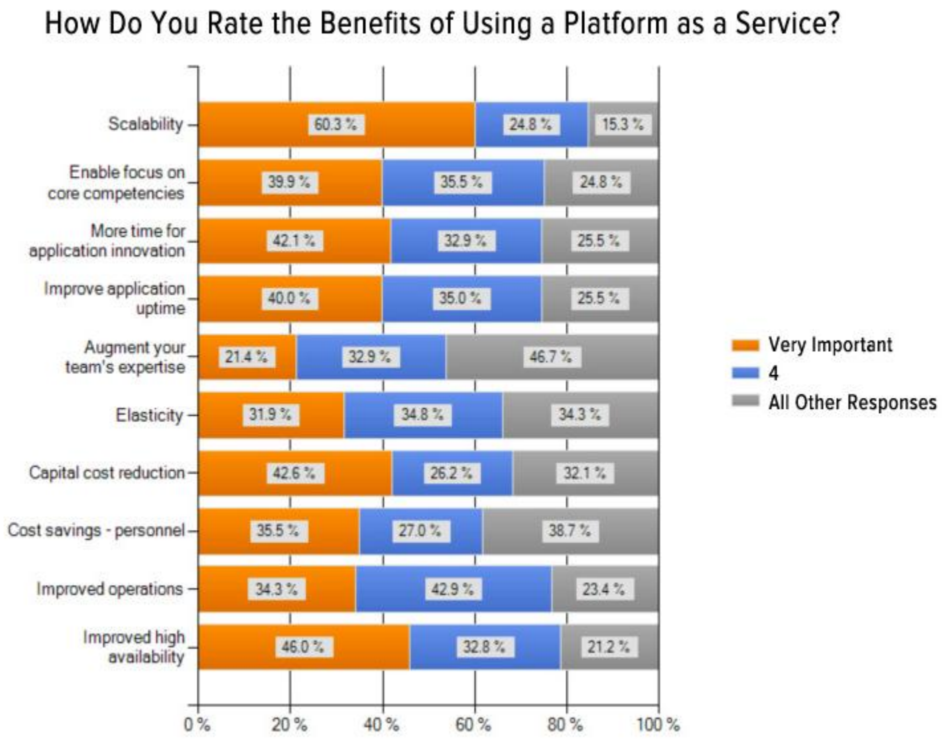


Figure 7: Benefit Rating of Using PaaS[30]

- Nearly 5% of the surveyed people were using PaaS for Medical Application which leaves a lot of room for research and innovation in health sector using PaaS.

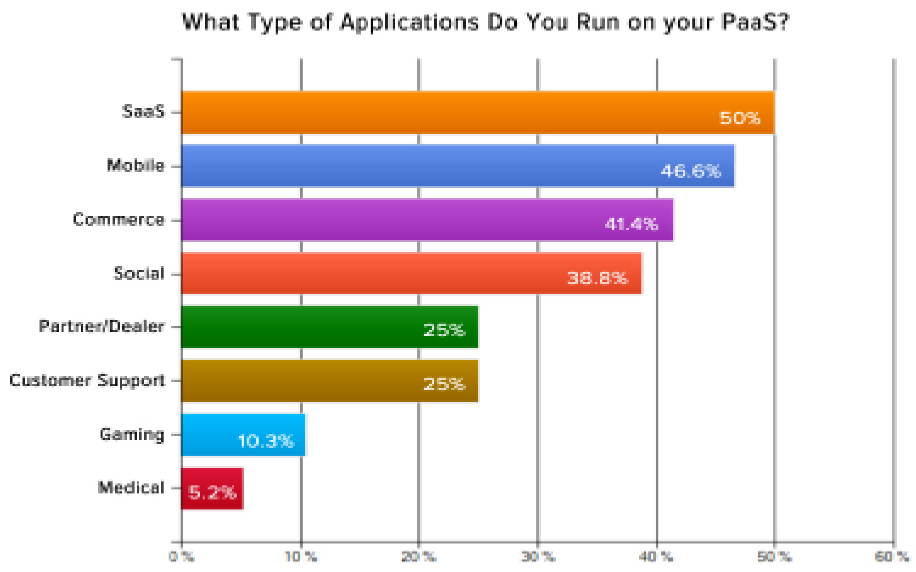


Figure 8: Type of Applications Run on PaaS [30]

Examples of PaaS include: Engine Yard, Heroku, AppFog, Google App Engine , OpenShift, Windows Azure Cloud Services[31].

3.2.1 Single-tenant PaaS (ST-PaaS) [23]

In ST-PaaS, an IaaS instance is dedicated to a particular user. The relationship between MT-PaaS and ST-PaaS is a trade-off. The running cost of MT-PaaS is smaller than that of ST-PaaS, but thedevelopment cost is higher.

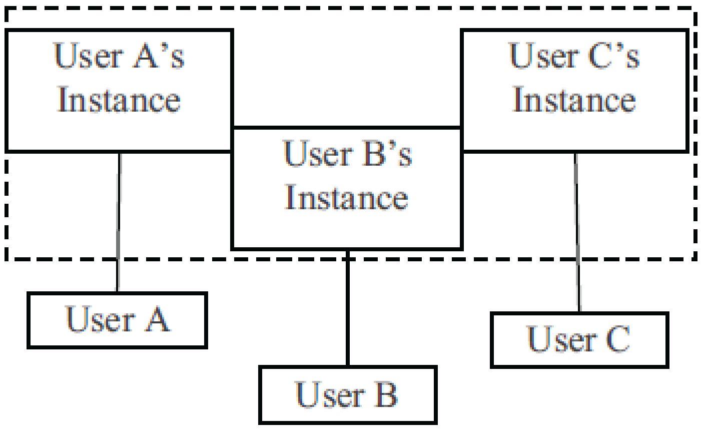


Figure 9: Structure of ST-PaaS [23]

3.2.2 Multi-tenant PaaS (MT-PaaS) [23]

In MT-PaaS, an instance of IaaS is shared by multiple tenant users. In principle, the number of instances in MT-PaaS will be smaller than that in ST-PaaS, which reduces the cost.

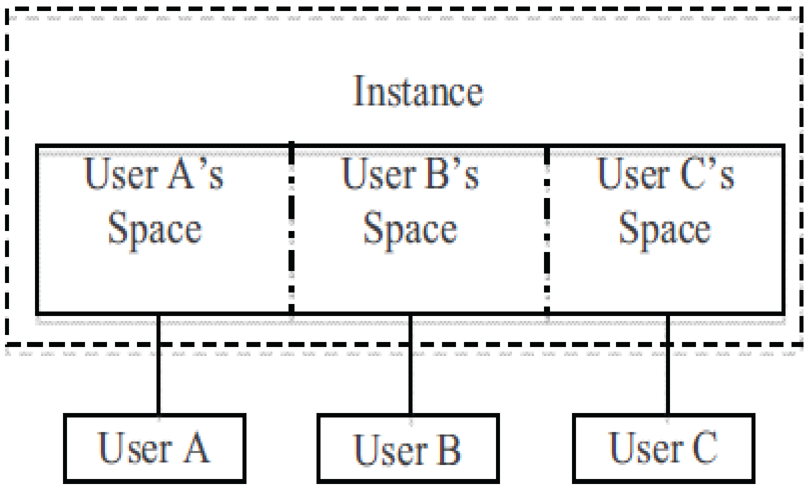


Figure 10: Structure of MT-PaaS [23]

3.3 Software as a Service

In this model the user directly access and interact with the application which is deployed over the cloud. The user does not need to install the application on his/her own computer in order to run that application, instead the user is able to directly access the application which is already running on the cloud of the service provider. This service model is of great use from business point of view, as businesses are able to get the same results as that of commercial software with a much reduced cost. It also frees the user from the burden of software support and maintenance. The main benefit of SaaS is non-involvement of licensing risk and also no issue of version compatibility. It also decreases the hardware and software cost. Customer does not require any technical knowledge while using paid software and underlying infrastructure. Common example of SaaS application software is CRM, Google Docs, ERP, etc. SaaS also reduces the IT support cost.

Enterprises have to select the right service model based on their specific requirements. The selection has to be done considering various factors such as cost benefit analysis, relevant risks, security and controls and the criticality of the data and services. Typically, enterprises would choose the model which offers them the best savings with the required security as appropriate to the criticality of the services provided. Quite often, the non-critical services/applications are the first to be migrated both as a test case and also considering the lower risks.

\

Chapter 4

IaaS SOLUTIONS

4.1 Eucalyptus

Eucalyptus is an (Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems) open-source implementation of Amazon EC2 and compatible with business interfaces. Eucalyptus project started at California University Santa Barbara, and mainly was used to build open-source private cloud platform initial release of Eucalyptus came out in October 2010.

4.2 OpenNebula

OpenNebula is an open source toolkit used to build private, public and hybrid Clouds. It works with Xen, KVM and VMware virtualization solution and it is currently working on supporting VirtualBox.

4.3 Nimbus

The Nimbus project is considered as a science Cloud Computing solution providing Infrastructure as a service. Nimbus is attached to the Culumbus project and it supports different virtualization implementations: Xen and KVM.

4.4 AbiCloud

AbiCloud is a private Cloud solution developed by Abiquo. It enables users to build Infrastructure as a service Cloud environment. The supported virtualization techniques of AbiCloud are VirtualBox, VMWare, XEN, and KVM.

4.5 Delta Cloud

Deltacloud is a vendor neutral API and an open source project initiated by Red Hat and is now an Apache incubator project. Deltacloud abstracts the differences between clouds and maps a cloud client's application programming interface (API) into the API of a number of popular clouds, including Amazon EC2, GoGrid, OpenNebula, and Rackspace

4.6 OpenStack

OpenStack is a collaborative software project designed to create freely available code and needed standards for the benefit of both Cloud providers and Cloud customers. OpenStack is currently three projects: OpenStack Compute (deploy automatically provisioned virtual compute instances), OpenStack Object Storage (redundant storage of static objects) and OpenStack Image Service (provides discovery, registration, and delivery services for virtual disk images).

4.7 XenCloud Platform

The Xen Cloud Platform (XCP) is an open source server virtualization and cloud computing platform, delivering the Xen Hypervisor. The Xen Hypervisor provides an abstraction layer between server's hardware and operating systems including Windows and Linux network and storage support.

4.8 CloudStack

CloudStack is an open source software provided by Apache which pools computing resources such as network, computing nodes and storage to provide Infrastructure as a Service (IaaS) clouds. CloudStack also provides an interface for deployment, management, and configuration of cloud computing environments [12].

4.9 Comparison Table

Property	**OpenStack**	**Eucalyptus**	**Nimbus**	**OpenNebula**	**CloudStack**
Initial Release	October 2010	May 2008	January 2009	March 2008	May 2010
Developers	Rackspace Hosting, NASA and now More than 200 companies have joined	Eucalyptus Systems, Inc.	Kate Keahey, Tim Freeman, et al.	OpenNebula Community	Apache Software Foundation
Written in Language	Python	Java and C	Python and Java	C++, C, Ruby, Java, Shell script, lex and yacc	Java and C

Users	Enterprise, service providers and researchers	Enterprise	Scientific communities	Researchers on Cloud Computing and Virtualization	Service providers and enterprises
Architecture	Integration of OpenStack object storage and OpenStack compute -Minimum 6 servers	- Hierarchical - Five components - Minimum two servers	-Centralized - Three components - Minimum two servers	- Centralized - Three components - Minimum two servers	Two Components - Management Server -Cloud infrastructure
Supported Hosts	Linux and Windows	Linux	Windows	Linux	Linux
Storage	OpenStack Object Storage and Block storage	Storage Controller	- GridFTP, Comulus (new version of GridFTP) - SCP	-SCP - SQLite3	Primary Storage and Secondary Storage
Network	Quantum linked with OpenStack Compute (Nova)	DHCP server on the cluster controller	DHCP server installed on nodes	Manual configuration	Simple and Advanced Network
Load Balancer	Yes	Yes	Yes	Yes	Yes
VM migration	Manual Live Migration	No	No	Shared FS	Manual Live Migration
VMs Location	OpenStack Compute (Nova)	Node controller	Physical nodes	Cluster node	Hosts
Hypervisor	XEN, KVM and VMware	Xen, KVM and VMWare	XEN and KVM	XEN, KVM and VMWare	XEN, KVM and VMware

Interface	Web interface	- EC2 WS API - Tools as: HybridFox, ElasticFox	- EC2 WS API - Nimbus WSRF	- EC2 WS API - OCCI API	Web Interface
Used By	AT&T, CERN, Intel, Sony, Yahoo!, NASA, HP public cloud, etc.	NASA	STAR	Reservoir Project, NUBA	Sungard, Softlayer, Spotify,Amdoc s

Table 1: Comparison between IaaS[3][4][5][6][7]

4.10 <u>Why Private Cloud?</u>

Following are the reasons for utilizing private cloud for this project:

- Have higher Return On Investment than traditional infrastructure
- Are more customizable
- Can quickly respond to changes in demands
- Support rapid deployment
- Have increased security
- Focus on an organization's core business

4.11 <u>Why Open Source System?</u>

In general OSS has following advantages:

- Lowering the costs (i.e. no licensing headaches)
- Interchange ability & portability (general, avoiding vendor lock-in)
- Socio-organizational reasons
- Energy efficient

Based on the above mentioned points I have selected Eucalyptus for this project work.

Chapter 5

CLOUDSTACK

CloudStack[16] is an open source software provided by Apache to deploy private, public and hybrid IaaS cloud. It pools and manages computing resources such as storage, network, and computing nodes which collectively makes cloud infrastructure [12].

CloudStack[16] can set up on-demand, elastic cloud computing service as well as it can set up on-premise private cloud for use of employees. On-demand means “provision of cloud resource available to user whenever they requires”. Elastic means “how instantly an application can add or remove resources”.

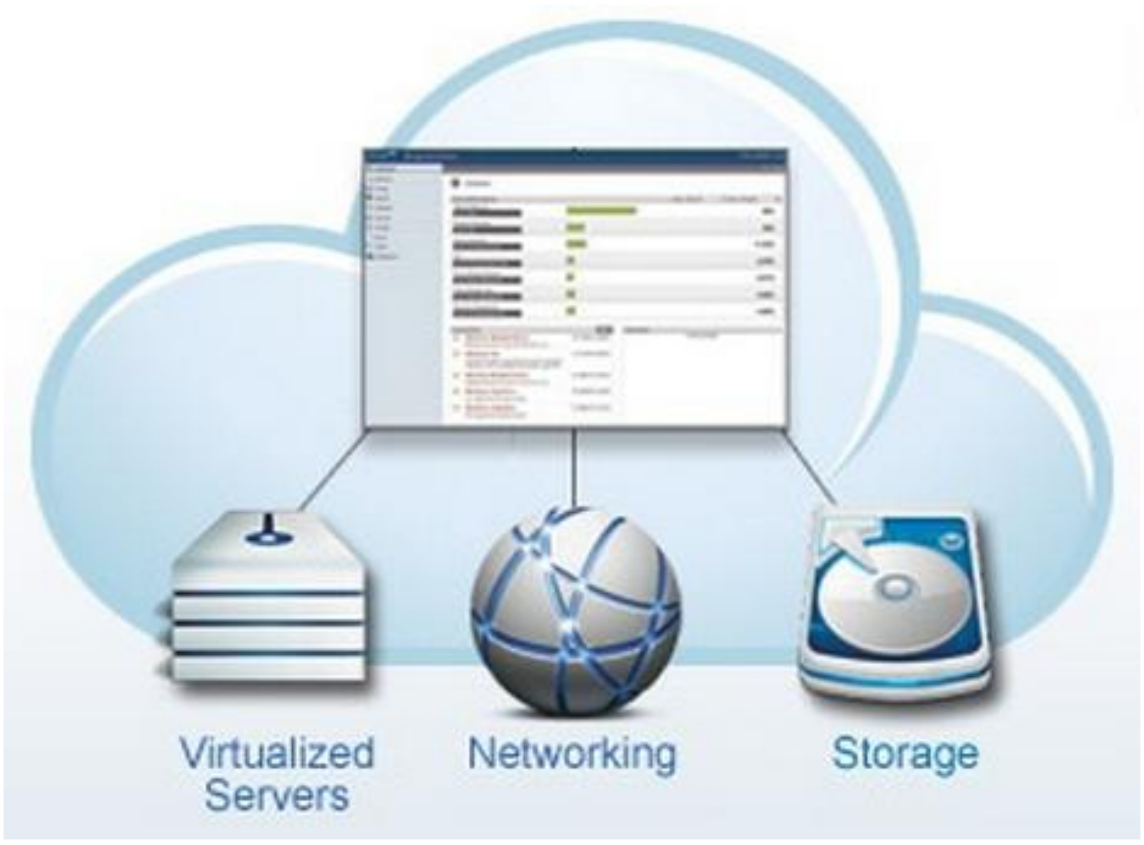


Figure 11: Overview of CloudStack [12]

5.1 CloudStack Architecture

A CloudStack[16] consists of two parts:

1. Management Server
2. Cloud Infrastructure

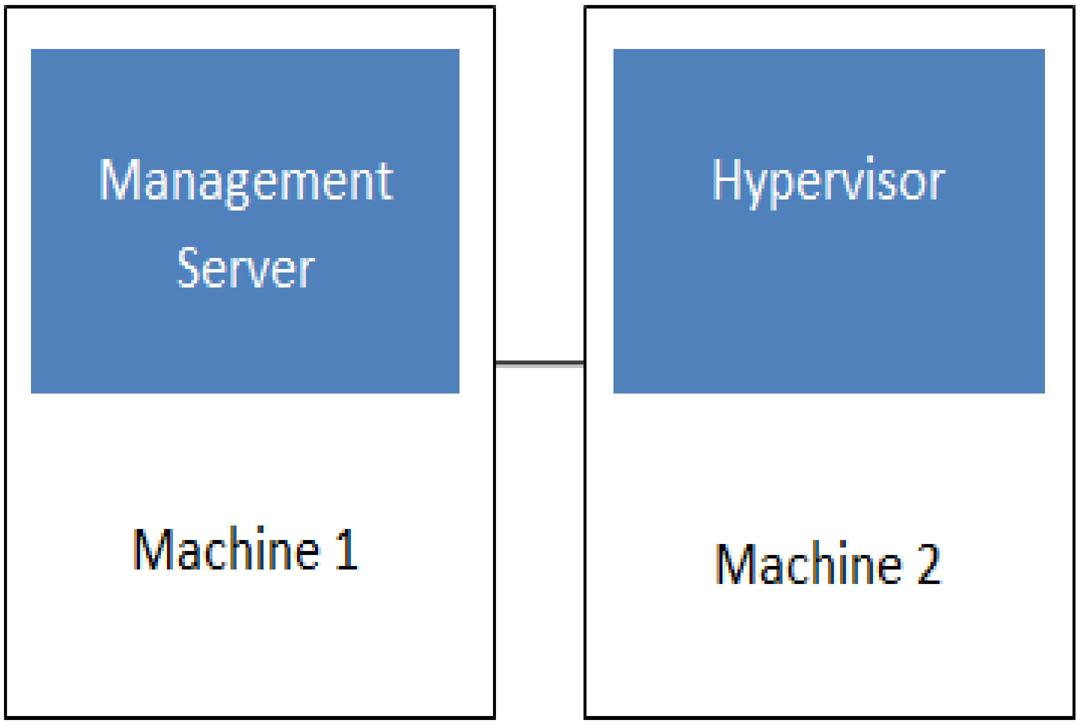


Figure 12: Basic Architecture of CloudStack Deployment [12]

5.2 <u>Management Server [12]</u>

The Management Server is responsible for the proper management of cloud resources. We can manage and configure our cloud infrastructure by interacting with Management Server through or UI or API. Also the Management Server can be run on dedicated server or VM. It insures the allocation of virtual machines to hosts and allocates storage and IP addresses to the virtual machines instances.

In short the Management Server [12]:

- Offers the web user interface to the administrator and end users.
- Provides the APIs[15] for CloudStack.
- Handles the assignment of guest VMs to specific hosts.
- Handles the assignment of private and public IP addresses to specific accounts.
- Handles the assignment of storage unit to guests as virtual disks.
- Management of templates, snapshots and ISO images, which are possibly replicated across data centers.
- Provides an interface to configure the cloud at a single point only.

