Final Project Report

on

Openstack Performance Monitoring Dashboard and Billing Module based on usage

by

Ashutosh Agarwal B80028502
Apoorva Kokate B80028532
Deepak Poojari B80028550
Pratik Saraf B80028560

Internal Project Guide

Archana Chougule

External Project Guide

Mr.Kunal Patil

Department of Information Technology
MAEER's MAHARASHTRA INSTITUTE OF TECHNOLOGY
PUNE - 411038

2014 - 2015

MAEER's
MAHARASHTRA INSTITUTE OF TECHNOLOGY, PUNE
DEPARTMENT OF INFORMATION TECHNOLOGY

C E R T I F I C A T E

This is to certify that
Ashutosh Agarwal B80028502
Apoorva Kokate B80028532
Deepak Poojari B80028550
Pratik Saraf B80028560
of B.E. (Information Technology)
have successfully completed their interim project report on

Openstack Performance Monitoring Dashboard and Billing Module based on usage

and have submitted this interim report in partial fulfillment of the requirements for the degree Bachelor of Engineering in Information Technology of the University of Pune for the academic year 2014-2015.

Archana Chougule
Internal Project Guide
Dept. of Information Technology, Pune
MAEER's MIT, Pune.

Dr.Debajyoti Mukhopadhyay
Dean (R&D) of MIT group & Head of,
Dept. of Information Technology, Pune.
MAEER's MIT, Pune.

Mr. Kunal Patil
e-Zest Solutions Ltd.
(External Project Guide)

Acknowledgement

We would like to express our deep gratitude and thankfulness to respected Prof. Sheetal Girase our B.E. Project Coordinator for her guidance, useful ideas and patience during selection of the project.
We are aslo very grateful to Archana Chougule our Internal Project Guide for his constant support and encourage us to made this project possible and also contributed with his valuable guidance and provided with perfect solution for every problem that arose.
We are also thankful to Prof. Dr. Debajyoti Mukhopadhyay, Dean(RD) ,HOD and all staff members of department of Information Technology of MIT ,Pune who extended their kind support and encouragement during the preparatory steps of this project. We would also like to express appreciation and thanks to all our friends who knowingly or unknowingly assisted us with their valuable suggestion and comments, and we are very grateful for their assistance.
We would also like to thank the institute for providing the required facilities, Internet access and important books.

Abstract

In cloud computing environment, Infrastructure as a Service (IaaS) takes the lowest tier in the cloud pyramid where most control and management is needed. IaaS clouds offer IT infrastructure resources for computing, storage and networking to cloud users. So we need a comprehensive intelligent monitoring tool to analyze the performances of machines on which the application is hosted. We propose to develop performance monitoring tool for openstack performance monitoring and displaying the results on dashboard, depending on results of monitoring tool we will also be computing the billing cost of the users.The monitoring tool collects the information of resource usage of the instances on which theF application is currently running and displays them on dashboard.The statistical report of dashboard will provide information about the performance monitoring depending upon the resource utilization and depending on that end users billing cost will be computed, which will then be reflected in the billing module.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

1.1.1 Introduction to cloud

Today's $21^{s\ t}$ century buzzword in IT industry is all about cloud computing. But what actually it means is given in the following definition.

Cloud :
CLOUD is abbreviated as Computing Location independent Online Utility that is available on demand.It is a new term but it encloses advantages of many other technologies which are pre-exists in the IT world such as virtualization ,grid computing ,utility computing etc which make cloud computing successful. Key attributes behind cloud computing are: Dynamic, abstraction, resource sharing and virtually infinite scalability. Cloud Computing has rapidly become a widely adopted paradigm for delivering services over the Internet.

Cloud computing comes in three forms: public clouds, private clouds, and hybrids clouds.

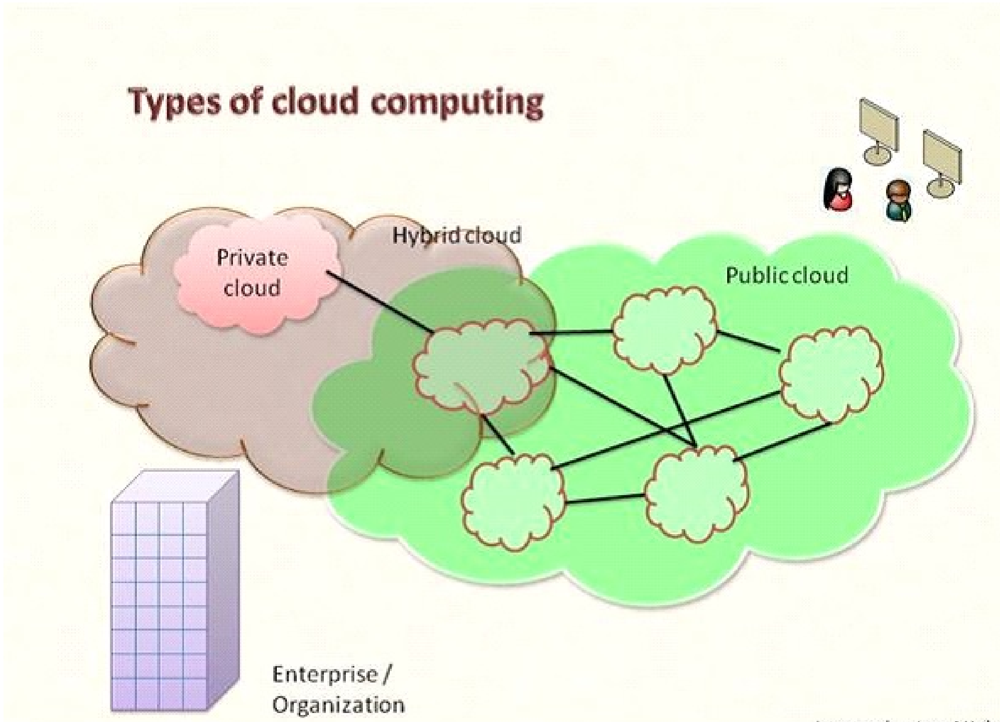

Figure 1.1 : Types of Cloud Computing

Public : A cloud is called a "public cloud" when the services are rendered over a network that is open for public use. Public cloud services may be free or offered on a pay-per-usage model.
Example Of Private Cloud:
Cloudstack ,Euclaptus,Openstack,Openebula, Nimbus,Openshift.

Private: Private cloud is cloud infrastructure operated solely for a single organization, whether managed internally or by a third-party, and hosted either internally or externally.
Examples of Public Cloud: Amazon Web Services, Microsoft azure,icloud,(google).

Hybrid: Hybrid cloud is a composition of two or more clouds (private, community or public) that remain distinct entities but are bound together, offering the benefits of multiple deployment models. Hybrid cloud can also mean the ability to connect collocation, managed and/or dedicated services with cloud resources

1.1.2 Openstack

In July 2010 , OpenStack was announced and the initial contributes of it are NASA and Rackspace.It is the fastest growing free open source software. Rackspace contributed their "Cloud Files" platform (code) to power the Object Storage part of the OpenStack, while NASA contributed their "Nebula" platform (code) to power the Compute part. OpenStack is a set of software tools for building and managing cloud computing platforms for public and private clouds.

OpenStack is a collection of open source software project that developers and cloud computing technologist can use to setup and run their cloud compute and storage infrastructure. Its services are available through Amazon EC2/S3 compatible APIs and hence the client tools written for AWS can also be used with OpenStack. It consist of three core software project:OpenStack Compute Infrastructure also called Nova; OpenStack Object Storage Infrastructure also called Swift and OpenStack Image Service Infrastructure also called Glance. Nova is the main part of Infrastructure as a service and it also is the computing Fabric controller for the OpenStack cloud .Enterprises/Organization can use Nova to host and manage their cloud computing systems. Nova manages all the activities that are needed to support life cycle of instances within the open stack.Swift offers a distributed,consistent virtual object containers in which lots of data can be store and from which data can be retrieve It is capable of storing large number of object distributed across nodes.Glance is a lookup and retrieval system for virtual machine images.

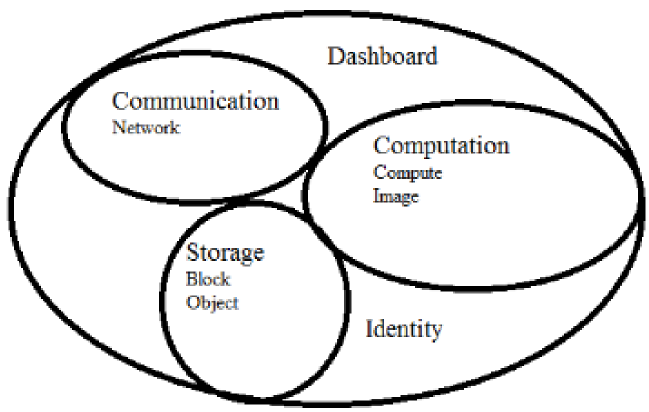

Figure : 1.2 Dashboard Of Openstack

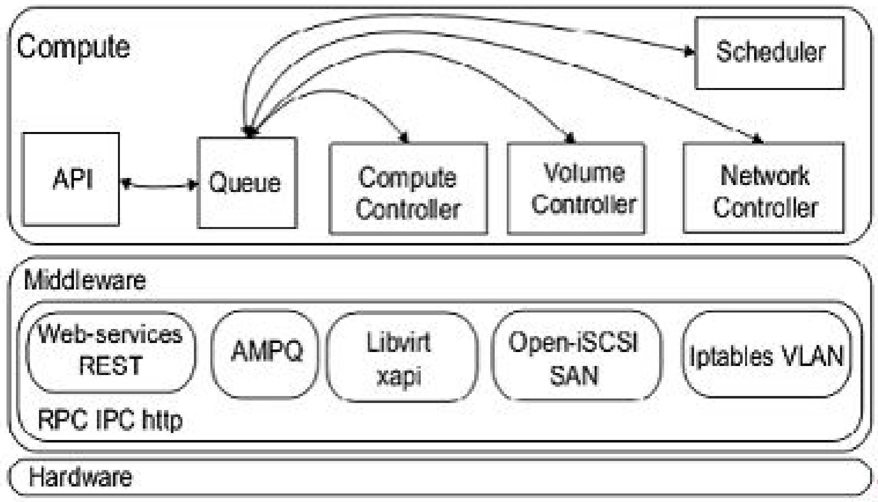

Figure : 1.3 compute Node's Networking

OpenStack is composed of seven main services which can be grouped into three principal areas: Communication, Storage, and Computation as depicted in Figure 1. These services are backed by two support services, namely Identity and Dashboard. Compute service manages the virtual disks and associated metadata in Image. Dashboard provides a web-based front-end to the Compute whereas Network provides virtual networking for Compute. Block Storage provides storage volumes for Compute. Image can store the actual virtual disk files in the Object Storage and all the services authenticate with Identity. The Compute service consists of the following components as depicted in Figure 2. Web-based API ensures the command and control aspects of the computation, storage and networking. The component Queue brokers the interaction between the Compute service components, i.e., Volume, Network and Compute Controllers, Scheduler and API. Compute Controller manages the life-cycle of computing instances (i.e. VMs) on the nodes. The Network Controller manages the networking resources and the Volume Controller ensures the interaction between the instances and the Block Storage