5.3 <u>Cloud Infrastructure [12]</u>

The Management Server handles one or more zones having host computers where guest virtual machines usually runs.

Cloud Infrastructure Organized into

5.3.1 Zone:

It is equivalent to datacenter. It comprises of secondary storage and one or more pods.

5.3.2 Pod:

It is one rack of hardware that comprises a layer-2 switch and clusters.

5.3.3 Cluster:

It comprises of primary storage and one or more hosts. A cluster is a bunch of XenServer servers, VMware cluster (preconfigured in vCenter) or a bunch of KVM servers. Within the same cluster it it possible to live-migrate the VM from one host to another, without interruption of service to user.

5.3.4 Host:

It’s a single node in the cluster. Host run in the form of virtual machines to provide cloud service. Hypervisor software is installed on each host for the management of guest VMs. To provide more capacity to guest VMs more hosts can be added at any point of time. Hosts are hidden from the end user i.e. they are not able to find, to which host their guest VM is assigned.

5.3.5 Primary Storage:

It is linked with a cluster. It consist the disk volumes of every virtual machine running on hosts under a single cluster. It supports NFS, iSCSI and Clusterd Logical Volume Manager (CLVM)

5.3.6 Secondary Storage:

It is linked with a zone. It consists of the templates, ISO images and disk volume snapshots.

All the items in secondary storage linked to zone are made available to all the hosts in the zone.

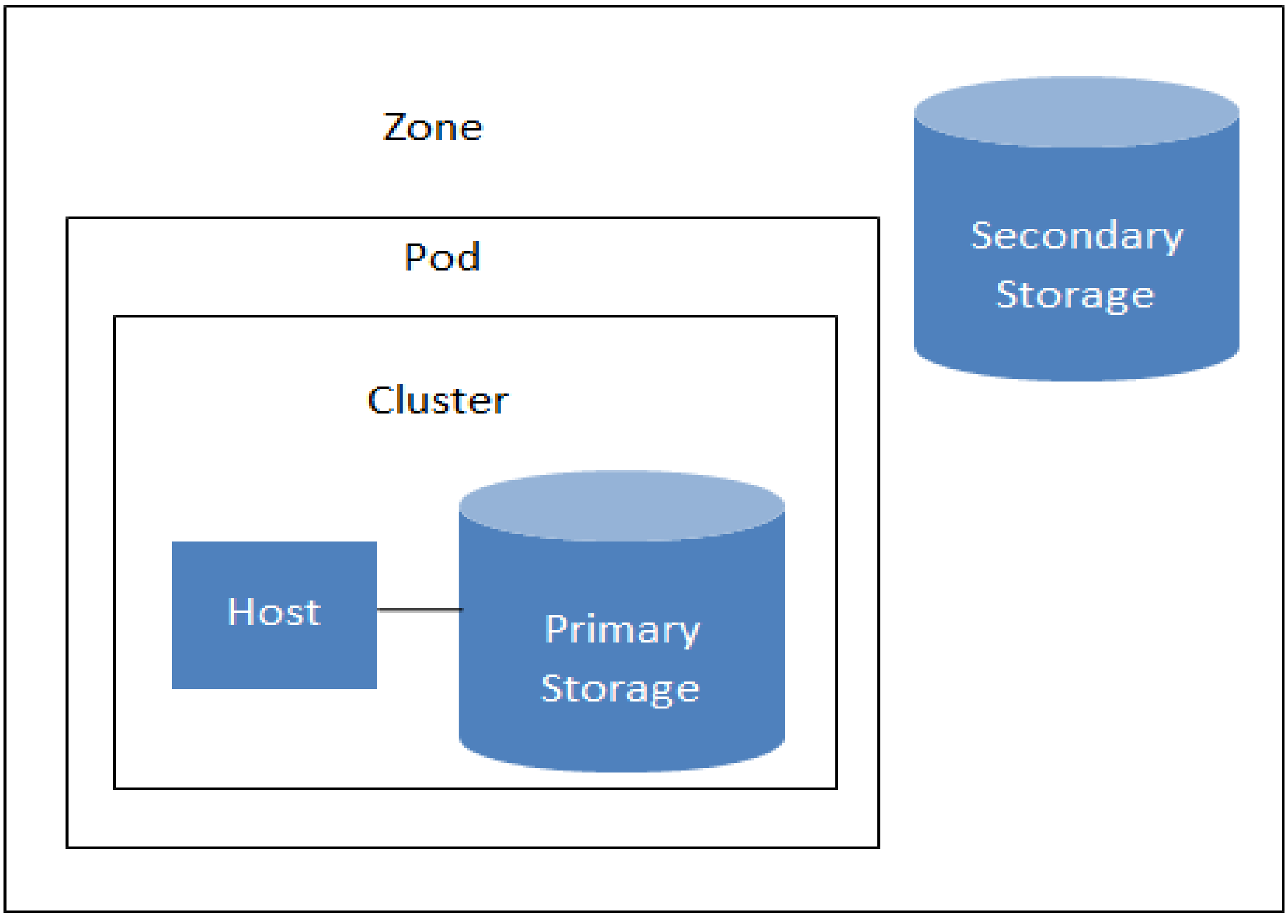


Figure 13: Cloud Infrastructure [12]

5.4 <u>**CloudStack Networking [12]**</u>

5.4.1 Basic Networking

In a basic network, configuration of the physical network is fairly easy. In general, to carry a traffic generated by guest VMs we have to configure only one guest network.

5.4.2 Advanced Networking

In advanced networking, it is recommended to set aside sufficient private IPs for accommodating the total number of users, plus sufficient for the required CloudStack System VMs.

5.5 <u>CloudStack Traffic [12]</u>

5.5.1 Guest

Guest traffic is generated when the end users run any VMs. Through guest network the guest VMs communicate with each other. This network will be isolated or shared. In isolated guest network, the administrator has to reserve VLAN range to provide isolation for each CloudStack account's network. While in shared network, single network is shared by all guest VMs.

5.5.2 Management

Management traffic is the one which is generated on account of communication between internal resources. This simply involves the direct communication between Management Server and any other component like hosts and system VMs that communicates directly with the Management Server. It is important to configure the of system VMs.

5.5.3 Public

When the VMs in the cloud get access to the Internet a public traffic is generated. For that purpose, public IP should be allocated. With help of CloudStack UI the end users can get these IPs for implementation of NAT between public network and guest network.

5.5.4 Storage

A separate Network Interface Controller (NIC) known as Storage NIC is used for storage network traffic. Storage network traffic is the one which is generated when VM templates and snapshots are sent between secondary storage servers and the secondary storage VM.

Chapter 6

CLOUDSTACK DEPLOYMENT

For implementing the CloudStack two Ubuntu 10.04 machines are used. The one machine will act as Management Server while the second machine will be KVM Hypervisor Host. Also the Database and Storage System is on Management Server itself.

The Conceptual View of CloudStack deployment is shown in below figure.

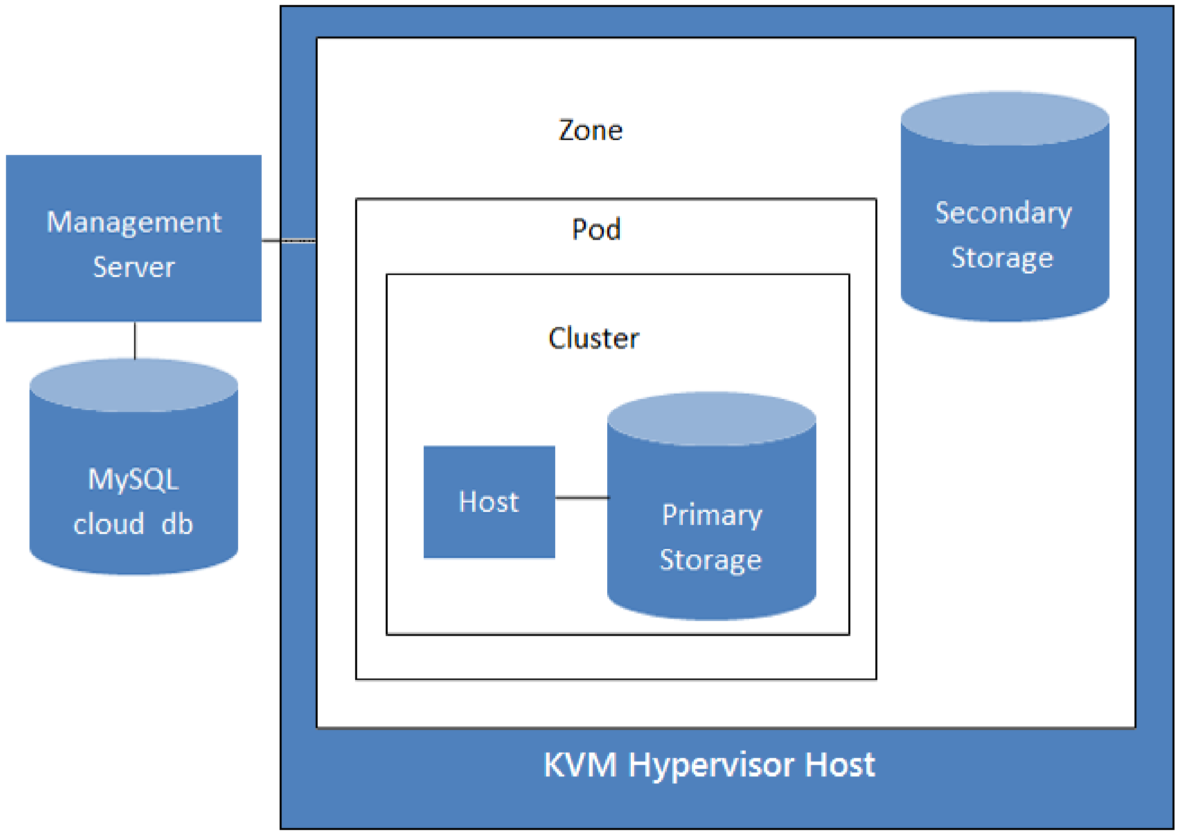


Figure 14: Conceptual View of Deployed CloudStack Cloud

6.1 Minimum System Requirements [12]

6.1.1 Management Server

	Minimum Requirements
OS	CentOS/RHEL 6.3+ or Ubuntu 10.04 or above
CPU	64-bit x86
RAM	4GB
HDD	50GB (if secondary storage on management server then 500GB is recommended)
NIC	1

Table 2: Management Server Requirements

Domain Name should be fully qualified and should return it by hostname command.

6.1.2 Host/Hypervisor

	Minimum Requirements
OS	CentOS/RHEL 6.3+ or Ubuntu 10.04 or above
CPU	64-bit x86
RAM	4GB
HDD	36 GB
NIC	1

Table 3: Host/Hypervisor Requirements

Requires Hardware virtualization support

Domain Name should be fully qualified and should return it by hostname command

6.2 <u>Management Server Installation [12] [21]</u>

Following steps describes the installation of Management Server. There are two ways of doing this installation

a) Single Management Server with MySQL on same server.

b) Multiple Management Server with MySQL installed on separate server rather than same management server machine.

6.2.1 Prepare the Operating System

The Server OS (Ubuntu or CentOS) should be prepared first to host the management server on it. The following steps confirms the fully prepared server OS for management server.

1. Login to Server OS as Root User
2. Verify the fully qualified hostname of the prepared Server OS with following command

```
hostname --fqdn
```

The above command should return the fully qualified domain name of system like "cldstkmgmt.lanslab.edu". If output of the above command is not in the specified format the correct it by editing the file "/etc/hosts".

3. Install NTP for Time Synchronisation

```
#apt-get install openntpd
```

6.2.2 Install the Management Server on the Host

Installation of Management Server can be done with RPM or DEB packages.

```
#apt-get install cloud-client
```

6.2.3 Install Database on Management Server

1. Install MySQL with following command

```
#apt-get install mysql-server
```

complete the MySQL installation by setting up the root database user's password.

2. Edit MySQL configuration

```
#nano /etc/mysql/my.cnf
```

insert the below lines in [mysqld] section after datadir line

```
innodb_rollback_on_timeout=1
innodb_lock_wait_timeout=600
max_connections=350
log-bin=mysql-bin
binlog-format = 'ROW'
```

3. Now setup the database the below command creates the 'cloud' user in the database.

```
cloud-setup-databases cloud:<dbpassword>@localhost \
          --deploy-as=root:<password> \
          -e <encryption_type> \
          -m <management_server_key> \
          -k <database_key>
```

Alternatively, the required version of CloudStack, tar.gz file can be downloaded from http://sourceforge.net/projects/cloudstack/files/CloudStack%20Acton/3.0.0/ and then untarring the file and running install.sh and choosing M for management server, D for database and A for agent or host setup.

6.2.4 Prepare NFS Share

CloudStack implements the primary and secondary storage using NFS share. The NFS share can be set using following commands:

1. Install nfs-kernel-server on Ubuntu Server

```
#apt-get install nfs-kernel-server
```

2. On storage server create NFS share using command

```
#mkdir -p /export/primary
#mkdir -p /export/secondary
```

3. Now configure the new directories as NFS exports. for this edit the file /etc/exports

```
#nano /etc/exports
```

then add the line

```
/export  *(rw,async,no_root_squash)
```

4. Export the /export directory

```
# exportfs –a
```

5. On Management Server create the mount point of secondary storage

```
#mkdir /mnt/secondary
```

6. Now mount the secondary storage on Management Server

```
# mount -t nfs nfsservername:/nfs/share/secondary /mnt/secondary
```

In our case its

```
# mount -t nfs 172.16.6.15:/nfs/share/secondary /mnt/secondary
```

7. Reboot the system

```
#reboot
```

6.2.5 Prepare VM Template [12]

Template should be created in secondary storage for creation of cloudstack system VMs.
On management server run cloud-install-sys-tmplt command to set template for specific hypervisor which is to be used in cloud infrastructure machine or which are being used with management server.
For KVM:

```
#/usr/lib/cloud/common/scripts/storage/secondary/cloud-install-sys-tmplt
-m /mnt/secondary -u http://download.cloud.com/templates/acton/acton-systemvm-
02062012.qcow2.bz2 -h kvm -F
```

6.2.6 Start Management Server

```
#cloud-setup-management
```

6.3 KVM Hypervisor Host Installation [12][22]

6.3.1 Prepare the Operating System

The Server OS (Ubuntu Server or CentOS) should be prepared first to host the KVM Hypervisor on it. The following steps confirms the fully prepared server OS for KVM Hypervisor.

1. Login to Server OS as Root User
2. Verify the fully qualified hostname of the prepared Server OS with following command

```
#hostname --fqdn
```

   The above command should return the fully qualified domain name of system like "cldstkkvm01.lanslab.edu". If output of the above command is not in the specified format the correct it by editing the file "/etc/hosts".

3. Install NTP for Time Synchronisation

```
#apt-get install opennntpd
```

6.3.2 Install the KVM Agent on the Host

```
#apt-get install cloud-agent
```

6.3.3 Configure KVM

libvirt and QEMU are the two parts of KVM which should be properly configured to get the things running.

QEMU Configuration

Edit the QEMU VNC configuration.

```
#nano /etc/libvirt/qemu.conf
```

uncomment the following line (simply remove # before the following line)

```
vnc_listen=0.0.0.0
```

Configure libvirt

1. For live migration events the libvirt has to listen on unsecure TCP connections. These settings are located in /etc/libvirt/libvirt.conf

Set the following parameters:

```
listen_tls = 0
listen_tcp = 1
```

```
tcp_port = “16059”
auth_tcp = “none”
mdns_adv = 0
```

2. Turning “listen_tcp” 0 in libvirtd.conf is not enough, we have to change the parameters as well:

modify **/etc/init/libvirt-bin.conf**

Change the following line (at the end of the file):

```
exec /usr/sbin/libvirtd -d
```

to (just add -l)

```
exec /usr/sbin/libvirtd -d -l
```

3. Restart libvirt

```
# service libvirt-bin restart
```

6.3.4 Configure Network Bridges

This can be done by editing the file /etc/network/interfaces.

```
#nano /etc/network/interfaces
```

Modify the interfaces file so finally it will look like:

```
auto lo
iface lo inet loopback
# The primary network interface
auto eth0.100
iface eth0.100 inet static
address 172.16.6.17
netmask 255.255.240.
gateway 172.16.1.1
dns-nameservers 172.16.1.1
dns-domain cldstkkvm01.lanslab.edu
# Public network
auto cloudbr0
iface cloudbr0 inet manual
bridge_ports eth0.200
bridge_fd 5
```

```
bridge_stp off
bridge_maxwait 1

# Private network
auto cloudbr1
iface cloudbr1 inet manual
bridge_ports eth0.300
bridge_fd 5
bridge_stp off
bridge_maxwait 1
```

Now restart the network

```
#sudo /etc/init.d/networking restart
```

6.3.5 Configure Firewall

The hypervisor needs to communicate with other hypervisors and the management server needs to reach the hypervisor.

For this purpose we have to open the following TCP ports (if you are using a firewall):

1. 22 (SSH)
2. 1798
3. 16509 (libvirt)
4. 5900 - 6100 (VNC consoles)
5. 49152 - 49216 (libvirt live migration)

To open this ports execute the following commands:

```
$ ufw allow proto tcp from any to any port 22
$ ufw allow proto tcp from any to any port 1798
$ ufw allow proto tcp from any to any port 16509
$ ufw allow proto tcp from any to any port 5900:6100
$ ufw allow proto tcp from any to any port 49152:49216
```

6.4 Cloud Infrastructure Provisioning [12]

6.4.1 Change Root Password

It is the first step after setting up the cloudStack environment. Root administrator is user account which is responcible for managing the cloudstack deployment. Through root administrator modification in the configuration settings in the basic cloudstack is possible.
So, after installing CloudStack it is very important to change the default password to newone which is unique
Steps:

1) Open web browser and open the CloudStack Dashboard's URL which is of the form *http://<management_server_ip:8080>/client*
2) Log in to UI with current root user ID and password being 'password'.
3) Click Accounts
4) Click on the admin account
5) click view users
6) click on the admin user
7) click on the button "Change Password"
8) Type new password and click on the button "OK"

6.4.2 Add a Zone

Before adding a Zone beware to log in to the Cloudstack UI
Steps
From the left navigation pane, Choose Infrastructure
Click on View More in the Zones container
Click on the Add Zone after that Zone creation wizard appear
choose appropriate network type as you wish like, Basic or Advanced
On the basis of choosen network type set respective configuration of that network such as Name, DNS 1 and 2, Internal DNS 1 and Internal DNS 2, Hypervisor, etc.

6.4.3 Add a Pod

In the newly created zone, cloudstack adds the first pod, also more pods can be added later configuration of Pod:

Pod Name which is the uique name for the new pod.
Reserved System Gateway the gateway for the hosts in the pod
Reserved System Netmask the netmask that defines the pod's subnet using CIDR notation
start/End Reserved System IP
This is the IP range which is used by management network of cloudstack for managing various system VMs

6.4.4 Add a Cluster

To add Cluster, create the cluster with configuration which is listed below

1) Hypervisor
   Hypervisor can be one of the type such as vSphere, KVM, XenServer, etc.
2) Cluster Name
   Give a name to newly created cluster a unique identifiable name.

6.4.5 Add more Hosts

New hosts can be added into cluster after its installation and configuration on separate machine. After complete installation of host they can be added to cloudstack infrastructure with configuration setup fields such as
HostName, User Name, Password and Host tags.

6.4.6 Add a Primary Storage

Each cluster has primary storage corresponds to each guest host of cloud.
Primary storage server can be configured by entering the following

Name

The storage device Name.

Protocol.

For XenServer, choose either NFS, iSCSI, or PreSetup.
For KVM, choose NFS, SharedMountPoint,CLVM, or RBD.
For vSphere choose either VMFS (iSCSI or FiberChannel) or NFS.

6.4.7 Add a Secondary Storage

Each zone has secondary storage associated with it.