1.1.3 Cloud monitoring

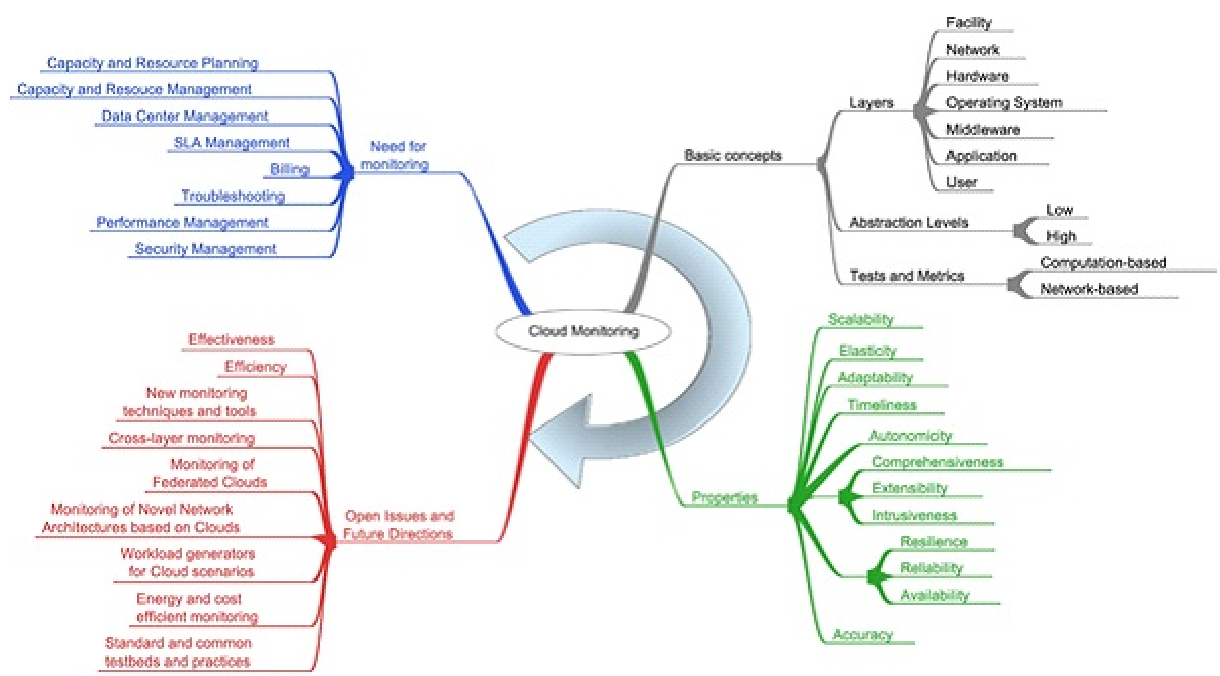

Figure :1.4 Cloud Monitoring

Monitoring of Cloud is a task of paramount importance for both Cloud Service Providers (called Providers in the following) and Cloud Service Consumers (called Consumers in the following). On the one side, it is a key tool for controlling and managing hardware and software infrastructures; on the other side, it provides information and Key Performance Indicators (KPI) for both platforms and applications. The continuous monitoring of the Cloud and of its Service Level Agreements (SLAs), - for example, in terms of availability, delay, etc. - supplies both the Providers and the Consumers with information such as the workload generated by the latter or the performance and Quality of Service (QoS) offered through the Cloud, also allowing to implement mechanisms to prevent or recover violations, for both the Provider and Consumers. Cloud Computing involves many activities for which monitoring is an essential task.

1.1.4 Private Cloud Monitoring

A private cloud, as an alternative to the public cloud, is the ideal choice for a network administrator who is looking for more control over the system. A private cloud can be monitored more accurately and directly than a service in the public cloud. All services and resources live in defined systems that are only accessible to the user and are protected from external access. Also, unlike the situation with many public clouds, the quality criteria for performance and availability in a private cloud can be customised, and compliance with these criteria can be monitored to ensure that they are achieved.
It is important that the existing physical network is monitored in the first instance, in order to understand the current workload of the systems used. Long-term analysis, trends and peak loads can be attained via extensive network monitoring evaluations, and resource availability can be planned according to demand. This is necessary to guarantee consistent IT performance across virtualised systems, and will ensure that the everyday user experience is the same or better when switching to the virtual systems in the cloud from the former physical system.

Need Of private cloud : 1)Confidentiality :
Private cloud is usually refred at places where confidential data is stored in the cloud .In private cloud the data is stored at your place or at a more secure place then the data that is stored at public cloud . 2)For long term usage:
The organization where cloud can be used for long term use,in such organization private cloude is refered because for longterm use private cloud is more cloud efficient then public cloud.

1.2 Need

1.2.1 Capacity and resource planning

In order to guarantee the performance required by applications and services, developers have to
(i) quantify capacity and resources (e.g. CPU, memory, storage, etc.) to be purchased, depending on how such applications and services are designed and implemented .
ii) determine the estimated workload.
However, while an estimation can be obtained through static analysis, testing and monitoring, the real values are unpredictable and highly variable. Cloud Service Providers usually offer guarantees in terms of QoS and thus of resources and capacity for their services as specified in SLAs, and they are in charge of their resource and capacity planning so that service and application developers do not have to worry about them. To this end, monitoring becomes essential for Cloud Service Providers to predict and monitor all the real-time parameters involved in the process of QoS assurance in order to properly plan their infrastructure and resources for respecting the SLAs.

1.2.2 Capacity and resource management

The first step to manage a complex system like a Cloud consists in having a monitoring system able to accurately capture its state. Over the years, virtualization has be-come a key component to implement Cloud Computing. Hiding the high heterogeneity of resources of the physical infrastructure, virtualization technologies introduced another complexity level for the infrastructure provider, which has to manage both physical and virtualized resource. Virtualized resources may migrate from a physical machine to another at any time. Hence, in Cloud Computing scenarios (specially in mobile ones) monitoring is necessary to cope with volatility of resources and fast changing network conditions (which may lead to faults).
E.g., Healthcare or other Strategic applications: when using IaaS, when concerns about QoS and QoP (Quality of Protection) become very critical , a resilient and trustworthy monitoring of the entire Cloud infrastructures is needed.

1.2.3 Data center management

Cloud services are provided through large scale data centers, whose management is a very important activity. Data center management activities (e.g. data center control) imply two fundamental tasks:
1. monitoring, that keeps track of desired hardware and software metrics.
2. data analysis, that processes such metrics to infer system or application states for resource provisioning, troubleshooting, or other management actions. In order to properly manage such data centers, both monitoring and data analysis tasks must support real-time operation and scale up to tens of thousands of heterogeneous nodes, dealing with complex network topologies and I/O structures.

1.2.4 SLA management

The unprecedented flexibility in terms of resource management provided by Cloud Computing calls for new programming models in which Cloud applications can take advantage of such new feature, whose underlying premise is monitoring. Moreover, monitoring is mandatory and instrumental in certifying SLA compliance. Finally, monitoring may allow Cloud Providers to formulate more realistic and dynamic SLAs and better pricing models by exploiting the knowledge of user-perceived performance.

1.2.5 Billing

One of the essential characteristics of Cloud Computing is the offer of measured services, allowing the Consumer to pay proportionally to the use of the service with different metrics and different granularity, according to the type of service and the price model adopted.
Examples of billing criteria are: for SaaS, the number of contemporary users, or the total user base, or application specific performance levels and functions; in PaaS services, the CPU utilization, or the task completion time; for IaaS, the number of VMs, possibly varying with different CPU/Memory setups.For each of the reported pricing models and service models, monitoring is necessary both from the Provider side for billing, and from the Consumer side for verifying his own usage and to compare different Providers, a non-trivial process requiring monitoring functionalities and tools.

1.2.6 Troubleshooting

The complex infrastructure of a Cloud represents a big challenge for troubleshooting (e.g. root cause analysis), as the cause of the problem has to be searched in several possible components (e.g. network, host, etc.), each of them made of several layers (e.g. real and virtual hardware, host and guest OS, etc.).
A comprehensive, reliable and timely monitoring platform is therefore needed for Providers to understand where to locate the problem inside their complex infrastructure and for Consumers to understand if any occurring performance issue or failure is caused by the Provider, network infrastructure, or by the application itself .

1.2.7 Performance management

Despite the attention paid by Providers, some Cloud nodes may fail to meet the expectation. If a Consumer adopts a public Cloud to host a mission critical service or for a scientific application, performance variability and availability become a concern. Therefore, from a Consumers perspective, monitoring the perceived performance is necessary to adapt to the changes or to apply corrective measures. For instance, a Consumer may decide to host applications at multiple Clouds to ensure high-availability, switching between Clouds depending on the measured performance. Monitoring is then necessary since it may considerably improve the performance of real applications and affect activity planning and repeatability of experiments.

1.2.8 Security management

Cloud security is very important for a number of reasons. Security is considered as one of the most significant obstacles to the spread of Cloud Computing, especially considering certain kinds of applications (e.g. business-critical ones) and Consumers (e.g. governments). Research and developments have provided reviews and recommendations for Cloud security. For managing the security in Cloud infrastructures and services, proper monitoring systems are needed. Moreover, for hosting critical services for public agencies, Clouds have to satisfy strict regulations and prove it.

1.3 BASIC CONCEPTS

1.3.1 physical Layers

Cloud can be modeled in seven layers:

1.Facility
2.Network
3.Hardware
4.Operating System
5.Middleware
6.Application
7.The user

They are detailed in the following:
1.Facility:
At this layer we consider the physical infrastructure comprising the data centers that host the computing and networking equipment.

2.Network:
At this layer we consider the network links and paths both in the Cloud and between the Cloud and the user.

3.Hardware:
At this layer we consider the physical components of the computing and networking equipment.

4.Operating System (OS):
At this layer we consider the soft-ware components forming the operating system of both the host (the OS running on the physical machine) and the user (the OS running in the virtual machine).

5.Middleware:
At this layer we consider the software layer between the OS and the user application. It is typically present only in the Cloud systems offering SaaS and PaaS service models.