Secondary storage server can be configured by entering the following

Name

The storage device Name.

Protocol.

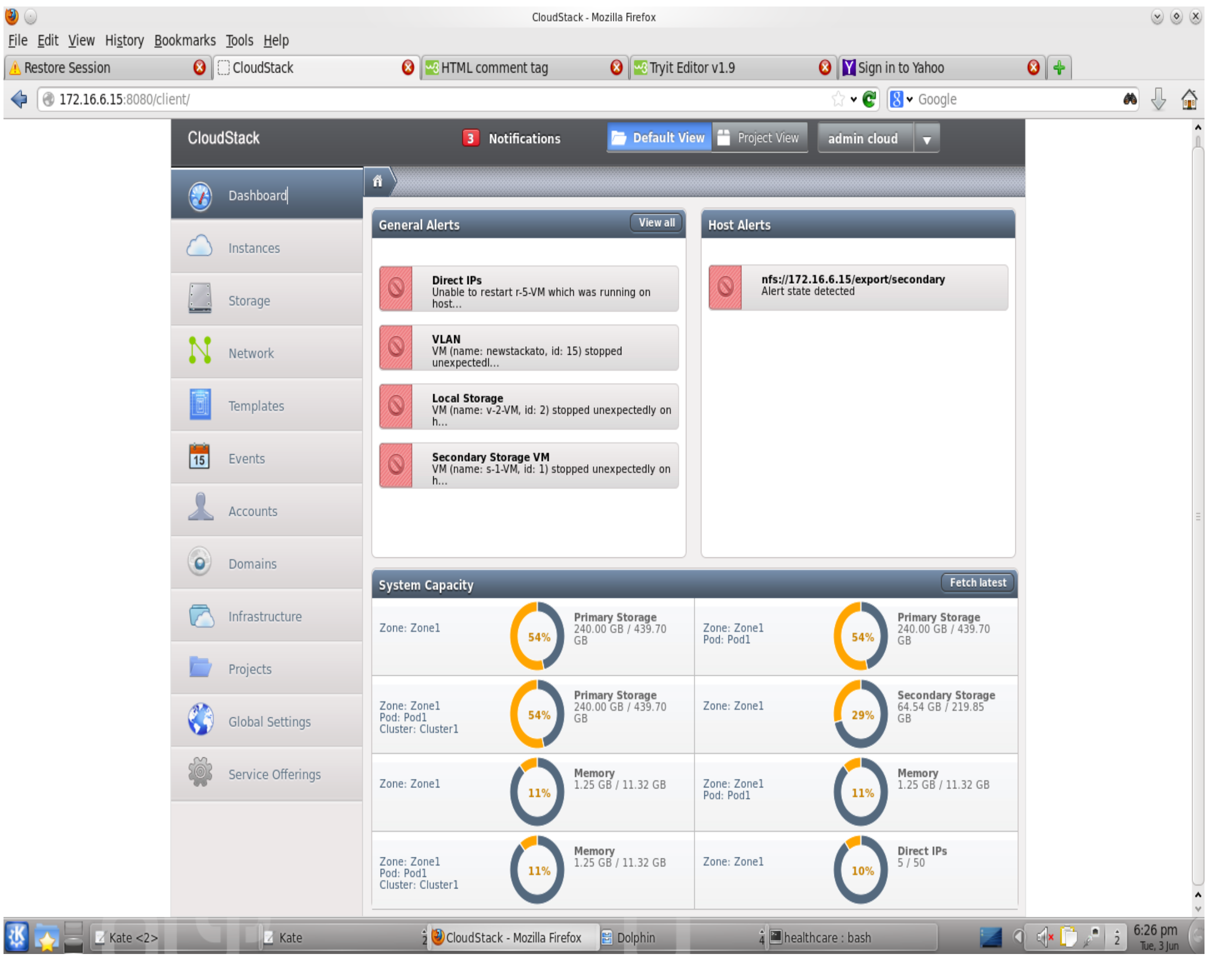
For XenServer, choose either NFS, iSCSI, or PreSetup.

For KVM, choose NFS, SharedMountPoint,CLVM, or RBD.

For vSphere choose either VMFS (iSCSI or FiberChannel) or NFS.

6.4.8 Initialize and Run the new cloud

After installing the complete CloudStack System we have added some OS ISO's which are to be used by guest VM's. Customized templates which has predefined OS's installed in it are also added for quick creation of guest VM's and accessing those VM's through GUI.



General Alerts

Direct IPs
Unable to restart r-5-VM which was running on host...

VLAN
VM (name: newstackato, id: 15) stopped unexpectedl...

Local Storage
VM (name: v-2-VM, id: 2) stopped unexpectedly on h...

Secondary Storage VM
VM (name: s-1-VM, id: 1) stopped unexpectedly on h...

Host Alerts

nfs://172.16.6.15/export/secondary
Alert state detected

System Capacity

Zone: Zone1	54%	Primary Storage 240.00 GB / 439.70 GB	Zone: Zone1 Pod: Pod1	54%	Primary Storage 240.00 GB / 439.70 GB
Zone: Zone1 Pod: Pod1 Cluster: Cluster1	54%	Primary Storage 240.00 GB / 439.70 GB	Zone: Zone1	29%	Secondary Storage 64.54 GB / 219.85 GB
Zone: Zone1	11%	Memory 1.25 GB / 11.32 GB	Zone: Zone1 Pod: Pod1	11%	Memory 1.25 GB / 11.32 GB
Zone: Zone1 Pod: Pod1 Cluster: Cluster1	11%	Memory 1.25 GB / 11.32 GB	Zone: Zone1	10%	Direct IPs 5 / 50

Figure 15: Dashboard of Cloud

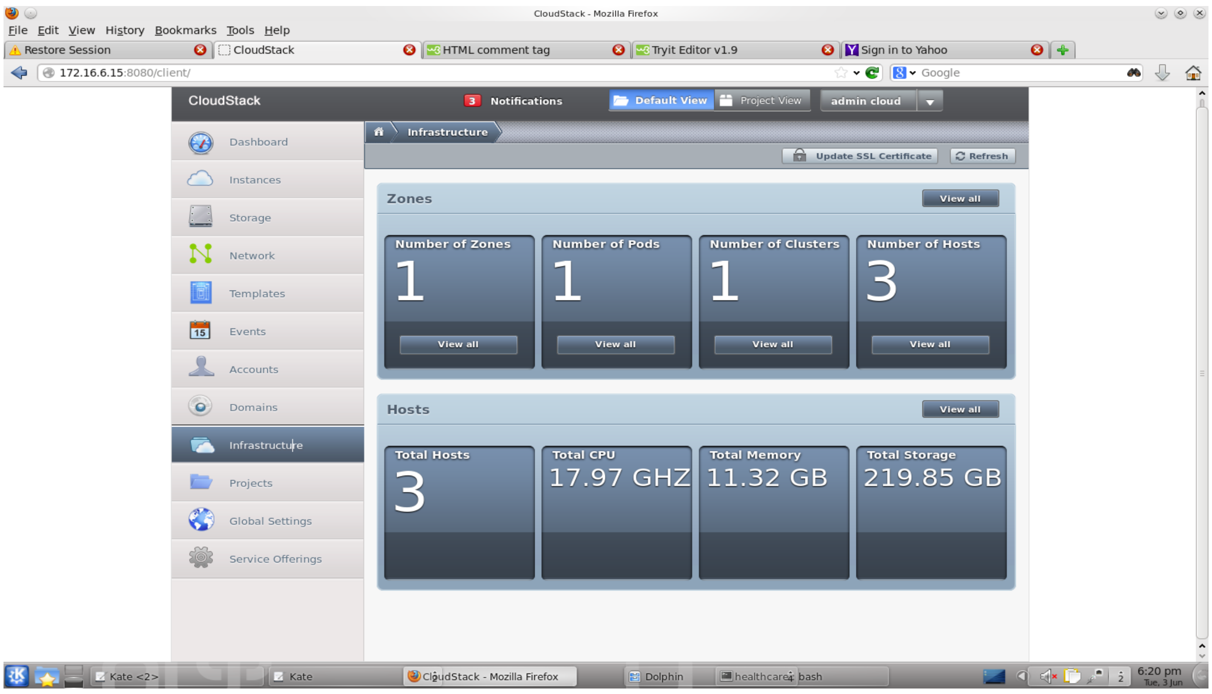
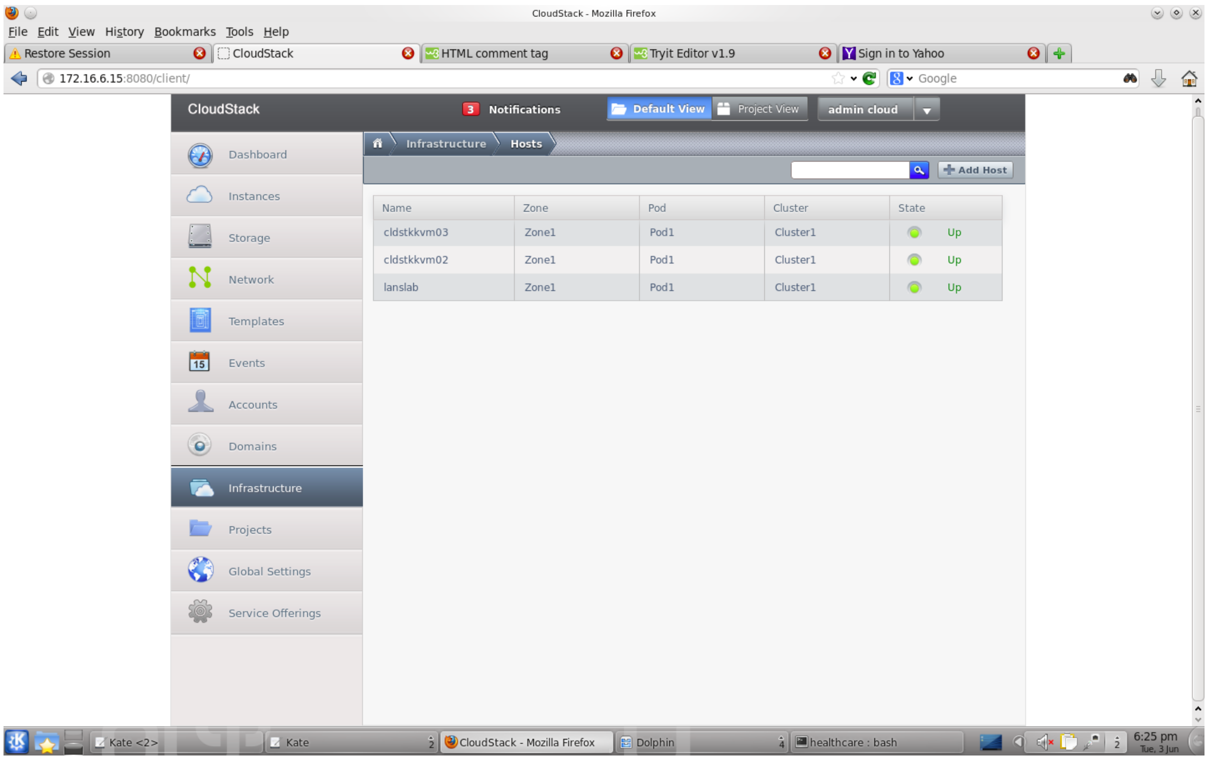
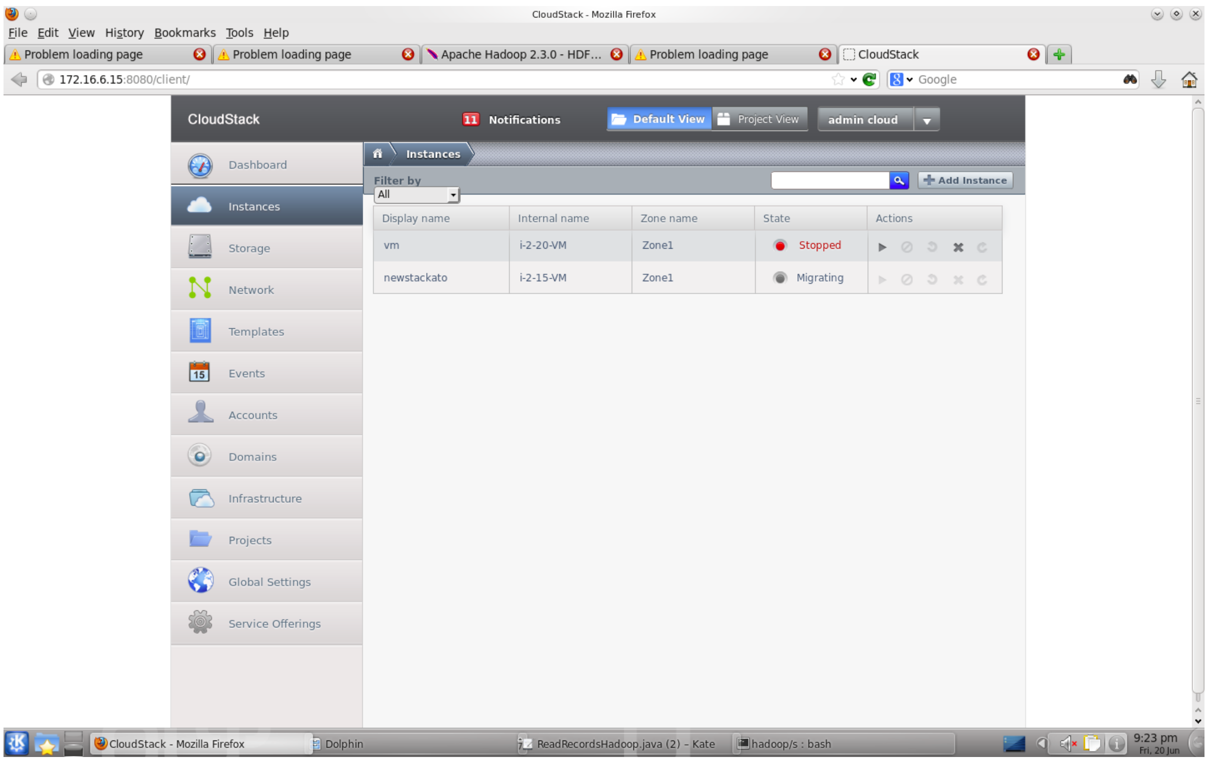


Figure 16: Cloud Infrastructure



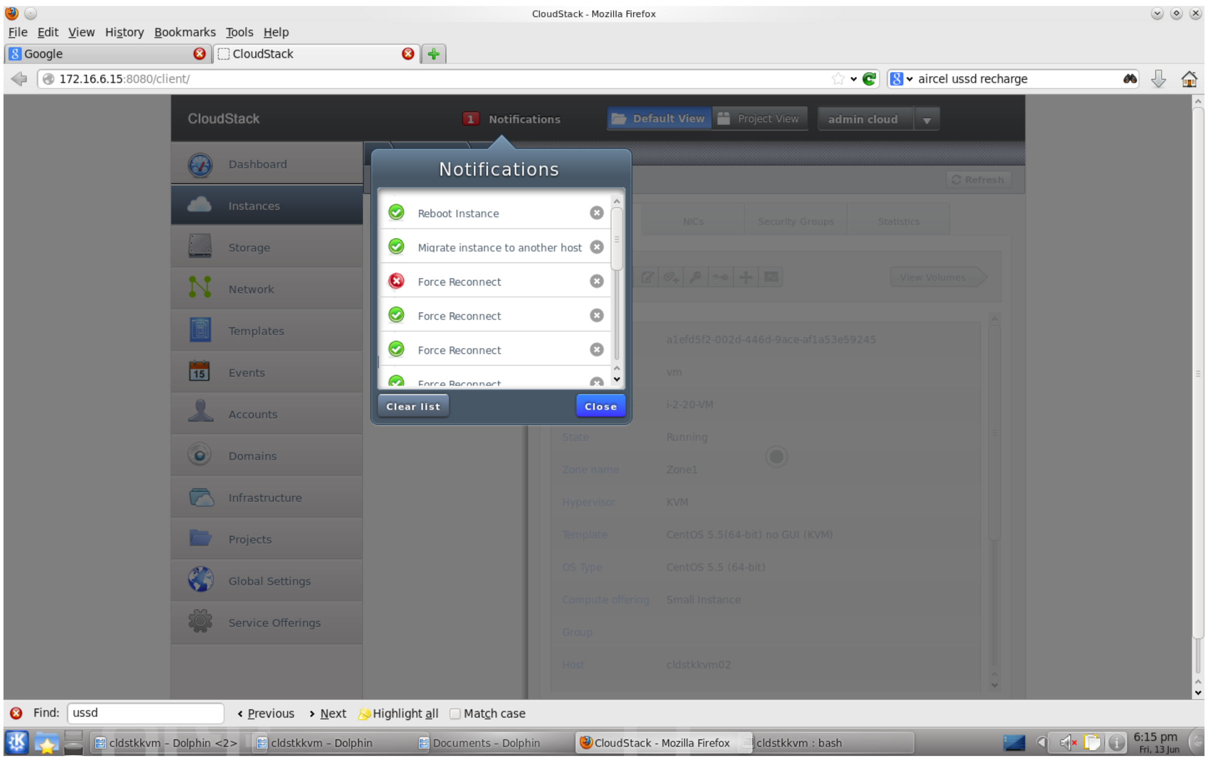
Name	Zone	Pod	Cluster	State
cldstkkvm03	Zone1	Pod1	Cluster1	● Up
cldstkkvm02	Zone1	Pod1	Cluster1	● Up
lanslab	Zone1	Pod1	Cluster1	● Up

Figure 17: List of Hosts



Display name	Internal name	Zone name	State	Actions
vm	i-2-20-VM	Zone1	Stopped	
newstackato	i-2-15-VM	Zone1	Migrating	

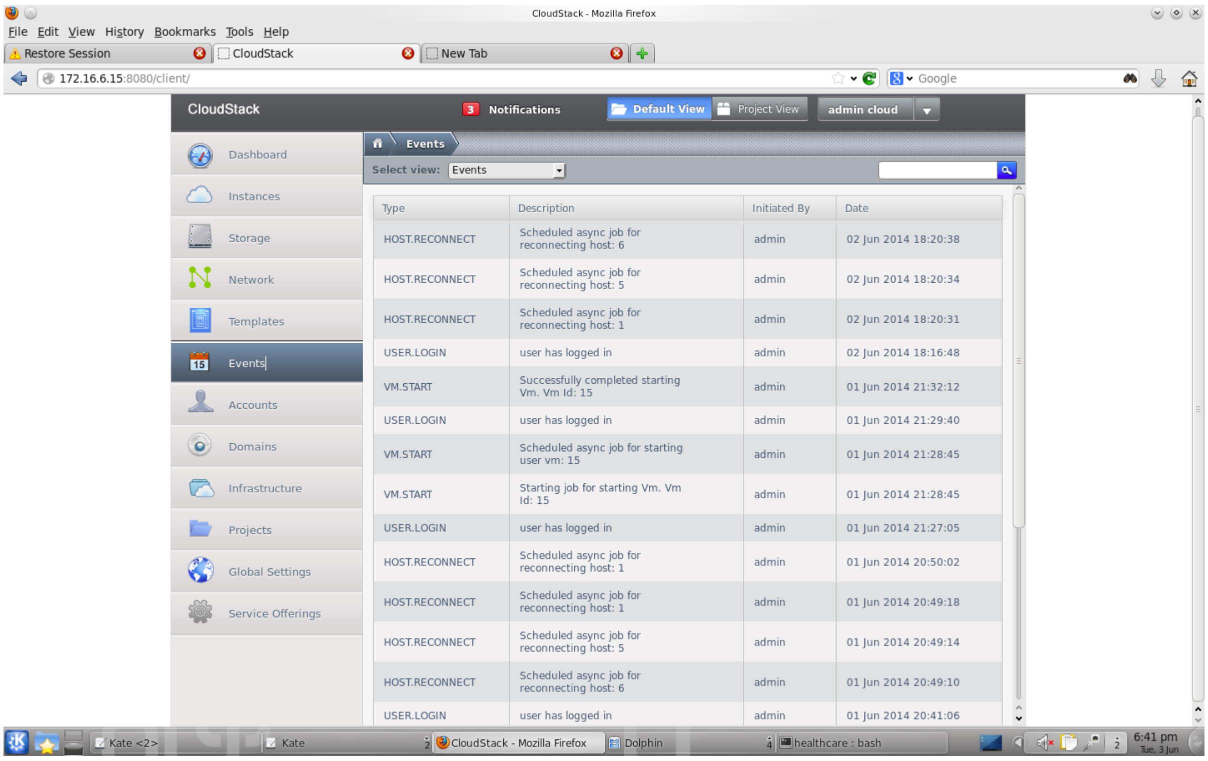
Figure 18: Virtual Machine Migration



Notifications

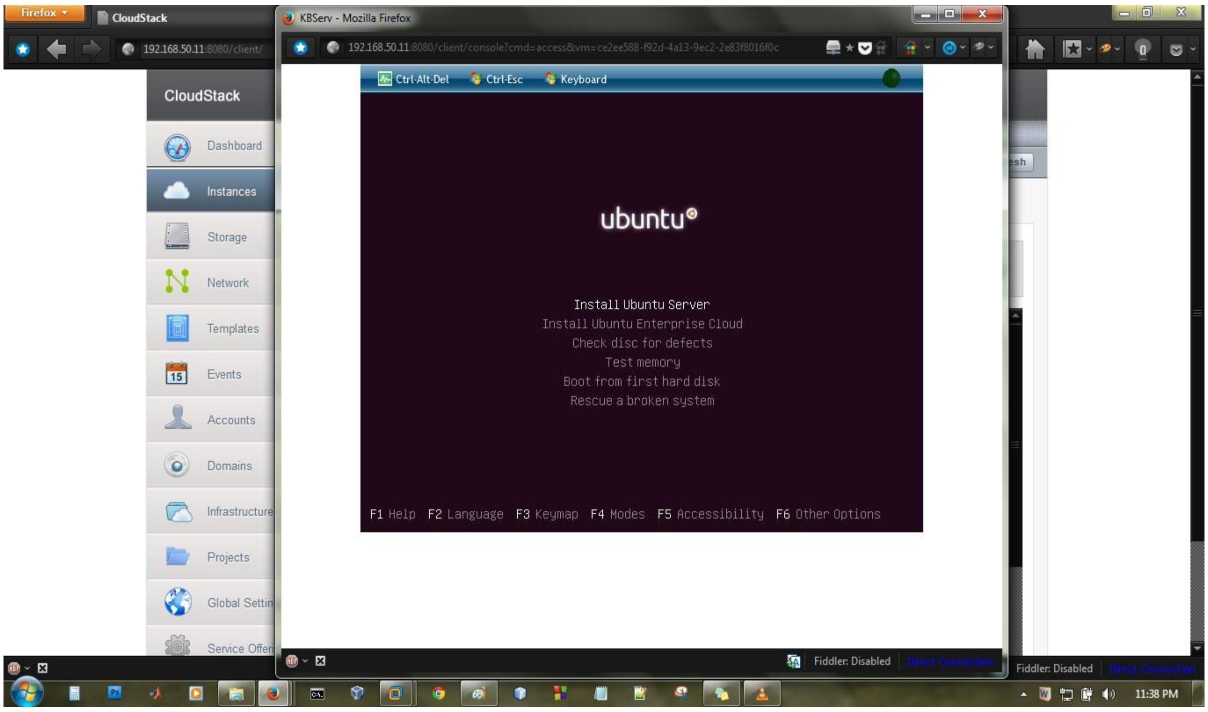
- Reboot Instance
- Migrate instance to another host
- Force Reconnect
- Force Reconnect
- Force Reconnect
- Force Reconnect

Figure 19: Successful VM Migration Notification



Type	Description	Initiated By	Date
HOST.RECONNECT	Scheduled async job for reconnecting host: 6	admin	02 Jun 2014 18:20:38
HOST.RECONNECT	Scheduled async job for reconnecting host: 5	admin	02 Jun 2014 18:20:34
HOST.RECONNECT	Scheduled async job for reconnecting host: 1	admin	02 Jun 2014 18:20:31
USER.LOGIN	user has logged in	admin	02 Jun 2014 18:16:48
VM.START	Successfully completed starting Vm. Vm Id: 15	admin	01 Jun 2014 21:32:12
USER.LOGIN	user has logged in	admin	01 Jun 2014 21:29:40
VM.START	Scheduled async job for starting user vm: 15	admin	01 Jun 2014 21:28:45
VM.START	Starting job for starting Vm. Vm Id: 15	admin	01 Jun 2014 21:28:45
USER.LOGIN	user has logged in	admin	01 Jun 2014 21:27:05
HOST.RECONNECT	Scheduled async job for reconnecting host: 1	admin	01 Jun 2014 20:50:02
HOST.RECONNECT	Scheduled async job for reconnecting host: 1	admin	01 Jun 2014 20:49:18
HOST.RECONNECT	Scheduled async job for reconnecting host: 5	admin	01 Jun 2014 20:49:14
HOST.RECONNECT	Scheduled async job for reconnecting host: 6	admin	01 Jun 2014 20:49:10
USER.LOGIN	user has logged in	admin	01 Jun 2014 20:41:06

Figure 20: Events Log



```
Install Ubuntu Server
Install Ubuntu Enterprise Cloud
Check disc for defects
Test memory
Boot from first hard disk
Rescue a broken system

F1 Help  F2 Language  F3 Keymap  F4 Modes  F5 Accessibility  F6 Other Options
```

Figure 21: Pop-Up window for accessing VM through GUI

Chapter 7

PaaS SOLUTIONS

7.1 Engine Yard

Engine Yard is made using PHP, Node.js and Ruby on Rails. It is projected for web application developers who desire to seize supremacy of cloud computing without worrying about the responsibility of operations management. It provides a set of services on top of AWS. It uses Amazon cloud to runs its platform, so the worth of the PaaS rests more with the management and orchestration than with providing software components. It additionally takes care of major operations tasks such as managing clusters, performing backups, load balancing, managing snapshots, and administering databases.

7.2 OpenShift

Red Hat OpenShift is highly customizable and is provided in three versions:

- OpenShift Enterprise- It is a private PaaS which runs in the data center.
- OpenShift Online- It is a public cloud-based hosting service.
- OpenShift Origin- It is a community open source application hosting platform.

It is based on open source applications and provides user with a wide variety of choice of languages, components and databases. It automates system administration tasks such as scaling, configuration and virtual server provisioning and for code management it supports git repositories.

7.3 App Engine

App Engine is designed by Google using Java, PHP, Go and Python. It is targeted for distributed web applications and developers. It provides a SDK for the four supported languages, it also provides plug-in for Eclipse. The java environment enables it to support other languages that make use of JRE. The PaaS offers runtime environments and managed infrastructure that can be scaled, but only when the applications fits the restrictions set by the App Engine.

7.4 Heroku

Heroku supports Java, Python, Node.js, Ruby, Clojure and Scala. The PaaS provides dynos (abstract computing environments), which are virtualized Unix-style containers that are responsible for running processes in an isolated environment. These are broadly classified in two,

- Web Dynos- which responds to all the HTTP requests.
- Worker Dynos- which responds to all the task requests.

Heroku gives best result for those applications that fits well into Twelve Factor App methodology.

7.5 <u>AppFog</u>

It is a multi framework and multi language platform which is suitable for multi-cloud deployments, which also includes private clouds. AppFog PaaS supports seven language namely Java, Python, PHP, Ruby, Erlang, Node and Scala. It offers four services namely RabbitMQ, Redis, PostgreSQL and MySQL along with third party add-on services.

7.6 <u>Azure</u>

Microsoft Windows Azure is diminishing the lines between infrastructure and platform as a service. Cloud Services provided by Windows Azure supports languages such as PHP, .NET, Java, Python, Node.js and Ruby. Application administration is done through the dashboard provided by Windows Azure or optionally it can also be done by a cli. In addition to SDK, one can also use Visual Studio by Microsoft for creating applications which can be deployed over the cloud. It also provides choice between Persistent storage namely SQL Database, Tables and Blobs.

7.7 <u>Caspio</u>

Caspio aims on carrying the desktop database like functionality to the cloud. It does not provide a fully useful software development environment. It is intended for creating databases that can be used for providing data entry forms and generating reports. Its visual development tool (point and click) minimizes coding efforts.

7.8 <u>Stackato</u>

ActiveState Stackato is a secure, stable and commercially supported enterprise Platform-as-a-Service (PaaS) that is built with and on top of various open source packages including Cloud

Foundry and Docker. Stackato can deploy applications written in Java, Python, Clojure, Dotnet, Go, Node, Perl, Php, Python and Ruby. It supports services like Filesystem, Memcached, Microsoft Sql Server, Mongodb, Mysql, Postgresql, Rabbitmq and Redis.

7.9 Comparison Table

Property	**AppFog**	**App Engine**	**Heroku**	**OpenShift**	**Stackato**
Vertical Scaling	Yes	No	Yes	Yes	Yes
Horizontal Scaling	Yes	Yes	Yes	Yes	Yes
Auto Scaling	No	Yes	No	Yes	Yes
Java	Yes	Yes	Yes	Yes	Yes
Clojure	No	No	Yes	No	Yes
Node	Yes	No	Yes	Yes	Yes
Perl	No	No	No	Yes	Yes
Python	No	Yes	Yes	Yes	Yes
Go	No	Yes	No	No	Yes
Php	Yes	Yes	Yes	Yes	Yes
Ruby	Yes	No	Yes	Yes	Yes
Scala	No	No	Yes	No	No
Dotnet	No	No	No	No	Yes
BuidPacks	No	No	No	Yes	Yes
Jenkins	No	No	No	Yes	No
Mongodb	Yes	No	No	Yes	Yes
Mysql	Yes	No	No	Yes	Yes
Postgresql	Yes	No	Yes	Yes	Yes
Filesystem	No	No	No	No	Yes
Memcached	No	No	No	No	Yes
Microsoft Sql Server	No	No	No	No	Yes
Rabbitmq	Yes	No	No	No	Yes
Redis	Yes	No	No	No	Yes

GoogleCloud Datastore	No	Yes	No	No	No
Google Cloud Sql	No	Yes	No	No	No
Google Cloud Storage	No	Yes	No	No	No
Hosting	Public and Private	Public	Public	Public and Private	Private

Table 4: Comparison between PaaS [27][32][36][37][38]

Chapter 8

STACKATO

Stackato[36] is a Platform-as-a-Service commercially supported by ActiveState that incorporates open source components such as Dockers and Cloud Foundry. It runs on top of cloud infrastructure, as a application middleware and automatically configures the requirements such as web frameworks, language runtimes, messaging and data services. It also provides administrators with facility to configure and monitor user roles, scaling, application components, and memory usage through the web interface or command line.

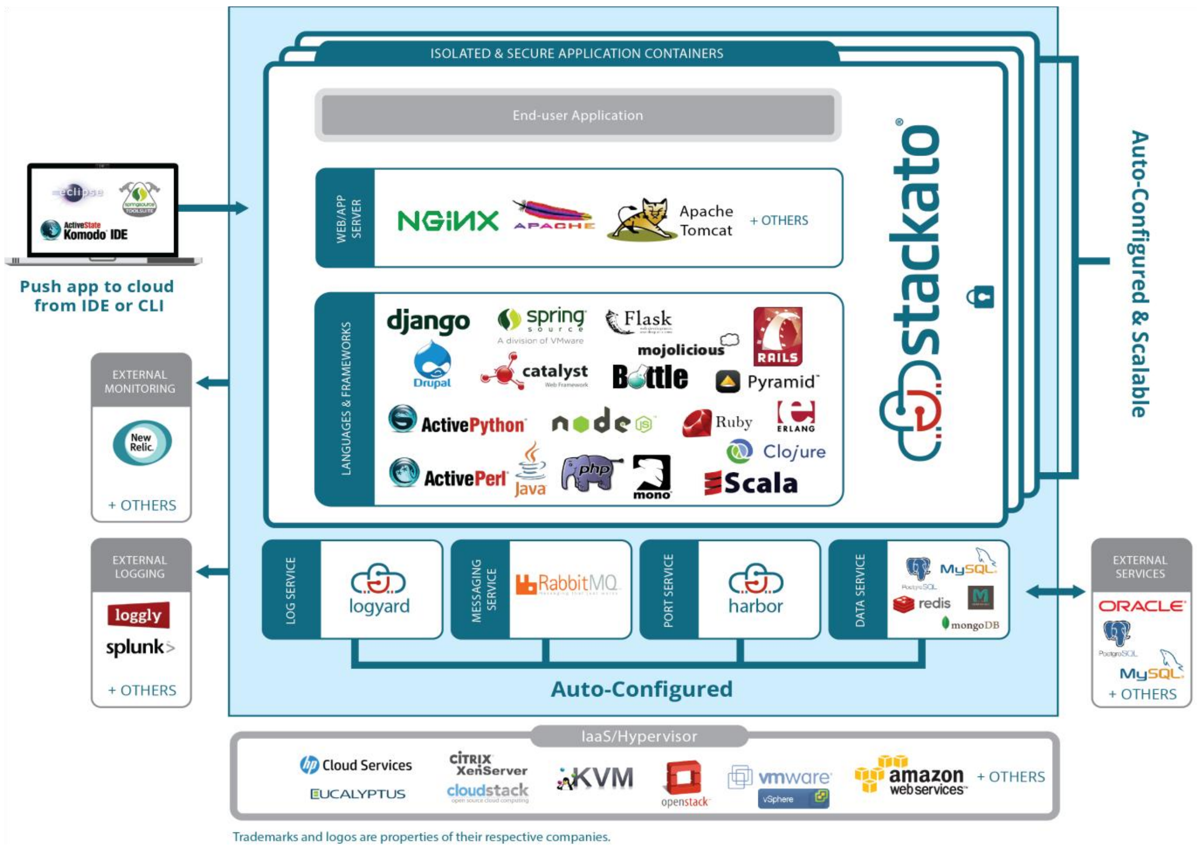


Figure 22: Stackato Platform Overview [36]

8.1 Stackato Architecture [40]

8.1.1 Cloud Controller

It is responsible for managing all the components of the system. The Health Monitor attached to it is used to keep track of availability of Droplet Execution Agents. Communication between Cloud Controller and Stackato VMs is done over NATS, which is a lightweight distributed queuing messaging and publish-subscribe system. This system handles the synchronization of the components and establishes other communication channels between the components, like TCP connections between databases and applications. Cloud Controller is connected to all the VMs through NATS during cluster configuration.

8.1.2 Router

It is responsible for mapping of application instances running on the Droplet Execution Agents to application URLs. When a user tries to connect to the application via web, then he/she is transparently redirected to an internal URL and port. All the connections from Stackato's client are routed to the Cloud Controller.

8.1.3 Droplet Execution Agents (DEAs)

Droplet Execution Agents are the worker nodes of the system. Each DEA hosts many apps within separate Linux containers. From the Cloud Controller application droplets are pulled which are then launched inside a pre-allocated Linux container. In case of unresponsiveness of DEA, the Health Manager which monitors DEA, notices the unresponsive behaviour of DEA and notifies the Cloud Controller, which then redeploys the assigned applications to another healthy Droplet Execution Agents.

8.1.4 Services

Cloud Controller can automatically provision database, messaging, file system, and other services. A special environment variable which shows the connection information in the application container bounds the services to the applications. Any deployed user application can be bound to any services requested by that user, so this allows multiple applications to bind to a service and vice versa. Services can share VMs, or can be run on separate VMs depending on the requirement. Provision for using external services by applications is also available. For example, if an existing external database cluster connected to the Droplet Execution Agents, then the applications can connect directly to that database, as it would in a traditional scenario.

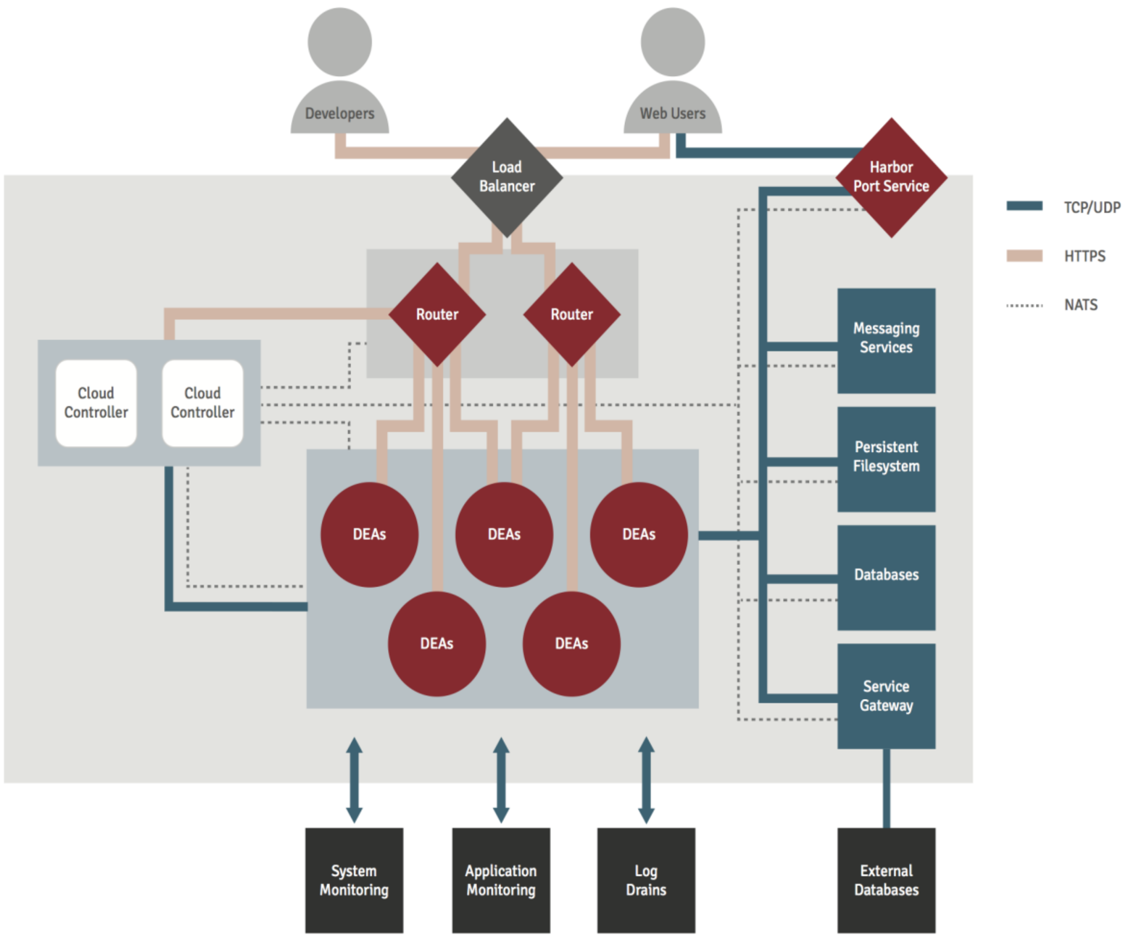


Figure 23: Stackato Architecture [40]

8.2 <u>Stackato Security [41]</u>

The security at platform level is equally important as at infrastructure level. Stackato has been designed with security as its prime focus to protect applications from being breached and bringing down the cloud completely.

8.2.1 Docker Containers

Docker is used for Stackato's LXC (Linux Containers), which ensures that users' applications are safe and secure. Docker containers enable customers to deploy applications in a safe way. No two applications can interact with each other on the PaaS unless they are intentionally allowed to interact with each other. The isolated application only sees its own files and processes.

8.2.2 AppArmor

To provide an extra layer of security each container runs AppArmor which is an alternative to SELinux. It prevents an intruder from breaking out of the container, in a scenario where the intruder is somehow able to access the root level of the container.

8.2.3 Secure Sockets Layer

Stackato supports Secure Sockets Layer across the stack as well as the API itself is accessed over HTTP Secure (HTTPS) by default. Stackato is delivered with a self-signed certificate as SSL requires a certificate on the server. This certificate can be updated at any point of time.