6.Application:

At this layer we consider the application run by the user of the Cloud system.

7.User:
At this layer we consider the final user of the Cloud system and the applications that run outside the Cloud (e.g. a web browser running on a host at the users premise).

In the context of Cloud monitoring, these layers can be seen as where to put the probes of the monitoring system. In fact, the layer at which the probes are located has direct consequences on the phenomena that can be monitored and observed.

1.3.2 Abstraction levels

In Cloud Computing, we can have both high-level and low-level monitoring, and both are required.

High-level monitoring is related to information on the status of the virtual platform. This information is collected at the middleware, application and user layers by Providers or Consumers through platforms and services operated by themselves or by third parties. In the case of SaaS, high-level monitoring information is generally of more interest for the Consumer than for the Provider (being closely related to the QoS experienced by the former). On the other hand, low-level monitoring is related to information collected by the Provider and usually not exposed to the Consumer, and it is more concerned with the status of the physical infrastructure of the whole Cloud (e.g. servers and storage areas, etc.). In the context of IaaS, both levels are of interest for both Consumers and Providers.

For low-level monitoring specific utilities collect information at the hardware layer (e.g., in terms of CPU, memory, temperature, voltage, workload, etc.), at the operating system layer and at middleware layer (e.g., bug and software vulnerabilities), at the network layer (e.g., on the security of the entire infrastructure through firewall, IDS and IPS), and at the facility layer (e.g. on the physical security of involved facilities through mon-itoring of data center rooms using video surveillance and authentication systems).

1.3.3 Tests and metrics

Monitoring tests can be divided in two main categories:
1.Computation-based
2.Network-based

Computation-based

Tests are related to the following metrics: server throughput, defined as the number of requests (e.g. web page retrieval) per second; CPU Speed; CPU time per execution, defined as the CPU time of a single execution; CPU utilization, defined as the CPU occupation of each virtual machine memory page exchanges per second, defined as the number of memory pages per second exchanged through the I/O; memory page exchanges per execution, defined as the number of memory pages used during an execution; disk/memory throughput; throughput/ delay of message passing between processes; duration of specific predefined tasks; response time; VM startup time; VM acquisition/release time; execution/access time, up-time. All of them can be evaluated in terms of classical statistical indicators (mean, median, etc.) as well as in terms of temporal characterization and therefore stability/variability/predictability.

Network-based

Tests are related to the monitoring of network-layer metrics. This set includes round-trip time (RTT), jitter, throughput, packet/data loss, available bandwidth, capacity, traffic volume, etc. Using these metrics, several experimental studies in literature compared legacy web hosting and Cloud-based hosting.

1.4 Areas of Application

Domain : Cloud
Sub Domain: Cloud Monitoring
Topic: Performance monitoring of computational metrics in openstack

Application :

This tool is used to monitor the computational metrics in cloud(openstack cloud) and provide billing of individual instances This tool also provide recommendation to administrator so that he can make intelligent decisions to make future decisions which at automatically generated through analysing reports generated This tool also monitors individual applications running on instance using various computational metrics

Chapter 2

Literature Survey

2.1 DARGOS

Description	Parameter	Limitation	Advantages
1)DARGOS is a completely distributed and highly efficiently cloud monitoring architecture to disseminates resource monitoring information. 2)DARGOS ensures an accurate measurment of physical and virtual resources.In the cloud keeping at the same time a low overhead. 3)DARGOS is flexible and adaptable and it also allows defining monitoring tool.	1)Iaas monitoring: zone, average cpu 2)Host Monitoring host,cpu1, meomery usage 3)service monitoring i) Apache id,uptime,req byte,total access total Kb Idle busy workes,Mysql id,uptime,s/w queries,flash tables,threads question,open,open table	1)It issues with realibility, avaibility of services. 2)lack of operating standard 3)It is problem in data security and privacy	1)Its perform Iaas Monitoring 2) Its perform service monitoring 3)DARGOS ensures an accurate measurment of physical and virtual resources include keeping at some time a low overhead

Figure : 2.1 DARGOS

2.2 Zenoss

Description	Parameter	Limitation	Advantages
1.Zenoss Service Dynamics provides cloud monitoring capabilities for common cloud platforms that help IT Operations teams quickly and effectively monitor their cloud resources. For enterprises and service providers who are deploying private or hybrid clouds, 2. Zenoss offers ZenPacks for OpenStack. 3.These ZenPacks provide a tenant view for your public cloud clients, and allow your clients to monitor the performance and availability of their services as part of their own unified environment	1)SwiftObject Server - Async Pending 2)Swift Object Server - Disks 3)Swift Object Server - Quarantine 4)Swift Object Server - Replication Time 5)Swift Object Server - Load Averages 6)Swift Object Server - Process Churn 7)Swift Object Server - Disk Usages 8)Swift Object Server - Disk Sizes 9)Swift Object Server - Processes	1)Zenoss Core doesnt support as many diverse devices as HP OpenView or Argent Extended Technologies, nor does it monitor Microsoft Exchange or SQL Server as closely as a commercial tool does." 2)getting started was challenging as Zenoss provided no context-sensitive help to guide us through a truly staggering number of configuration options.	1)Comprehensive cloud computing monitoring of networks, devices, and relationships. 2)Immediate, agentless modeling of each server as it is instantiated. 3)Comprehensive modeling of every device, including hardware, software, virtualization, and network relationships, to ensure the right monitoring policies are always applied. 4)Comprehensive monitoring of cloud infrastructures using functions built into operating systems and applications by cloud vendors. 6)End-to-end cloud operations management reporting, alerting, and action scripting.

Figure : 2.2 Zenoss

2.3 PCNOMS

Description	Parameter	Limitation	Advantages
the current pcnoms version acts as the integration layer, by retrieving gathering and preparing relevant information for the visualization layer	1)node information gather 2) cluster integrator 3) monitoring data integrator 4) vm monitor 5)monitoring tool server 6) user interface 7)database	only acts as a integration tool with view and infrastructure layer	1) compatible with nagios at the view layer and with eucalyptus at the infrastructure Layer 2) provides a clear sepration via abstraction between infrastructure details and monitoring information by cloud users.

Figure : 2.3 PCNOMS

2.4 NAGIOS

Description	Parameter	Limitation	Advantages
1)Nagious monitors your entire IT infrastructure to ensure system and services are functioning properly. 2)In the event of faliure nagious can alert technical staff of the problem. 3)allowing them to begin remediation process before outages affect bussiness process and end users or customer.	1)diskusage 2)ram 3)network 4)application	1)operate only at view level 2)dosent provide suggestion only alert's.	1)can be used with any opensourse cloud. 2)if there is any malfunctioning in the software.

Figure : 2.4 NAGIOS

2.5 MUNIN

Description	Parameter	Limitation	Advantages
1)munin is a network resorse monitoring tool. 2)munin has master node architecture in which the master connect to all nodes at a regular interval and ask them for the data . . 3)IT then stores the data in RDD file and update the graph.	1)server and client monitoring. 2)instance monitoring. 3)disk,network, process,processor.	1)not a real time system. 2)client is unknown when network is down. 3)only display graph by day,week,month and year.	1)plugin's can be written in any language. 2)munin provide a nice baseline for graphing. 3)better for longterm graphical display.

Figure : 2.5 MUNIN

Chapter 3

Project Statement

3.1 Project Definition

In cloud computing environment, Infrastructure as a Service (IaaS) takes the lowest tier in the cloud pyramid where most control and management is needed. IaaS clouds offer IT infrastructure resources for computing, storage and networking to cloud users. These resources can further be divided into instances with the help of open source project (Openstack) which operates at Iaas layer to create a private cloud.

We propose to develop a performance monitoring tool which can monitor the individual instances (both local and remote) of the cloud and also individual applications based on set of metrics. The set of metrics (parameters) are C.P.U utilization vs. time, Memory vs. time, Bandwidth vs. time , Disk vs. time in real time. The same information is then reflected into interactive dashboards where the Administrator will get the daily, monthly, weekly reports and graphs as needed by him [5].

After that we will also compute the cost of each individual instances of the cloud based on the resource utilization so, that the owner can get the exact overview of the cost of each instance and then can take the necessary actions to ensure that users pay only for what they use.

3.2 Future Scope

1. **System can be enhanced to take some action along with monitoring.**

Our monitoring tool performs the functionalities of monitoring and recommendation. But further updates should be made in the tool so that the tool not only monitors the issues but also configures the virtual machines and the software to make the system compatible for any particular scenario. The monitoring tool must also be able to handle and prevent various threats to the virtual machines so as to ensure the safe running of the applications. If at any point of time, a certain component of the virtual machine starts malfunctioning or gets corrupted, the monitoring tool must be able to fetch the disk image of the previous ACTIVE state and then RESET that virtual machine to that state ,preventing the crashing of any virtual machine and hence any application. The monitoring must also be programmed to keep the software updated. A functionality of Software-Update must be made available in the tool which will automatically search for all the update and will download and install it either instantly or by users permissions. This will help the user to use the latest version of his software.

2.**Enhancing the monitoring tool from the security perspective.**

This functionality could be helpful to provide additional security to the software. The monitoring tool could be programmed further to detect the attacks of the various virus such as malwares, worms etc. and resolve these issues before they could even enter the system. This could be done by equipping the tool with various Intrusion Detection System such as Host IDS and Network IDS which will ensure the security of the Host as well as detect the vulnerability inside the network

3. **Energy Saving Mechanism**

In the cloud monitoring of QoS parameters in Services Computing as well as in Clouds has been a functionality provided by all contemporary systems. As the optimization of energy consumption becomes a major concern for system designers and administrators, it can be considered as another QoS metric to be monitored. Todays Data-centers supporting Cloud computing

consume an enormous amount of power , representing a financial burden for their operating organizations, an infrastructure burden on power utilities, and an environmental burden on society. Cloud computing is an emerging paradigm for business computing and as its popularity grows, its potential energy impact grows in significance. This increased usage of Cloud computing, together with the increasing energy costs and the need to reduce carbon emissions call for energy-efficient technologies to sustain Cloud datacenters
.

Chapter 4

System Requirement and Specification

4.1 Project Scope

The openstack performance monitoring tool performs monitoring of various instances application wise. The requirment specified in the SRS document considered with oragnization need for a new performance monitoring tool which reports Infrastrcture usage of the instance and compares them for making feasible for the administrator to make intelligent decisions by giving him appropriate suggestion .It also used by end users to know his performance usage and billings costs and pay for what you use.