8.2.4 Secure Shell

Direct access to the container is provided by SSH (Secure Shell). It provides complete access to the process space, file systems, environment, hostname and network. Stackato’s SSH support enables the user to SSH into the container for environment monitoring, for troubleshooting purpose and low-level debugging. Any changes made in one container will not impact any other running containers.

8.2.5 Secure Copy

SCP (Secure Copy) is also fully supported by Stackato, it allows safe transferring of files to and from the container. Any changes made will not persist and are specific to the container. The container is ephemeral, so once the container goes away, all the changes will go away with it. The application instances, running in containers, should not store any state information, as this will restrict that application’s ability to scale beyond a single instance. State information should be the domain of the provisioned data-services that Stackato provides.

8.2.6 Database Operations via dbShell

Stackato gains access to the underlying data via SSL tunnel, which is created to access an interactive shell (MySQL, MongoDB, PostgreSQL). This feature is used to import data into the databases.

8.2.7 Privileged Access

Within user’s application containers he/she can be granted sudo privileges, which allow total access to install any software or packages within the container. Due to this uncontrolled power, this feature is limited to trusted users only. Admin can revoke or grant sudo privileges to the users through Stackato’s Web Console.

8.3 Advantages

The PaaS manages infrastructure resources on its own, provide better infrastructure utilization and eliminate time consuming configuration tasks such as from manual configuration of individual application environments. It provides a platform that enables developer to provide their applications as service offerings. The process of application management, deployment and scaling is simplified by PaaS which increases developer’s productivity. Development teams are able to get a quick access to application hosting service, which automatically assembles the software required by the application at runtime.

Resources sharing between applications can be easily done this reduces the number of VMs required which in turn reduces the cost. Most of the Private PaaS such as Stackato provide a central place (web console) from where all the applications can be managed, eliminating the concern of being outside IT governance.. Stackato is independent from the underlying infrastructure. VM images for different hypervisors exists so if user changes virtualization platforms, or move from a private cloud to a public IaaS, than one can easily create a new PaaS cluster over the new infrastructure and can import all user and application data.

And finally, using Stackato to build one’s own PaaS keeps applications and data where they should be, i.e. under the direct control of the organization.

Chapter 9

STACKATO DEPLOYMENT

9.1 Minimum Host System Requirements

	Minimum Requirements
CPU	64-bit x86 with VT-x enabled (x86 virtualization).
RAM	3GB
HDD	20GB

Table 5: Host System Requirements

The Stackato VM uses 2GB of memory by default. So there should be sufficient memory remaining on the host system to run the host operating system and any other applications required.

9.2 Micro Cloud VM

By default, the Stackato VM starts as a single node "micro cloud" with a base set of roles already running and ready to use. This can be used as a test bed for pushing applications still in development. If program deploys successfully to a Stackato micro cloud, it will deploy to any Stackato PaaS of the same version. The application hosting environment is the same in all aspects except scale.

1) Download a Stackato VM in the KVM hypervisor format.
2) Convert the image to qcow2 format

```
$ qemu-img convert -f raw -O qcow2 stackato.img stackato.qcow2
```

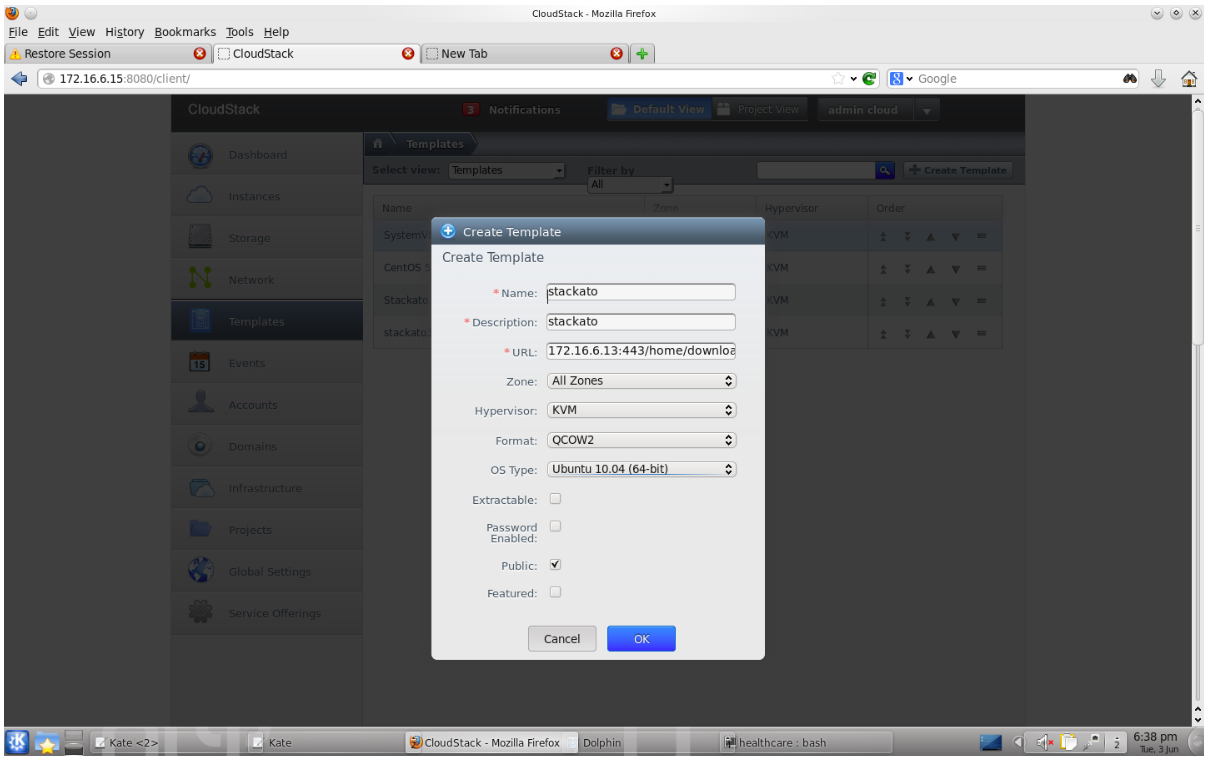
3) cd to the directory where staccato.qcow2 image is stored and run the following command to make a web server

```
$ python -m SimpleHTTPServer 8080
```

9.3 Create Stackato Template

A CloudStack Template of Stackato will enable us to create instances of stackato very easily.

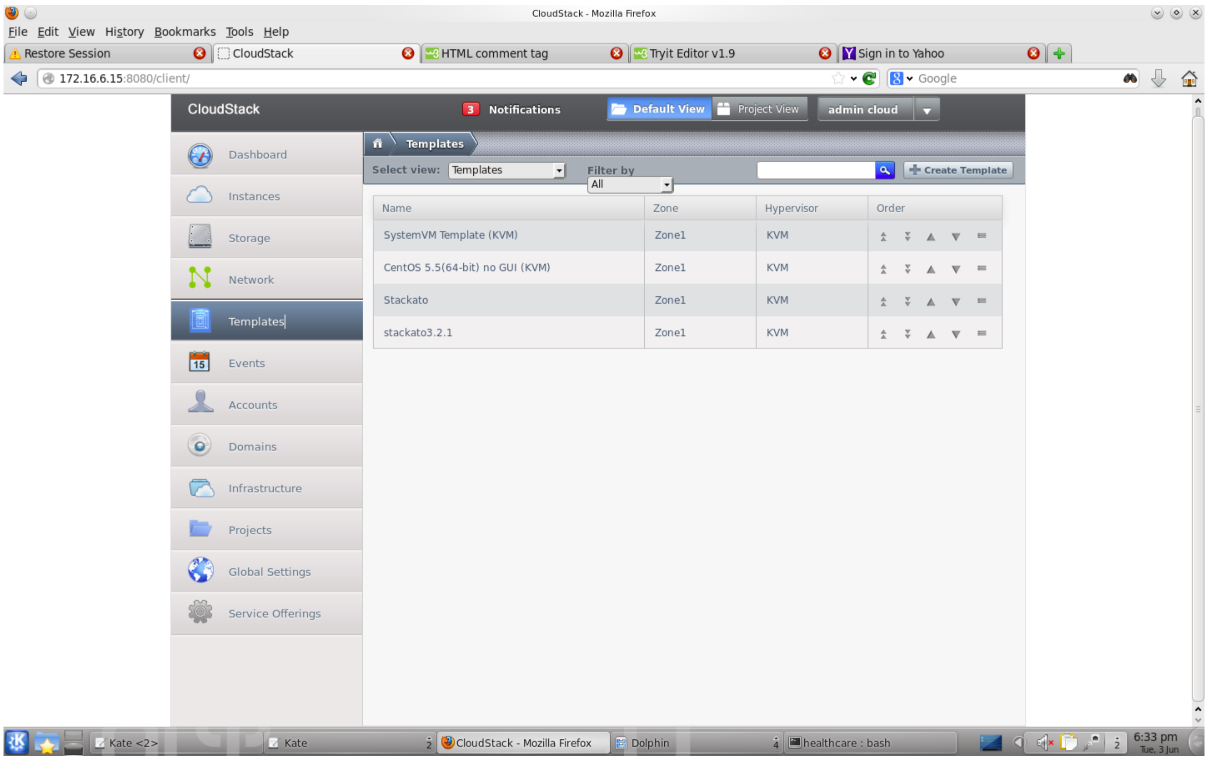
1) Open Cloudstack console (xxx.xx.x.xx:8080/client) and navigate to Templates tab and click on create template.
2) In the URL field, specify the URL where the staccato.qcow2 image is hosted.



Create Template

*Name:	stackato
*Description:	stackato
*URL:	172.16.6.13:443/home/downloa
Zone:	All Zones
Hypervisor:	KVM
Format:	QCOW2
OS Type:	Ubuntu 10.04 (64-bit)
Extractable:	☐
Password Enabled:	☐
Public:	✓
Featured:	☐

Figure 24: Stackato Template

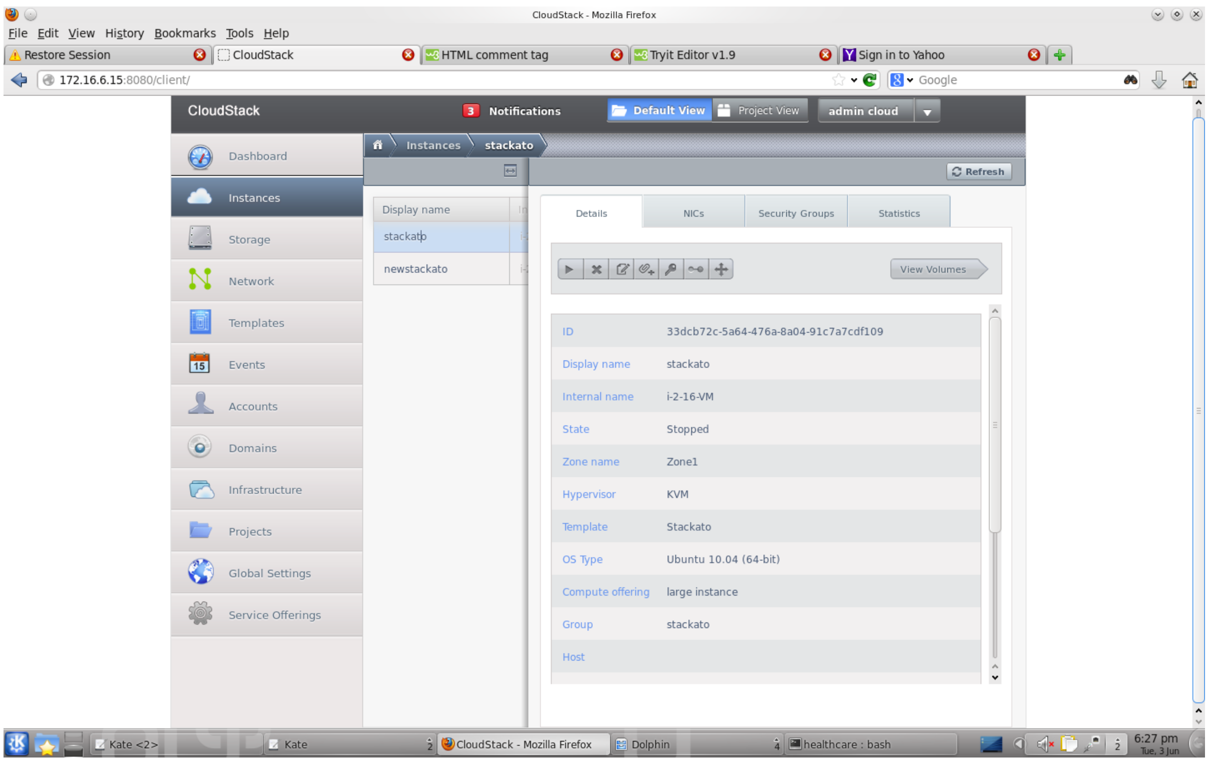


Name	Zone	Hypervisor	Order
SystemVM Template (KVM)	Zone1	KVM	
CentOS 5.5(64-bit) no GUI (KVM)	Zone1	KVM	
Stackato	Zone1	KVM	
stackato3.2.1	Zone1	KVM	

Figure 25: List of Templates

9.4 Create Stackato Instance

1) Open Cloudstack console (xxx.xx.x.xx:8080/client) and navigate to Instances tab and click on add instance.
2) Choose template option, and select the stackato template.
3) Choose the large instance (3 GB) in compute offering.
4) Choose medium disk (20 GB) in data disk offering.

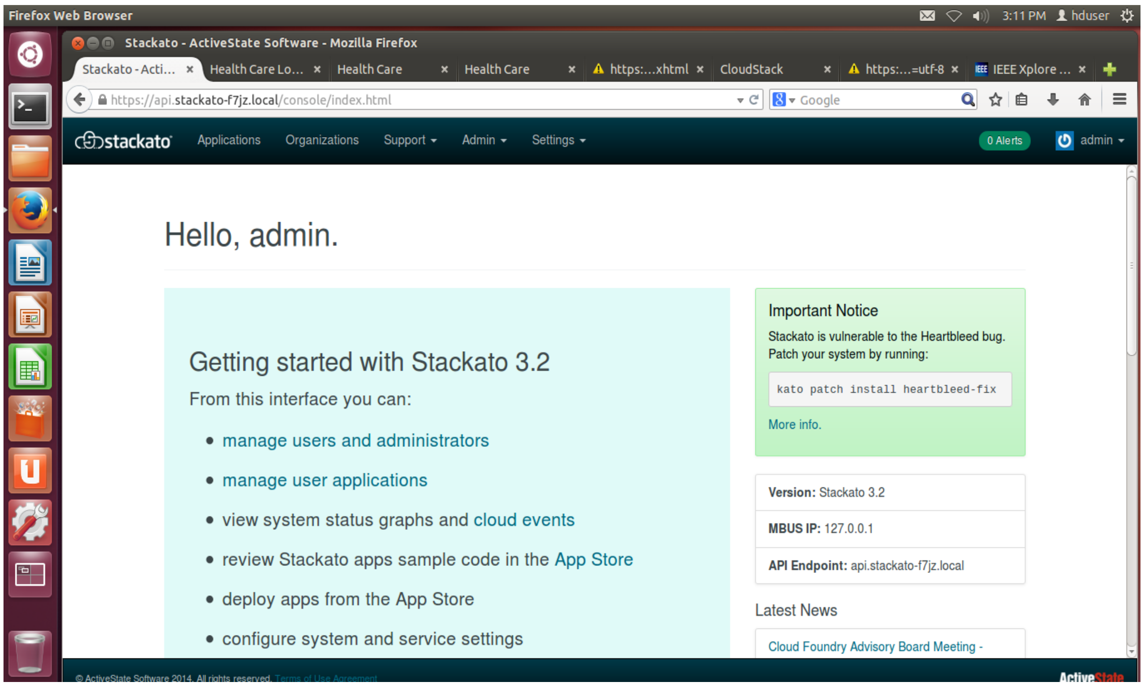


ID	33dcb72c-5a64-476a-8a04-91c7a7cdf109
Display name	stackato
Internal name	i-2-16-VM
State	Stopped
Zone name	Zone1
Hypervisor	KVM
Template	Stackato
OS Type	Ubuntu 10.04 (64-bit)
Compute offering	large instance
Group	stackato
Host	

Figure 26: Stackato Instance

9.5 Stackato Management Console

1) Open Stackato console (staccato-xxxx.local) and create the first admin user and first organization for the system.
2) Use the above credentials for creating new users, organizations, system configuration and pushing applications.



Hello, admin.

Getting started with Stackato 3.2

From this interface you can:

- manage users and administrators
- manage user applications
- view system status graphs and cloud events
- review Stackato apps sample code in the App Store
- deploy apps from the App Store
- configure system and service settings

Important Notice

Stackato is vulnerable to the Heartbleed bug. Patch your system by running:

```
kato patch install heartbleed-fix
```

More info.

Version: Stackato 3.2

MBUS IP: 127.0.0.1

API Endpoint: api.stackato-f7jz.local

Latest News

Cloud Foundry Advisory Board Meeting -

© ActiveState Software 2014. All rights reserved. Terms of Use Agreement

Figure 27: Stackato Management Console

9.5.1 Adding Organizations

Upon creating the first user staccato automatically sets up first organization, if one wish to add more organizations then it can be done by logging into management console and going to organization view.

9.5.2 Adding Spaces

Minimum of one space is required for deploying application. This can be done by clicking on add space in organization view.

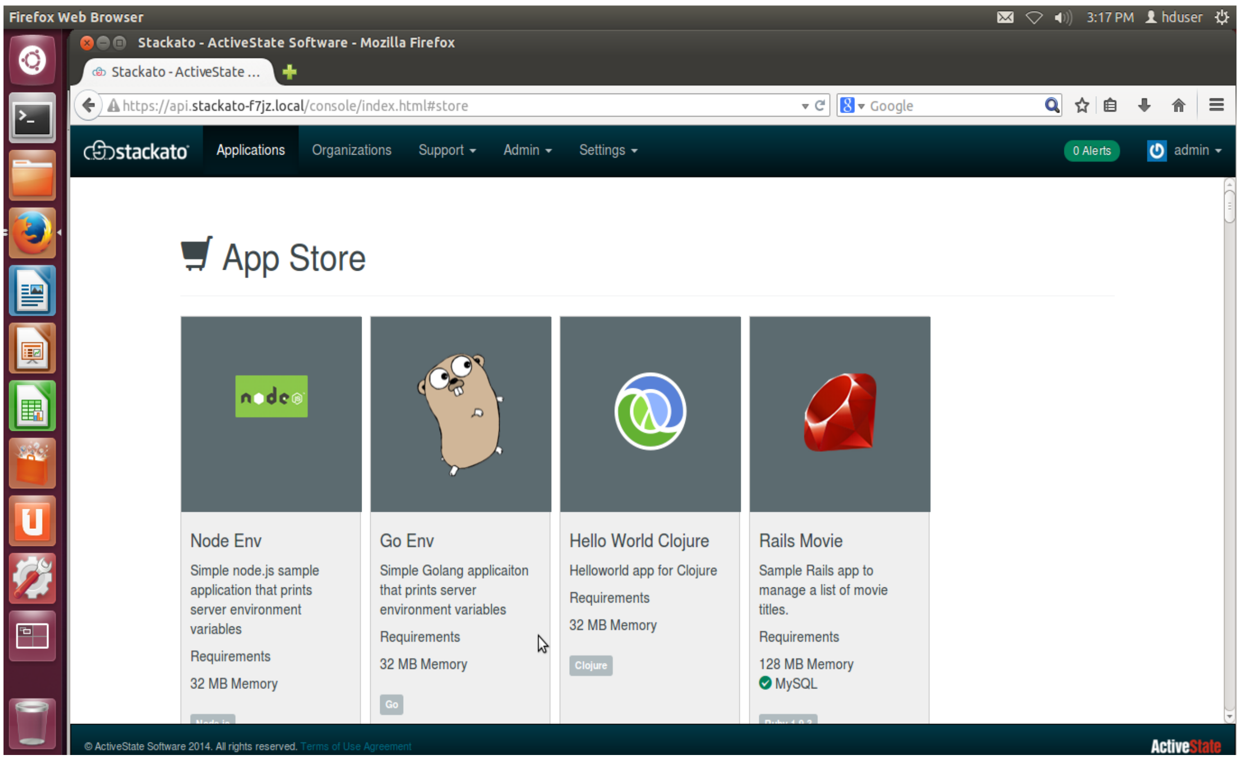
9.5.3 Adding Users

More users can be added from web interface or command line interface

1) Login to management console and go to User section and add user with its role.
2) $ stackato add-user username [--passwd password]

9.6 App Store

Stackato App Store consists of many ready to run applications which can be easily deployed on the staccato micro cloud. The app store pulls the source code of the application from Git repository.