4.1.1 System Requirements (Hardware)

Controller Node:

1 processor, 2 GB memory, and 5 GB storage

Compute Node:

Compute Node: 1 processor, 512 MB memory, and 5 GB storage

Compute Node:

1 processor, 2 GB memory, and 10 GB storage

CPU :

64 bit processor

Networking:

1 Gbps or 10 Gbps is suggested internally.

For OpenStack Object Storage, an external network should connect the outside world to the proxy servers, and the storage network is intended to be isolated on a private network or multiple private networks.

4.1.2 System Requirements (Software)

Operating system: OpenStack Object Storage currently runs on Ubuntu, RHEL, CentOS, Fedora, openSUSE, or SLES.

4.1.3 User Requirements (Hardware)

CPU :

1)64 bit processor
2)at least 500 MHz
3)Ram: at least 512 MB

4.1.4 User Requirements (Software)

Browser:

Google Chrome, Mozilla Firefox, Internet Explorer, etc.

Chapter 5

Design

5.1 Architectural Diagram

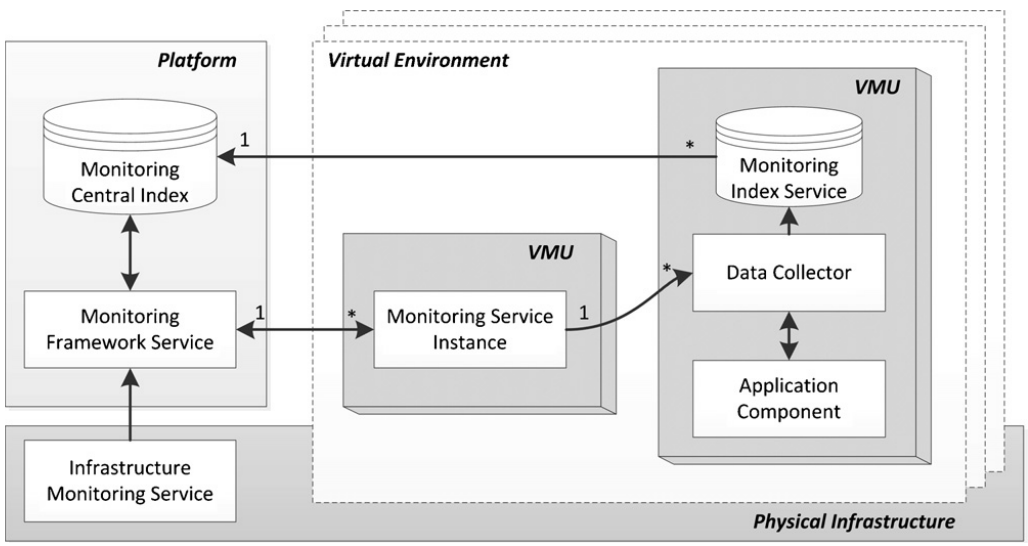

Figure 5.1: Monitoring infrastructure architecture.

In modern Cloud environments and service-based platform that encompass a virtualization layer, a set of virtual Machine units (VMUs) that host different components or services is instantiated to fulfill the needs of particular applications. The mechanism that we propose consists of six components that are deployed across the different Cloud Layers.

5.1.1 Paas layer

Monitoring Framework Service (MFS): It is the major component of the mechanism while it orchestrates the monitoring of all applications on the virtual environment and has access to the aggregated information through the Monitoring Central Index. Another key aspect of the MFS is the evaluation of the monitoring data against specific policies which will allow the reconfiguration of the overall monitoring mechanism or trigger corrective actions for recovering from any abnormal conditions.

5.1.2 IaaS layer

Infrastructure Monitoring Service is component collects low level information regarding the execution of the VMUs on the physical nodes and publishes respective reports to the monitoring central Index

5.1.3 Saas Layer

Monitoring service Instance (MSI): It is located within a virtual Machine Unit (VMU) and unique for every set of application components that comprise the application workflow deployment. It exposes an interface towards the Monitoring Framework service in order for latter to initiate the applications monitoring. During that action, configuration Parameters are being transferred from the MFS to MSI related with specific application workflow monitoring. Monitoring Index Service, It serves the role of local repository of the monitoring Index Service are being refreshed every time that the Data Collector provides a new sets of parameters values Data Collector. The outcome of this execution is the set of parameters (name, unit, etc) formatted in plain XML and fed into local Monitoring Index.

5.2 UML Diagrams

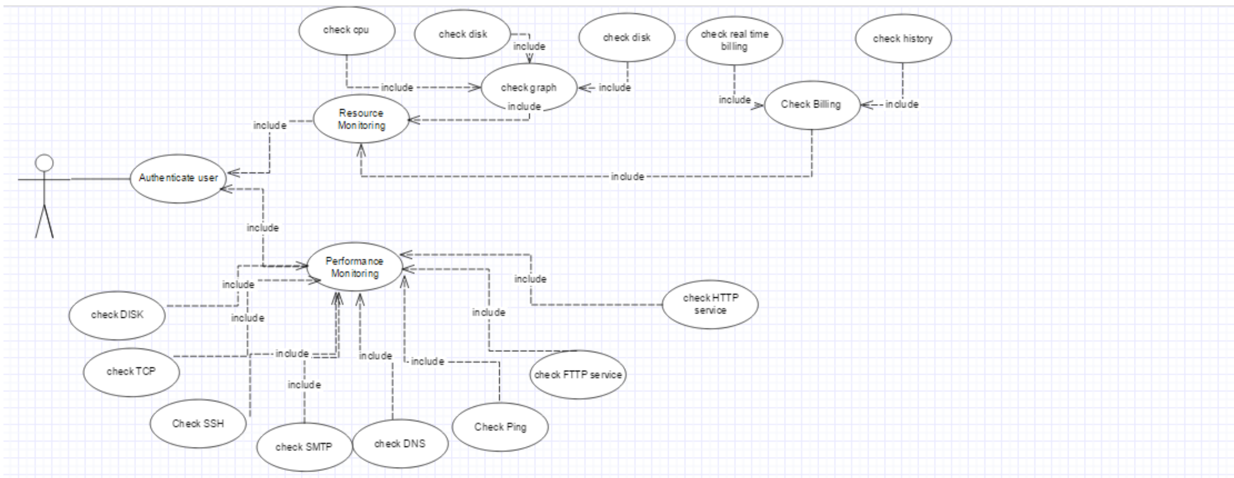

Figure 5.2: Use Case Diagram

This Use Case Diagram tells us about the function of the user once he logins into his Openstack account. The user can be two types:
The privileges defined for the admin and user are different. Admin accesses great authority over the system than remote user.
The user operation can be broadly divided into major classes:
1) Instance Creation Launching instance
2) Monitoring operations
textbf1) Instance Creation: This is the privilege given to admin. In order to create or launch a group of instances he has to create new project. Admin need to select appropriate configurations before launching a particular instance. Once the instance is launched block storage can be externally.

textbf2) Monitoring operations: The monitoring operations can be performed on Instances:

textbfInstance monitoring: status of an instance can be depicted two ways: Either via graphs or via reports. The reports are formed by three basic baselines: monthly, weekly and daily reports. Additional functionality of bill generation where the data collected is used to generate the bill to indicate the cost of the session. The graphs are used to depict real time usage of server resources such as: bandwidth vs. time, cpu vs. time, Load vs. time. This functionality helps us to monitor the change in the status of the applications / Instances and use this data for future use.

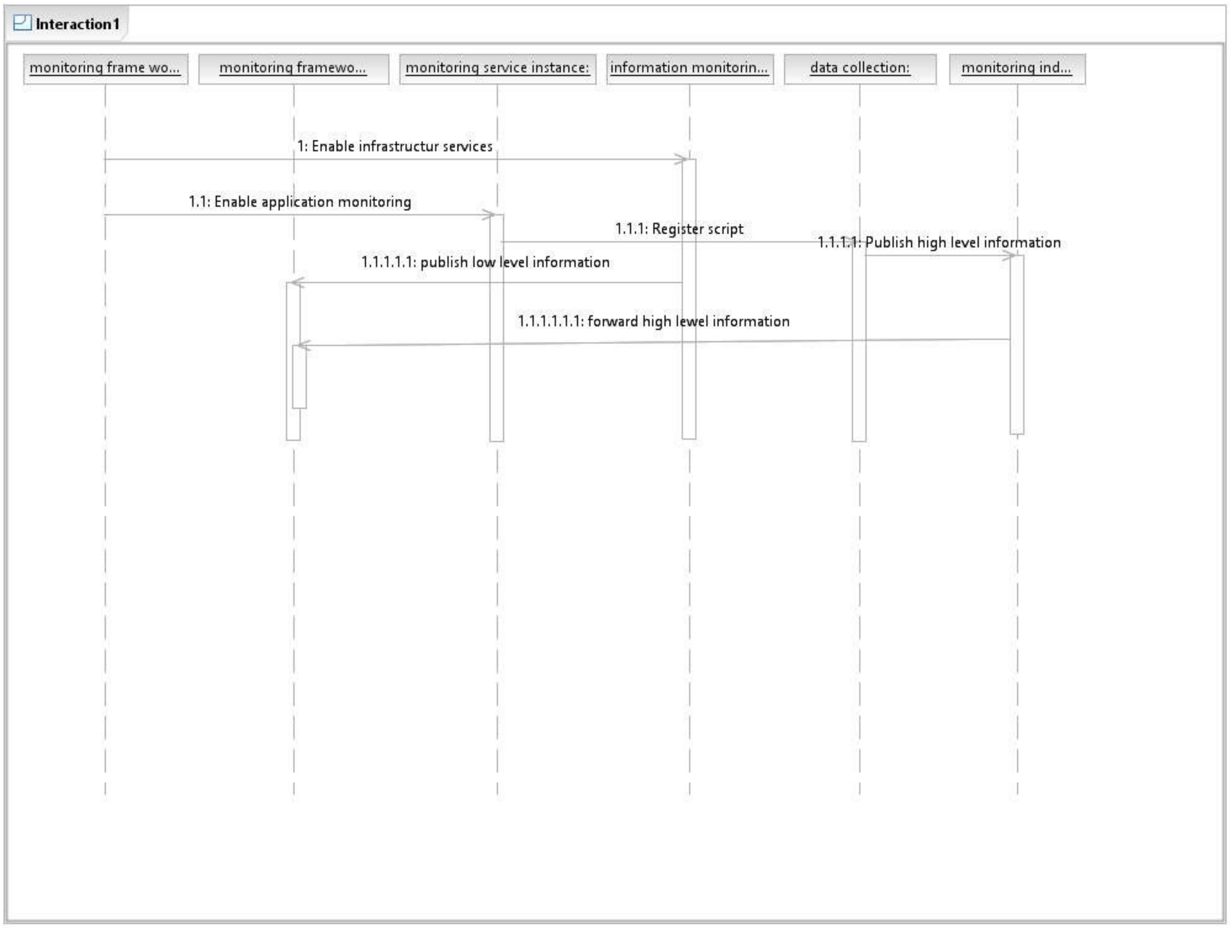

Figure 5.3: Sequence Diagram

The sequence diagram depicts the flow of interaction between the various entities in the sequences. The Monitoring Framework Services gives an asynchronous message to infrastructure monitoring. The Monitoring Framework Services gives an asynchronous message to Monitoring Services Instances to enable application monitoring.

Monitoring Service Instance sends an asynchronous signal to the data collector to register the script. Data Collector also gives asynchronous message to monitoring Index Service to publish high level information. Then Infrastructure Monitoring Service publishes the low level information to Monitoring Central Index and the same is done by Monitoring Central Index where complete monitoring is done.

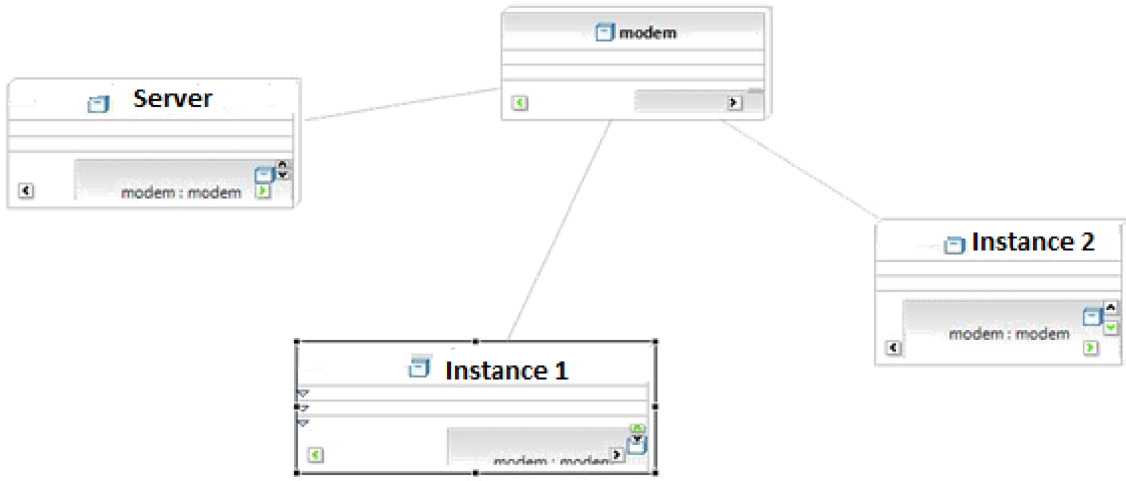

Figure 5.4: Deployment Diagram

The deployment diagram is used to show the hardware devices and the servers that are connected together to perform a specific task. Here, the main components are the private server and machines running the instances. The servers are operated by the user who can be an administrator or remote user respectively. The instances are run by local user who accesses it through internet to perform his task.

Chapter 6

Implementation Details

We have used shell scripts for polling data from ceilometers component in which three types of data(Delta, Gauge, Cumulative) are been stored in the json format for each record which uses MySQL database for its storage. We have converted several records to store in one real time json file which goes on updating and then finally, we have used canvas along with JavaScript for front end.