Firefox Web Browser
3:17 PM hduser
Stackato - ActiveState Software - Mozilla Firefox
Stackato - ActiveState ...
https://api.stackato-f7jz.local/console/index.html#store
Google
stackato
Applications
Organizations
Support
Admin
Settings
0 Alerts
admin
App Store

Node Env
Simple node.js sample application that prints server environment variables
Requirements
32 MB Memory

Go Env
Simple Golang applicaiton that prints server environment variables
Requirements
32 MB Memory
Go

Hello World Clojure
Helloworld app for Clojure
Requirements
32 MB Memory
Clojure

Rails Movie
Sample Rails app to manage a list of movie titles.
Requirements
128 MB Memory
✓ MySQL

© ActiveState Software 2014. All rights reserved. Terms of Use Agreement
ActiveState

Figure 28: Stackato App Store

9.7 Stackato Client

Stackato client helps to interact with the system from command line interface and also to push applications

1) Download the client for the platform (Windows, OS X, Linux x86, Linux x64)
2) Unzip the archive in a suitable directory.
3) Add the executable to system/shell $PATH by:
   - moving it to a directory in the $PATH,
   - creating a symlink from a directory in the $PATH, or
   - creating a shell alias for the executable.
4) Confirm that the client is installed correctly by running stackato help.

9.7.1 Target and Login

1) stackato target command is used to connect client to the PaaS

```
$ stackato target https://api.stackato-xxxx.local
```

2) After targeting stackato authentication is provided by stackato login command

```
$ stackato login
```

9.7.2 Selecting Organization and Space

1) If you have multiple organization then organization can be switched by following command

```
$ stackato switch-org organization_name
```

2) Similarly if multiple space exists then they can be switched by following command

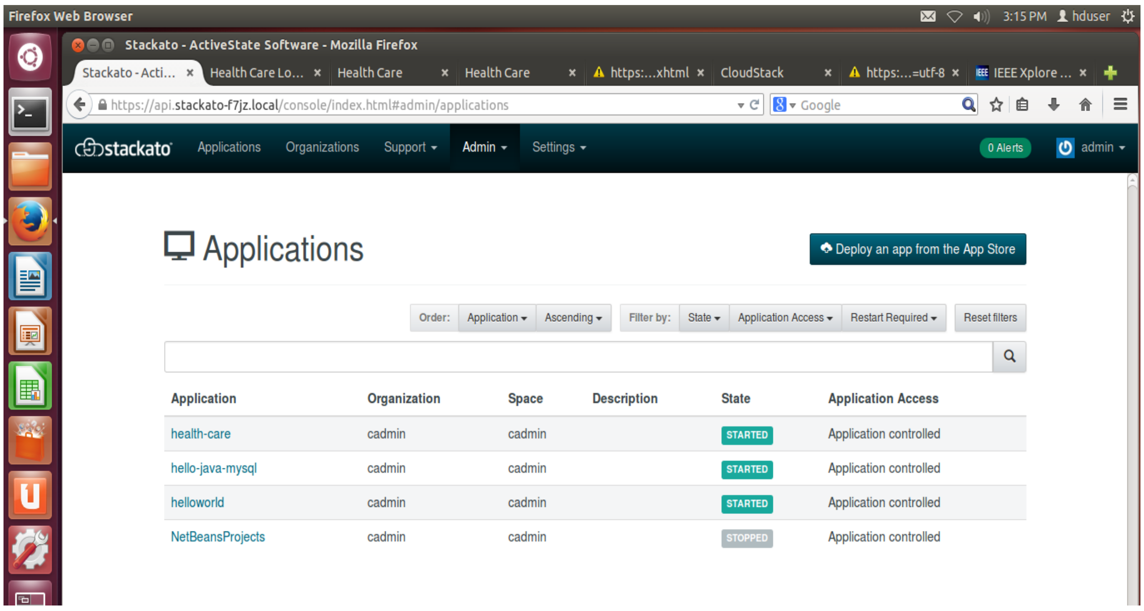
```
$ stackato switch-space space_name
```

9.7.3 Pushing Application Code

Change the current directory to the root directory of source code and use stackato push command to deploy the application.

```
$ stackato push
$ stackato push -n
```

The –n option stands for no-prompt and is used if stackato.yml or manifest.yml config file exists in the directory.


Firefox Web Browser
3:15 PM hduser
Stackato - ActiveState Software - Mozilla Firefox
https://api.stackato-f7jz.local/console/index.html#admin/applications
stackato
Applications
Organizations
Support
Admin
Settings
0 Alerts
admin
Applications
Deploy an app from the App Store
Order: Application Ascending
Filter by: State Application Access Restart Required
Reset filters

Application	Organization	Space	Description	State	Application Access
health-care	cadmin	cadmin		STARTED	Application controlled
hello-java-mysql	cadmin	cadmin		STARTED	Application controlled
helloworld	cadmin	cadmin		STARTED	Application controlled
NetBeansProjects	cadmin	cadmin		STOPPED	Application controlled

Figure29: List of Deployed Application

Chapter 10

HEALTHCARE APPLICATION

Health is the biggest asset and most important concern in anyone’s life. Everyone in his or her lifetime faces one or other health related problems. Taking care of one’s health problems through the services provided by doctor, hospitals and any other similar service is the most logical and justified step that one can take. Whenever we talk about health care, we are implicitly referring to all services and goods which are used in one’s recovery. Therefore a good healthcare system always target to provide quality health services to as many people as possible.

An efficient system with reliable patient record, and secure health flow is required for the care to reach to the right patient at right time. In the developing countries like India the health flow in the hospitals is based on an OPD (Out Patient Data) Card. The patients have to register for a doctor standing in long queues. Once the doctor sees a patient, prescription is written manually on the card, or may be suggested to move to another department for further investigations and tests. There is no mapping between the OPD card and the patient. This is a major security flaw as the OPD card of one patient can be utilized by another patient. As all the patient information is written on the OPD card only, then if the OPD card is lost all patient records are lost. There is a need to store information digitally where it can be securely accessed and managed. The records could be either be retained electronically on some external source or retained on a mobile device frequently retained by a patient.

This can be easily achieved by integrating the cloud services in the healthcare system.The elasticity of cloud will allow infinite number of users to access the cloud at the same time. With introduction of cloud computing in health sector there will be substantial workload shift where the local devices no more have to do heavy task while running applications. Instead group of computers which contributes to particular cloud will do that heavy task. This will reduce the need of hardware and software on user's side. The user only has to access cloud's interface through simple web browser and network and then the cloud will take care of the rest [26].

The existing system of healthcare with one application server with database server has issues like [25]

1) These types of server do not provide implicit security which is to be handled explicitly.

2) Also for managing this type of system the organisation needs qualified IT staff.

3) On-demand scaling of system is not possible which ensures the smooth running of application in case of huge number of users is accessing the service.

4) Needs huge investment in procuring the dedicated servers which are costly.
Due to all these reasons, it is not possible to use a single server for the healthcare purpose.

But, with introduction of cloud there are various advantages like [25]
1) The time and efforts required by the IT for managing system will tremendously reduce as cloud provider such as Stackato provides central interface to manage and troubleshoot problems and it also itself configures the run time environment of application. This provides developer with more time for providing a better and quality service to user without worrying about the underlying environment.
2) Scalability provided by cloud easily enables the application to scale when number of user increases. It enables to scale up or scale down the computing capacity as per the requests.
3) Customer has to pay as they use.
4) Inbuilt security features are provided by the cloud for example Stackato provides feature such as apparmor, docker container etc.
6) Load Balancer ensures the proper usage of the underlying resources and no extra care is to be taken for load balancing

10.1 <u>Architecture for Secure Healthcare System</u>

The Secure Healthcare System is comprises of many components with each one of them having their own importance. The key idea behind this system is to takes into consideration of all the security and other healthcare system related issues and use state-of-art technologies to deliver a robust, scalable, secure, manageable and easy to use system. In the proposed system use of various technologies like Hadoop, NFC cards, CloudStack, Stackato and Kerberos are made. In the system patient retains their individual health card which safely stores the patient information digitally and this information is shared only between the authenticated users and doctors. Secure keys which are used for this purpose of authentication are stored on the cloud and the data of all the patient is stored using hadoop.

The key components of the architecture are:

- **OpenMRS** – It is a collaborative open source project to develop software to support the delivery of healthcare in developing countries. It is intended as a platform that can be used by many organizations which eliminates the need of developing a system from scratch instead the existing OpenMRS system can be modified according to one's requirements. Here a hospital information system based on OpenMRS is used, where authorized doctors reviews the registered patients.

- **Hadoop Distributed File System-** It is open source software for large scale processing and storage of data on clusters of hardware. Here it is used for the purpose of storing records and backups.

- **Stackato-** The PaaS is used to deploy the healthcare application on the cloud which accesses a SQL database for retrieval and storage of patient and doctor secure keys. The SQL database retains security hash of the keys of the patient and access control rights of the doctor along with some other patient/ doctor details.

- **e-HealthCard-** The patient health information is retained in his Smartphone in a secure region. Hence, patient carries his/her Health-Card digitally, which justifies the use of term e-HealthCard. The secure region is implemented by an external microSD card.Here two secure elements (SE) – one for doctor and other for patient. Doctor's SE is used just for authentication whereas patient's SE is used for both authentication and storing recent health records.

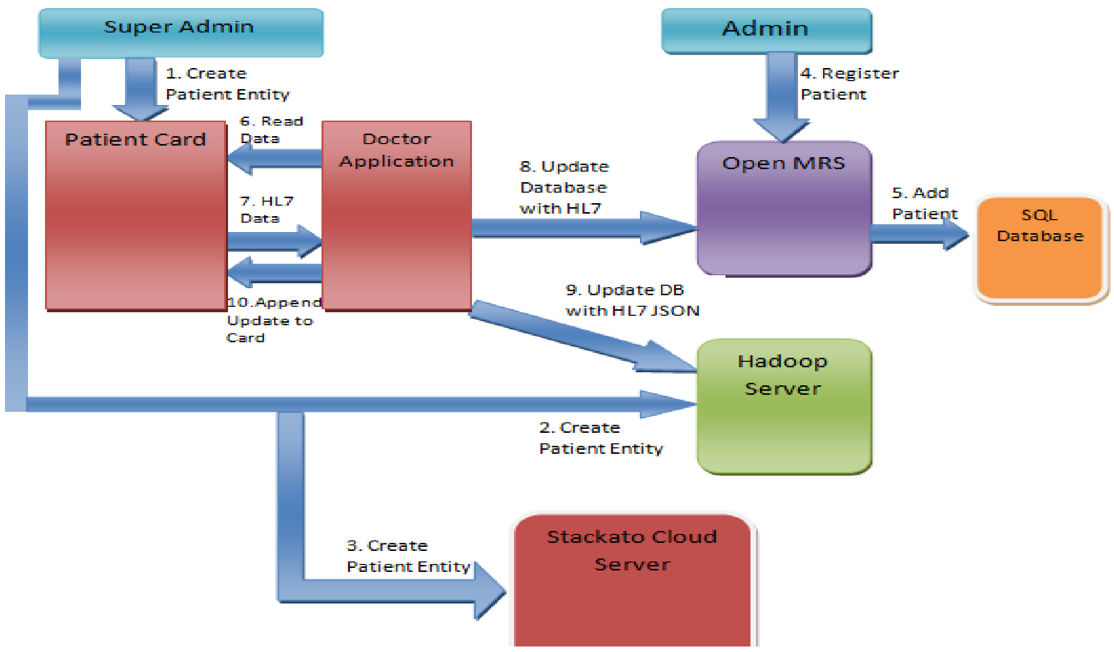


Figure 30: Flow Diagram of Proposed Secure Healthcare System

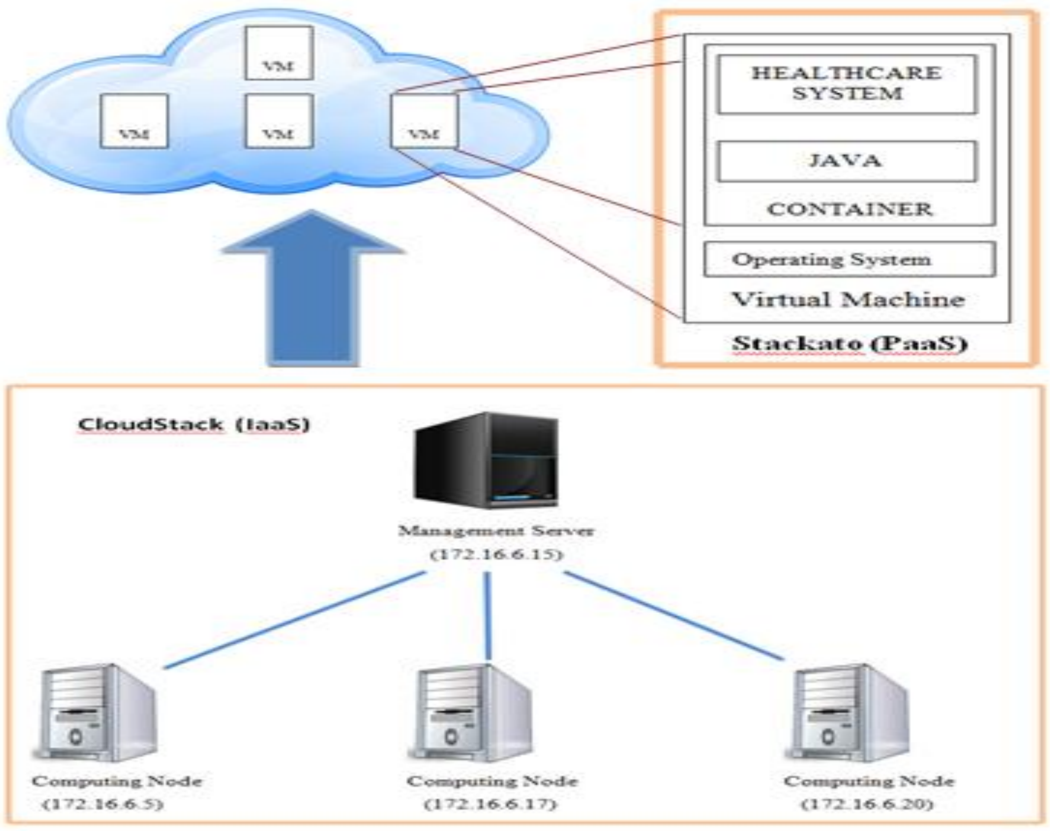


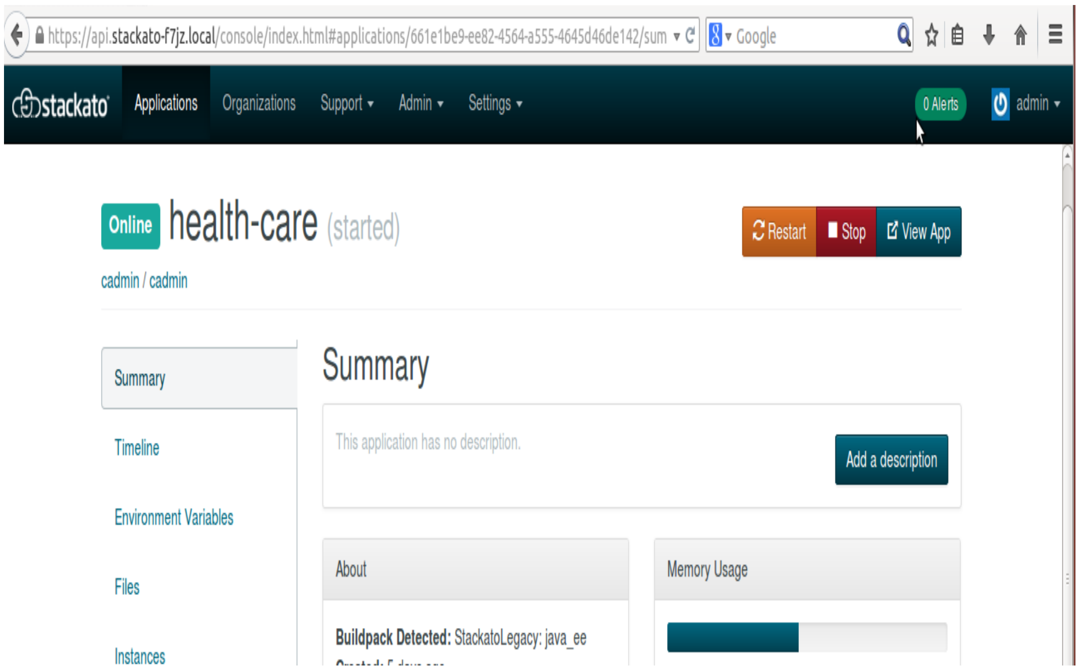
Figure 31: Deployed System Architecture

10.2 <u>Healthcare Application</u>

As shown in figure 30, the super admin needs to interact with the cloud to create new patient entry so in order to provide an interface between the super admin and the cloud a healthcare application is created and deployed on the cloud using Stackato (PaaS).

- The PaaS is used to deploy the healthcare application on the cloud which accesses a SQL database for retrieval and storage of patient and doctor secure hash keys. The SQL database retains hash of the security keys of the patient and access control rights of the doctor along with some other patient/ doctor details. Figure 32, shows the successful deployment of the application on stackato.
- The healthcare web application is made using JSF (Java Server Faces) and it accesses SQL database. Figure 33, shows the login page of the application.
- After successful login as show in figure 34, the application displays the list of all the registered users and also enables the admin to register new users.
- Figure 35, shows the editing option provided by the application to the admin in order to modify the data according to the requirements.

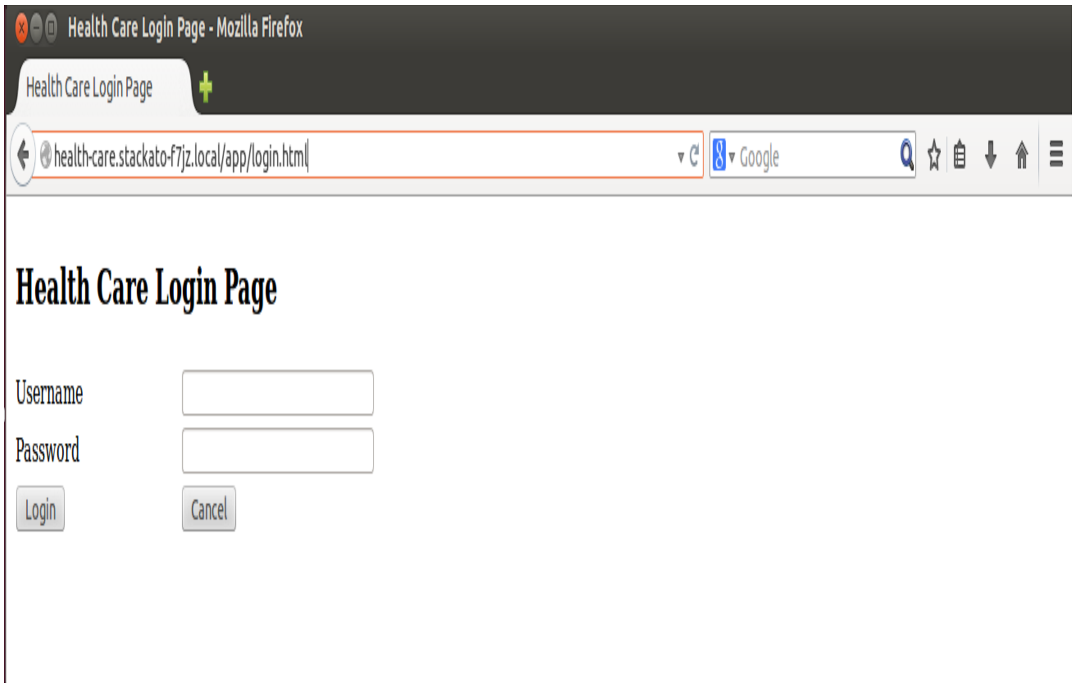
- The details of any particular user can be viewed by clicking on the user_id of the selected user. Figure 36, shows the detail view of selected user.
- Further for security purpose it is required to check the access rights and secure hash key of a user. Figure 37, shows the retrieval of access rights and secure hash key by entering the user id of the user as an input.


https://api.stackato-f7jz.local/console/index.html#applications/661e1be9-ee82-4564-a555-4645d46de142/sum

This application has no description.