6.1 Screen-Shots

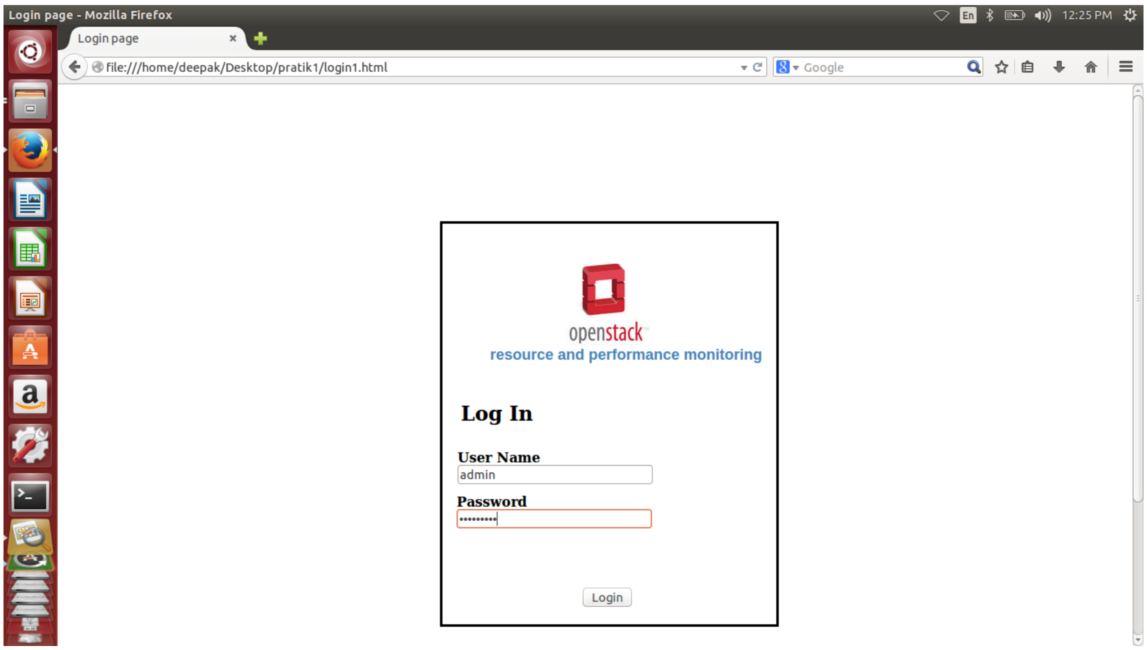

Figure : 6.1 Login Page

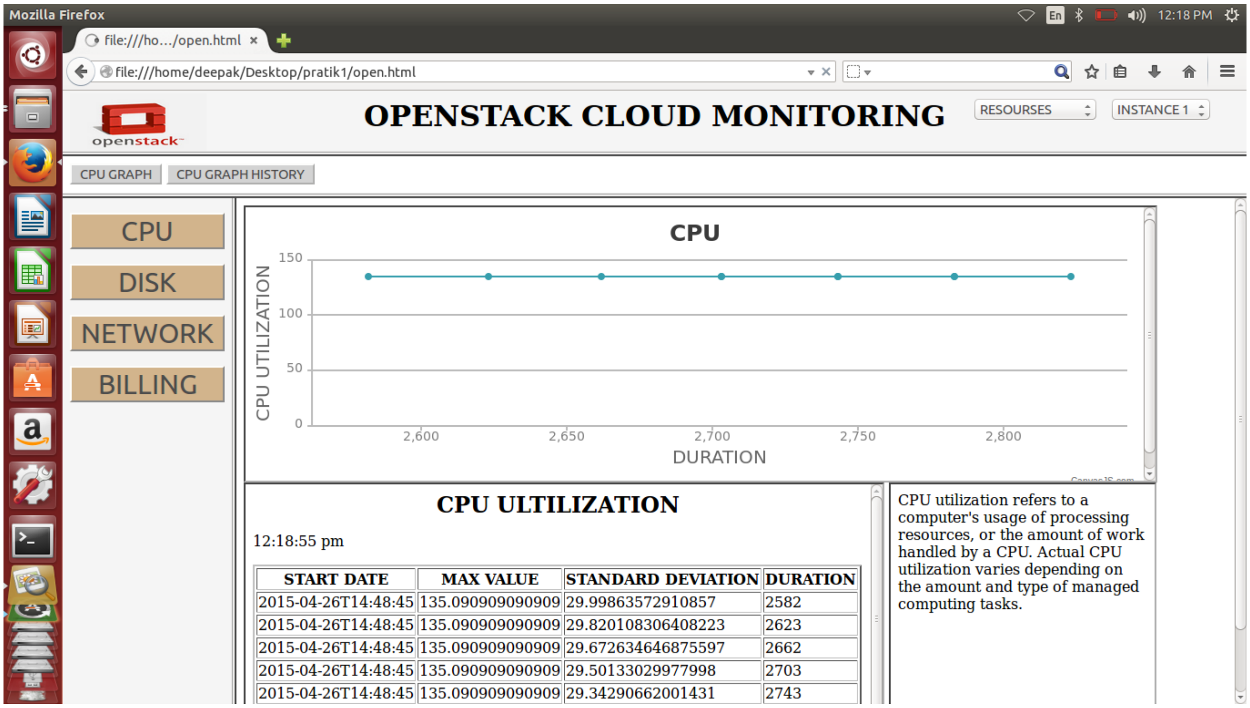

Figure : 6.2 CPU Graph

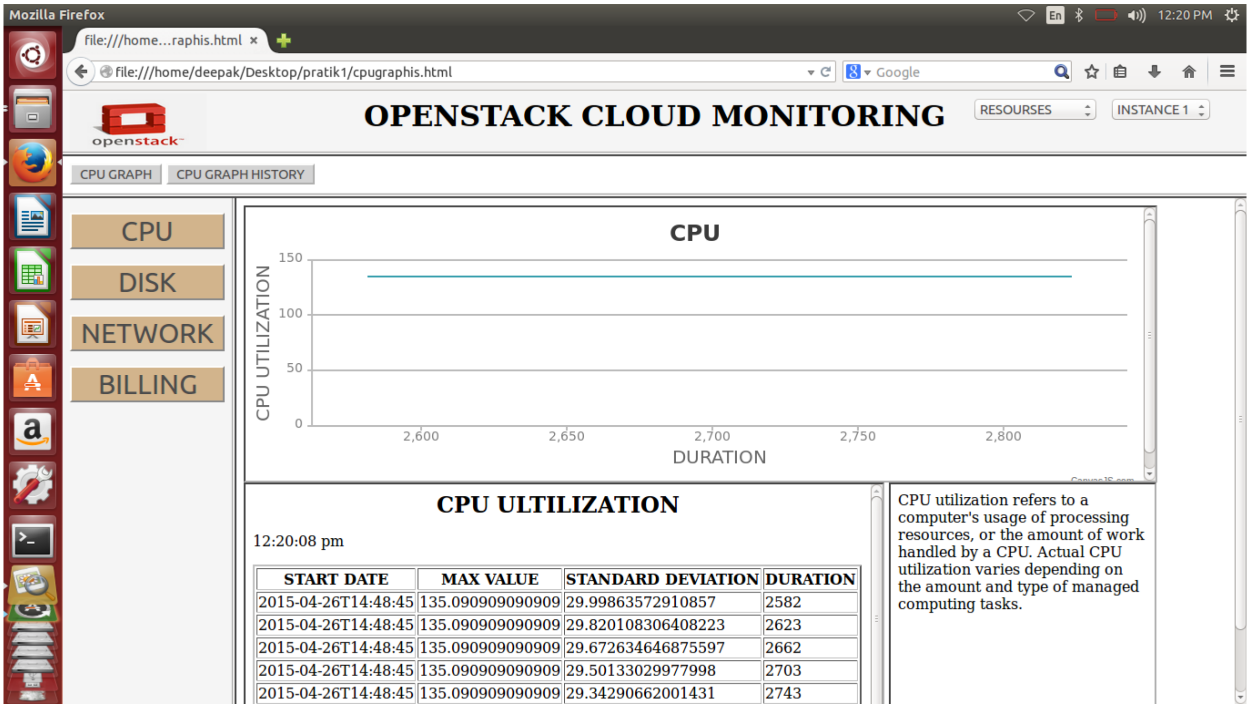

Figure : 6.3 Click On CPU History Button

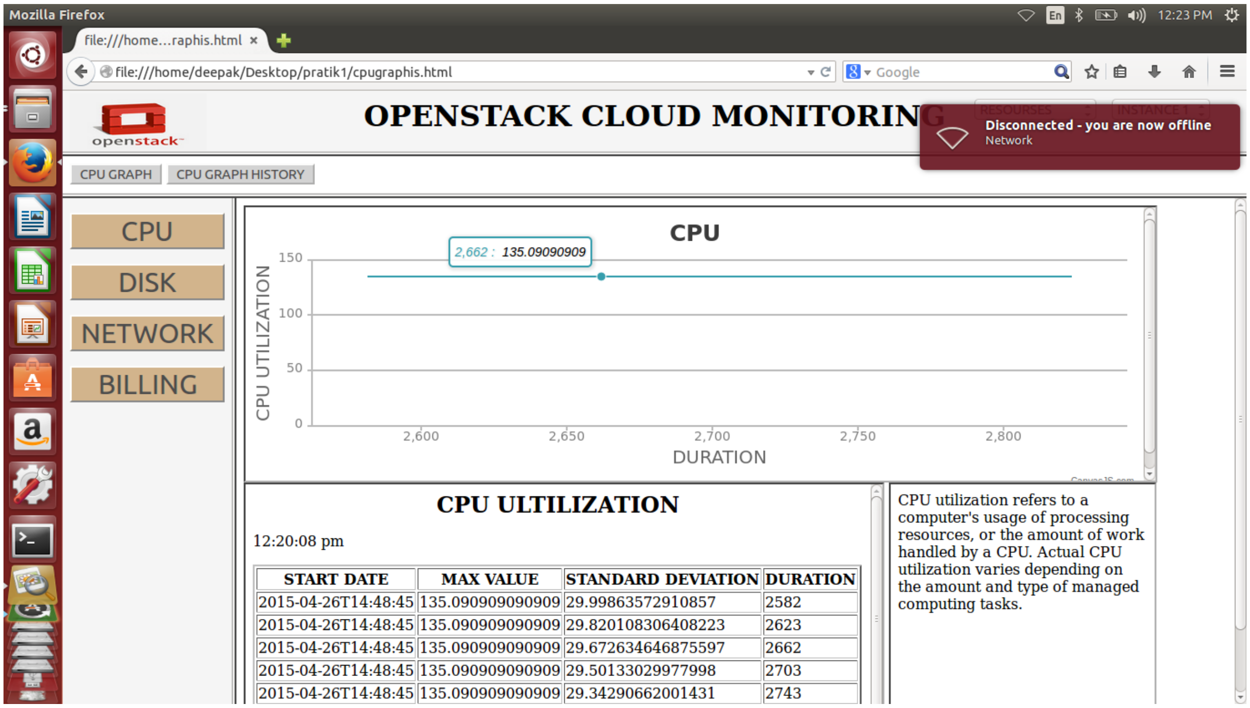

Figure 6.4 CPU History

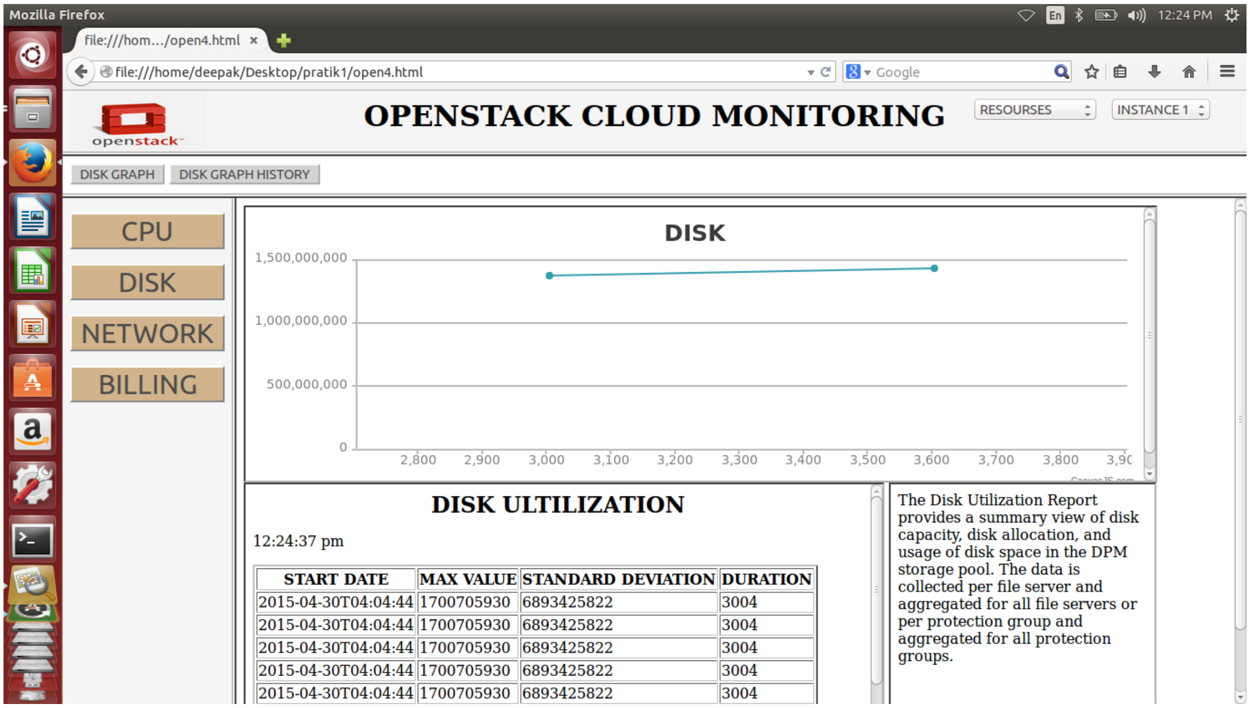

Figure 6.5 Click On DISK Button

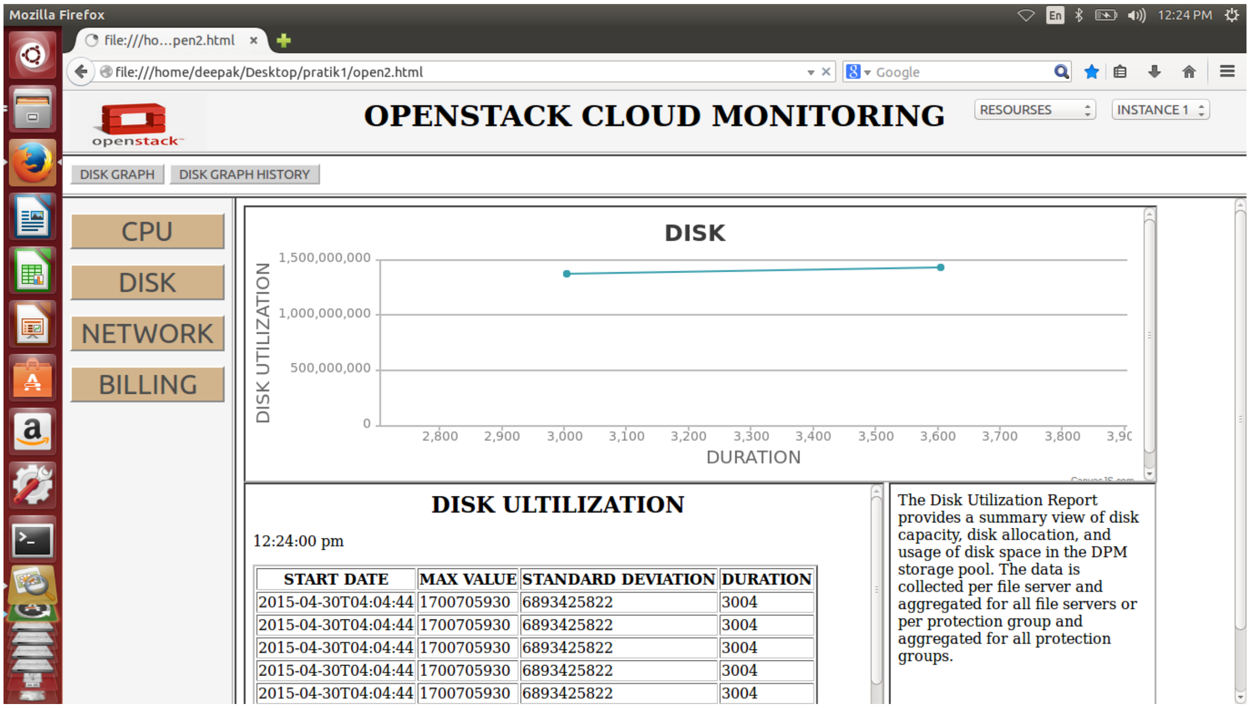

Figure 6.6 DISK Graph

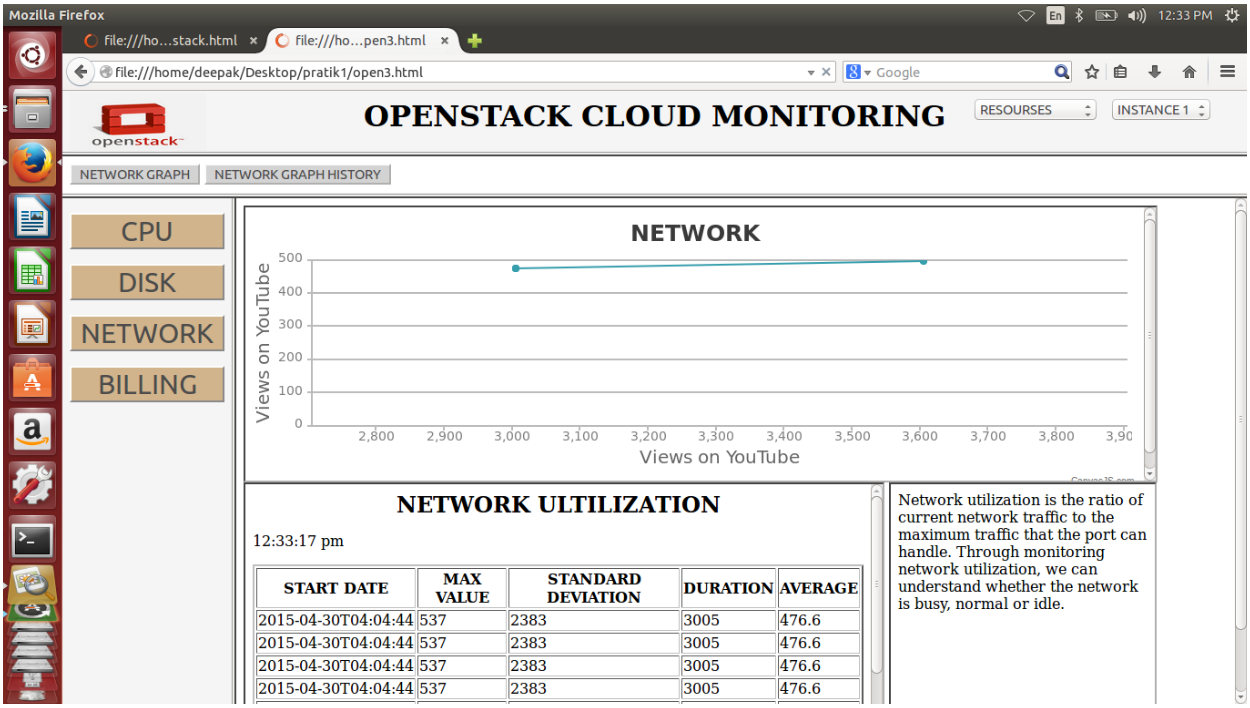

Figure 6.7 Network Graph

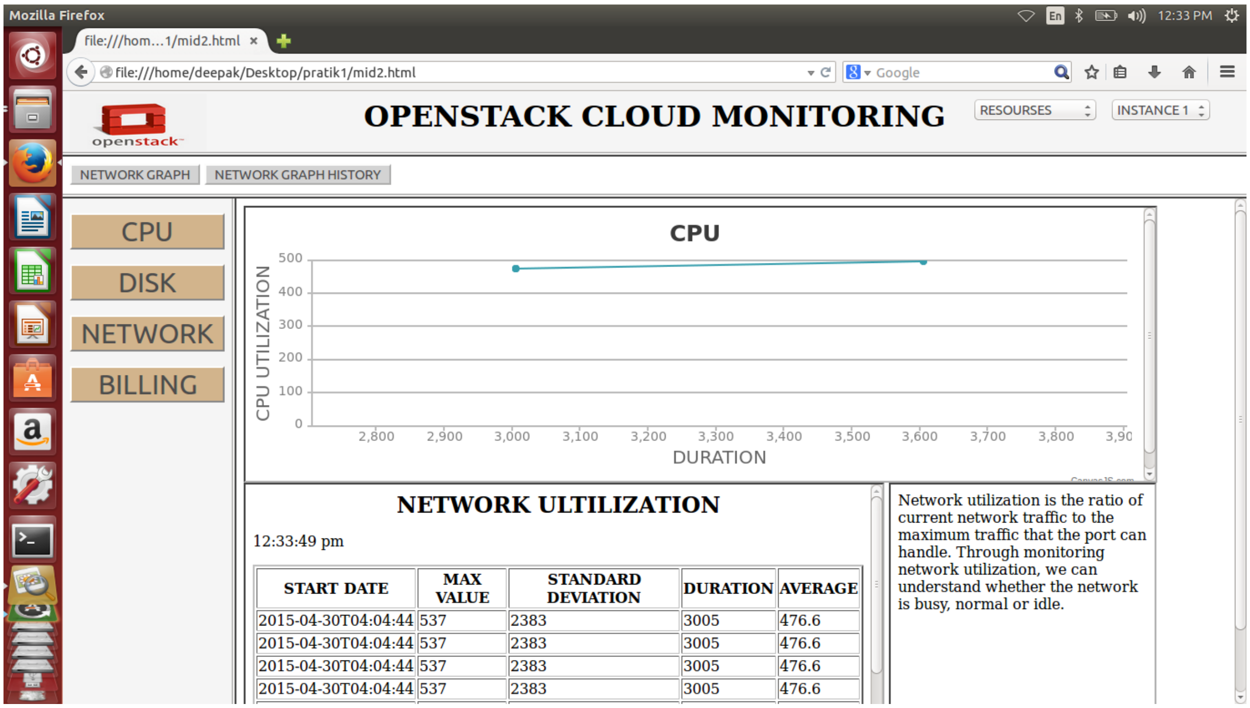

Figure 6.8 Network Graph History

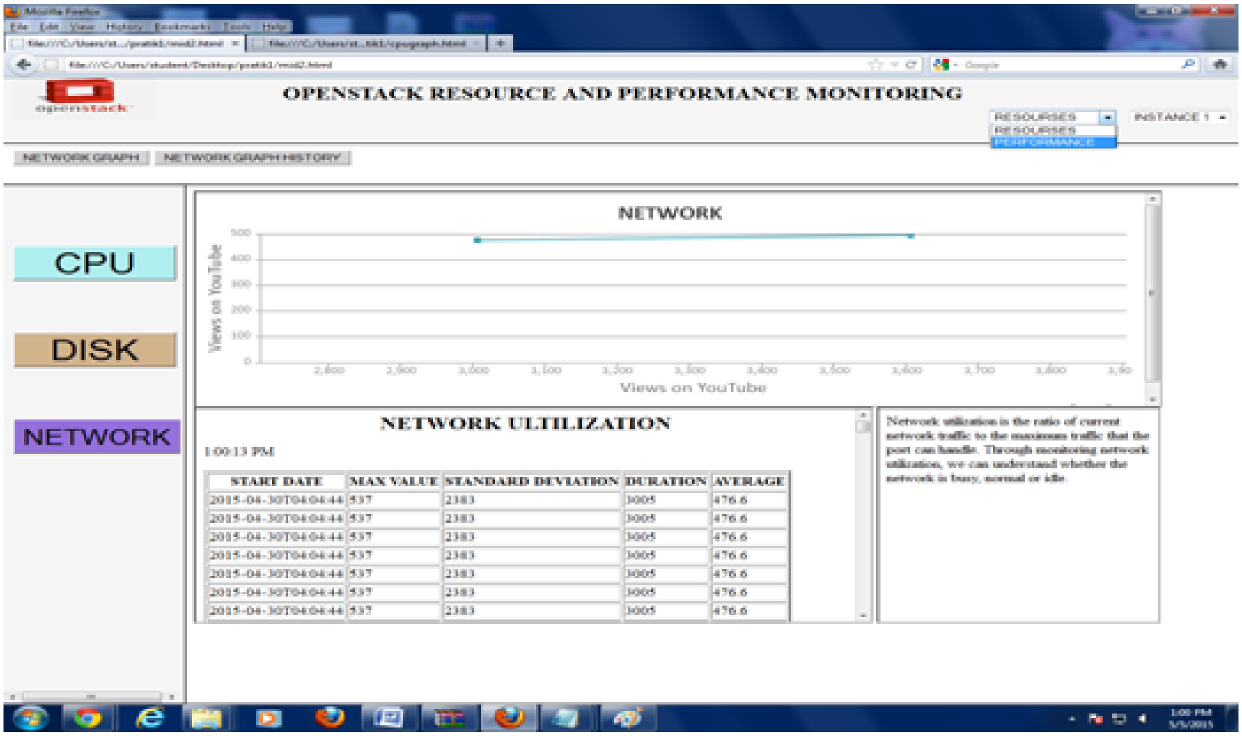

Figure 6.9 Performance Monitoring

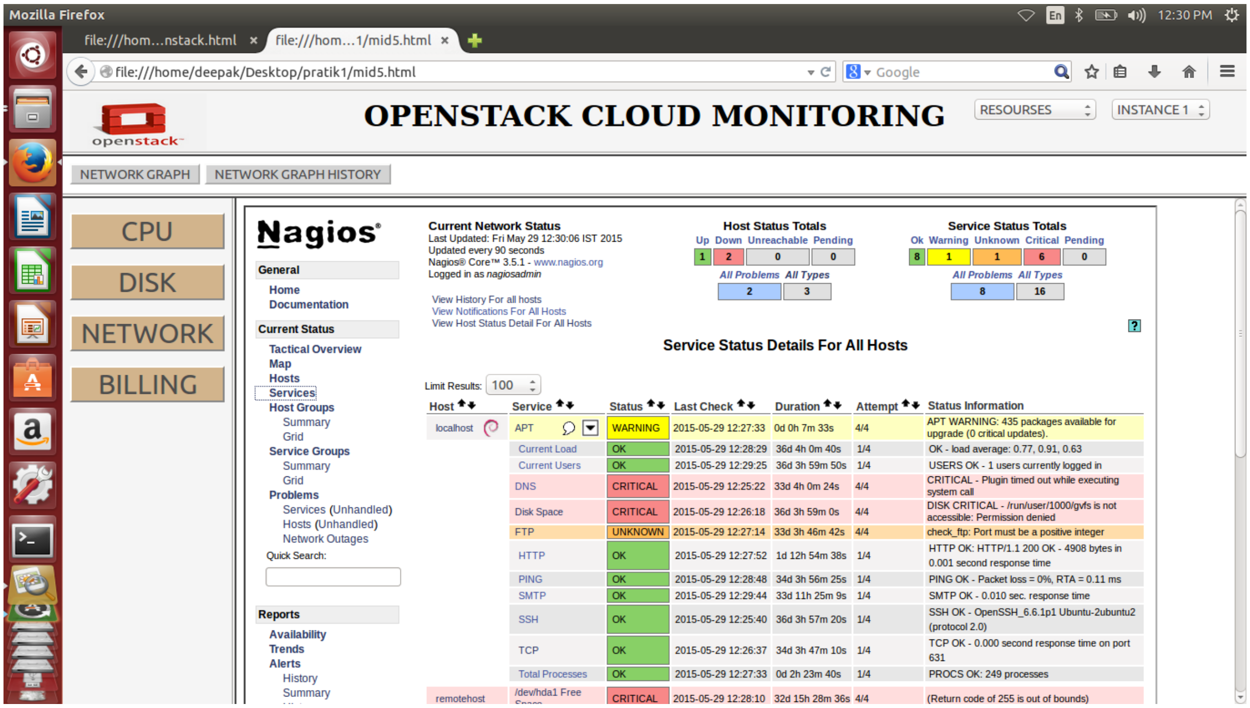

Figure 6.10 Nagios Website

6.2 White-Box Testing

White-box testing is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs.