Buildpack Detected: StackatoLegacy: java_ee

Figure 32: Healthcare Application Deployed on Stackato


Health Care Login Page - Mozilla Firefox

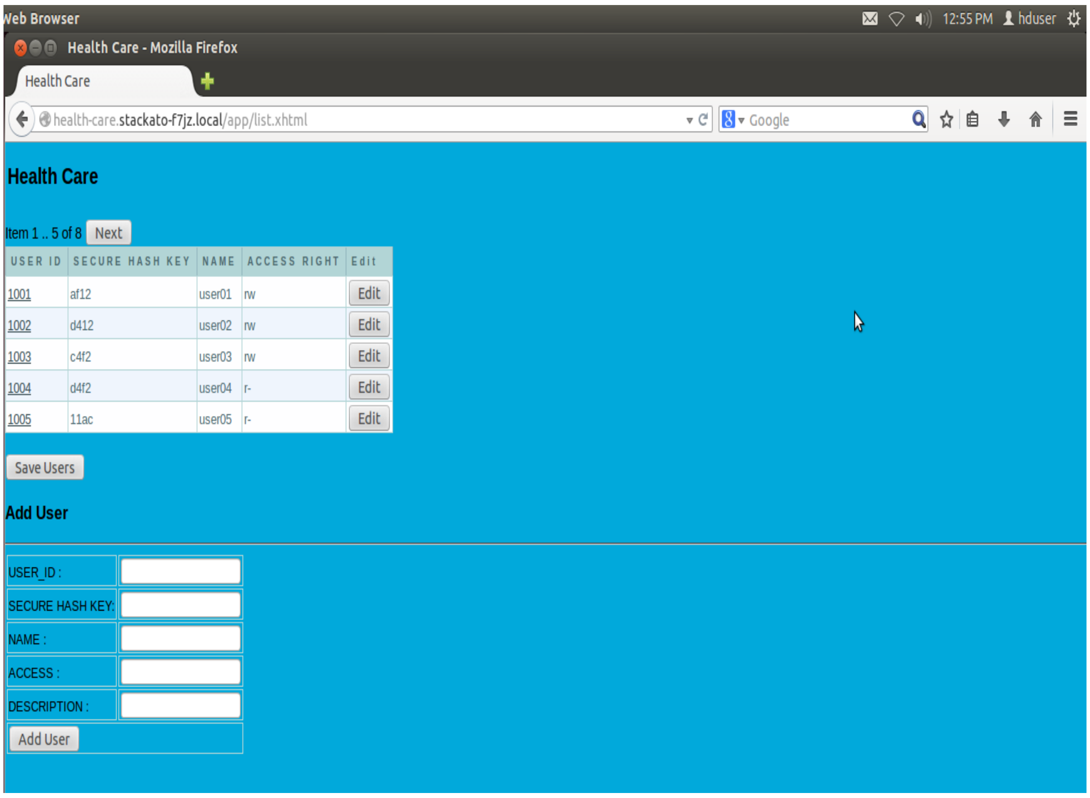
health-care.stackato-f7jz.local/app/login.html

Health Care Login Page

Username

Password

Figure 33: Healthcare Login Page



Health Care

Item 1 .. 5 of 8 Next

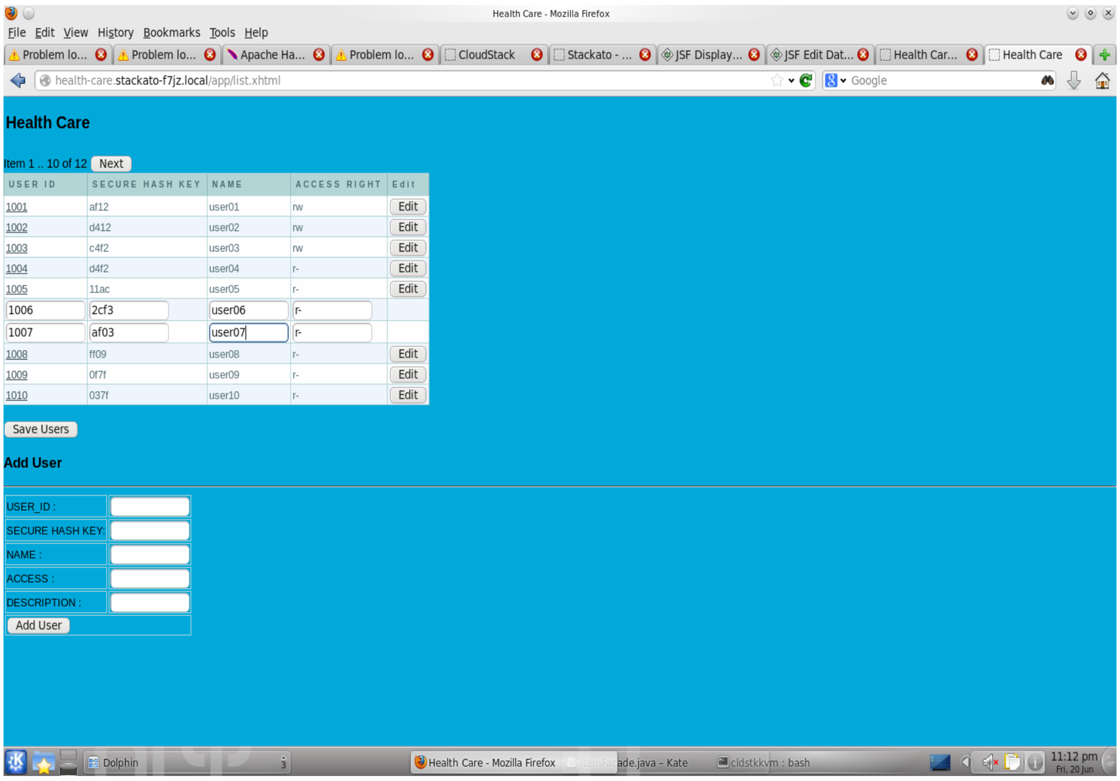
USER ID	SECURE HASH KEY	NAME	ACCESS RIGHT	Edit
1001	af12	user01	rw	Edit
1002	d412	user02	rw	Edit
1003	c4f2	user03	rw	Edit
1004	d4f2	user04	r-	Edit
1005	11ac	user05	r-	Edit

Save Users

Add User

USER_ID :	
SECURE HASH KEY:	
NAME :	
ACCESS :	
DESCRIPTION :	
Add User	

Figure 34: Healthcare View and Add Data



Health Care

Item 1 .. 10 of 12 Next

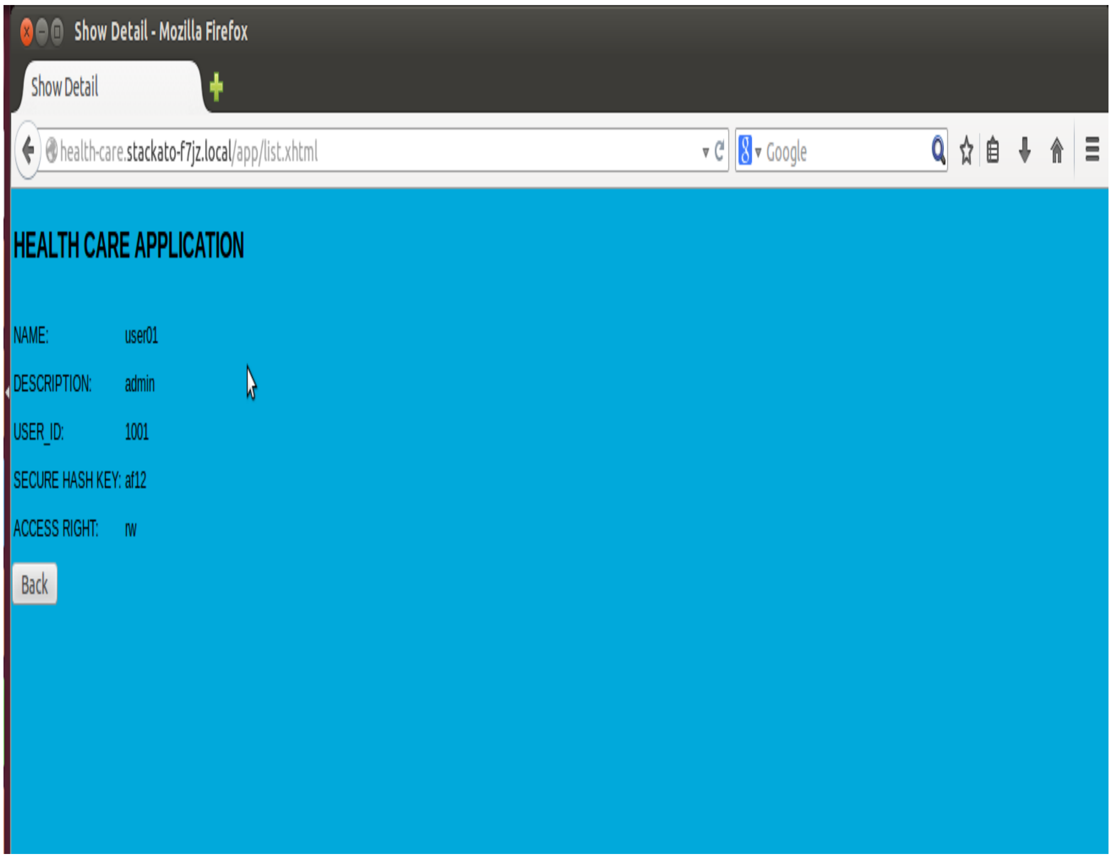
USER ID	SECURE HASH KEY	NAME	ACCESS RIGHT	Edit
1001	af12	user01	rw	Edit
1002	d412	user02	rw	Edit
1003	c4f2	user03	rw	Edit
1004	d4f2	user04	r-	Edit
1005	11ac	user05	r-	Edit
1006	2cf3	user06	r-	
1007	af03	user07	r-	
1008	ff09	user08	r-	Edit
1009	0f7f	user09	r-	Edit
1010	037f	user10	r-	Edit

Save Users

Add User

USER_ID :	
SECURE HASH KEY:	
NAME :	
ACCESS :	
DESCRIPTION :	
Add User	

Figure 35: Healthcare Edit Data


Show Detail - Mozilla Firefox
Show Detail
health-care.stackato-f7jz.local/app/list.xhtml
Google

HEALTH CARE APPLICATION

NAME: user01

DESCRIPTION: admin

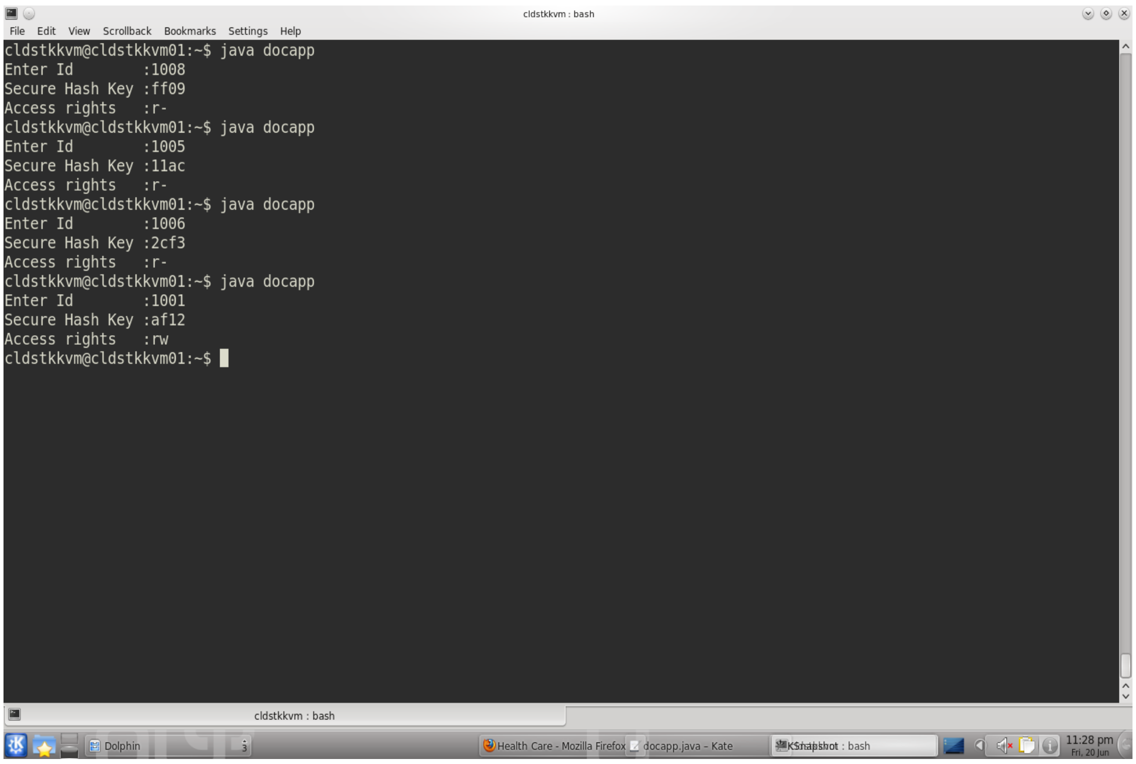
USER_ID: 1001

SECURE HASH KEY: af12

ACCESS RIGHT: rw

Back

Figure 36: Detail View



```
cldstkkvm@cldstkkvm01:~$ java docapp
Enter Id        :1008
Secure Hash Key :ff09
Access rights   :r-
cldstkkvm@cldstkkvm01:~$ java docapp
Enter Id        :1005
Secure Hash Key :11ac
Access rights   :r-
cldstkkvm@cldstkkvm01:~$ java docapp
Enter Id        :1006
Secure Hash Key :2cf3
Access rights   :r-
cldstkkvm@cldstkkvm01:~$ java docapp
Enter Id        :1001
Secure Hash Key :af12
Access rights   :rw
cldstkkvm@cldstkkvm01:~$
```

Figure 37: Secure Hash Key and Access Rights retrieval

The PaaS provides a platform for easy deployment and management of the applications, which reduces the time and efforts required by the IT for managing the system. This provides developer with more time for providing a better and quality service to the user. PaaS easily handles the large number of requests by scaling the computing capacity as per the requests and the load balancer ensures the proper usage of the underlying resources.

10.2.1 Application Testing

To test and compare the application deployed over Stackato and on a server for its scalability, response time and other related performance metrics Apache Jmeter [43] is used. The following parameter values were set for the purpose test, number of users was set to 500 and the time period between the requests was set to 2 seconds and loop count was set to infinite loop.

After 30 minutes following results were obtained

- **Response Time**

Average Response Time of server was about 19.4 seconds whereas that of cloud was 0.96 seconds which is about 1/20th time as compared to the server.

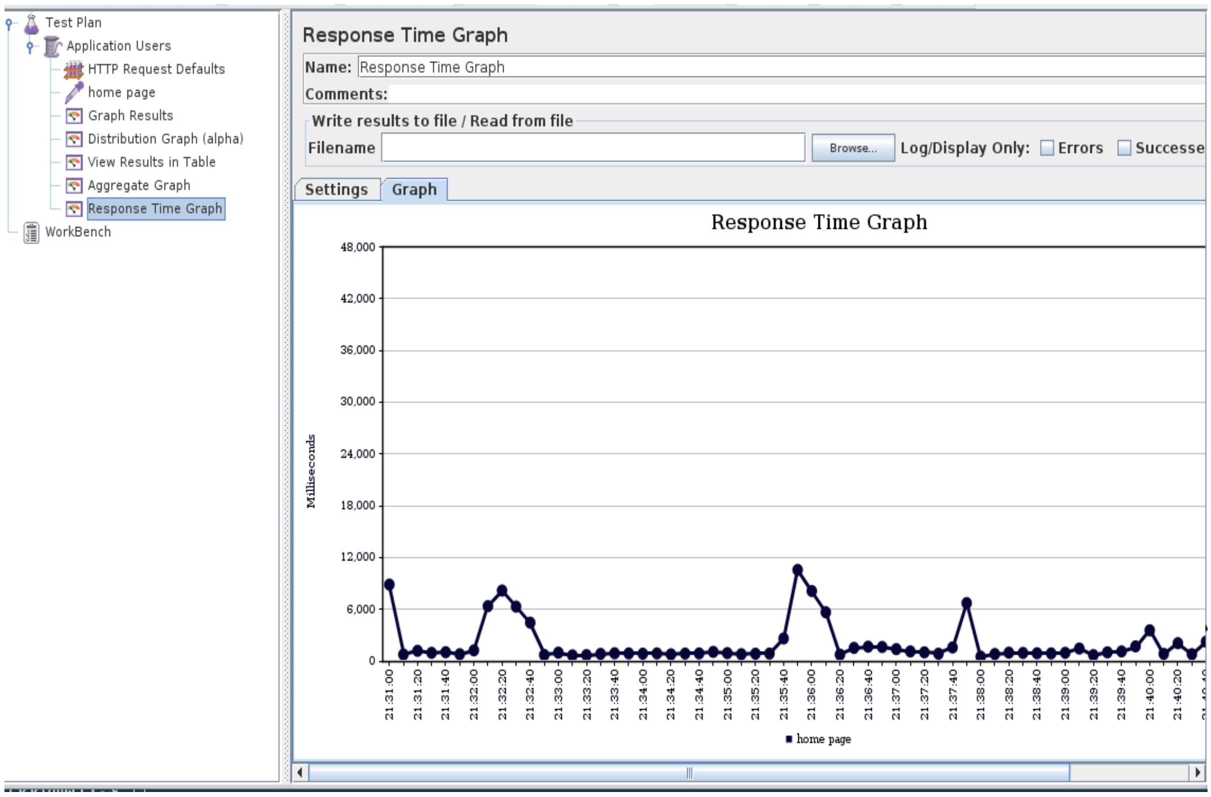
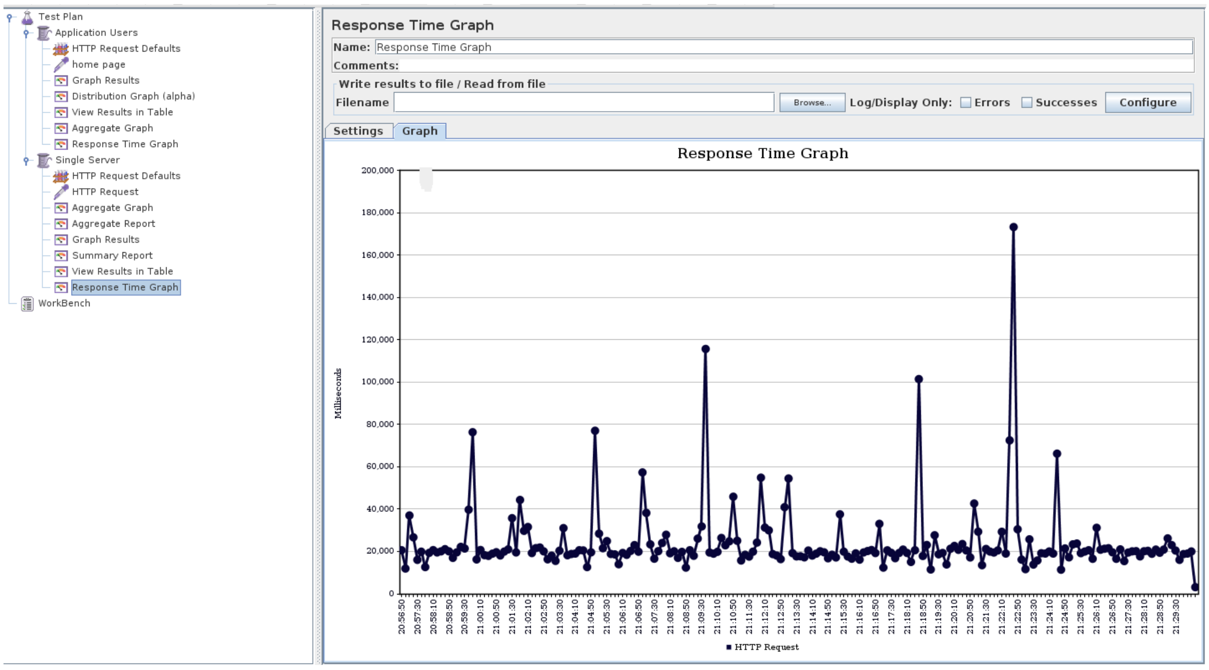


Figure 38: Response Time Graph for application on Stackato



Name: Response Time Graph
Comments:
Filename
Log/Display Only: ☐ Errors ☐ Successes

Figure 39: Response Time Graph for application on Server

- **Error Percentage**

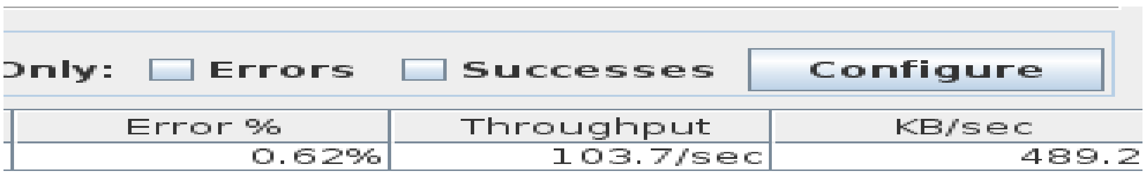
Error percentage for application running on cloud was 0.62% and on server was 85.12% which is about 137 times more than that of cloud.

- **Throughput**

The number of requests per second for cloud was 103.7/sec and that of server was 19.5/sec

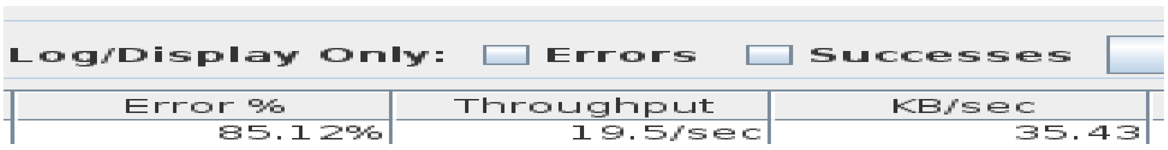
- **Data Rate**

The data rate for cloud was 489.2 KB/sec and for server was 35.43 Kb/sec


Only: ☐ Errors ☐ Successes Configure

Error %	Throughput	KB/sec
0.62%	103.7/sec	489.2

Figure 40: Error%, Throughput and Data Rate for application over Stackato

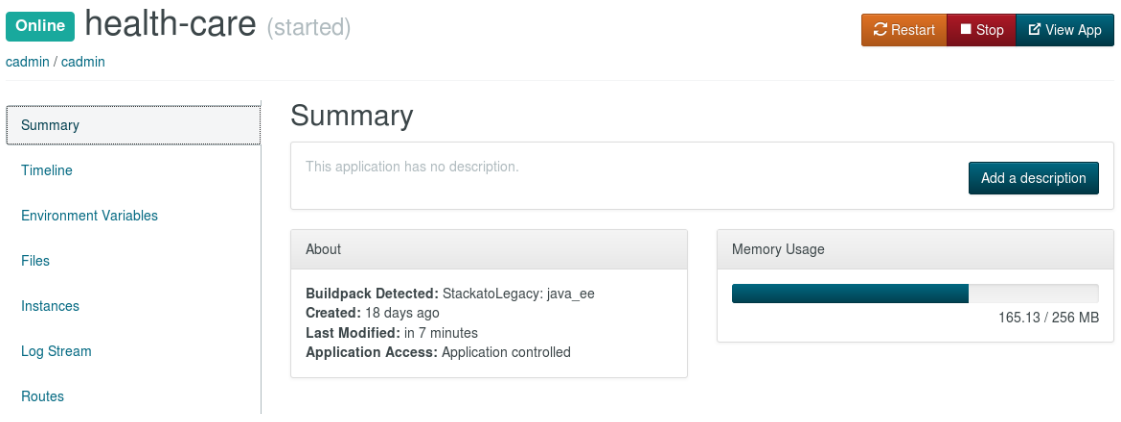

Log/Display Only: ☐ Errors ☐ Successes

Error %	Throughput	KB/sec
85.12%	19.5/sec	35.43

Figure 41: Error%, Throughput and Data Rate for application over Server

- **Scalability**

The application was initially set to use 256 MB, but as the number of requests increased the Stackato automatically scaled the application in order to support the increase in demand.



health-care (started)

cadmin / cadmin

Summary

This application has no description.

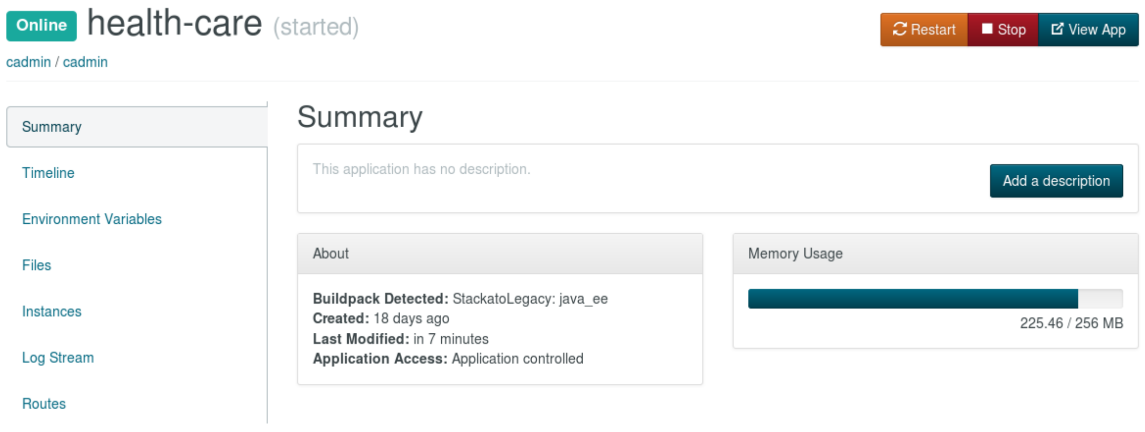
About

Buildpack Detected: StackatoLegacy: java_ee
Created: 18 days ago
Last Modified: in 7 minutes
Application Access: Application controlled

Memory Usage

165.13 / 256 MB

Figure 42: Initial Memory Usage



health-care (started)

cadmin / cadmin

Summary

This application has no description.

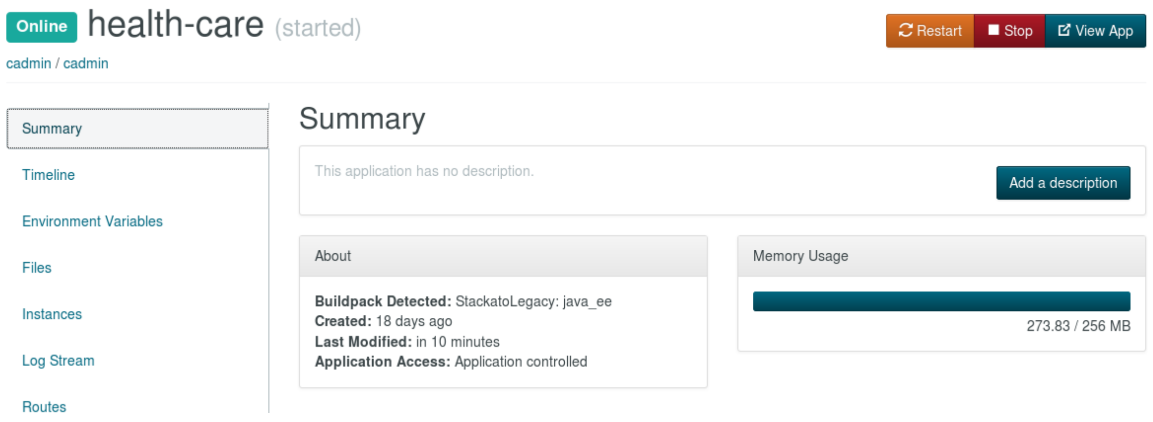
About

Buildpack Detected: StackatoLegacy: java_ee
Created: 18 days ago
Last Modified: in 7 minutes
Application Access: Application controlled

Memory Usage

225.46 / 256 MB

Figure 43: Intermediate Memory Usage



health-care (started)

cadmin / cadmin

Summary

This application has no description.

About

Buildpack Detected: StackatoLegacy: java_ee
Created: 18 days ago
Last Modified: in 10 minutes
Application Access: Application controlled

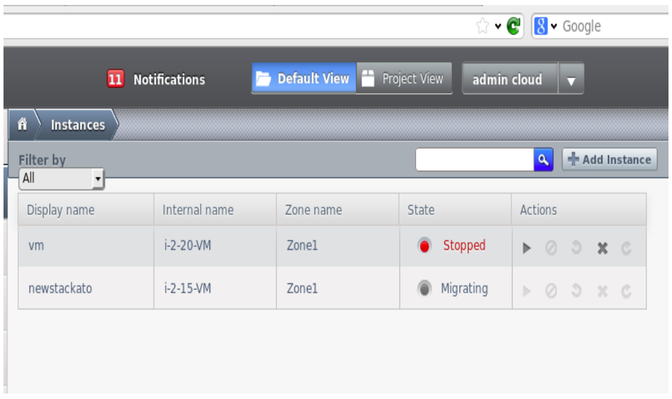
Memory Usage

273.83 / 256 MB

Figure 44: Scaled Memory Usage

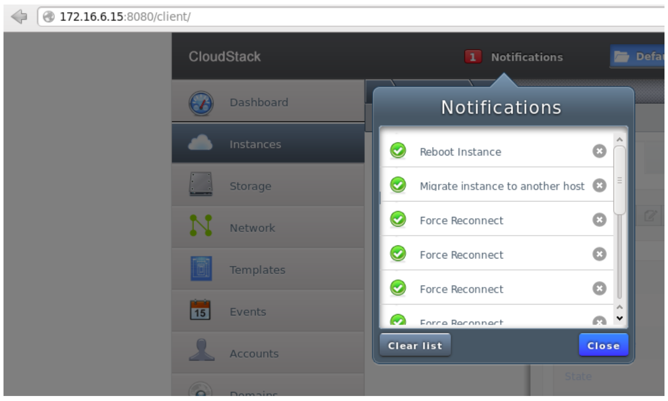
- **Stackato VM Migration**

There is a possibility of an unfortunate event where the host on which the stackato VM is running goes down, to ensure that the system is resilient to such an event migration of the running stackato instance to another host can be done.



Display name	Internal name	Zone name	State	Actions
vm	i-2-20-VM	Zone1	Stopped	
newstackato	i-2-15-VM	Zone1	Migrating	

Figure 45: Stackato VM Migration



Reboot Instance
Migrate instance to another host
Force Reconnect
Force Reconnect
Force Reconnect
Force Reconnect

Figure 46: Notification of Stackato VM Migration

- **Result s Summary**
  1. As compared to the single server the PaaS provides better response time, throughput and data rate, also the error percentage in case of PaaS is lesser than that of a single server.
  2. Scalability is i.e. once the application reaches to the maximum of its resource limit then it cannot scale after that limit and does not response to the incoming requests.
  3. VM migration is also not supported by the single server i.e. if the server fails or goes down than all the application running on it goes down and they cannot be resumed.

Chapter 11

CONCLUSION AND FUTURE WORK

- **Conclusion**

In simple terms cloud can be defined as a large pool of computing and software resources that are delivered on demand, as service. Virtualization and clustering are the essence of the cloud virtualization provides resource pooling and clustering provides scalability. The benefits provided by cloud such as low cost software, security, resilience, resource pooling, scalability, measured service, etc. clearly outweighs the legacy system.

The PaaS provides building blocks such as programming languages and an environment for easy and quick deployment of applications onto the cloud infrastructure. It eradicates the time consuming phase of hardware and other configurations. The PaaS such as stackato provides a platform for easy development, deployment and management of the applications, which reduces the time and efforts required by the IT for managing the system. This provides developer with more time for providing a better and quality service to the user. It easily handles the large number of requests by scaling the computing capacity as per the requests.

There is a need to make the current healthcare system automated, secure, scalable and resilient this is achieved by integrating cloud services in the healthcare system.The elasticity of cloud allows number of users to access the cloud at the same time. The resiliency of cloud promises uninterrupted service to the user.

- **Future Work**

Efforts are being made to integrate the deployed cloud with Hadoop, Apache Hadoop is an open-source software framework for storage and large-scale processing of data-sets on clusters. It would be beneficial when the database grows to large scale and data clusters are needed for storing all patient's records. Addition of Kerberos Authentication Server for authentication of end users for added security is also one area where further work can be done. The deployed system can be scaled by adding more resources which will provide more robust and high end services to the user. The process of VM migration can be automated and the system can be deployed and tested on faster and high end machines for improved results. To better understand the real challenges, requirements and to measure the performance of the system a tie-up with a hospital can be made where the system can be deployed and tested with real patients and doctors and according to their needs further modifications can be made to the system.

REFRENCES

[1] Divyashikha Sethia, Daya Gupta, Tanuj Mittal, Ujjwal Arora, Huzur Saran, "NFC Based Secure Mobile Healthcare System", Sixth International Conference on Communication Systems and Networks (COMSNETS), pp. 1-6, 2014.

[2] W. Lu, J. Jackson, J. Ekanayake, R. Barga, and N. Araujo, "Performing large science experiments on Azure: Pitfalls and solution", IEEE International Conference on Cloud Computing (CloudCom), pp. 209 – 217, 2010.

[3] Meenakshi Bist, Manoj Wariya and Amit Agarwal, "Comparing Delta, Open Stack and Xen Cloud Platforms: A Survey on Open Source IaaS", 3rd IEEE International Advance Computing Conference (IACC), pp. 96 - 100, 2013.

[4] Gregor von Laszewski, Javier Diaz, Fugang Wang, Geoffrey C. Fox," Comparison of Multiple Cloud Frameworks", IEEE Fifth International Conference on Cloud Computing, pp. 734 – 741, 2012.

[5] Xiaolong Wen, Genqiang Gu, Qingchun Li, Yun Gao, Xuejie Zhang, "Comparison of Open-Source Cloud Management Platforms: OpenStack and OpenNebula", 9th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 2457 - 2461, 2012

[6] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The Eucalyptus Open-Source Cloud-Computing System", CCGRID '09: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and theGrid. Washington, DC, USA: IEEE Computer Society, pp. 124–131, 2009.

[7] P. Sempolinski, and D. Thain, "A Comparison and Critique of Eucalyptus, OpenNebula and Nimbus", IEEE Second International Conference on Cloud Computing Technology and Science (Cloud Com), pp. 417–426, 2010.

[8] Cloud Concepts, http://en.wikipedia.org/wiki/Cloud_computing

[9] Cloud Service Models, http://blog.teamgrowth.net/index.php/virtualization/cloud-computing/introduction-to-cloud-computing-and-service-models

[10] T. Grance, and P. Mell, "The NIST definition of Cloud Computing," National Institute of Standards and Technology (NIST), 2009.

[11] Stefan Wind, "Open Source Cloud Computing Management Platforms Introduction, comparison, and recommendations for implementation", IEEE Conference on Open

Systems (ICOS), pp. 175 – 179, 2011.

[12] CloudStack Installation Guide, http://cloudstack.apache.org/docs/en-US/Apache_CloudStack/4.0.2/html/Installation_Guide/index.html

[13] Apache CloudStack, http://en.wikipedia.org/wiki/Apache_CloudStack

[14] CloudStack Documentation, https://cwiki.apache.org/confluence/display/CLOUDSTACK/Home

[15] CloudStack API Documentation, http://cloudstack.apache.org/docs/api/

[16] Apache CloudStack, http://cloudstack.apache.org/

[17] CloudStack's Major Users, http://buildacloud.org/users.html

[18] F. Gomez-Folgar, A. Garcia-Loureiro, T. F. Pena and R. Valin, "Performance of the CloudStack KVM Pod primary storage under NFS version 3", 10th IEEE International Symposium on Parallel and Distributed Processing with Applications, 2012

[19] AskUbuntu Website, http://askubuntu.com/

[20] Stackoverflow Website, http://stackoverflow.com/

[21] CloudStack Management Server Installation, http://ranafaisal.wordpress.com/2013/02/20/cloudstack-management-server-installation/

[22] CloudStack KVM Host Installation, http://ranafaisal.wordpress.com/2013/02/21/cloudstack-kvm-host-installation/

[23] S. Kibe, S. Watanabe, K. Kunishima R. Adachi, M. Yamagiwa and M. Uehara, "PaaS on IaaS" , in Advanced Information Networking and Applications (AINA), IEEE 27th International Conference, pp. 362-367, 2013.

[24] Essential Characteristics of Cloud, http://www.isaca.org/groups/professional-english/cloud-computing/groupdocuments/essential%20characteristics%20of%20cloud%20computing.pdf

[25] Rabi Prasad Padhy, Manas Ranjan Patra, Suresh Chandra Satapathy, "Design and Implementation of a Cloud based Rural Healthcare Information System Model", Universal Journal of Applied computer Science and Technology, Vol 2 (1), pp. 149-157, 2012.

[26] M.Deepa Lakshmi, J.P.M. Dhas, "An Open Source Private Cloud Solution for Rural Healthcare", International Conference on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN), pp. 67-674, 2011.

[27] Chengtong Lv, Qing Li, Zhou Lei, Junjie Peng, Wu Zhang, Tingting Wang "PaaS: A Revolution for Information Technology Platforms", International Conference on Educational and Network Technology (ICENT), pp. 346 – 349, 2010.

[28] The Next Wave of Technologies: Opportunities from Chaos, By Phil Simon, Chapter 4- Cloud Computing, pp 64.

[29] Cloud Service Providers List 2014, http://www.spamina.com/eng/cloud_hosting_providers_list.php

[30] The State of PaaS: 2012 Presented by Engine Yard, http://pages.engineyard.com/rs/engineyard/images/PaaS_Market_Report_2012.pdf

[31] PaaS Providers List: 2014, http://www.tomsitpro.com/articles/paas-providers,1-1517.html

[32] Heroku, https://devcenter.heroku.com/

[33] OpenShift, https://www.openshift.com/developers/documentation

[34] Google App Engine, https://developers.google.com/appengine/?csw=1

[35] AppFog, https://docs.appfog.com/

[36] Stackato, http://www.activestate.com/stackato

[37] Zeng Shu-Qing , Xu Jie-Bin, "The Improvement of PaaS Platform", First International Conference on Networking and Distributed Computing (ICNDC), pp.- 156 – 159, 2010.

[38] Teodor-Florin Fortis, Victor Ion Munteanu, Viorel Negru, "Towards a service friendly cloud ecosystem".

[39] Sasko Ristov and Marjan Gusev, “Security Evaluation of Open Source Clouds”, IEEE EUROCON, pp. 73-80, 2013.

[40] Stackato Architecture, http://www.activestate.com/stackato/why-private-paas/paas-architecture

[41] Stackato Security, http://www.activestate.com/stackato/why-private-paas/cloud-security

[42] Apache Jmeter, http://jmeter.apache.org/