6.2.1 Unit Testing

Unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures and operating procedures are tested to determine if they are fit for use. The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits.

6.2.2 Integration Testing

Integration testing in the software testing model comes before system testing and after the unit testing has been done. The way that integration testing works is by getting the individual modules that have been through the unit testing phase and integrating each module into the group.

6.2.3 System Testing

System testing is simply testing the system as a whole; it gets all the integrated modules of the various components from the integration testing phase and combines all the different parts into a system which is then tested. Testing is then done on a system as all the parts are now integrated into one system the testing phase will now have to be done on the system to check and remove any errors.

6.3 Black-Box Testing

Specific knowledge of the application's code/internal structure and programming knowledge in general is not required. The tester is aware of what the software is supposed to do but is not aware of how he does it. For instance, the tester is aware that a particular input returns a certain output but is not aware of how the software produces the output in the first place.

6.4 System Components

(1) Web Portal.
(2) MySQL Databases
(3) Openstack Software
(4) Nagios Performance tool

6.4.1 Need Of Components

(1) **Web Portal:** We need user to interact with our System.
(2)**Database:** To store monitoring data generated every second by our Instances and to access it using ceilometer.
(3)**Openstack:** To act as a platform to launch our Instances and help to generate monitoring data.
(4)**Nagios:** With help of this tool we can monitor the performance of the entire instance.

6.4.2 Test Cases

Sr.No	Description	Expected Result	Actual Result	Pass Or Fail
1	Click on the .html file	The browser should open	The browser opens	Pass
2	On the page there should be a text eld for user id	Verify that text box for user id exists	The text box for user id exists	Pass
3	On the page there should be a text eld for password	Verify that text box for password exists	The text box for password exists	Pass
4	Enter the correct user id and password in the user id and password text box respectively and click Log in	The system should allow user to login with correct user id and password only	The system allows user to login with correct user id and password only	Pass
5	Login the Account	The Dashboard should be displayed ;showing the default data and graph of C.P.U. data for Instance 1	The Dashboard is displayed ;showing the default data and graph of C.P.U. data for Instance 1	Pass

Sr.No	Description	Expected Result	Actual Result	Pass Or Fail
6	Click on the Disk tab	The real-time graph for Disk Usage should be shown	The real-time graph for Disk Usage is shown	Pass
7	Click on the HISTORY button For DISK	A table showing the DISK data should be displayed.	A table showing the DISK data is displayed.	Pass
8	Click on the REAL TIME DATA button for DISK	A table showing the real time DISK data should be displayed	The system allows user to login with correct user id and password only	Pass
9	Click on the Network tab	The real-time graph for Network Usage should be shown	The real-time graph for Network Usage is shown	Pass
10	Click on the HISTORY button for NETWORK .	A table showing the NETWORK data should be displayed	A table showing the NETWORK data is displayed.	Pass
11	Click on the REAL TIME DATA button for NETWORK	A table showing the real time NETWORK data should be displayed	A table showing the real time NETWORK data should be displayed	Pass
12	Change the instance	Its real-time data (graphs and tables) should be accessible	Its real-time data (graphs and tables) is accessible	Pass
13	Change the choice to PERFORMANCE button	Shows the entire Performance summary of the instance	Shows the entire Performance summary of the instance.	Pass

Chapter 7

System Requirement and Specification

7.1 Project Scope

The openstack performance monitoring tool performs monitoring of various instances application wise. The requirment specified in the SRS document considered with oragnization need for a new performance monitoring tool which reports Infrastrcture usage of the instance and compares them for making feasible for the administrator to make intelligent decisions by giving him appropriate suggestion .It also used by end users to know his performance usage and billings costs and pay for what you use.

7.1.1 System Requirements (Hardware)

Controller Node:

1 processor, 2 GB memory, and 5 GB storage

Compute Node:

Compute Node: 1 processor, 512 MB memory, and 5 GB storage

Compute Node:

1 processor, 2 GB memory, and 10 GB storage

CPU :

64 bit processor

Networking:

1 Gbps or 10 Gbps is suggested internally.

For OpenStack Object Storage, an external network should connect the outside world to the proxy servers, and the storage network is intended to be isolated on a private network or multiple private networks.

7.1.2 System Requirements (Software)

Operating system: OpenStack Object Storage currently runs on Ubuntu, RHEL, CentOS, Fedora, openSUSE, or SLES.

7.1.3 User Requirements (Hardware)

CPU :

1)64 bit processor
2)at least 500 MHz
3)Ram: at least 512 MB

7.1.4 User Requirements (Software)

Browser:

Google Chrome, Mozilla Firefox, Internet Explorer, etc.

7.1.5 Risk Analysis

Risk Monitoring involves regularly assessing each of the identified risk to decide whether or not the risk is becoming more or less probable and whether the effects of the risk have been changed. Risk Monitoring was a continuous process throughout the development phase.

7.1.6 Risk Planning

For scheduling risk we had decided the strategy that as everything is not properly defined, any requirement which comes to us will be kept by us and we will check its feasibility.
There is no alternative of the process risk; we need to study current structure and standard of the current system deeply.
Changing requirement risk, strategy has been decided that traceability information should be maintained every time the requirement added.

Chapter 8

Planning and Scheduling

Sr. No.	Deliverables	Submission Date	Review Date
1	Group Formation	2nd Week of June	-
2	Company Sponsorship	1st Week of July	-
3	Guide Allocation	2nd Week of July	-
4	Topic Selection	2nd week of August	3rd week of August
5	Literature Survey	3rd Week of Aug-2nd week of Sept	2nd week of September
6	Design	3rd week of September	1st week of October
7	Design Report	2nd Week of October	2nd Week of October
8	Implementation	2nd week of March	2nd week of March
9	Testing	2nd week of April	2nd week of April
10	Final Report Generation	30th April 2015	29th May 2015

References

1) On Measuring Disturbances in the Force: Advanced Cloud Monitoring Systems/ L.M. Vaquero, S. Sae Lor, J.M. Alcaraz-Calero, D. Niyato, S. Clayman, D. Audsin, R. Nadarajan/Received 28 July 2011/Received in revised form29 November 2011/Accepted 30 November 2011/Available online 8 December 2011
2) A service framework for energy-aware monitoring and VM management in Clouds Gregory Katsaros a,, Josep Subirats b, J. Oriol Fit b, Jordi Guitart b, Pierre Gilet c, Daniel Espling d/Received 3 April 2011/Received in revised form18 September 2012/Accepted 7 December 2012/Available online 21 December 2012
3) A Self-adaptive hierarchical monitoring mechanism for Clouds/Gregory Katsaros, George Kousiouris, Spyridon V. Gogouvitis, Dimosthenis Kyriazis, Andreas Menychtas, Theodora Varvarigou School of Electrical and Computer Engineering, National Technical University of Athens, 15773 Athens, Greece/ Received 28 July 2011/Received in revised form/29 November 2011/Accepted 30 November 2011/Available online 8 December 2011
4) David Josephsen-BUILDING A M ONITORING INFRASTRUCTURE WITH NAGIOS-ISBN 0- 132-23693-1-Prentice Hall-Upper Saddle River, NJ , Boston , Indianapolis , San Francisco ,New York ,Toronto , Montreal , London , Munich ,Paris ,Madrid , Cape Town , Sydney , Tokyo , Singapore , Mexico City-February 2007
5) User Guide:OpenStack Architecture Design Guide,October 11, 2014
6) Oleg Litvinski and Abdelouahed Gherbi-Experimental Evaluation of OpenStack Compute Scheduler-ScienceDirect Procedia Computer Science 19 (2013) 116 123 L.M. Vaquero, S. Sae Lor, J.M. Alcaraz-Calero, D. Niyato, S. Clayman, D. Audsin, R. Nadarajan- On Measuring Disturbances in the Force: Advanced Cloud Monitoring Systems-Editorial / Future Generation Computer Systems 29 (2013) 20072008

7) Dirk Paessler, Paessler AG-Monitoring private clouds-Network Security, November 2012
8) Gregory Katsaros , George Kousiouris, Spyridon V. G ogouvitis, Dimosthenis Kyriazis, Andreas Menychtas, Theodora Varvarigou-A Self-adaptive hierarchical monitoring mechanism for Clouds-The Journal of Systems and Software 85 (2012) 1029 1041-Received 28 July 201/Received in revised form /29 November 2011 /Accepted 30 November 2011 /Available online 8 December 2011
9) Md. Tanzim Khorshed, A.B.M. Shawkat Ali , Saleh A. Wasimi-A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing-Future Generation Computer Systems 28 (2012) 833851- Received 15 August 2011,Received in revised/form 11 January 2012 ,/Accepted 18 January 2012 ,/Available online 27 January 2012
10) A.Meera a *, S.Swamynathan b-Agent based Resource Monitoring system in IaaS Cloud Environment-Selection and peer-review under responsibility of the University of Kalyani, Department of Computer Science and Engineering doi: 10.1016/j.protcy.2013.12.353A.
11) Giuseppe Aceto, Alessio Botta, Walter de Donato, Antonio Pescap-Cloud monitoring: A survey Received 25 October 2012/ Received in revised form 15 February 2013 /Accepted 3 April 2013 Available online 10 April 2013
12) Sonali Yadav-Comparative Study on Open Source Software for Cloud Computing Platform: Eucalyptus, Openstack and Opennebula/Research Inventy: International Journal Of Engineering And Science Vol.3, Issue 10 (October 2013), PP 51-54 Issn(e): 2278-4721, Issn(p):2319-6483, www.Researchinventy.Com
13) Kevin Jackson Cody Bunch-OpenStack Cloud Computing Cookbook Second Edition First published: September 2012 /Second Edition: October 2013 Production Reference: 2111013 Published by Packt Publishing Ltd. Livery Place 35 Livery StreetBirmingham B3 2PB, UK. ISBN 978-1-78216-758-7
14) James Turnbull-Pro Nagios 2.0 ISBN-13: 978-1-59059-609-8 /ISBN-10: 1-59059-609-9 Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1 Publication-2006
15) John B. Carter, John K. Bennett ,Willy Zwaenepoel -Implementation and Performance of Munin Appeared in Proceedings of the Thirteenth Symposium on Operating System Principles, pp. 152164, October 1991.